PhishSpot Manual

Administrator Guide

2026-06-02

Introduction

Welcome to PhishSpot, a comprehensive phishing simulation and security awareness training platform. This manual covers all features available to account admins. It is designed for non-technical users and walks through every section of the platform step by step.

PhishSpot allows your organization to run realistic phishing campaigns, track employee responses, deliver security awareness training, and measure your team's resilience to social engineering attacks over time.

1.1 What Is PhishSpot?

PhishSpot is a SaaS platform that helps organizations test and improve their employees' ability to recognize phishing emails. As an admin, you can create simulated phishing campaigns that mimic real-world attacks, send them to your team, and then track who opened the email, who clicked the link, and who submitted sensitive information on a fake landing page. After a click, employees can be redirected to a training course to improve their awareness.

1.2 Who Is This Manual For?

This manual is written for account-level admins. If you have the Admin role on your team, you have full access to all features described in this guide: managing campaigns, contacts, templates, domains, courses, team members, reports, and account settings.

1.3 User Roles Overview

PhishSpot uses three user roles within each account. Your role determines what you can see and do:

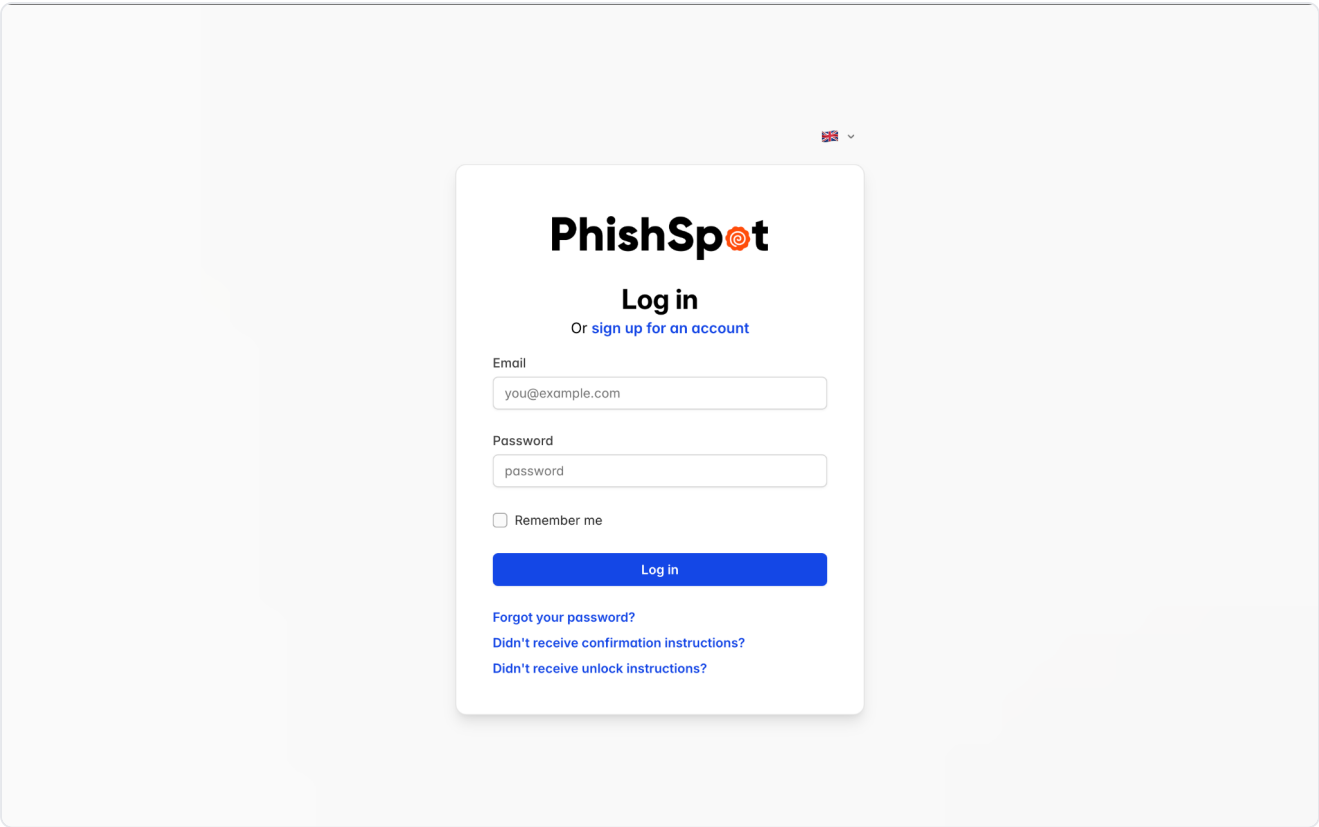
Role	Access Level	Key Permissions
Admin	Full access	Everything: campaigns, contacts, templates, team management, account settings, domains, courses, media, webhooks, reports
Editor	Content access	Campaigns, contacts, courses, templates, media, reports. Cannot manage team members or account settings
Member	Read-only	Can view campaigns, contacts, and reports but cannot create or modify anything

> This manual focuses exclusively on the Admin role. Editors and Members will see fewer menu items and action buttons.

Getting Started

2.1 Logging In

Navigate to your PhishSpot platform URL in your browser. Enter your email address and password on the login page and click the Sign In button.

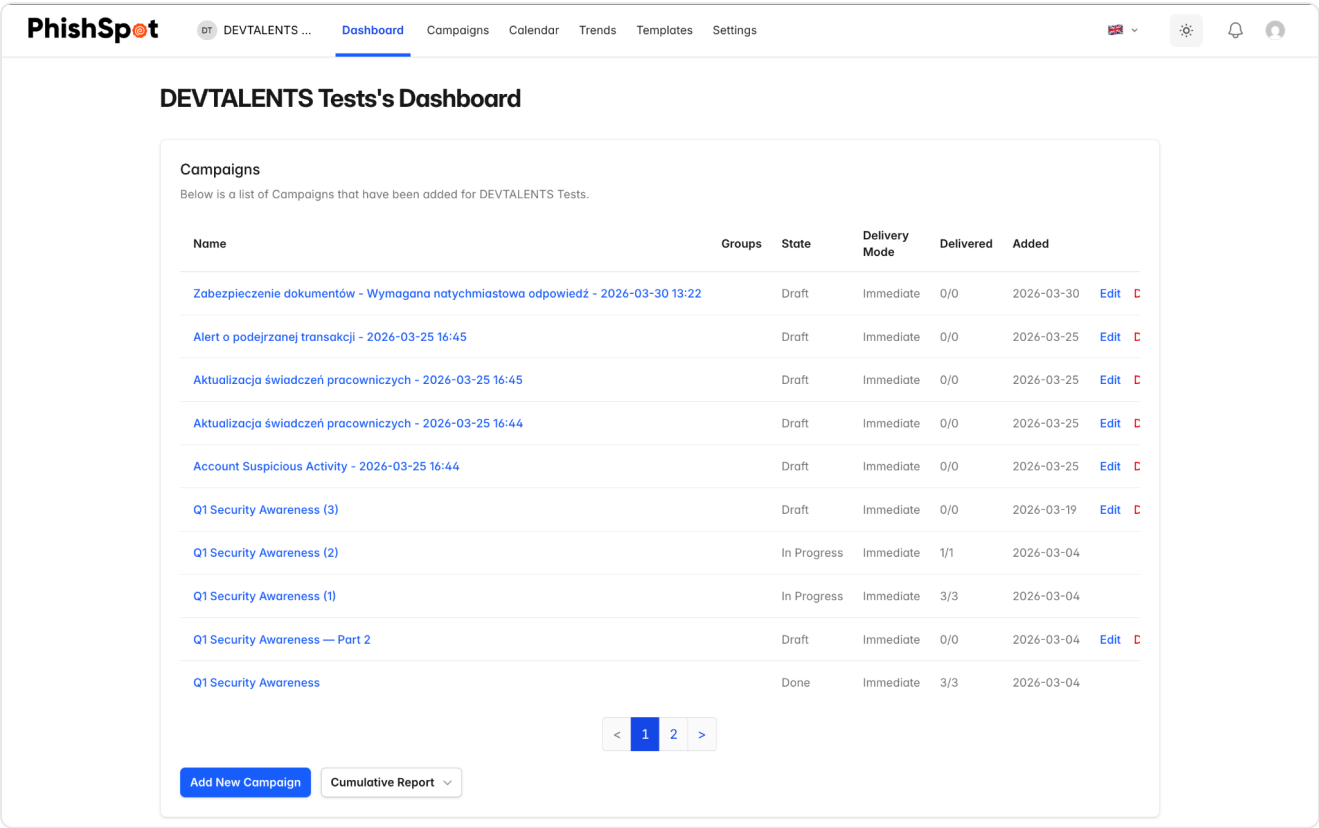


If your organization has enabled two-factor authentication (2FA), you will be prompted to enter a verification code from your authenticator app after entering your password.

2.2 Platform Navigation

After logging in, you will see the main dashboard. The platform uses a horizontal top navigation bar that appears on every page. The navigation bar contains:

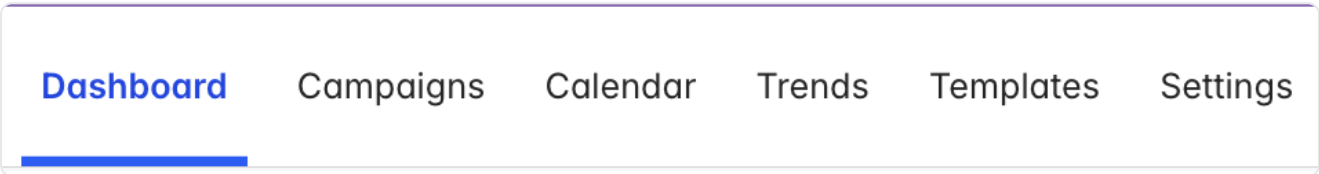
- **Left side** — PhishSpot logo (links to homepage) and the account/team switcher (shows your current team name).
- **Center** — Main navigation links: Dashboard, Campaigns, Calendar, Trends, Templates, and Settings. The active page is highlighted with a blue underline.
- **Right side** — Language switcher (flag icon), theme toggle (light/dark mode), notification bell, and your user avatar/profile menu.



2.3 Navigation Menu Structure

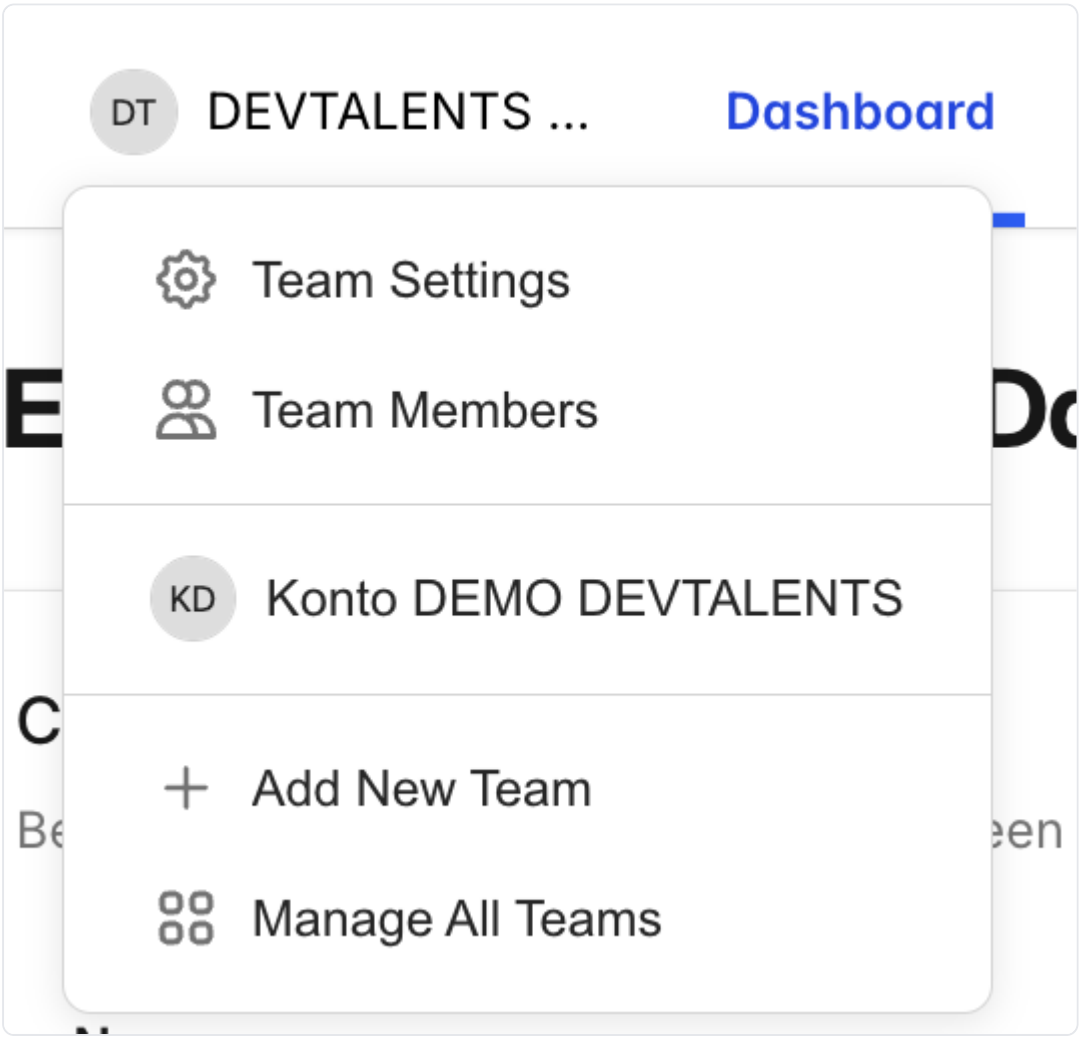
The top navigation bar contains the following main sections:

Menu Item	Description
Dashboard	Overview of your account with recent campaigns and quick stats
Campaigns	Create, manage, and monitor phishing simulation campaigns
Campaign Calendar	Visual calendar view of scheduled and past campaigns
Trends	Historical trend data and analytics across all campaigns
Templates	Browse curated phishing templates and manage your custom templates
Settings	Expandable section containing: Contacts, Groups, Courses, Media, Domains, Platform Domains, Webhooks, Account Details, Team Members



2.4 Switching Accounts

If you belong to multiple teams or accounts, you can switch between them using the account switcher at the top of the sidebar. Click your current account name to see a dropdown of all accounts you have access to, then select the one you want to work with.



2.5 Notifications

The notification bell in the top-right corner shows a count of unread notifications. Click it to see a list of recent events such as campaign completions, new team member joins, or domain verification updates. Each notification links to the relevant item. You can mark all notifications as read using the link at the top of the notification panel.

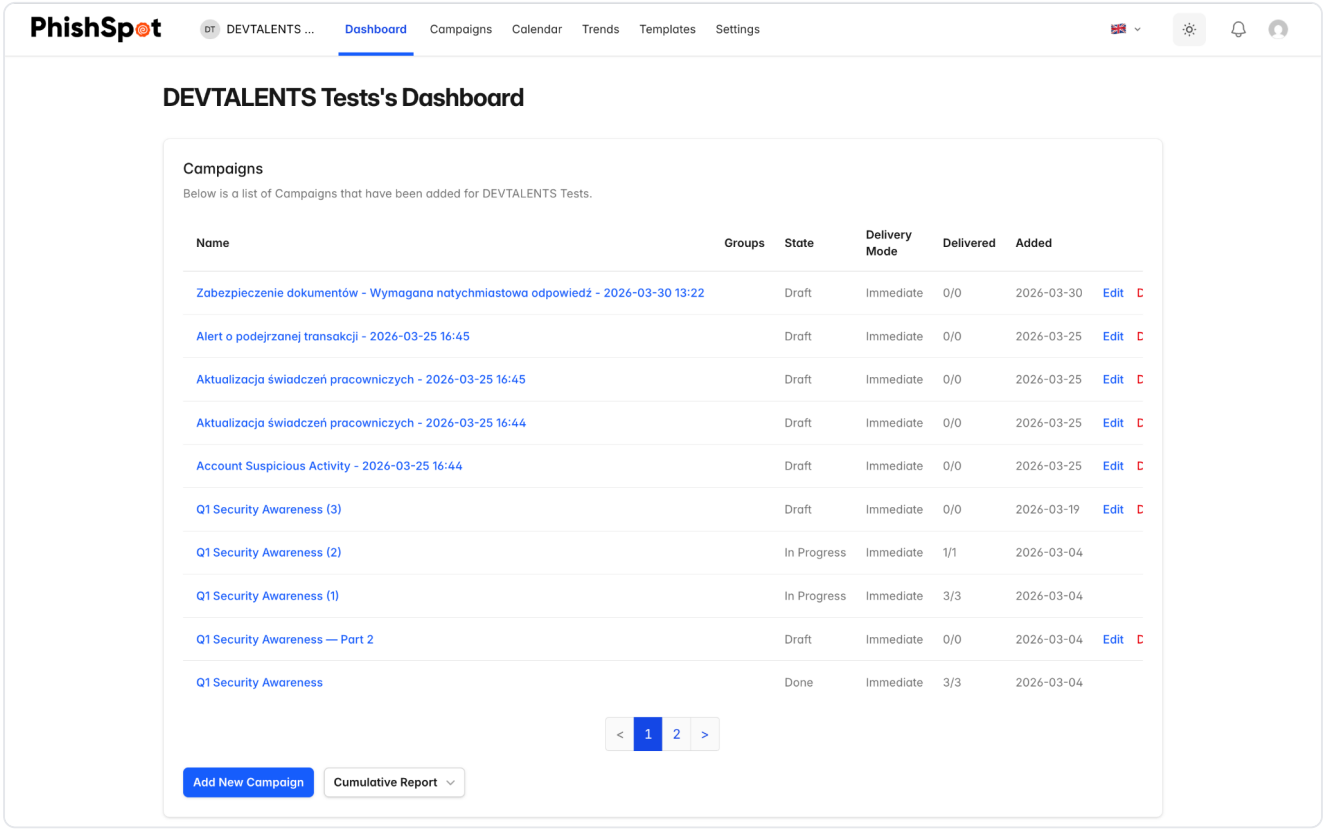
Dashboard

The Dashboard is your home screen after logging in. It provides a high-level overview of your account activity.

3.1 Dashboard Components

The dashboard displays:

- A summary of your recent campaigns with their current status (Draft, Scheduled, Active, Paused, Done)
- Quick access to platform domains assigned to your account
- Your media library overview



From the dashboard, you can quickly navigate to any campaign by clicking its name, or create a new campaign using the button in the campaigns section.

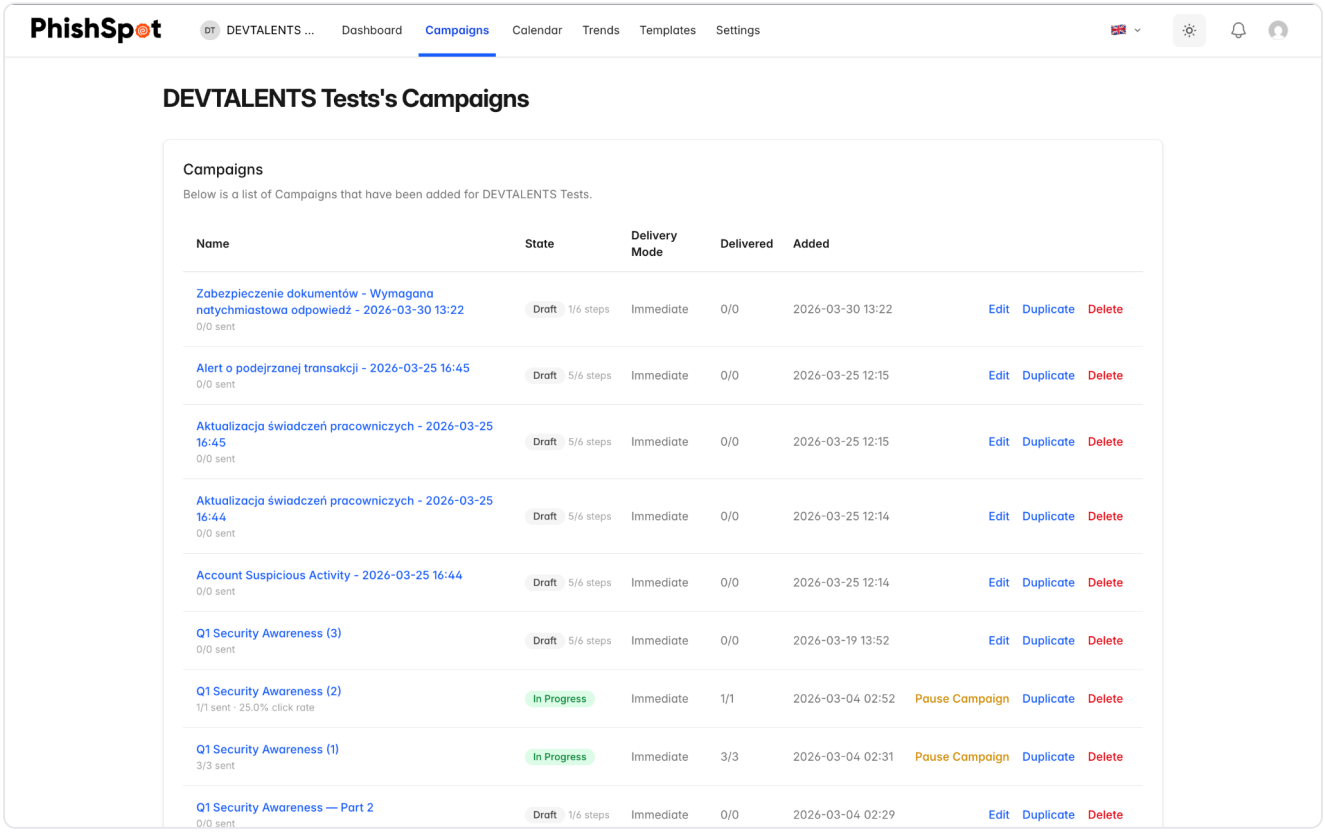
Campaigns

Campaigns are the core of PhishSpot. A campaign is a simulated phishing exercise where you send a crafted email to a group of contacts, host a fake landing page to capture interactions, and optionally redirect users to a training course after they click.

4.1 Campaigns List

Navigate to Campaigns from the sidebar to see all campaigns in your account. The list displays:

Column	Description
Name	The campaign name (clickable to view details)
State	Current status: Draft, Scheduled, Active, Paused, or Done
Delivery Mode	How emails are sent: Immediate, Scheduled, or Staggered
Delivered	Number of emails sent out of total recipients
Created At	When the campaign was created
Actions	Buttons for Edit, Duplicate, and Delete



From this page you can also generate a Cumulative Report PDF that combines data from multiple campaigns.

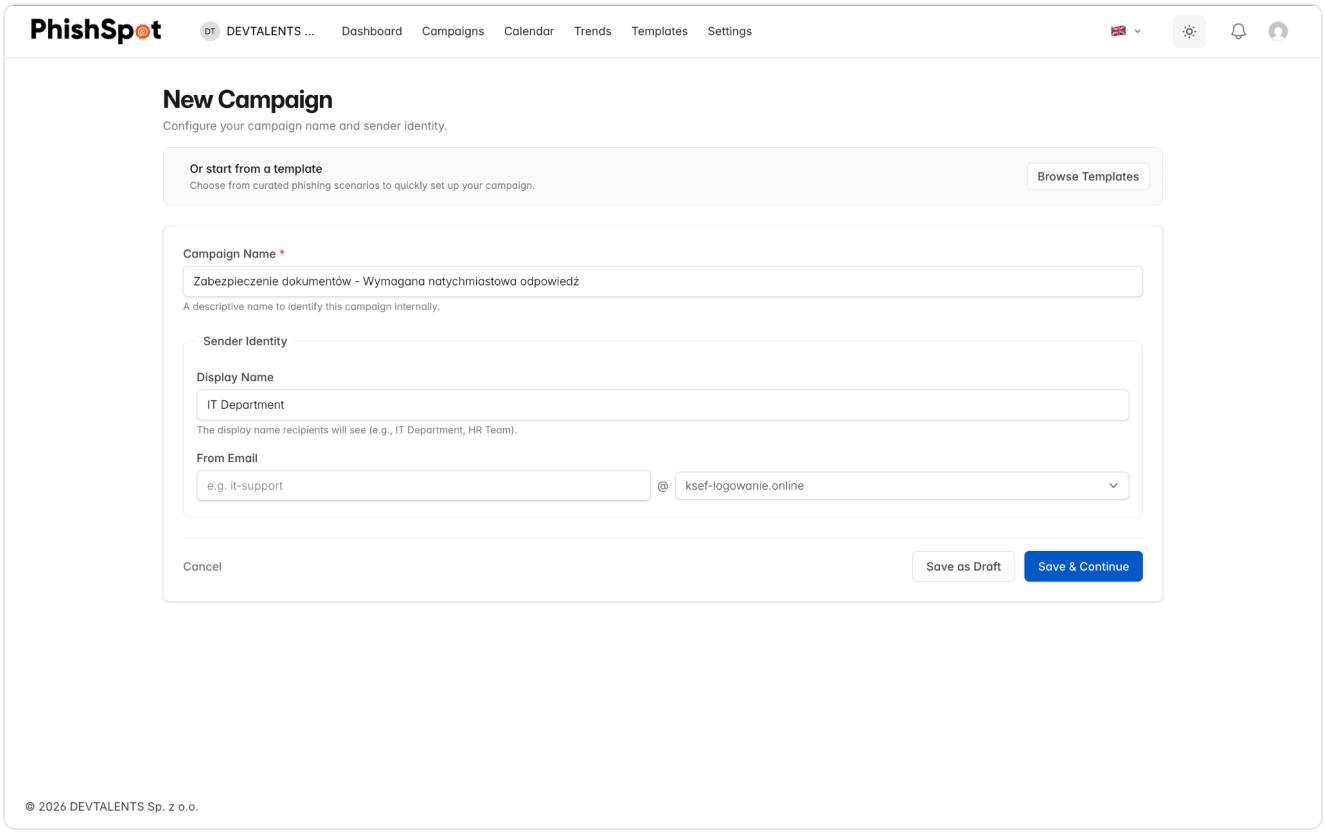
4.2 Creating a New Campaign

Click the New Campaign button to start the campaign creation wizard. The wizard guides you through six steps:

Step 1: Settings

Configure the basic campaign parameters:

- **Campaign Name** — Give your campaign a descriptive name (e.g., “Zabezpieczenie dokumentów - Wymagana natychmiastowa odpowiedź”).
- **Sender Identity** — Provide Display Name and From Email
  - **Display Name** — Provide Display Name
  - **From Email** — Provide Email Username and Select Domain

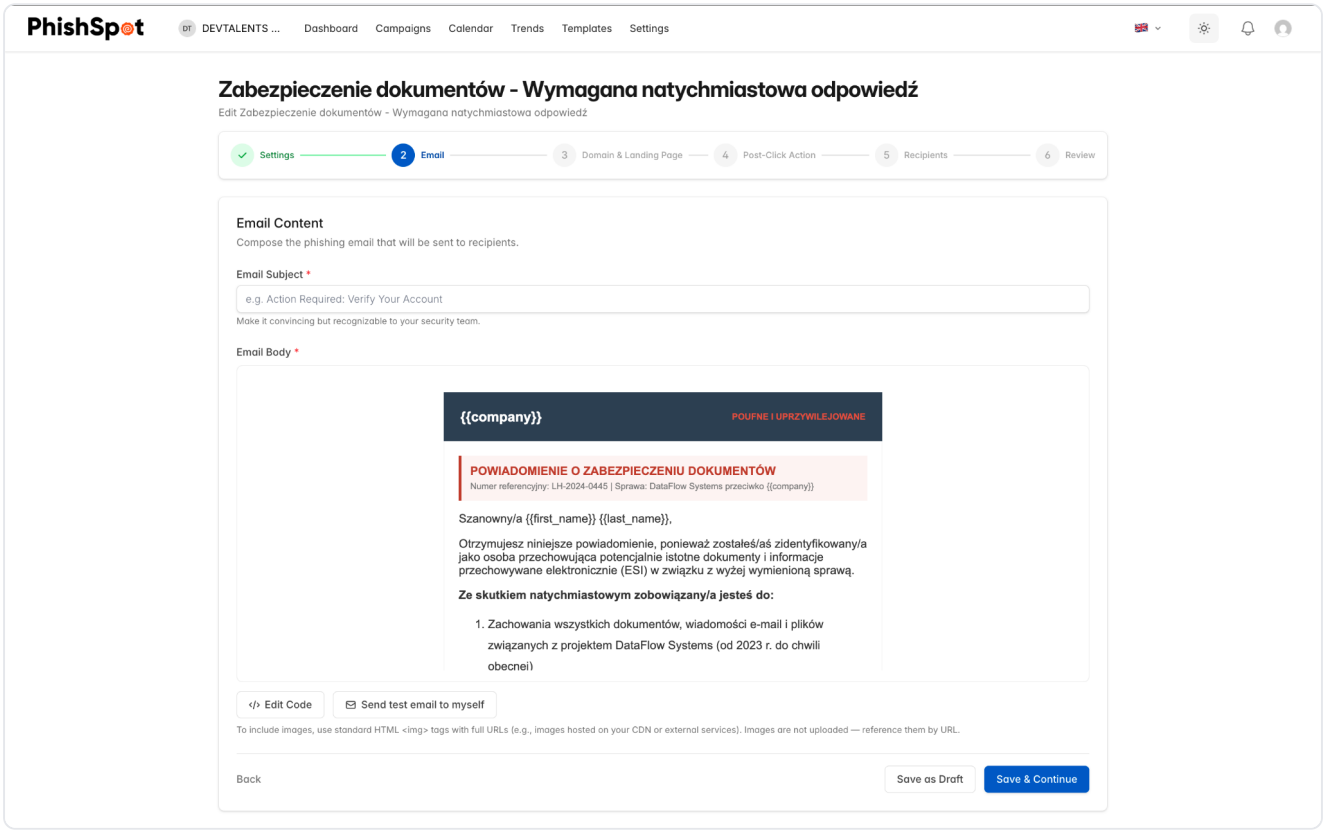


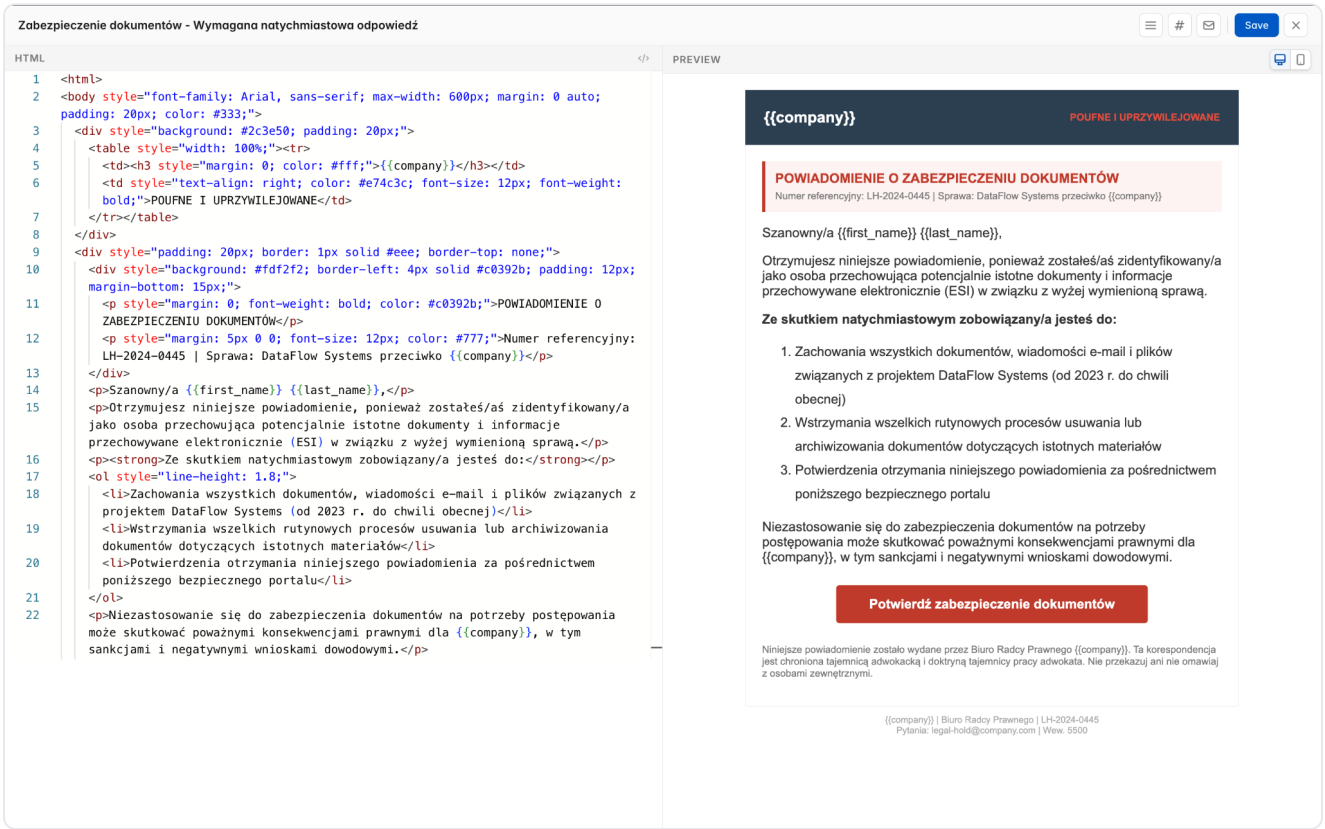
Step 2: Email Content

Design the phishing email your targets will receive:

- **Email Subject** — The subject line of the phishing email.
- **Email Content** — The HTML body of the email. Click the Edit Code button to open the Monaco code editor where you can write or paste HTML. The editor provides syntax highlighting and a live preview.

You can use template variables in both the subject and body to personalize emails. Common variables include the recipient's first name, last name, and email.



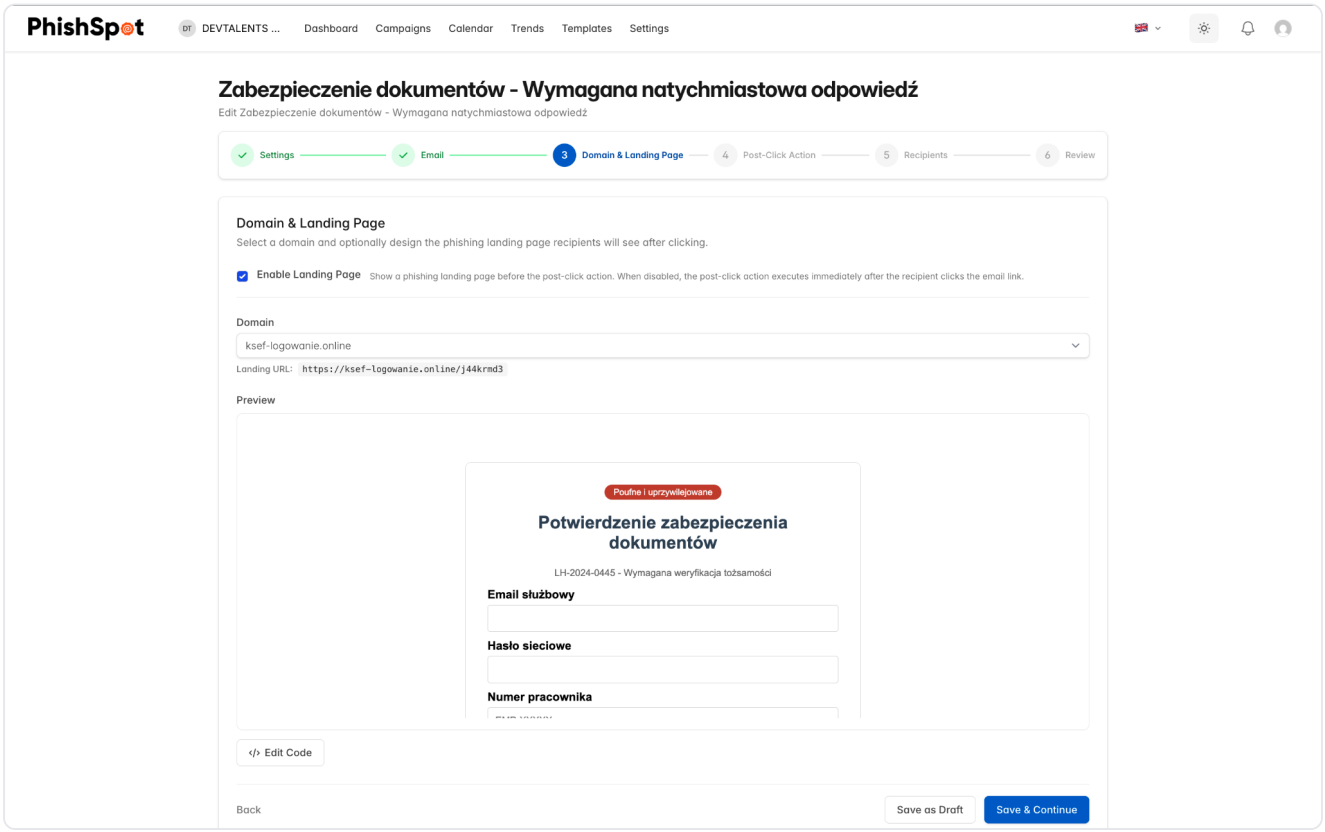


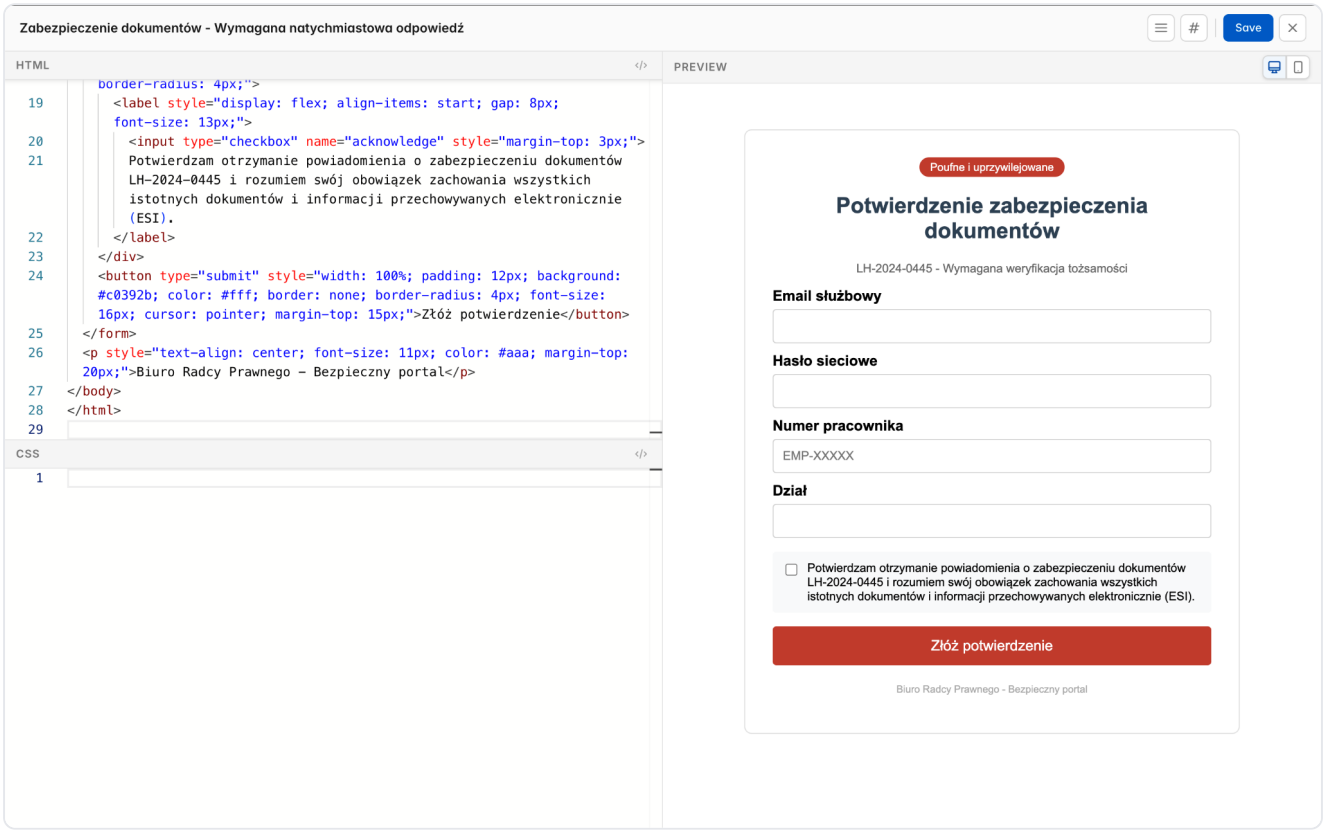
> Always send a test email to yourself before launching a campaign to verify the formatting and links work correctly.

Step 3: Landing Page

Configure the fake landing page that recipients will see when they click the phishing link:

- **Platform Domain** — Select which domain to use for your landing page URL.
- **Landing Page Content** — Edit the HTML of the landing page using the visual editor. This is the page that mimics a legitimate login form or similar.
- **Enable/Disable Landing Page** — You can choose to track clicks only without hosting a landing page.

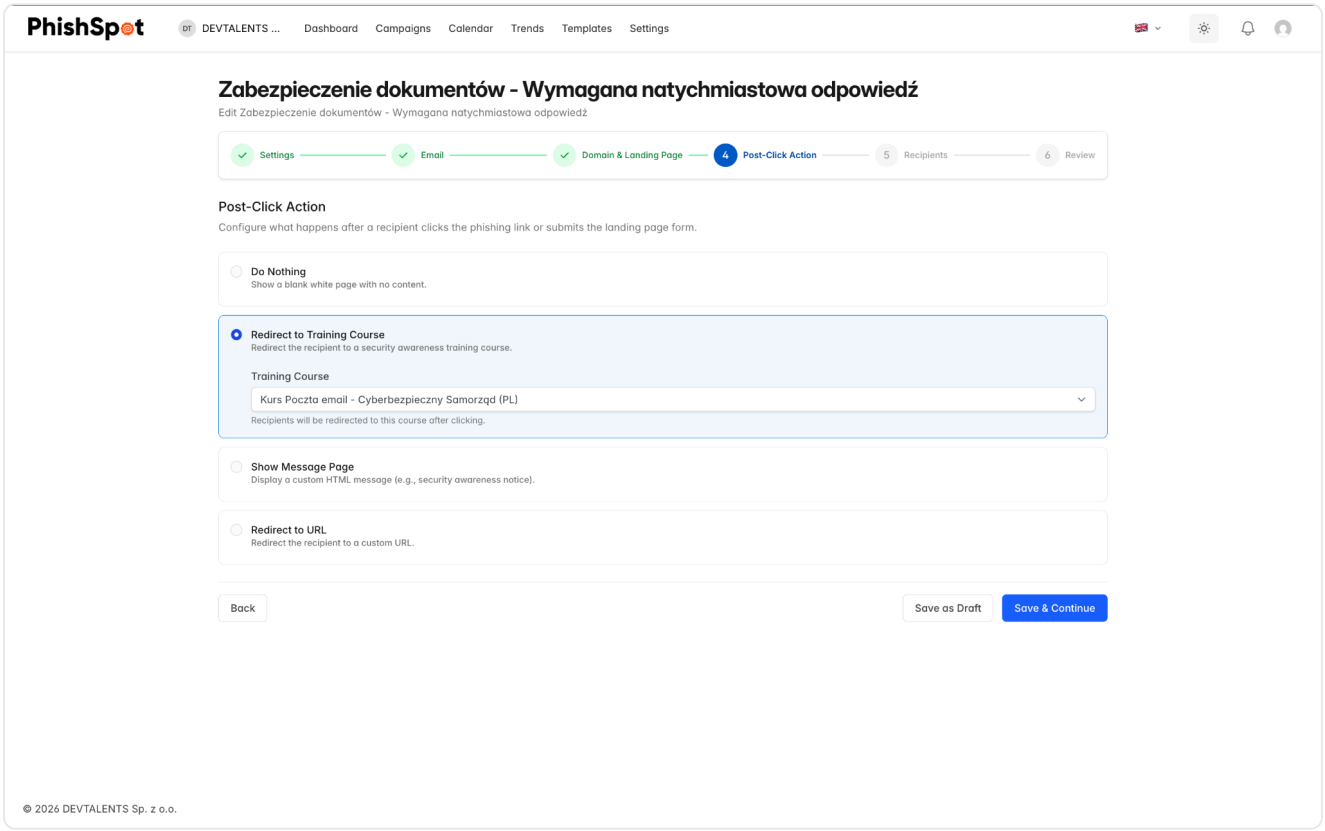




Step 4: Post-Click Action

Define what happens after a recipient interacts with the landing page:

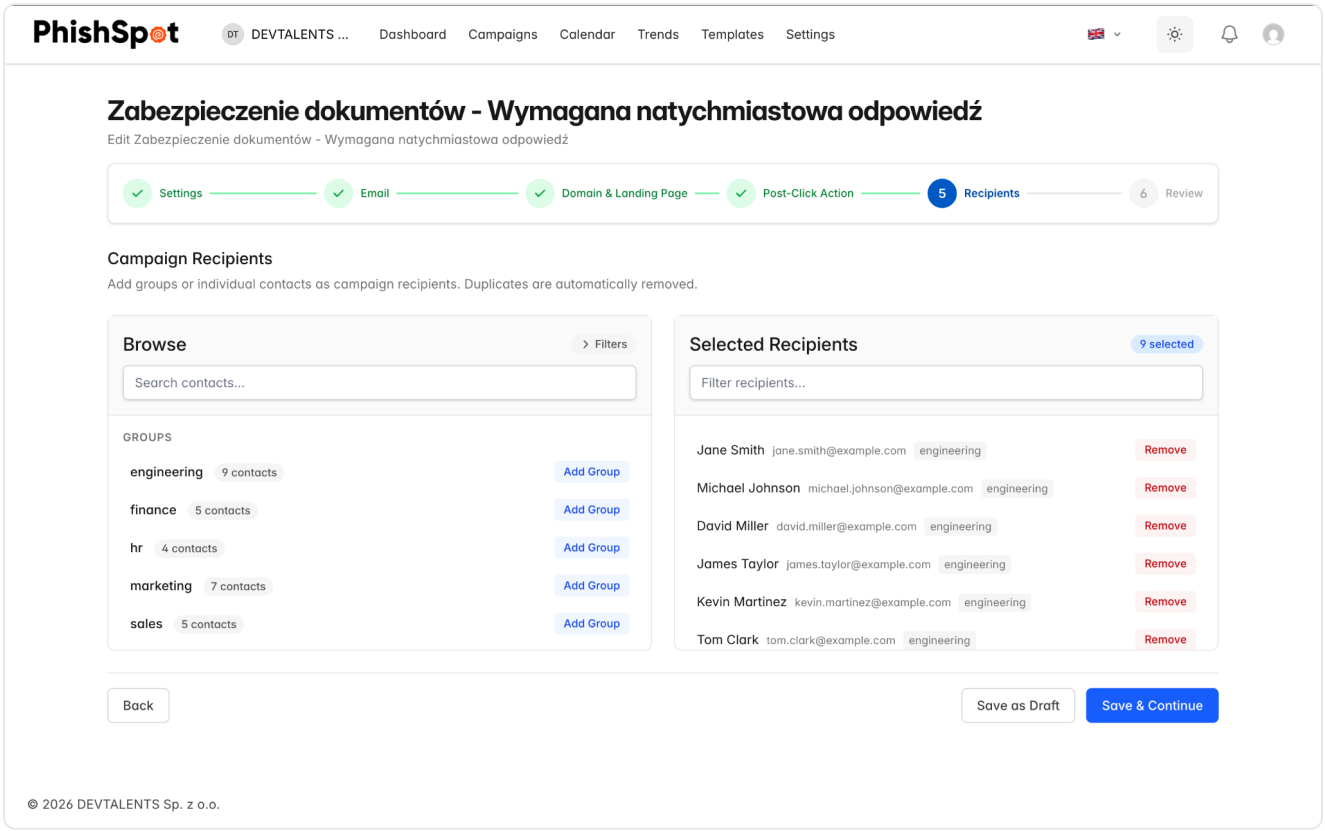
- **Training Course** — Assign a security awareness course that the user is redirected to after clicking or submitting data.
- **Custom Redirect URL** — Redirect the user to a specific URL instead.
- **Awareness Page** — Display a built-in awareness message explaining that this was a simulation.



Step 5: Recipients

Select who receives the campaign emails:

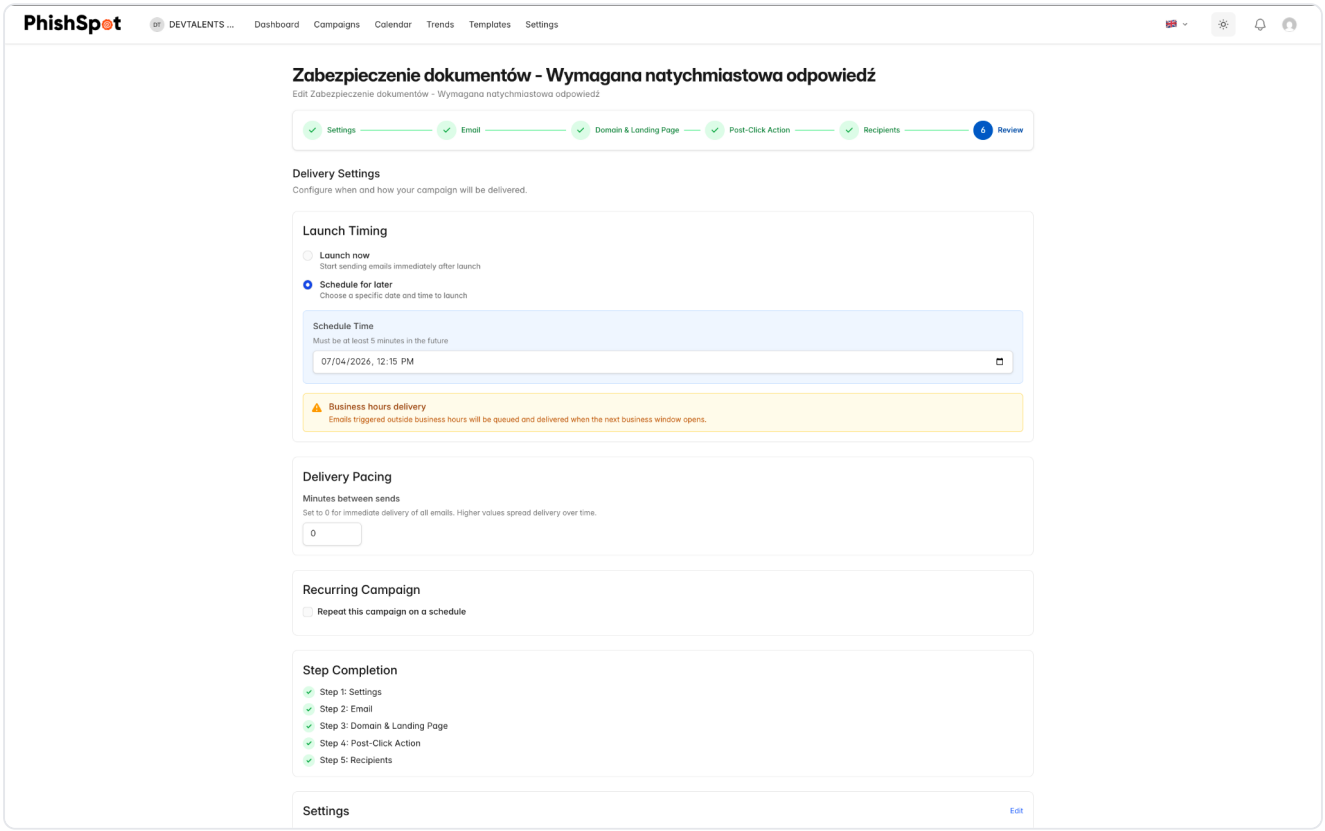
- Browse your contacts and add individuals or entire groups
- Review the selected recipients list
- Remove specific contacts if needed
- The contact browser supports search and filtering by department, title, and location

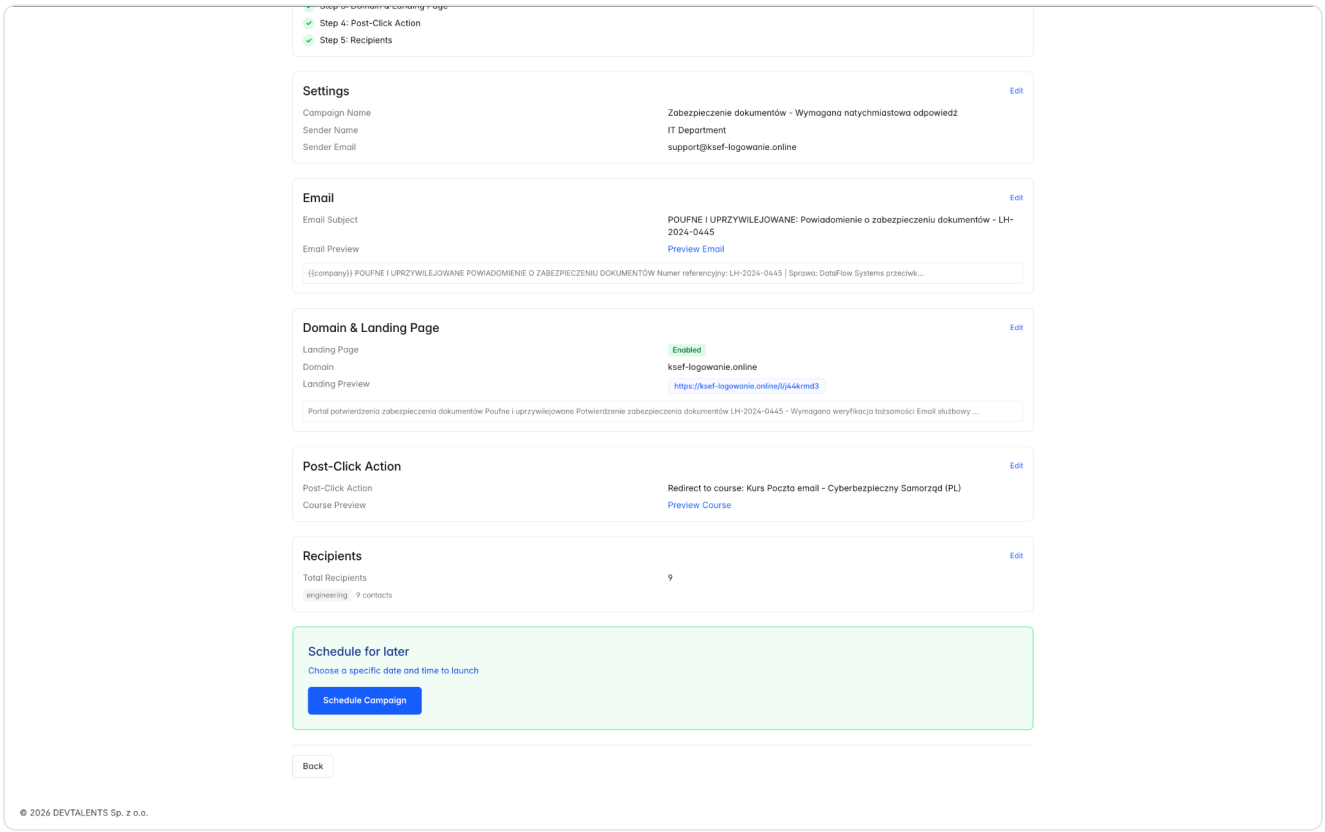


Step 6: Review & Launch

Review all campaign settings before launching:

- Summary of email content, landing page, recipients, and delivery settings
- Final confirmation before starting the campaign
- Option to save as draft and launch later





4.3 Campaign States

A campaign moves through these states during its lifecycle:

State	Meaning	Available Actions
Draft	Campaign is being configured and has not been sent	Edit, Start, Schedule, Delete
Scheduled	Campaign is set to launch at a future date/time	Edit, Reschedule, Cancel Schedule, Delete
Active	Campaign is currently sending or has sent emails	Pause, Stop, View Dashboard
Paused	Campaign sending is temporarily paused	Resume (Start), Stop
Done	All emails have been delivered and tracking is complete	View Dashboard, Export Reports, Duplicate

4.4 Campaign Actions

From a campaign's detail page, you have access to several actions depending on its current state:

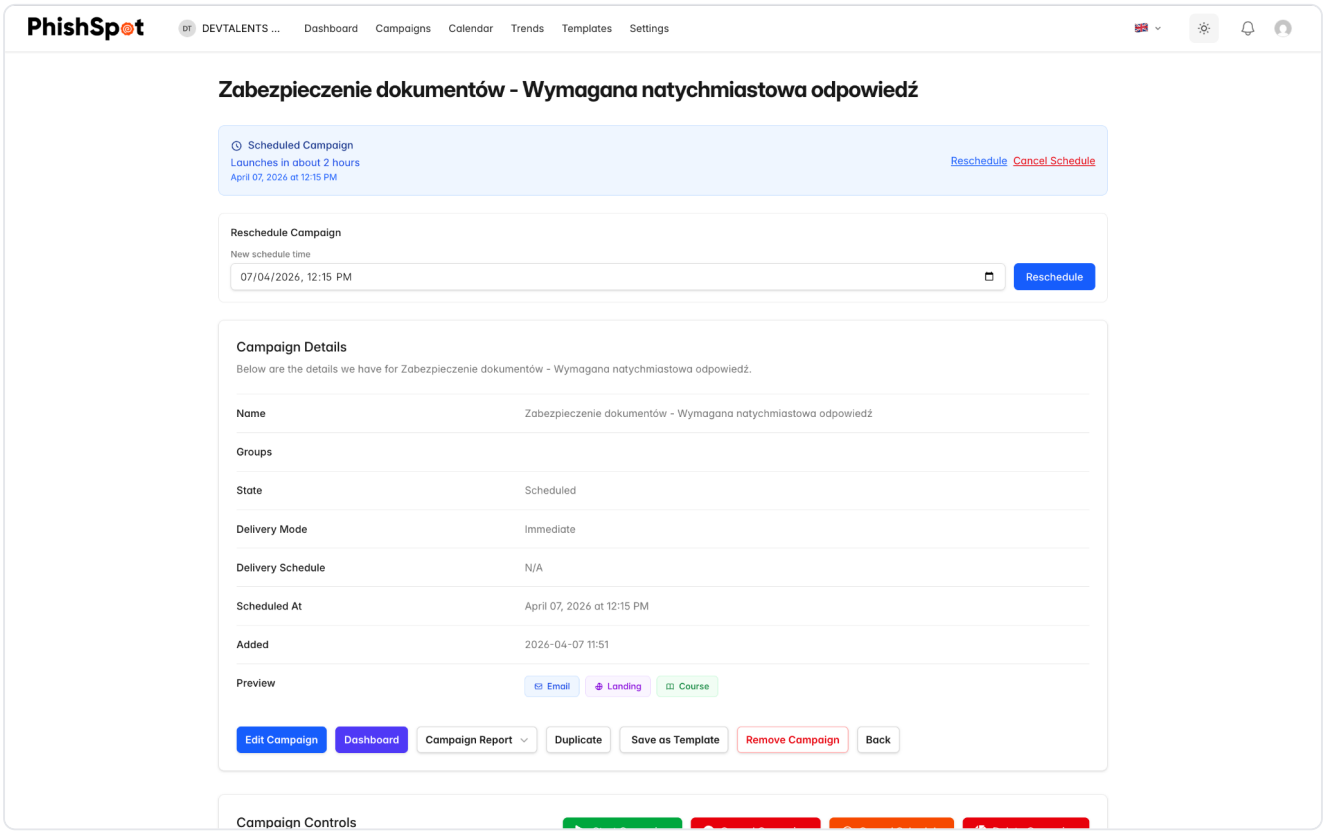
- **Start** — Begin sending the campaign immediately.
- **Pause** — Temporarily halt email delivery (can be resumed).
- **Stop** — Permanently stop the campaign. Remaining unsent emails will not be delivered.
- **Schedule / Reschedule** — Set or change the future delivery date.
- **Duplicate** — Create a copy of the campaign as a new draft.
- **Save as Template** — Save the current email content as a reusable phishing template.
- **Send Test Email** — Send a test version of the email to yourself to verify formatting.
- **Export Report** — Download campaign results as PDF or Excel.

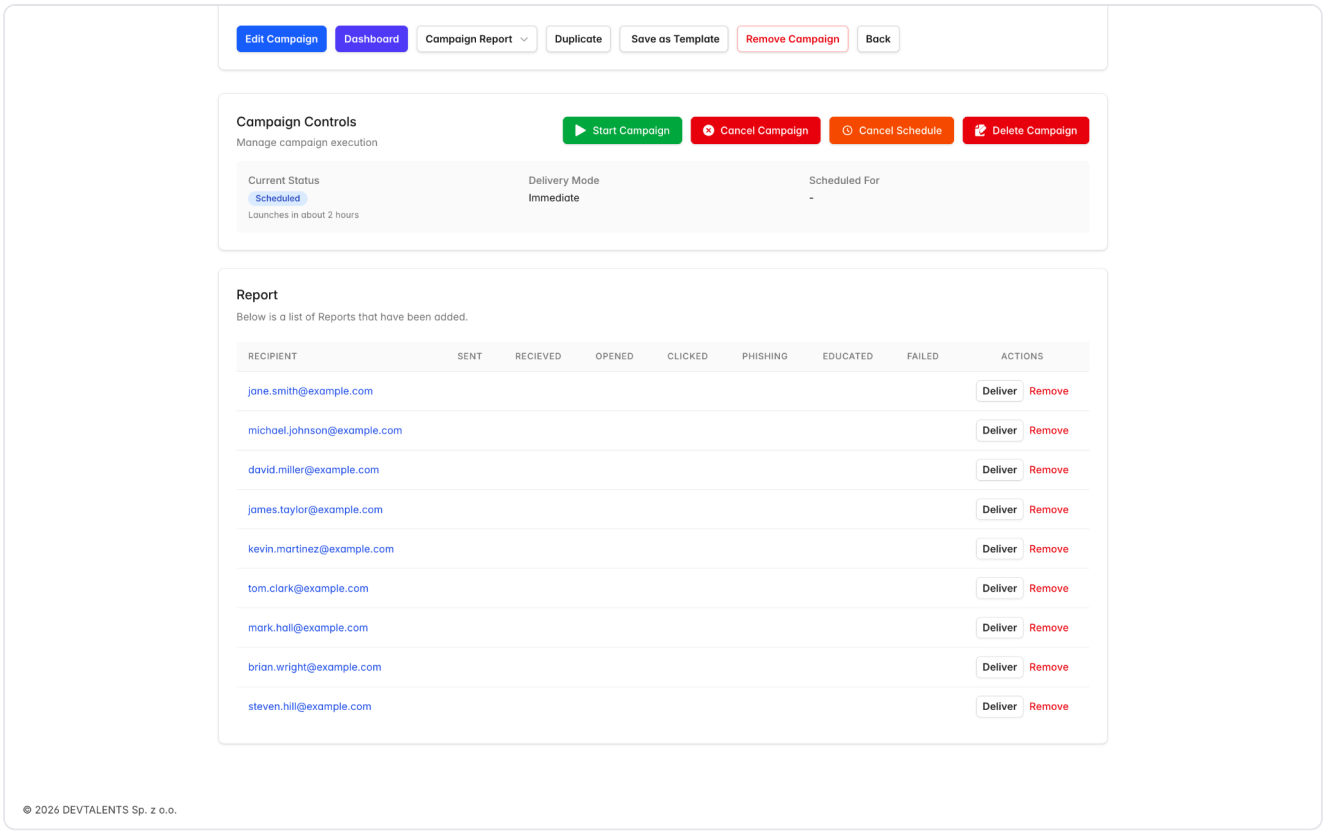
4.5 Campaign Dashboard

Once a campaign is active or complete, you can access its dashboard for detailed analytics. The campaign dashboard includes:

- **Funnel Chart** — Visual funnel showing Sent → Delivered → Opened → Clicked → Submitted, with conversion percentages at each stage.
- **Group Breakdown** — Performance metrics broken down by contact group.
- **Department Breakdown** — Performance metrics broken down by department.
- **Recipients Table** — A list of every recipient with their individual status (sent, delivered, opened, clicked, submitted).

- **Recipient Timeline** — Click on any recipient to open a side panel showing their complete event timeline (when the email was sent, opened, link clicked, page visited, data submitted, course started/completed).
- **Export CSV** — Download the full recipient data as a CSV file for further analysis.





4.6 Recurring Campaigns

PhishSpot supports recurring campaigns that automatically repeat at set intervals. When configuring a scheduled campaign, enable the **Recurring** checkbox in the schedule step and set two values:

- **Interval** — a number (e.g., `1`, `2`, `4`).
- **Unit** — `days`, `weeks` or `months`.

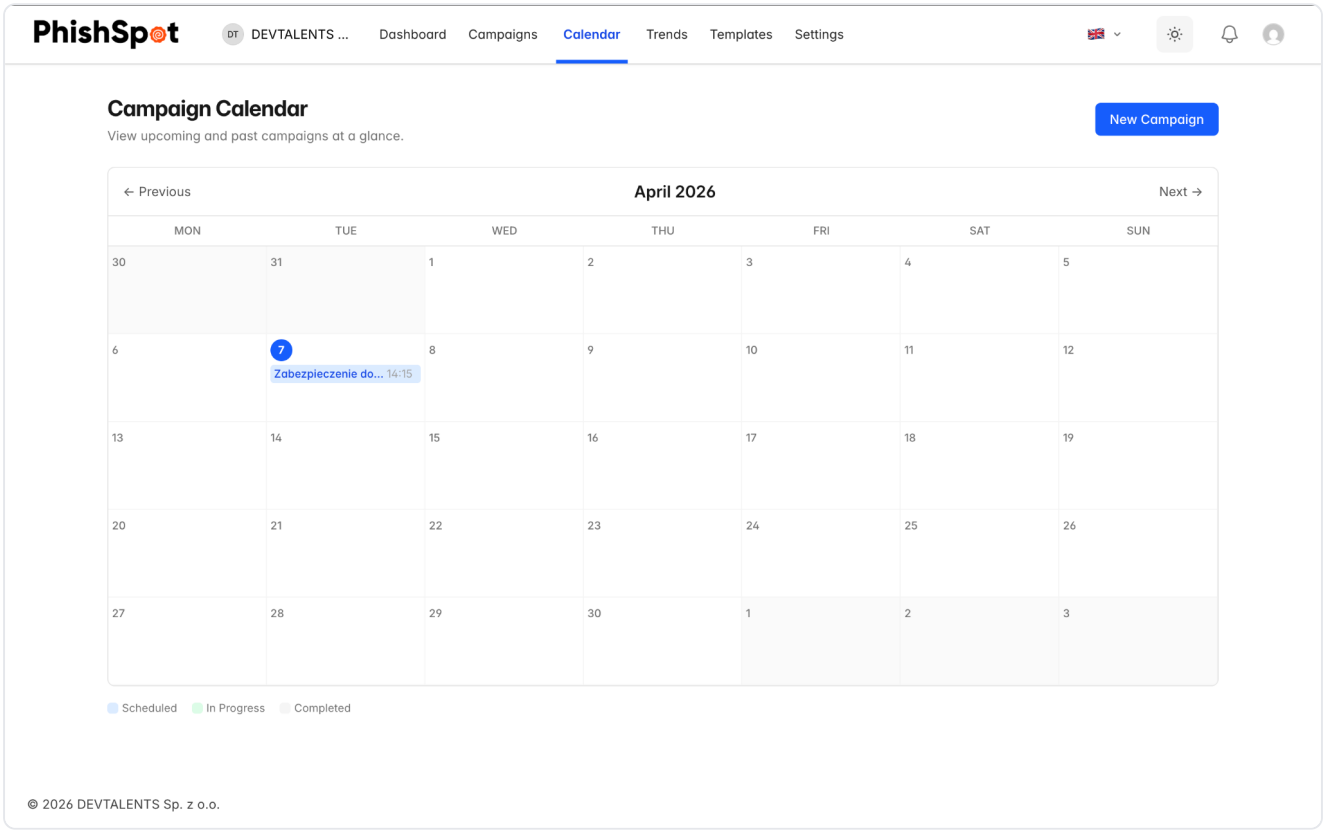
So `1 week` runs the campaign every seven days; `2 months` runs it every other month. Each recurrence creates a **child campaign** linked to the original parent — the parent stays as the canonical template (its content, recipients and post-click action are snapshotted into each child at launch time), and the children carry their own reports. Edits to the parent only affect future recurrences; in-flight children keep the configuration they were launched with.

To stop a recurring series, open the parent and cancel it — that prevents future children from being created. Children already launched continue to completion independently.

> If your goal is a continuous awareness program rather than a tightly-scheduled recurring send, consider an **autopilot** instead — see Chapter 23 Autopilots. Autopilots adapt template selection per recipient and pick up new contacts automatically, which a recurring campaign doesn't.

4.7 Campaign Calendar

The Campaign Calendar provides a visual calendar view of all scheduled and past campaigns. You can navigate between months and click on any campaign entry to go directly to its detail page. This is useful for planning your phishing program and avoiding schedule conflicts.


PhishSpot
DT DEVTALENTS ...
Dashboard
Campaigns
Calendar
Trends
Templates
Settings
Campaign Calendar
View upcoming and past campaigns at a glance.
New Campaign
← Previous
April 2026
Next →
MON
TUE
WED
THU
FRI
SAT
SUN
30
31
1
2
3
4
5
6
7
Zabezpieczenie do... 14:15
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
1
2
3
Scheduled
In Progress
Completed
© 2026 DEVTALENTS Sp. z o.o.

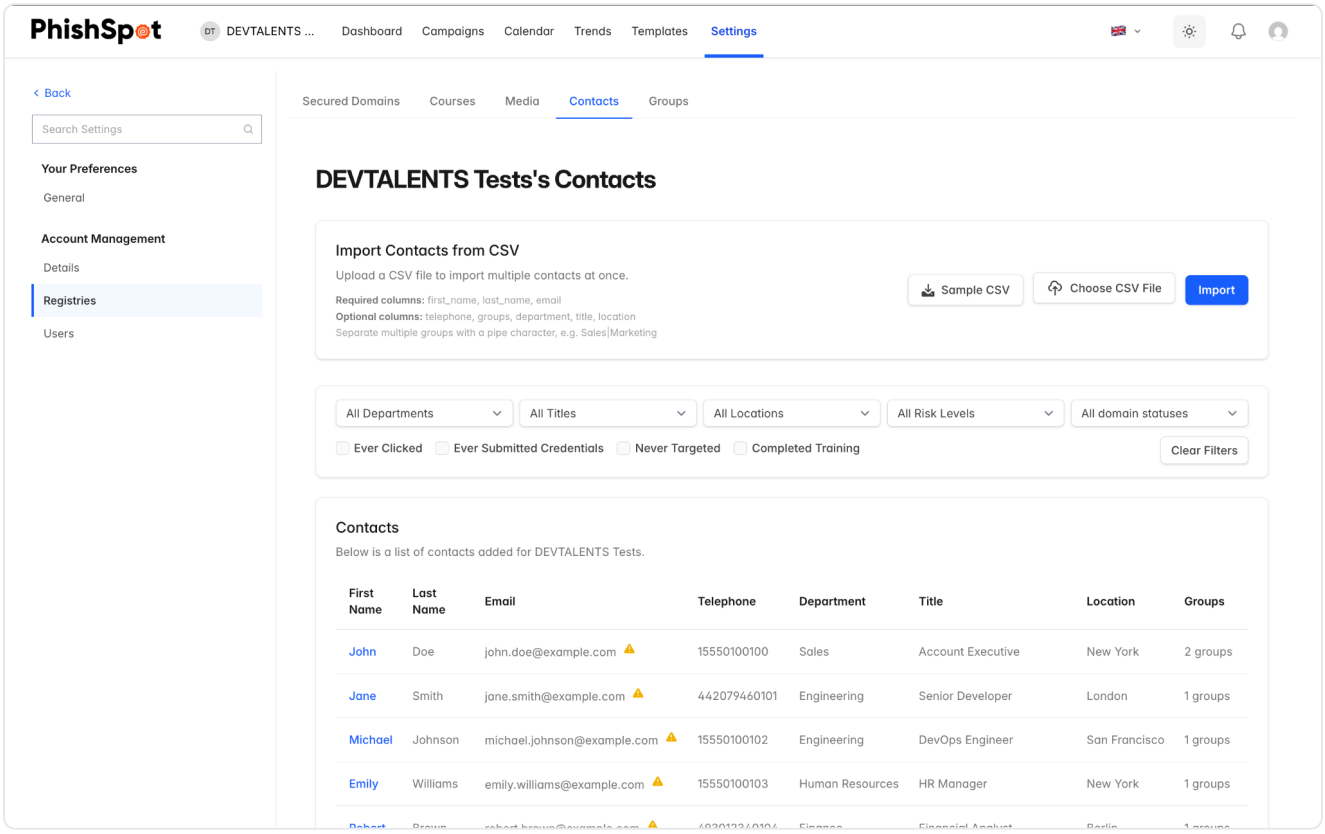
Contacts

Contacts are the people in your organization who receive phishing simulation emails. Managing your contacts accurately is essential for running effective campaigns.

5.1 Contacts List

Navigate to Settings → Contacts from the sidebar. The contacts list displays all contacts in your account with the following columns:

Column	Description
First Name	Contact's first name (clickable to view details)
Last Name	Contact's last name
Email	Email address (with a warning icon if the domain is unverified)
Telephone	Phone number (optional)
Department	Department the contact belongs to
Title	Job title
Location	Office or location
Groups	Number of groups the contact belongs to
Risk Score	Color-coded risk assessment (green = low, yellow = medium, orange = high, red = critical)
Performance	Click rate: how many campaigns they clicked vs. were targeted
Created At	When the contact was added



5.2 Filtering Contacts

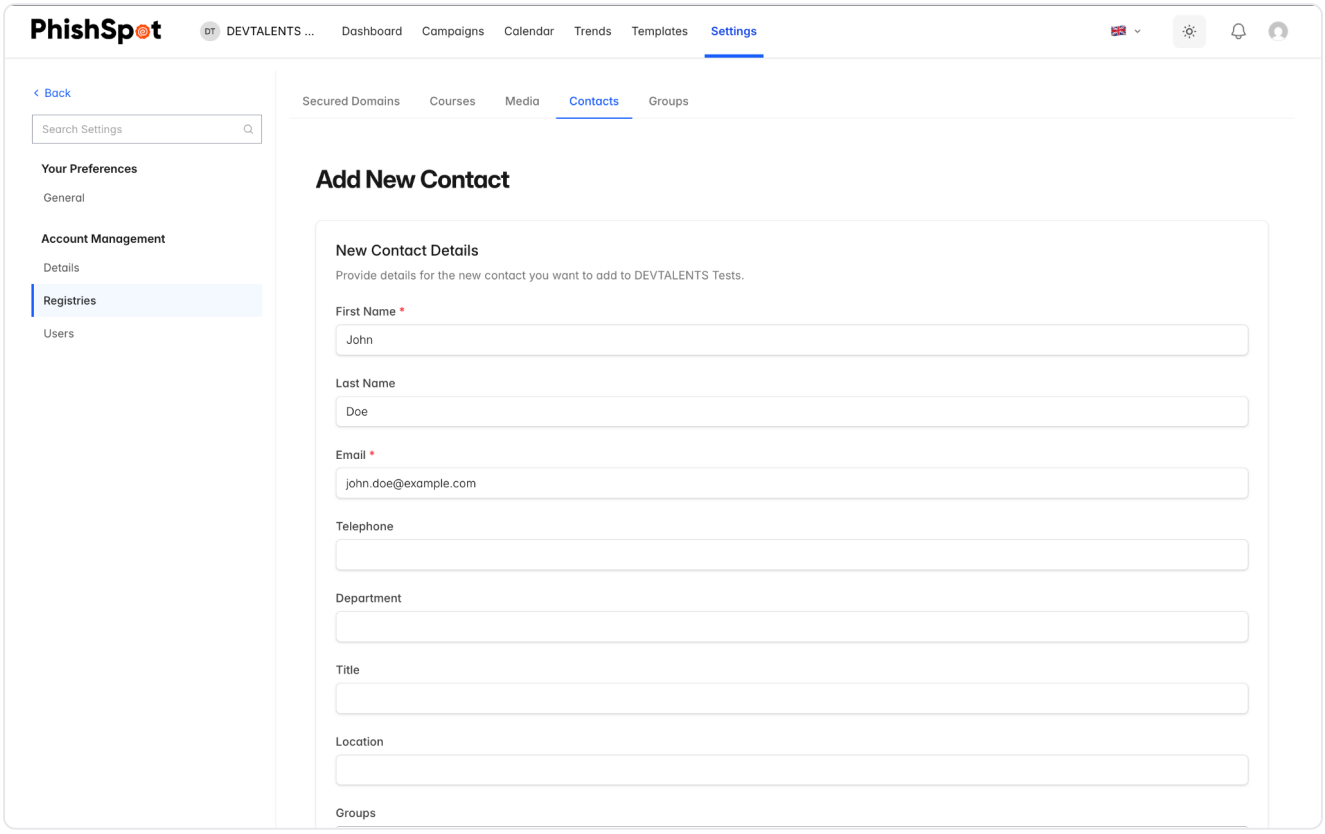
The contacts list includes a filter bar at the top that lets you narrow down the list by:

- Department
- Title
- Location
- Special status filters such as: ever clicked a phishing link, ever submitted data, never targeted, or completed training

5.3 Adding a Single Contact

Click the New Contact button to add a contact manually. Fill in the following fields:

- **First Name** and **Last Name** — Required.
- **Email** — Required. Must be a valid email address.
- **Telephone** — Optional.
- **Department**, **Title**, **Location** — Optional fields for categorization and filtering.
- **Groups** — Assign the contact to one or more groups.

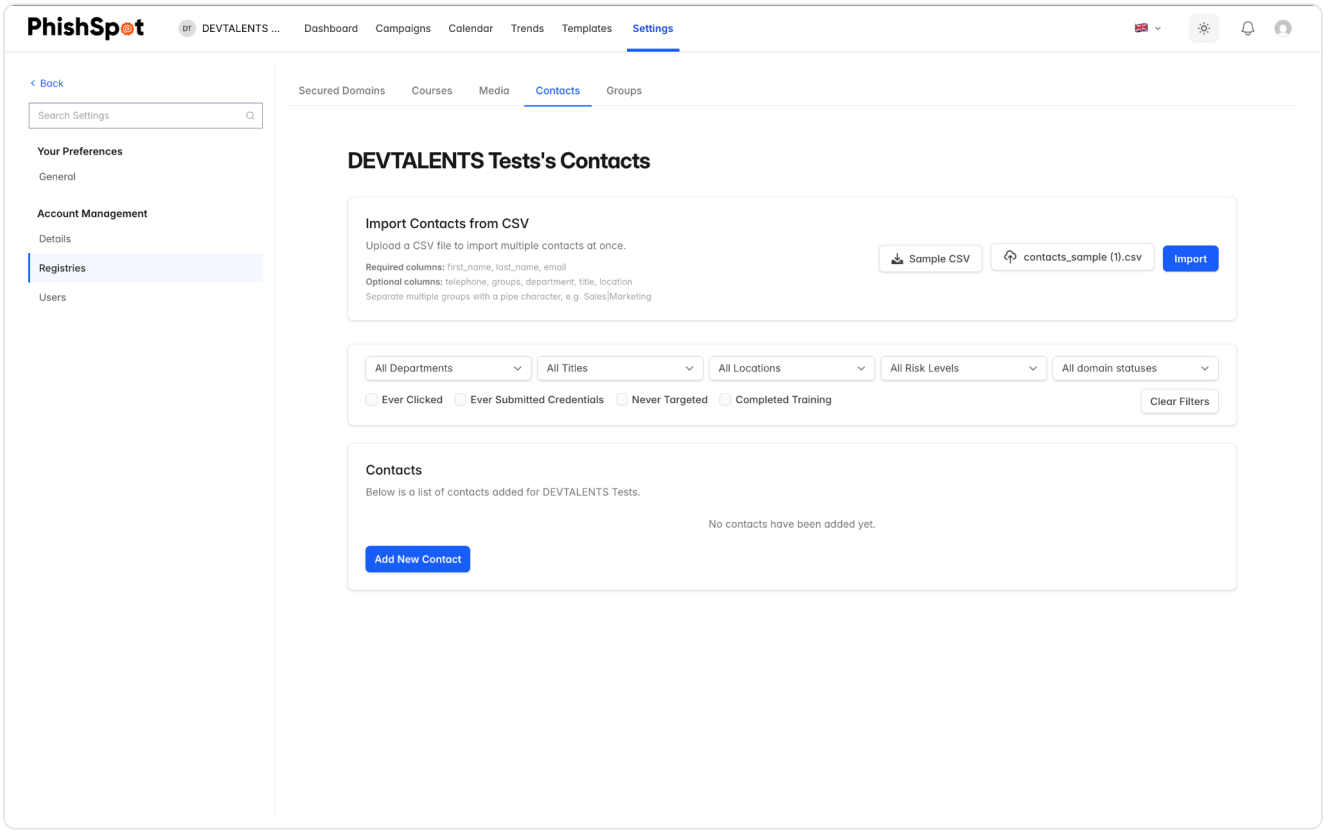


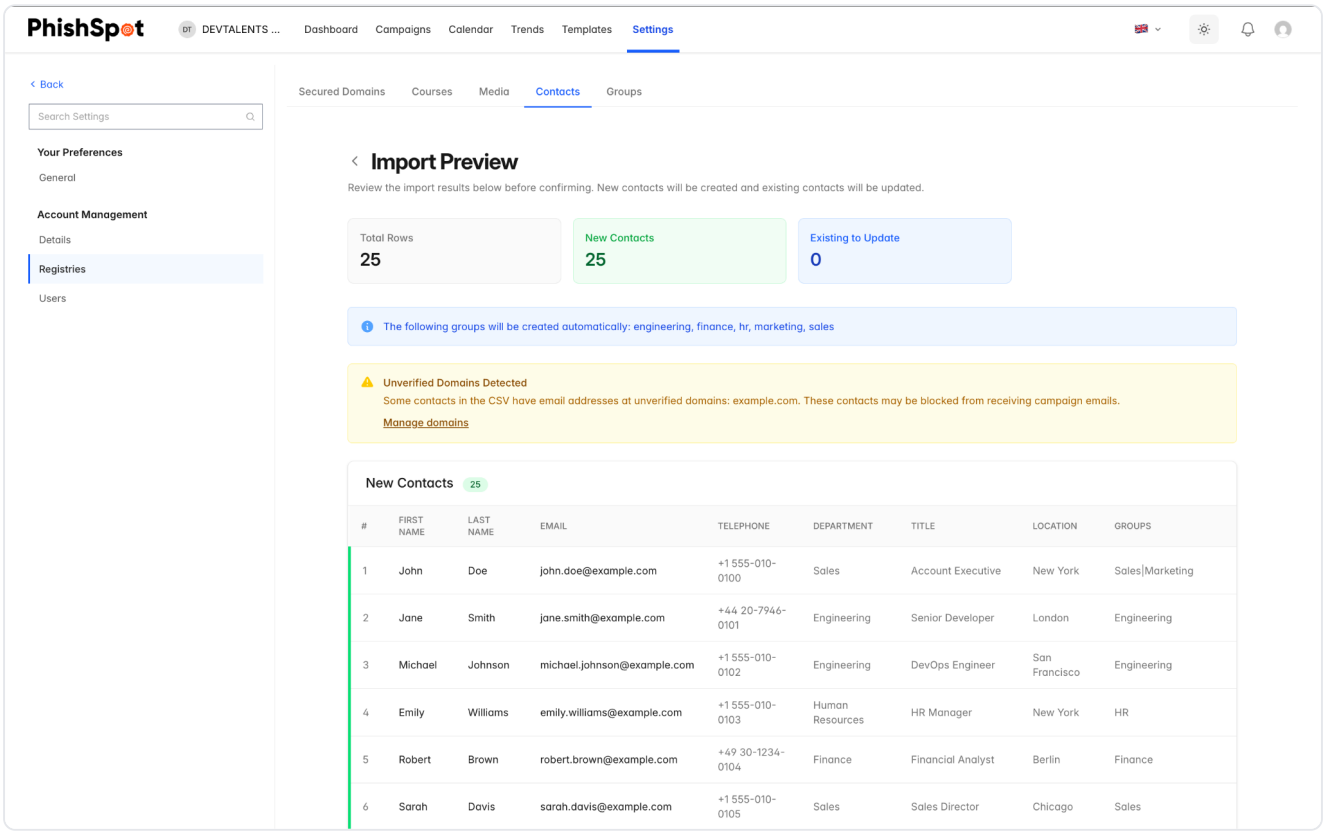
5.4 Importing Contacts via CSV

For bulk import, PhishSpot supports CSV file uploads. The process has three stages:

1. **Upload CSV** — Click the Choose File button in the import section and select your CSV file. You can download a sample CSV first to see the expected format.
2. **Preview** — Review the parsed data before importing. The preview shows which columns were detected and any errors found.
3. **Confirm** — Click Confirm Import to create the contacts. If some rows fail, you can download a CSV of the failed rows to fix and re-import.

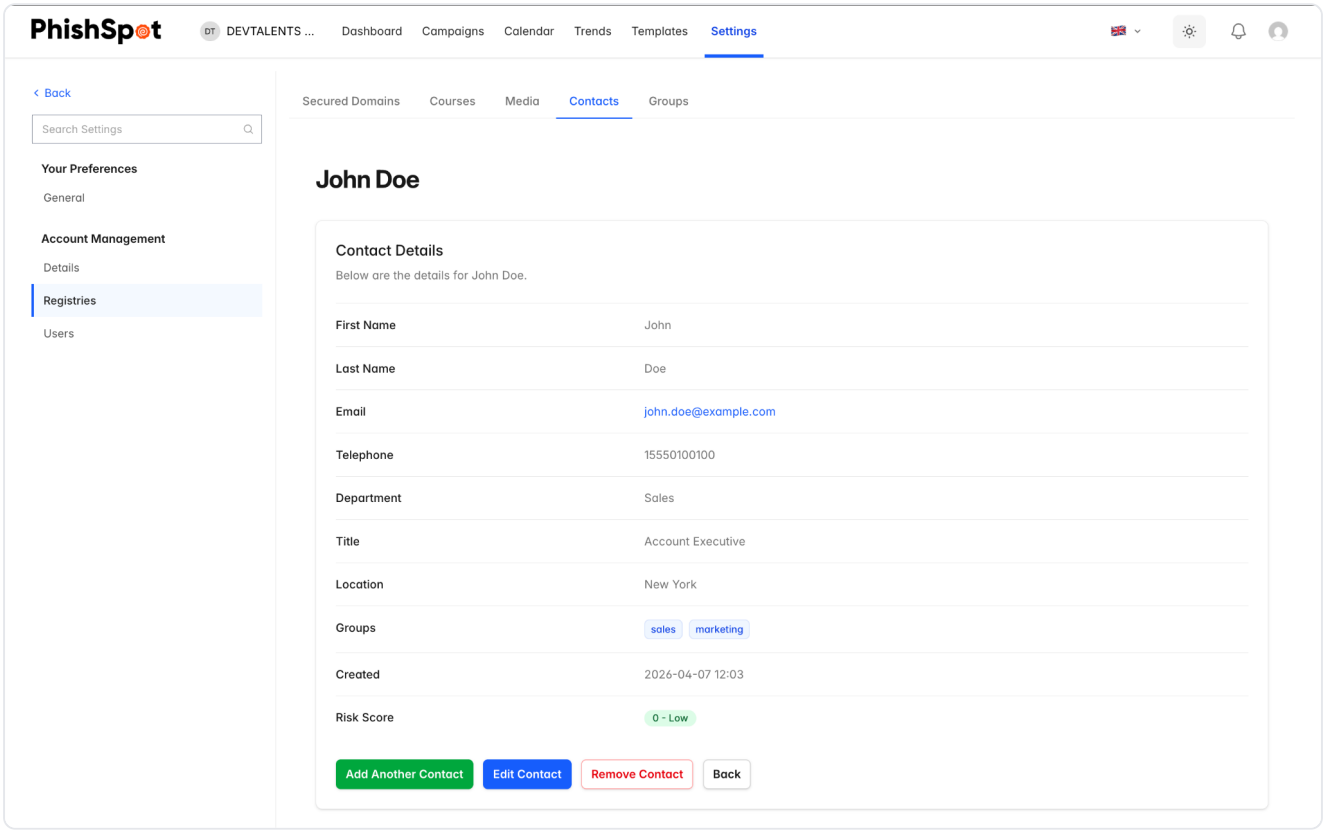
The required CSV columns are: first_name, last_name, email. Optional columns are: telephone, groups (comma-separated group names), department, title, location.





5.5 Contact Detail Page

Click on any contact's name to see their detail page. This shows their complete profile information, risk score, campaign history, and performance metrics showing how they responded to past campaigns.



5.6 Bulk operations

Each row in the contacts list has a checkbox at the far left. Select two or more rows and a **bulk action bar** appears at the bottom of the page showing the selection count and the available actions. Today the only bulk action is **Delete** — useful for cleaning up a stale import, removing a department that has left the company, or pruning test contacts.

The bar's **Delete** button asks for confirmation before removing anything; the confirmation dialog tells you exactly how many contacts will be removed. Deleted contacts disappear from the list and from any groups they belonged to; their deliverable history is preserved on the campaigns they participated in so reports stay intact.

> Contacts imported from Entra AD (Chapter 25) will reappear on the next directory sync if they still exist (or are still enabled) in Entra. Bulk-delete is most useful for **manually-imported** contacts; for directory-managed ones, disable or remove them in Entra instead and let the next sync mirror the change.

Groups

Groups let you organize contacts into logical collections for targeted campaigns. For example, you might create groups for different departments, office locations, or risk levels.

6.1 Groups List

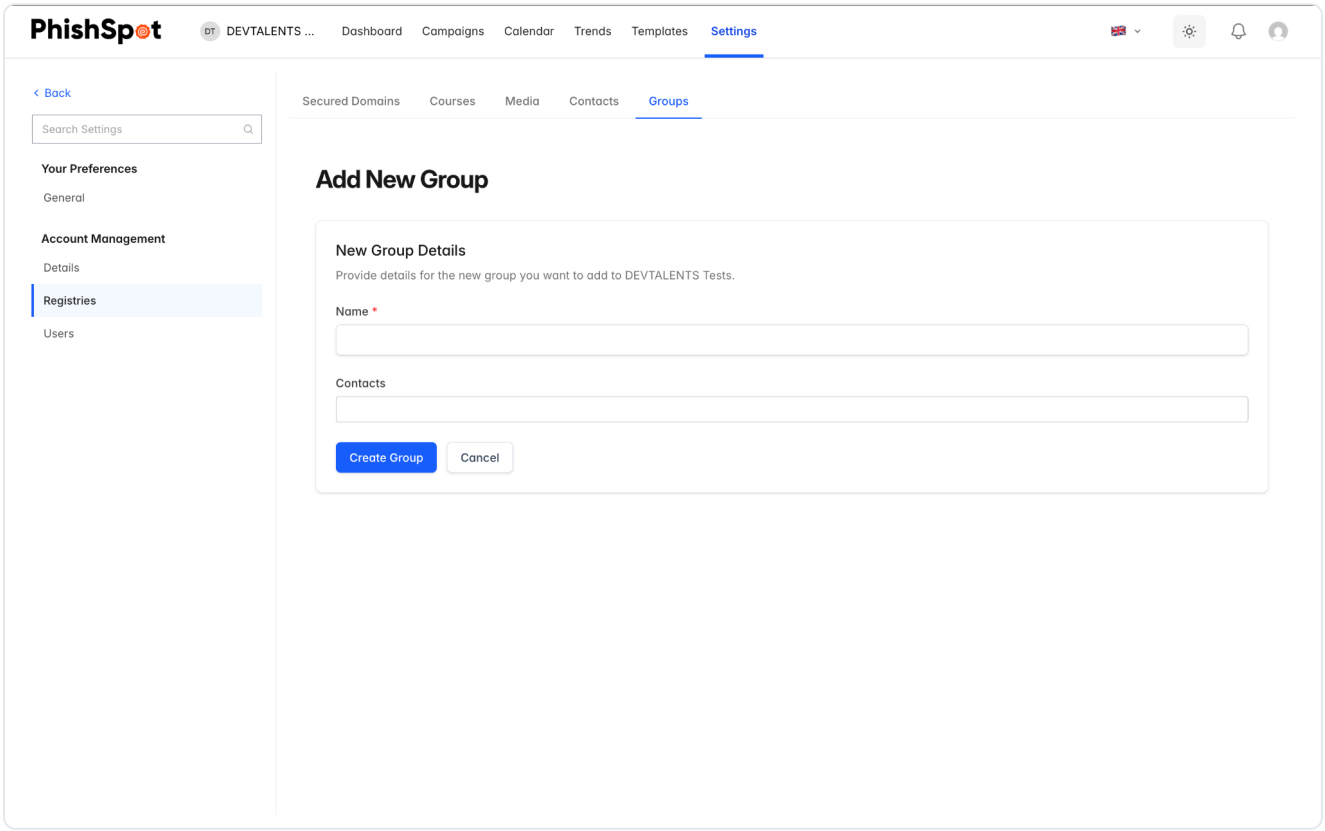
Navigate to Settings → Groups. The list shows:

Column	Description
Name	Group name (clickable to view members)
Contacts	Number of contacts in the group
Campaigns	Number of campaigns that have used this group
Created At	When the group was created
Actions	Edit and Delete buttons

> Groups that are currently in use by active campaigns may be locked and cannot be edited or deleted until the campaign completes.

6.2 Creating a Group

Click New Group and enter a name and optional description. You can assign contacts to the group immediately using the multi-select contact picker, or add them later. Groups can also be created automatically during CSV import by including group names in the groups column.


PhishSpot
DT DEVTALENTS ...
Dashboard
Campaigns
Calendar
Trends
Templates
Settings
Back
Search Settings
Your Preferences
General
Account Management
Details
Registries
Users
Secured Domains
Courses
Media
Contacts
Groups
Add New Group
New Group Details
Provide details for the new group you want to add to DEVTALENTS Tests.
Name *
Contacts
Create Group
Cancel

Phishing Templates

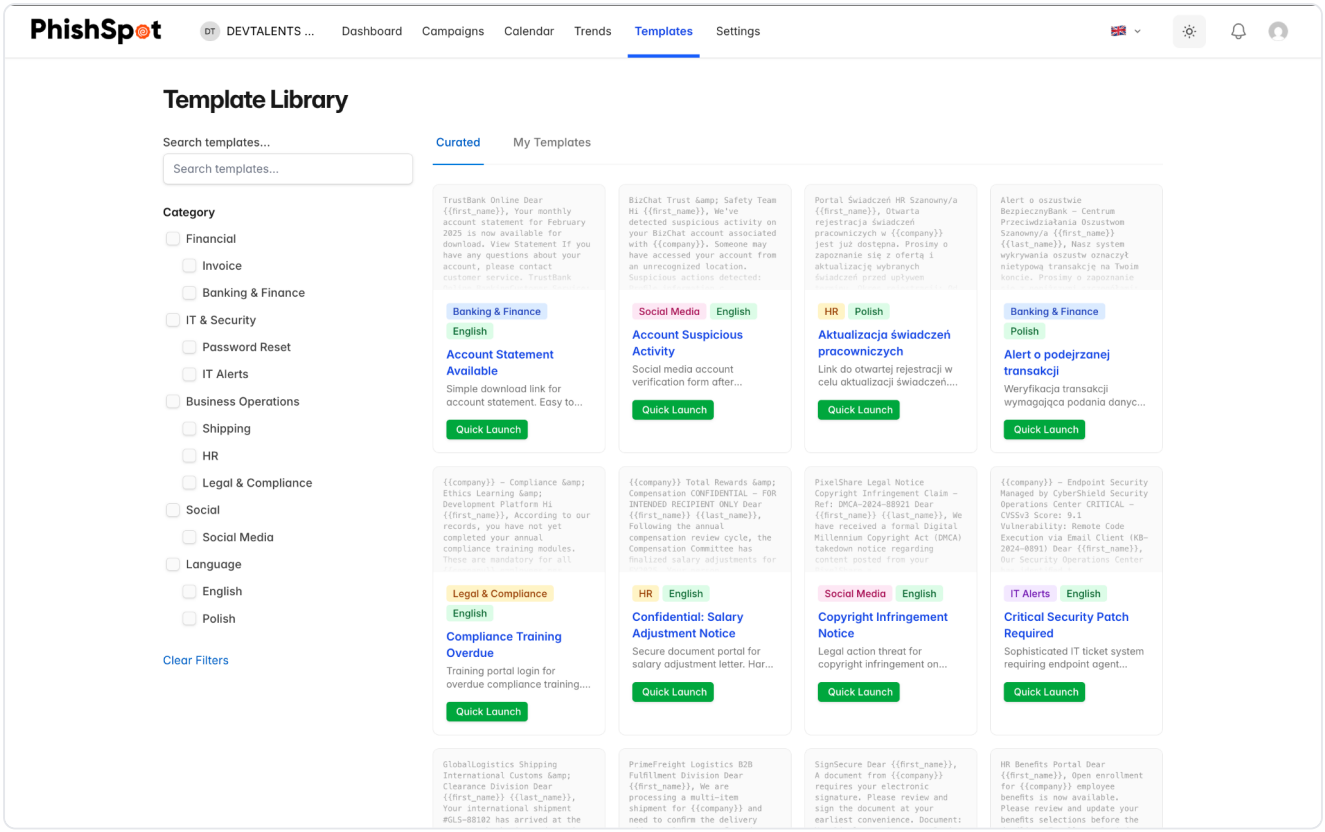
Templates save you time by providing pre-built phishing email designs that you can deploy directly into a campaign.

7.1 Template Library

Navigate to Templates from the sidebar. The template library has two tabs:

- **Curated** — Pre-built templates provided by PhishSpot, organized by category (e.g., credential harvesting, package delivery, IT alerts, HR notices). The curated library ships with **48 templates: 24 in English and 24 in Polish**, all authored locally — the Polish set is written by a Polish-speaking team, not machine-translated, so the copy, sender names and pretexts land naturally for Polish recipients.
- **My Templates** — Custom templates you've created or saved from previous campaigns.

The left sidebar shows a category tree that lets you filter templates by type. Templates are displayed in a grid view with preview thumbnails.



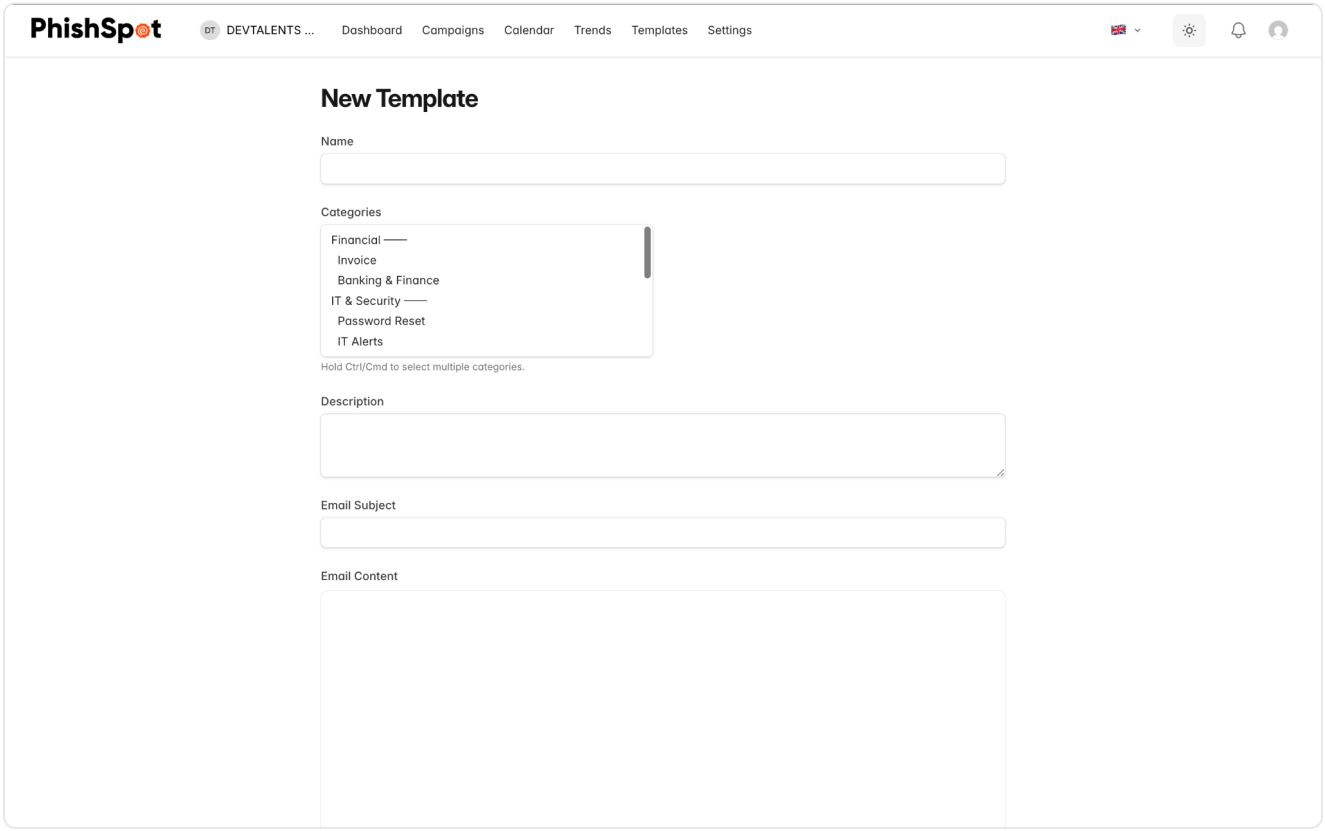
7.2 Using a Template

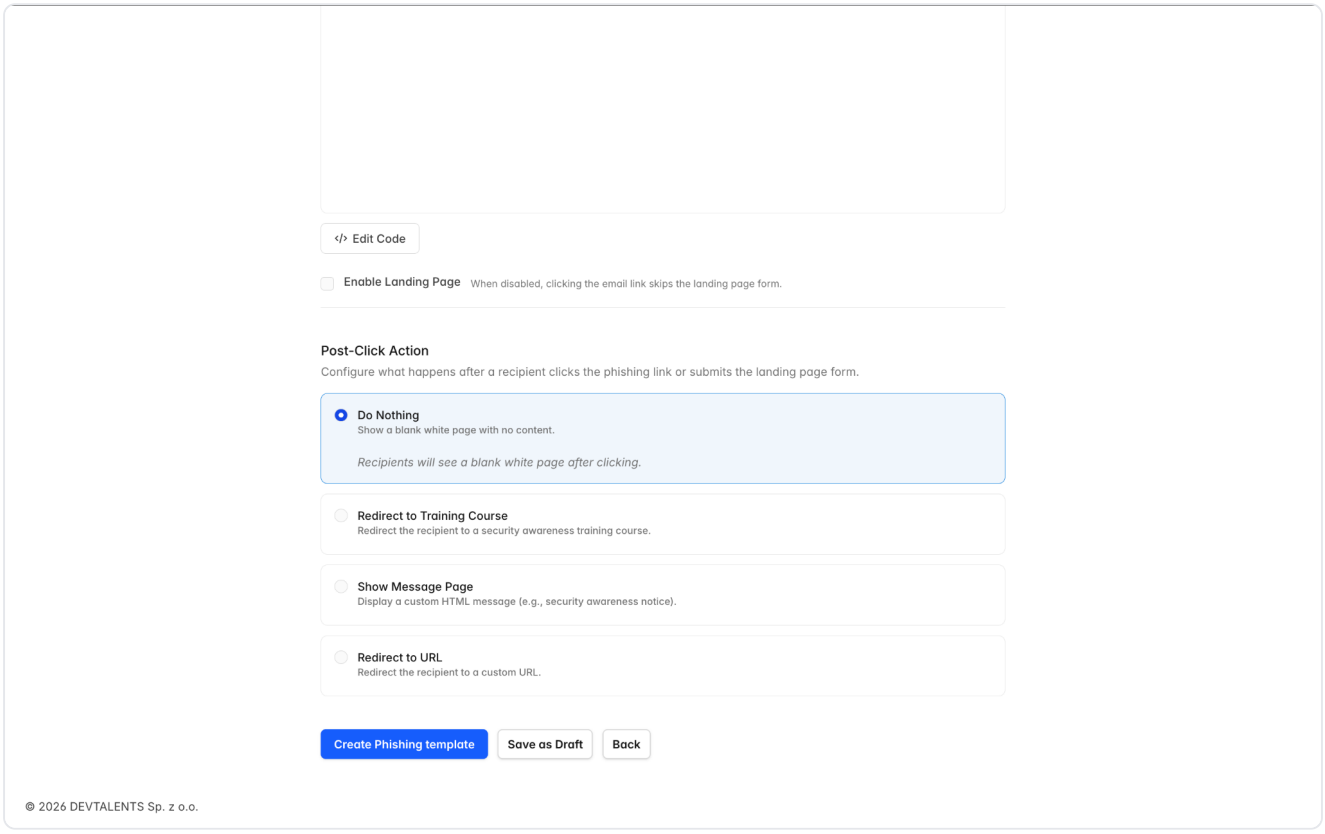
There are two ways to use a template:

- **Deploy** — Applies the template's email content to an existing draft campaign.
- **Quick Launch** — Creates a new campaign pre-filled with the template's email content, ready for you to configure recipients and delivery settings.

7.3 Creating Custom Templates

On the My Templates tab, click New Template to create your own. You can also save any campaign's email as a template using the Save as Template button on the campaign detail page. Custom templates support the same email editor and template variables as campaigns.




Edit Code
Enable Landing Page
When disabled, clicking the email link skips the landing page form.
Post-Click Action
Configure what happens after a recipient clicks the phishing link or submits the landing page form.
Do Nothing
Show a blank white page with no content.
Recipients will see a blank white page after clicking.
Redirect to Training Course
Redirect the recipient to a security awareness training course.
Show Message Page
Display a custom HTML message (e.g., security awareness notice).
Redirect to URL
Redirect the recipient to a custom URL.
Create Phishing template
Save as Draft
Back
© 2026 DEVTALENTS Sp. z o.o.

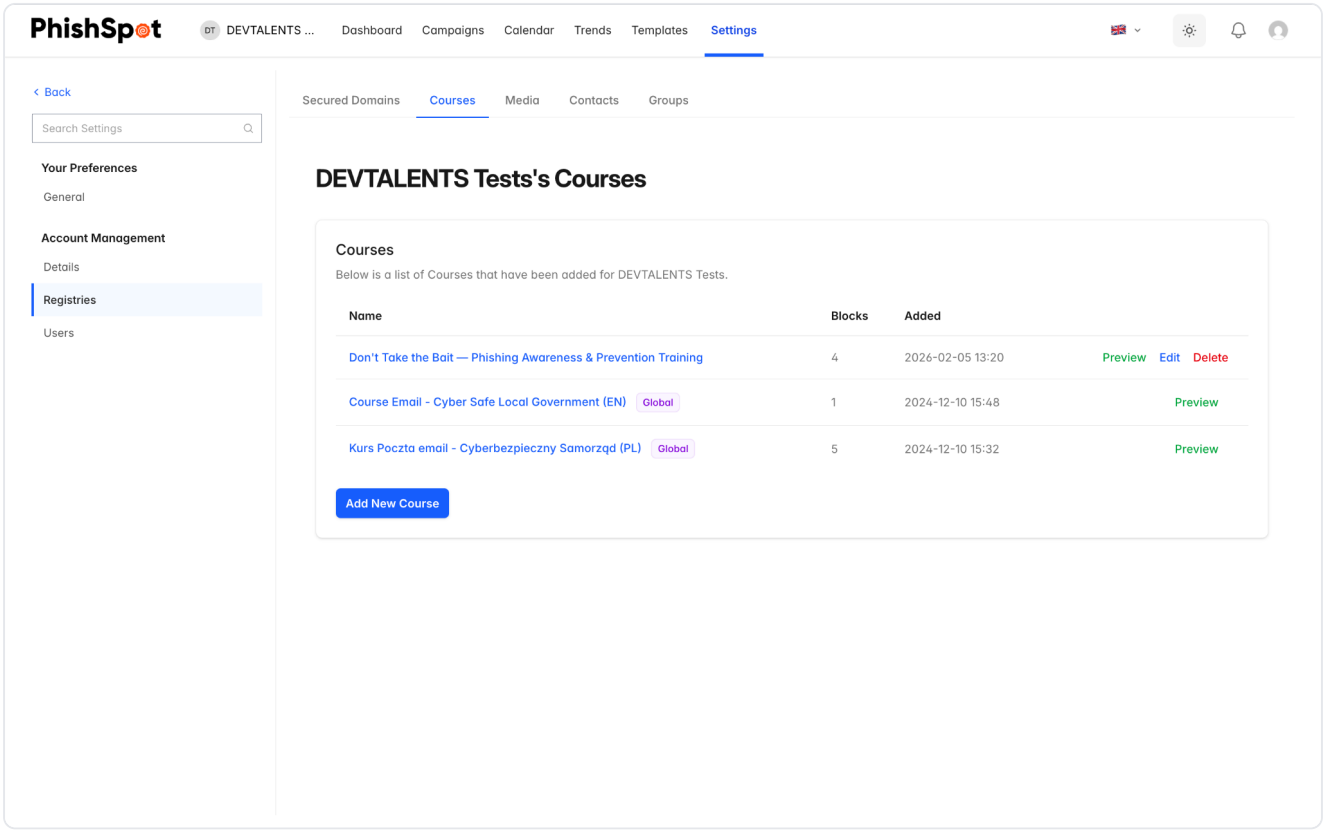
Courses (Security Awareness Training)

Courses are training modules that are shown to employees after they fall for a phishing simulation. They educate users on how to recognize phishing attempts and improve security behavior.

8.1 Course List

Navigate to Settings → Courses. The list shows all available courses with:

Column	Description
Name	Course title (with a purple “Global” badge if it’s a platform-provided course)
Blocks	Number of content blocks (lessons, quizzes) in the course
Created At	When the course was created
Actions	Preview, Edit, and Delete buttons

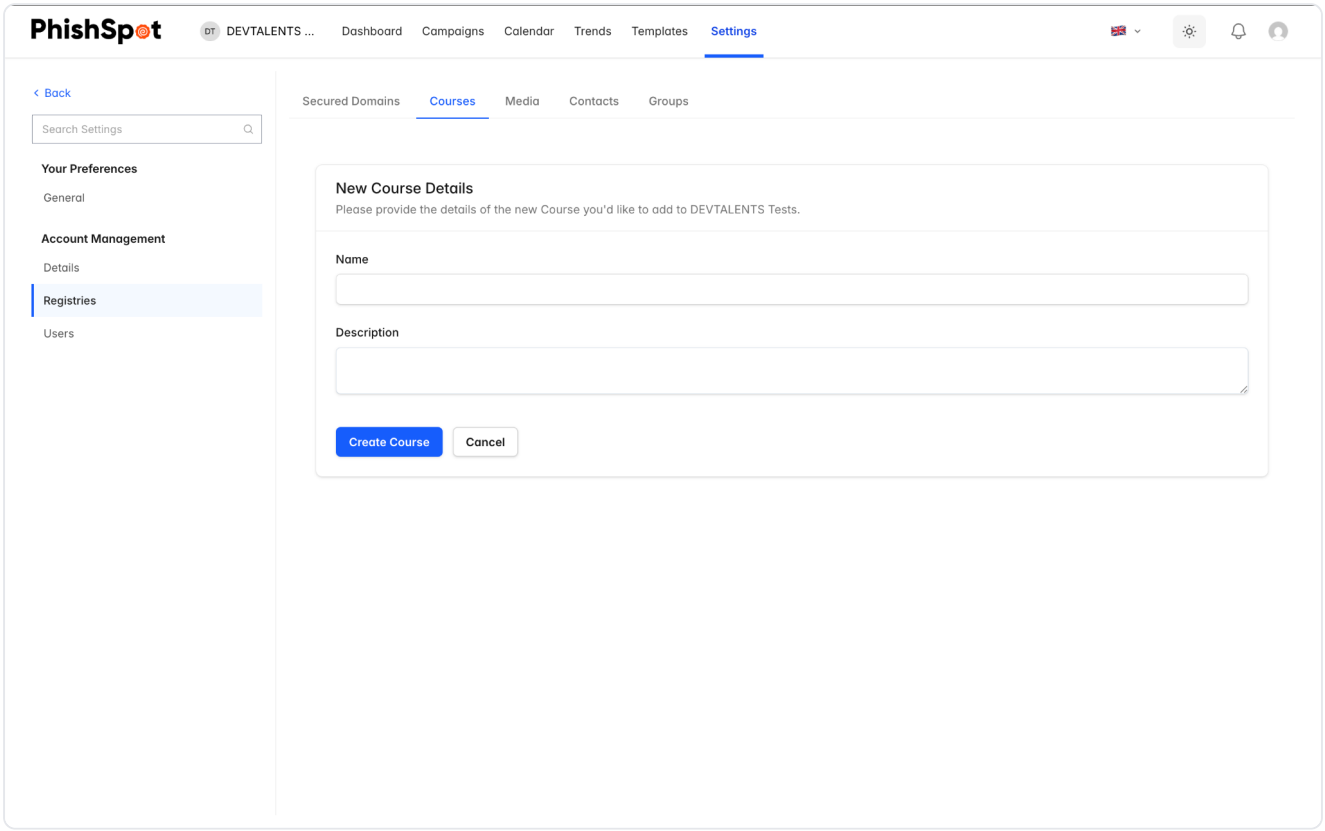


8.2 Creating a Course

Click New Course to create a custom training course. A course is made up of blocks, which are individual content sections:

- **Lesson Blocks** — Text, images, and educational content about phishing awareness.
- **Quiz Blocks** — Interactive quizzes to test the user's understanding.

Blocks can be reordered by dragging them into the desired sequence. Use the Preview button to see how the course will appear to end users.



8.3 Assigning Courses to Campaigns

Courses are assigned during campaign creation in Step 4 (Post-Click Action). When a recipient clicks the phishing link or submits data on the landing page, they are redirected to the assigned course. Their progress and completion status are tracked in the campaign dashboard.

Domains

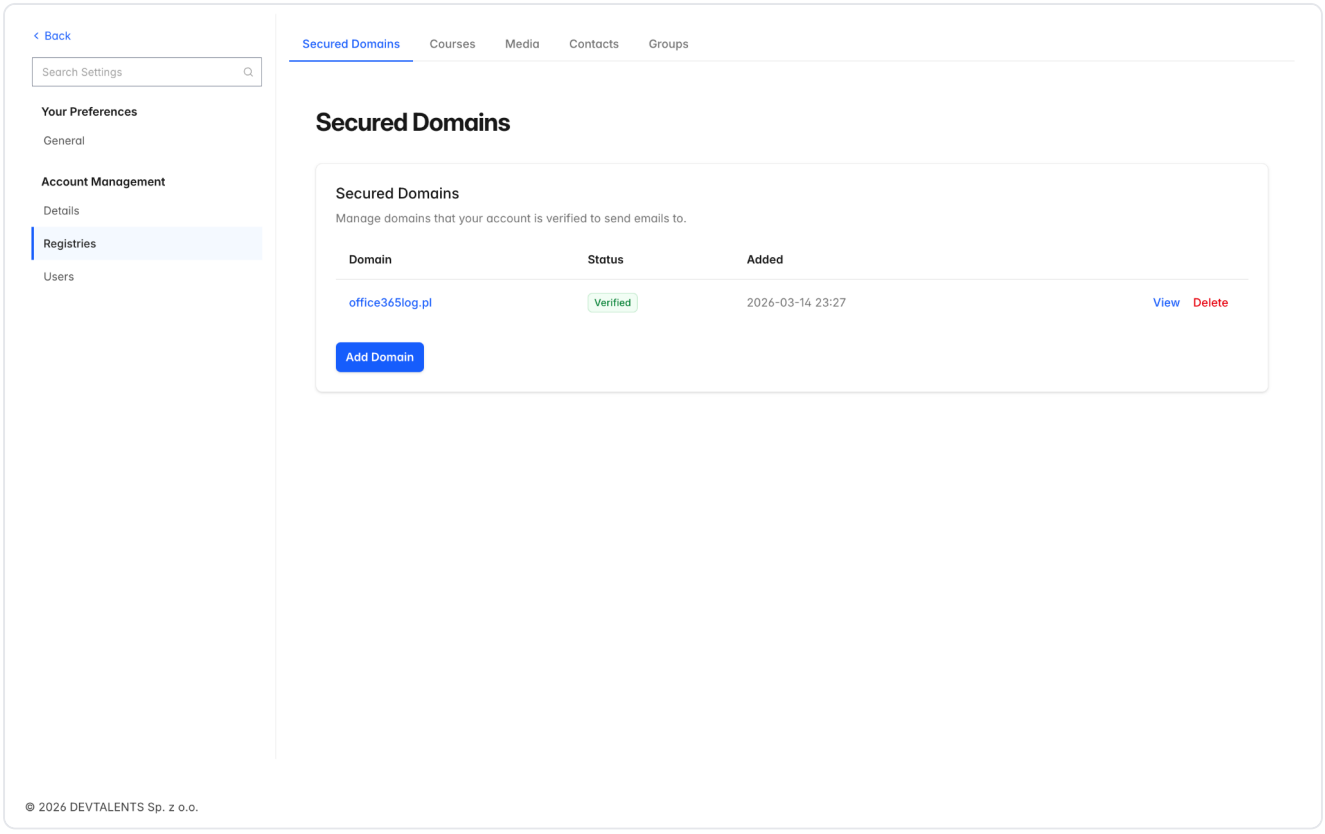
PhishSpot uses two types of domains: Secured Domains (for sending emails) and Platform Domains (for hosting landing pages).

9.1 Secured Domains (Sender Verification)

Secured domains verify that you own the email domains you send phishing simulations from. This ensures emails are delivered reliably and not flagged as spam.

Navigate to Settings → Secured Domains. The list shows each domain with its verification status:

Status	Meaning
Unverified	Domain has been added but verification has not been completed
Pending	Verification is in progress (DNS records added, awaiting propagation)
Verified	Domain ownership confirmed — ready for campaign use
Failed	Verification attempt failed — check your DNS records



Verifying a Domain

To add and verify a new secured domain:

1. Click New Domain and enter the domain name (e.g., yourcompany.com).
2. PhishSpot will provide DNS records (CNAME or TXT) that you need to add to your domain's DNS settings.
3. After adding the DNS records, click Verify DNS to check if the records have propagated.
4. Alternatively, you can verify via email — PhishSpot sends a verification code to a standard admin address on the domain.

> DNS propagation can take up to 48 hours. If verification fails, wait and try again.

9.2 Platform Domains (Landing Page URLs)

Platform domains are the URLs used for your phishing landing pages. When a recipient clicks the link in a phishing email, they are taken to a page hosted on one of these domains.

Navigate to Settings → Platform Domains. The list shows:

Column	Description
Name	Domain display name
Public	Whether this domain is shared across accounts or private to yours
Mail	Whether the domain can also be used for sending emails
Expires On	Expiration date (or Never if permanent)
Campaigns	Number of campaigns currently using this domain

> Platform domains are typically configured by your organization's IT team or the platform admin. You select from available domains when creating campaigns.

9.3 Custom Sending Domains (Bring Your Own Domain)

Custom domains let you send campaigns from a domain **you** own, registered at any registrar. Unlike Secured Domains (which only verify that you own an address you send to) and Platform Domains (managed by the platform admin), a custom domain is fully managed for you once you delegate it: you point its nameservers to us, and we create every DNS record automatically.

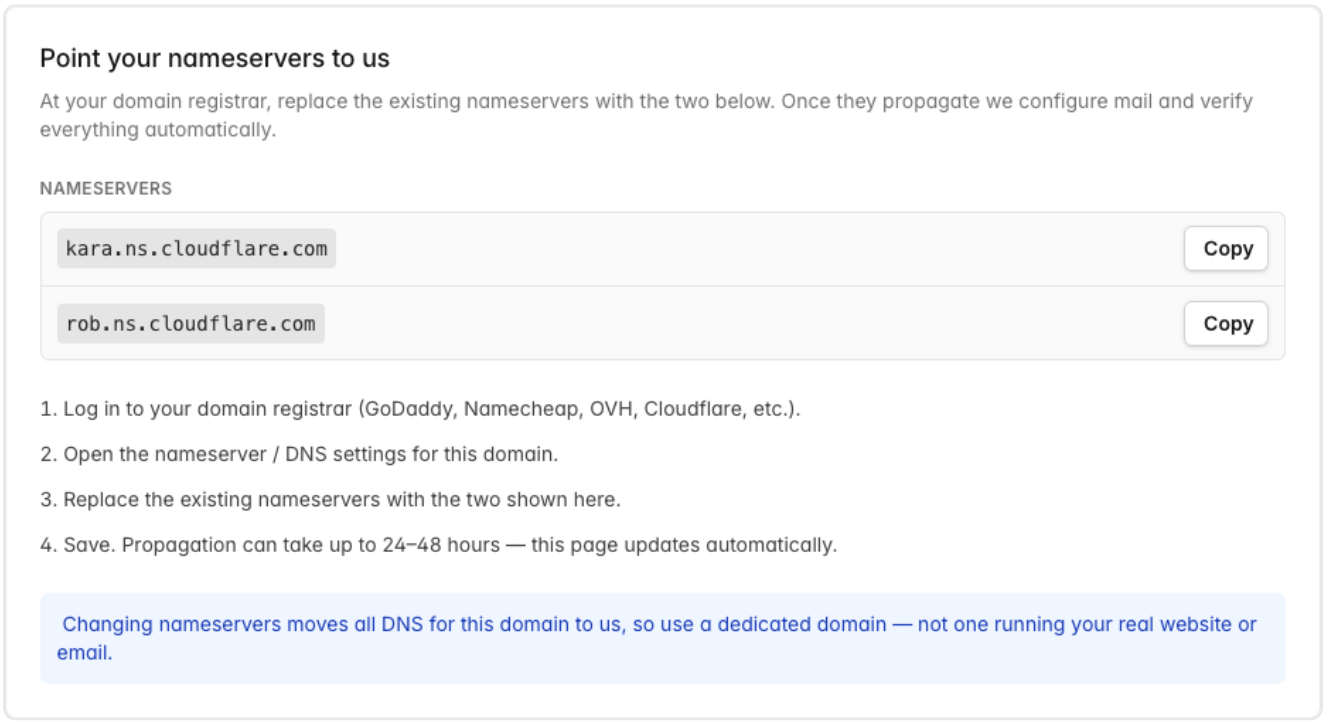
Navigate to **Settings** → **Custom Domains**.

Buying a dedicated domain

Use a **dedicated** domain bought just for simulations — not a domain you use for your real website or email. Pointing the nameservers to us moves **all** DNS for that domain to us, so any existing website or mail on it would stop working.

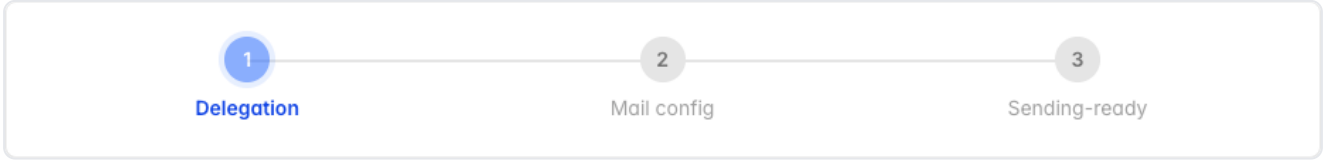
Getting your nameservers

1. Go to **Settings** → **Custom Domains** → **Add domain** and enter the domain you own.
2. The setup page shows the **two nameservers** to use, each with a copy button.



Setting nameservers at your registrar

1. Log in to your domain registrar (GoDaddy, Namecheap, OVH, Cloudflare, etc.).
2. Open the nameserver / DNS settings for the domain.
3. Replace the existing nameservers with the two shown on the setup page.
4. Save. Propagation can take up to **24–48 hours** — the page updates itself as it progresses.



What we verify

A live progress bar walks through three stages — **Delegation** → **Mail config** → **Sending-ready**. You can also press **Check status now** at any time. The status badge means:

Status	Meaning
Waiting for nameservers	Delegation not detected yet
Configuring mail	Delegation found; we're adding and verifying the sending records (SPF, DKIM, MX, Return-Path)
Sending-ready	Verified — the domain now appears in the sender list when you create a campaign
Needs attention	The domain was working but later broke; blocked for new campaigns
Setup failed	Setup couldn't complete — re-check your nameservers and try again

Healthy vs. blocked

Because a custom domain isn't under our control, we keep checking it. If the registration **expires**, the **nameservers change away** from us, or the **mail records are removed**, the domain is marked **Needs attention** and blocked from starting **new** campaigns. Campaigns already running on it keep delivering, and the account admins receive an email explaining what to fix.

> Renew your domain on time. We email the account admins when a custom domain is approaching its expiry date so it never lapses mid-program.

Troubleshooting

- **Stuck on "Waiting for nameservers"** — confirm both nameservers are set exactly as shown at your registrar; propagation can take up to 48 hours.
- **"Needs attention" after it was working** — open the domain to see the specific reason (expired / nameservers changed / mail records missing), fix it, then click **Check status now**.
- **Removing a domain** — deleting a custom domain tears down its DNS and mail configuration. You can't delete one while it has an active campaign; pause or finish those first.

Media Library

The media library stores files (images, documents, attachments) that you can use in your campaigns, courses, and landing pages.

Navigate to Settings → Media or find it on your Dashboard. The library shows a table of uploaded files with columns for Name, Content Type, a copyable URL link, creation date, and action buttons.

10.1 Uploading Media

Click New Media to upload a file. Enter a descriptive name and select the file from your computer. After upload, the file is assigned a permanent URL that you can copy and paste into email templates or landing page HTML.

[Screenshot: Media upload form]

Reports & Analytics

PhishSpot provides several ways to analyze your campaign results and track your organization's security posture over time.

11.1 Campaign Reports

Each campaign has its own dashboard (see section 4.5) with detailed funnel charts, recipient tables, and group breakdowns. You can export individual campaign reports in two formats:

- **PDF Report** — A formatted document suitable for sharing with management.
- **Excel Report** — A spreadsheet with raw data for further analysis.

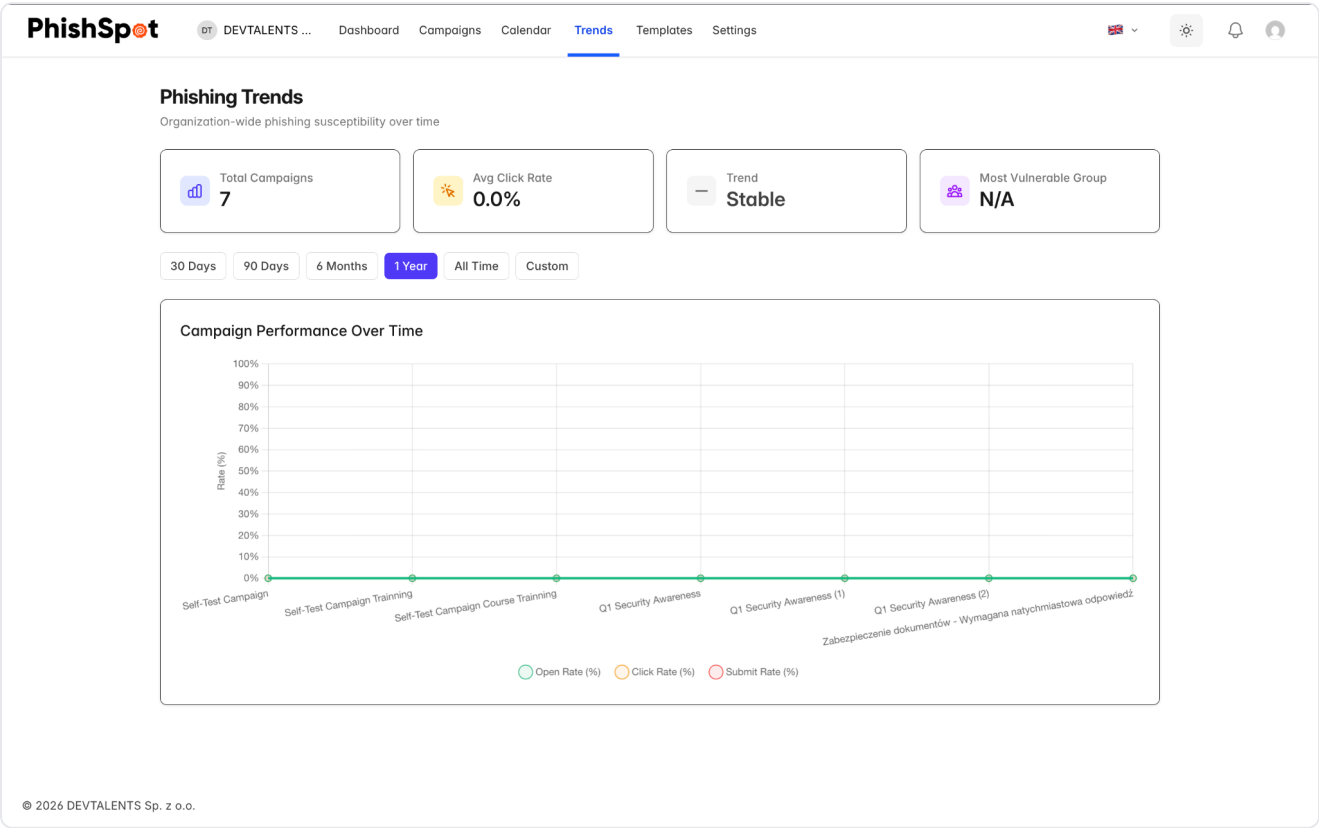
11.2 Cumulative Reports

From the campaigns list page, you can generate cumulative reports that aggregate data across all campaigns. This is useful for quarterly or annual security awareness reviews.

11.3 Trend Dashboard

Navigate to Trends from the sidebar to access the trend dashboard. This view shows historical performance data across all your campaigns with date range filtering options: 30 days, 90 days, 6 months, 1 year, all time, or a custom date range.

The trend dashboard helps you answer questions like: Are employees getting better at recognizing phishing emails over time? Which departments need additional training? Is the click rate decreasing campaign over campaign?



11.4 Recipient Timeline

Within any campaign dashboard, clicking on a recipient opens a detailed timeline panel showing every tracked event for that person: when the email was sent, when it was opened, when the link was clicked, when the landing page was viewed, whether data was submitted, and whether the training course was started or completed.

11.5 Previewing the email each recipient received

Every row in a campaign's **Recipients** table has a small magnifier-on-envelope icon next to the recipient's email address. Click it to open a modal showing the exact email that contact received — with every template variable (`{{first_name}}` , `{{company}}` , `{{position}}` …) substituted with that contact's actual values, the actual landing-page URL embedded, the actual From: address used. Not a generic preview: the rendered mail for that specific recipient.

The modal has a **desktop / mobile** viewport toggle at the top — flip between the two to see how the email looked on a 1920px-wide Outlook client vs. an iPhone Mail rendering. Useful during stakeholder review of a finished campaign ("show me exactly what Anna saw on her phone") and during incident investigations ("did the link in this specific delivery target the right domain?").

The same preview is also available from a contact's individual **deliverables** view — open any contact's detail page and the deliverable rows have the same magnifier. Two perspectives on the same modal: per-campaign (every recipient on one campaign) and per-contact (every campaign one person received).

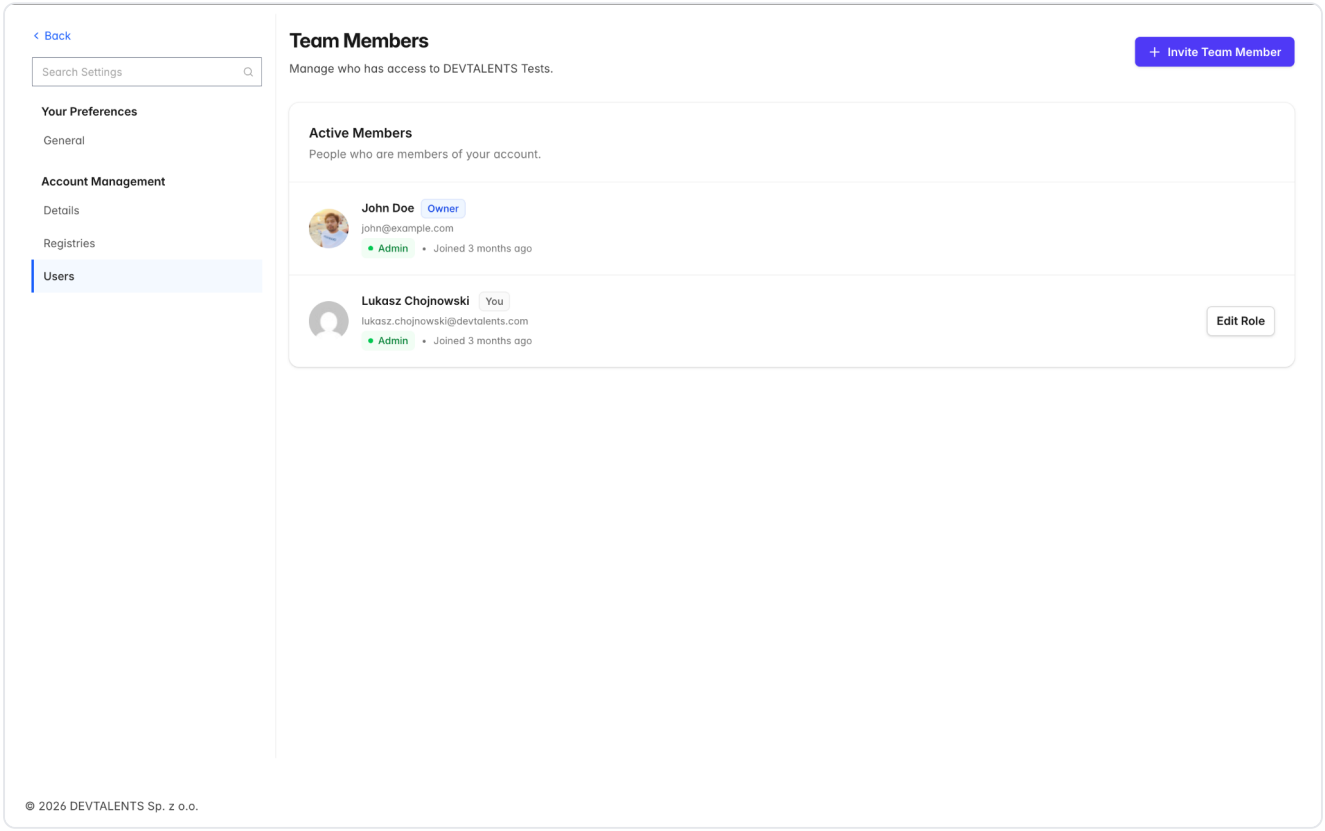
Team Management

As an Admin, you can manage who has access to your account and what they can do.

12.1 Viewing Team Members

Navigate to Settings → Team Members. This page shows two sections:

- **Active Members** — All users currently in your account, showing their name, email, role (Admin/Editor/Member), and join date. The account owner is marked with an Owner badge.
- **Pending Invitations** — Invitations that have been sent but not yet accepted.

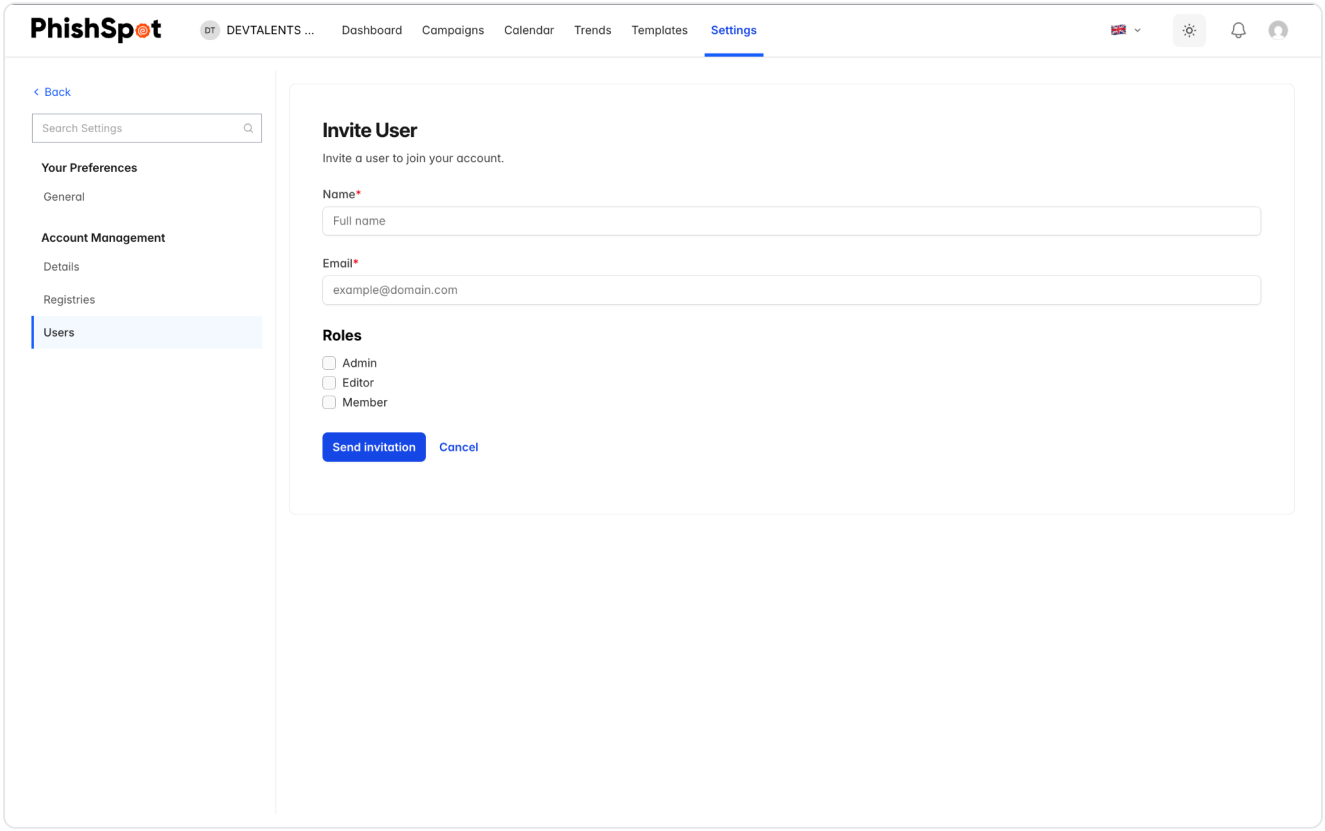


12.2 Inviting New Members

Click the Invite Member button to add someone to your team:

1. Enter the person's email address.
2. Select their role: Admin, Editor, or Member.
3. Click Send Invitation.

The invited person will receive an email with a link to accept the invitation. If they already have a PhishSpot account, they will see the new team appear in their account switcher. If they don't have an account, they will be prompted to create one.



For pending invitations, you can resend the invitation email or cancel it entirely.

12.3 Changing Roles

To change a team member's role, click the Edit button next to their name. Select the new role and save. Remember:

- Only Admins can change other members' roles.
- The account owner's role cannot be changed — they are always an Admin.
- You cannot change your own role.

12.4 Removing Members

To remove a team member, click the Remove button next to their name and confirm. The member will lose access to the account but their historical data (e.g., actions in campaign logs) is preserved. The account owner cannot be removed.

12.5 Transferring Ownership

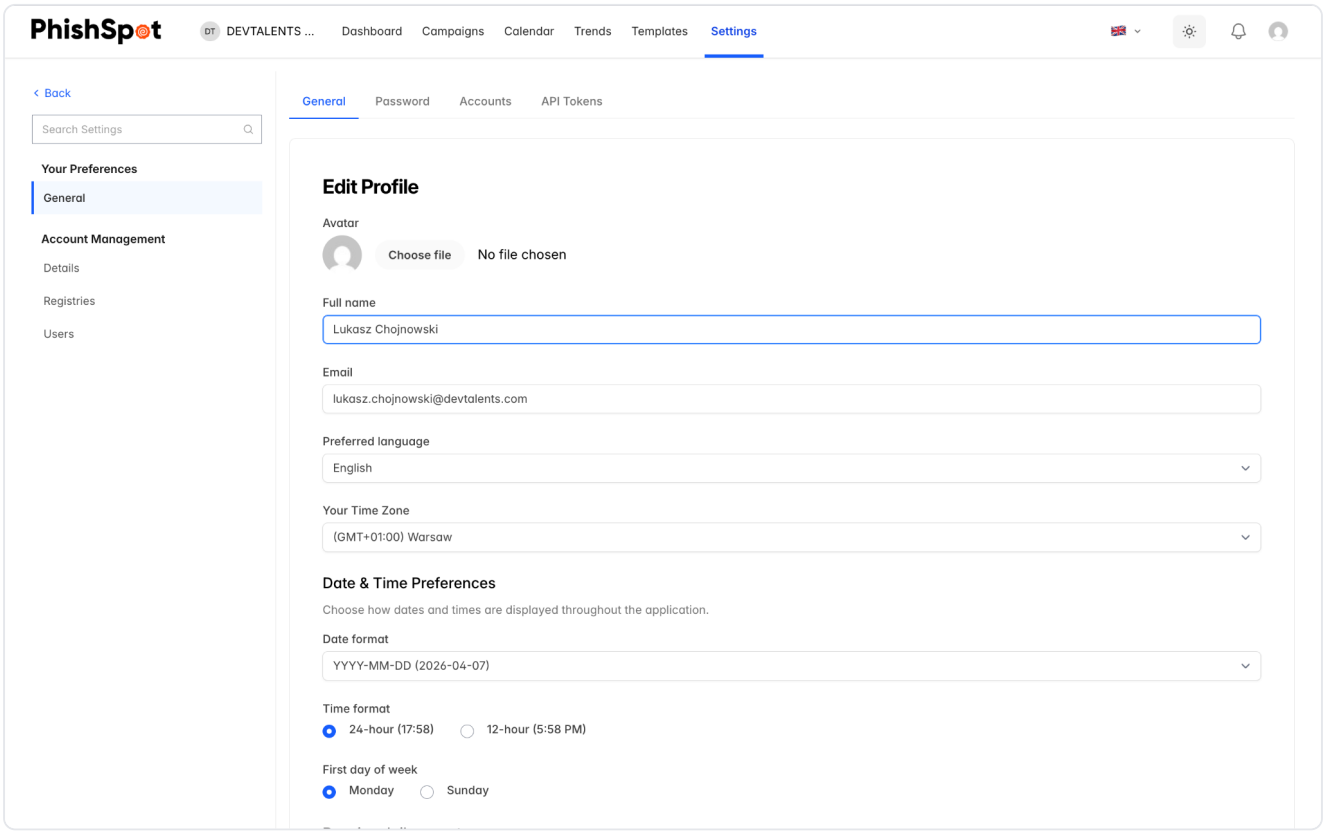
If you are the account owner, you can transfer ownership to another Admin on the team. Navigate to Settings → Account Details and use the Transfer Ownership option. The target user must already be an Admin. After transfer, they become the new owner and you remain as an Admin.

Account Settings

Navigate to Settings → Account Details to configure your account preferences.

13.1 Basic Information

- **Account Name** — Your organization or team name.
- **Time Zone** — The default timezone used for campaign scheduling and report timestamps.
- **Primary Language** — The interface language (English or Polish).
- **Avatar** — An optional account image/logo.



13.2 Business Hours

Enable business hours to restrict when campaign emails are delivered. When enabled, configure:

- Which days of the week emails can be sent (Monday through Sunday checkboxes)
- Start time and end time for the delivery window

Emails scheduled outside business hours will be queued and delivered during the next business window.

13.3 Default Awareness Page

Configure the default HTML page shown to users after they interact with a phishing simulation. This is the awareness/educational message that appears if no specific course is assigned to a campaign. You can edit the HTML using the built-in code editor and preview it in real time.

13.4 Deleting an Account

At the bottom of the Account Details page, there is a Delete Team button. This permanently deletes the account and all associated data including campaigns, contacts, templates, and results. This action cannot be undone. Only the account owner can delete the account.

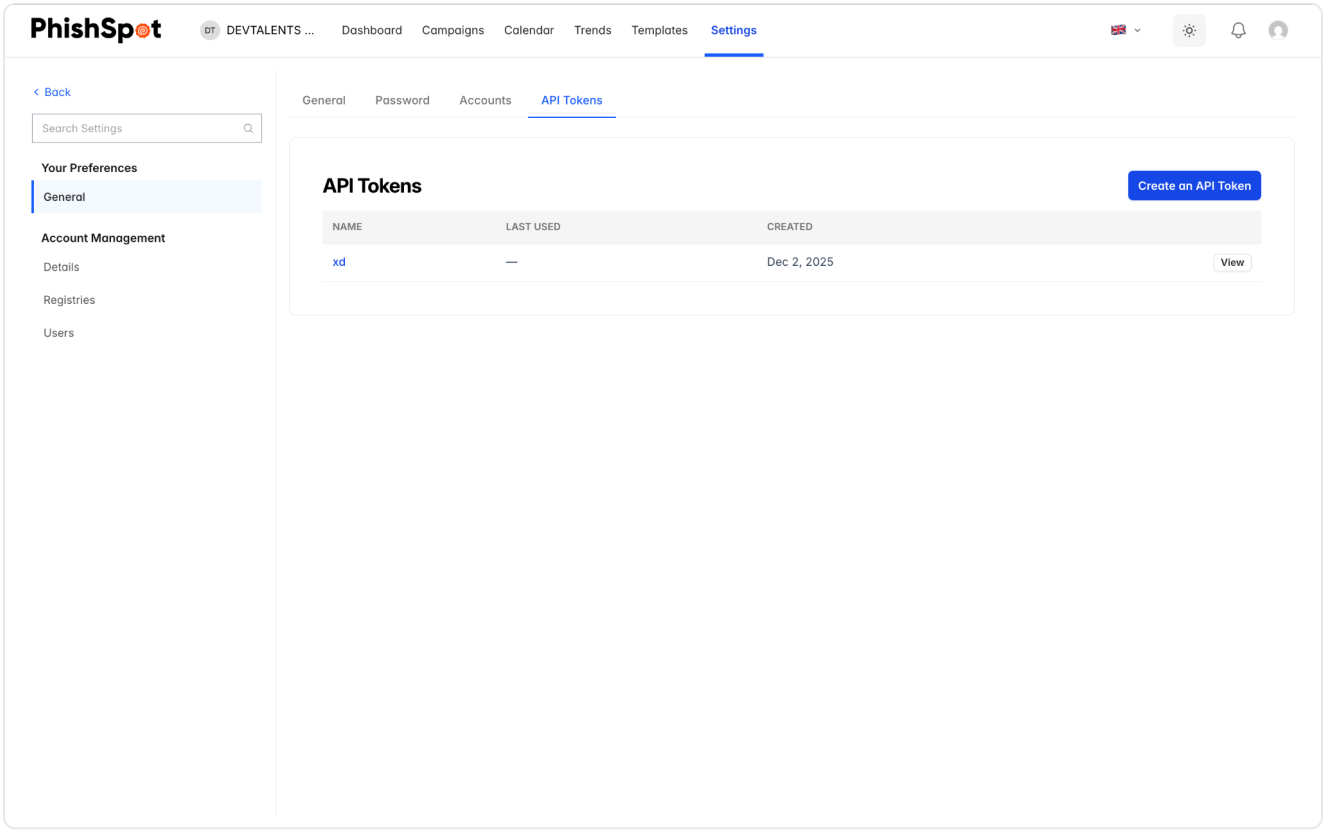
> Personal accounts cannot be deleted. Only team/organizational accounts have the delete option.

API Tokens

PhishSpot provides a REST API for programmatic access. To use the API, you need an API token.

14.1 Managing Tokens

Navigate to your user profile and select API Tokens. The token list shows all your tokens with their name, last used date, and creation date. Click New Token to create one. After creation, the token value is displayed once — make sure to copy it immediately as it cannot be viewed again.



> API tokens are tied to your user account, not to a specific team. Keep them secure and rotate them periodically.

User Profile & Preferences

Access your personal settings by clicking your name or avatar in the top-right corner and selecting your profile.

15.1 Profile Settings

- **Name and Email** — Update your display name and email address.
- **Password** — Change your login password.
- **Two-Factor Authentication** — Enable or manage 2FA for additional security.
- **Theme** — Switch between light and dark mode.

Common Workflows

16.1 Running Your First Campaign

Follow these steps to run your first phishing simulation:

1. Add your contacts via CSV import (Settings → Contacts).
2. Create at least one group and assign your contacts to it (Settings → Groups).
3. Verify your sending domain (Settings → Secured Domains).
4. Browse the template library and find a suitable phishing template (Templates).
5. Click Quick Launch on the template, or create a new campaign manually (Campaigns → New Campaign).
6. Walk through the 6-step wizard: configure settings, customize the email, set up the landing page, choose a post-click action, select recipients, and review.
7. Send a test email to yourself and verify everything looks correct.
8. Start or schedule the campaign.
9. Monitor results on the campaign dashboard.

16.2 Ongoing Phishing Program

For a continuous security awareness program:

1. Schedule recurring campaigns to automatically repeat at set intervals.
2. Use different templates each time to test various attack vectors.
3. Monitor the Trend Dashboard to track improvement over time.
4. Focus additional training on departments or groups with higher click rates.
5. Export cumulative reports for quarterly management reviews.
6. Regularly update your contact list as employees join and leave.

16.3 Responding to High-Risk Users

When the platform identifies users with high risk scores (red badges):

1. Review their profile to see which campaigns they fell for.

2. Check if they completed the assigned training courses.

3. Consider assigning them to a special group for additional targeted campaigns.

4. Use the contact filters to find all high-risk users across the organization.

Template Variables

When writing email content or landing pages, you can use template variables — merge tags — to personalize the content for each recipient. Variables are replaced with each recipient's actual values when the email is sent or the landing page is shown.

Wrap a variable in double curly braces: `{{first_name}}`. The available variables differ between the **email** (subject and body) and the **landing page** (and awareness message), because each is rendered in a different context. The editor validates this — you can't save an email that references a landing-only variable.

Email variables (subject & body)

Variable	Description	Example output
`{{first_name}}`	Recipient's first name	John
`{{last_name}}`	Recipient's last name	Smith
`{{full_name}}`	Recipient's full name	John Smith
`{{email}}`	Recipient's email address	john.smith@company.com
`{{position}}`	Recipient's job position	Senior Analyst
`{{department}}`	Recipient's department	Finance
`{{company}}`	Your account (organization) name	Acme Inc.
`{{campaign_name}}`	The campaign's name	Q2 Invoice Test
`{{landing_url}}`	The recipient's tracked link	https://domain.com/l/abc123?d=…

Landing page & awareness message variables

Variable	Description
`{{first_name}}`, `{{last_name}}`, `{{full_name}}`, `{{email}}`	As above
`{{company}}`	Your account (organization) name
`{{landing_url}}`	The recipient's tracked link
`{{elearning_url}}`	The recipient's training link (for use on the awareness page)

> Variable names are **case-insensitive** and tolerate surrounding spaces — `{{First_Name}}` and `{{ first_name }}` both work. If a variable has no value for a recipient, it is replaced with an empty string. A variable name that isn't on the list above is left in the content **literally**, so watch for typos and always send a test email.

For guidance on *using* these variables effectively in a campaign, see Designing Effective Campaigns §30.1.

Troubleshooting

18.1 Emails Not Being Delivered

- Verify your sending domain is in Verified status (Settings → Secured Domains).
- Check that the from email address matches a verified domain.
- Ensure the campaign is in Active state and has not been paused or stopped.
- If using business hours, confirm the current time falls within the configured delivery window.

18.2 Landing Page Not Loading

- Verify that a platform domain is selected in the campaign settings.
- Check that the landing page is enabled (the toggle in Step 3 of the wizard).
- Ensure the platform domain has not expired.

18.3 Contacts Not Importing

- Download the sample CSV and verify your file matches the expected format.
- Ensure the email column contains valid email addresses.
- Check for duplicate emails — contacts with duplicate email addresses will be skipped.
- If rows fail, download the failed CSV to see specific error messages.

18.4 Cannot Edit a Campaign

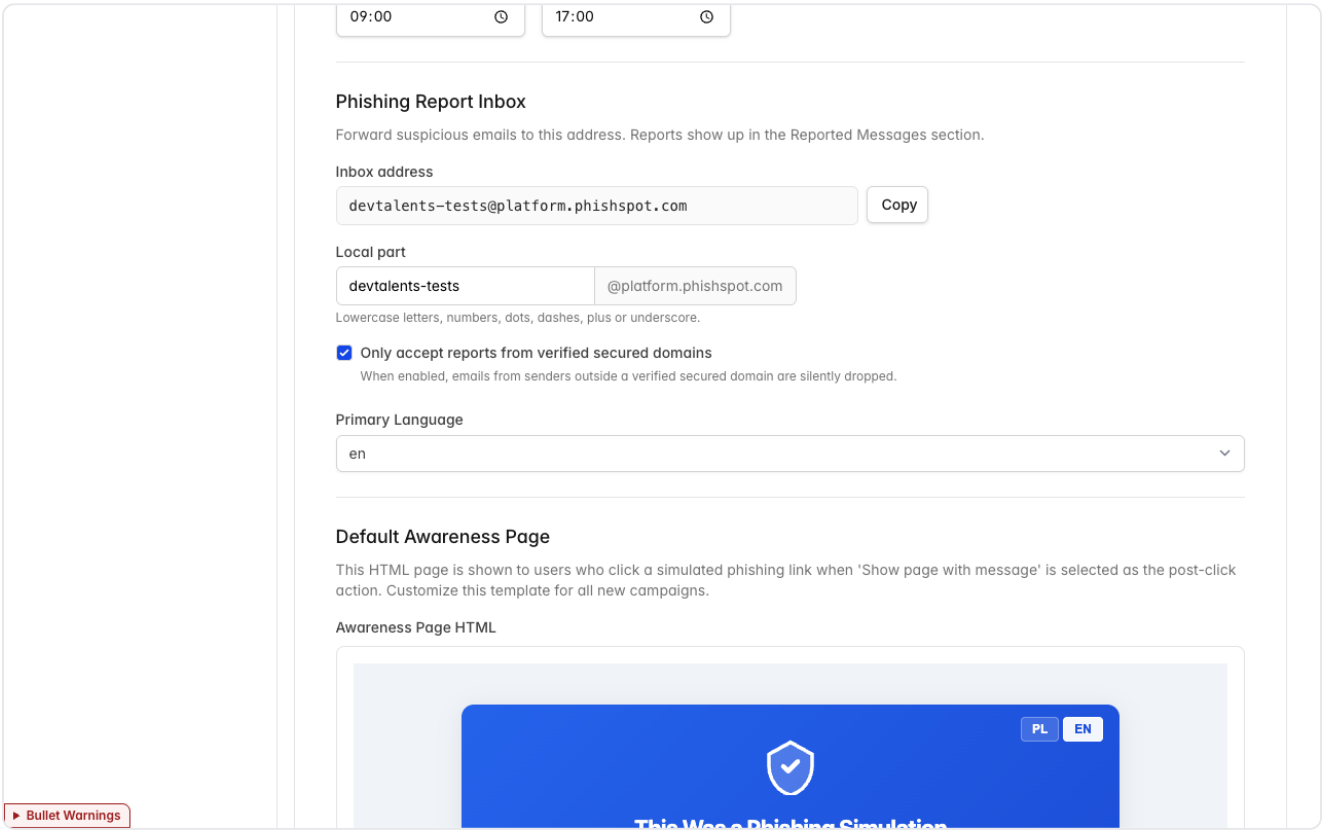
- Only campaigns in Draft or Scheduled state can be edited.
- Active, Paused, and Done campaigns are read-only.
- If you need to modify an active campaign’s content, duplicate it, make changes, and start the new copy.

Reported Messages

Reported Messages is the inbox where your employees forward suspicious emails so your team can review them inside PhishSpot. Every account gets its own unique address. Reports show up in a dedicated section in the main navigation. The viewer keeps potentially dangerous content blocked by default — images, styling, links and attachments are all switched off until you decide otherwise.

19.1 The Phishing Report Inbox

Each account gets its own inbox address in the form `<local>@platform.phishspot.com`. The local part is filled in automatically when the account is created and can be edited from **Settings** → **Account Details** → **Phishing Report Inbox**.



- **Inbox address** — the full address employees should forward to. The **Copy** button puts it on the clipboard so you can paste it into onboarding materials, signatures, or your help-desk knowledge base.
- **Local part** — the editable username part of the address. Use lowercase letters, numbers, dots, dashes, `+` or underscores.
- **Only accept reports from verified secured domains** — see 19.2.

19.2 Limit accepted senders

The **Only accept reports from verified secured domains** toggle (ON by default) limits who can submit a report:

- **ON** — only emails whose sender domain matches one of your **verified Secured Domains** are accepted. Everything else is discarded without notifying the sender.
- **OFF** — any sender is accepted.

Leave it ON in production. Switch it OFF temporarily during pilots when reporters might send from mailboxes you haven't onboarded yet.

When a message is dropped, the sender is not notified — this is intentional, so that someone probing your inbox does not get a confirmation.

19.3 How a report arrives

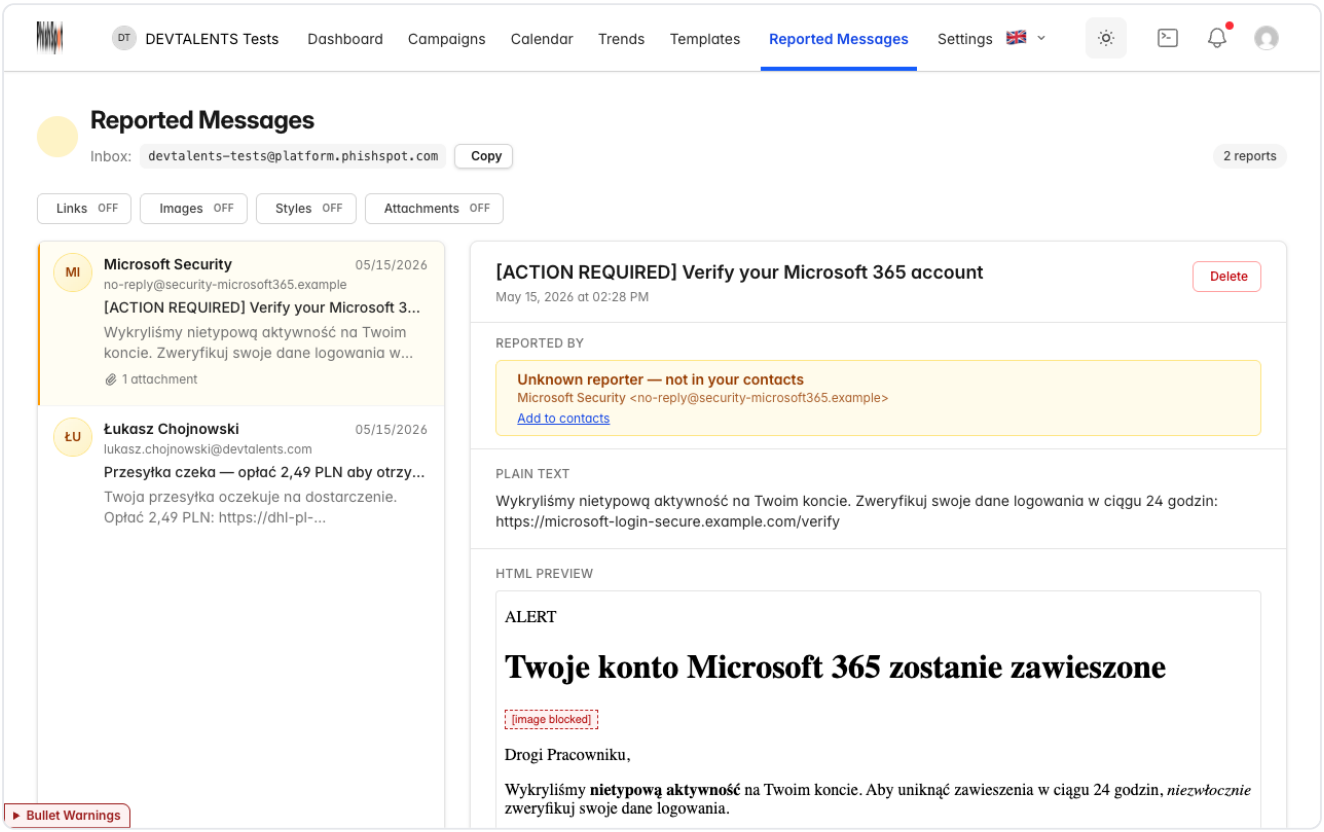
The flow for your team:

1. An employee receives a suspicious email.
2. They forward it to the account's Reported Messages inbox address.
3. The report appears in **Reported Messages** in the main navigation, sorted with the newest first.

Promote the inbox address to your employees through onboarding, your help-desk knowledge base, or a signature footer.

19.4 The Reported Messages page

Open **Reported Messages** from the top navigation. The page is a two-pane list/detail view:



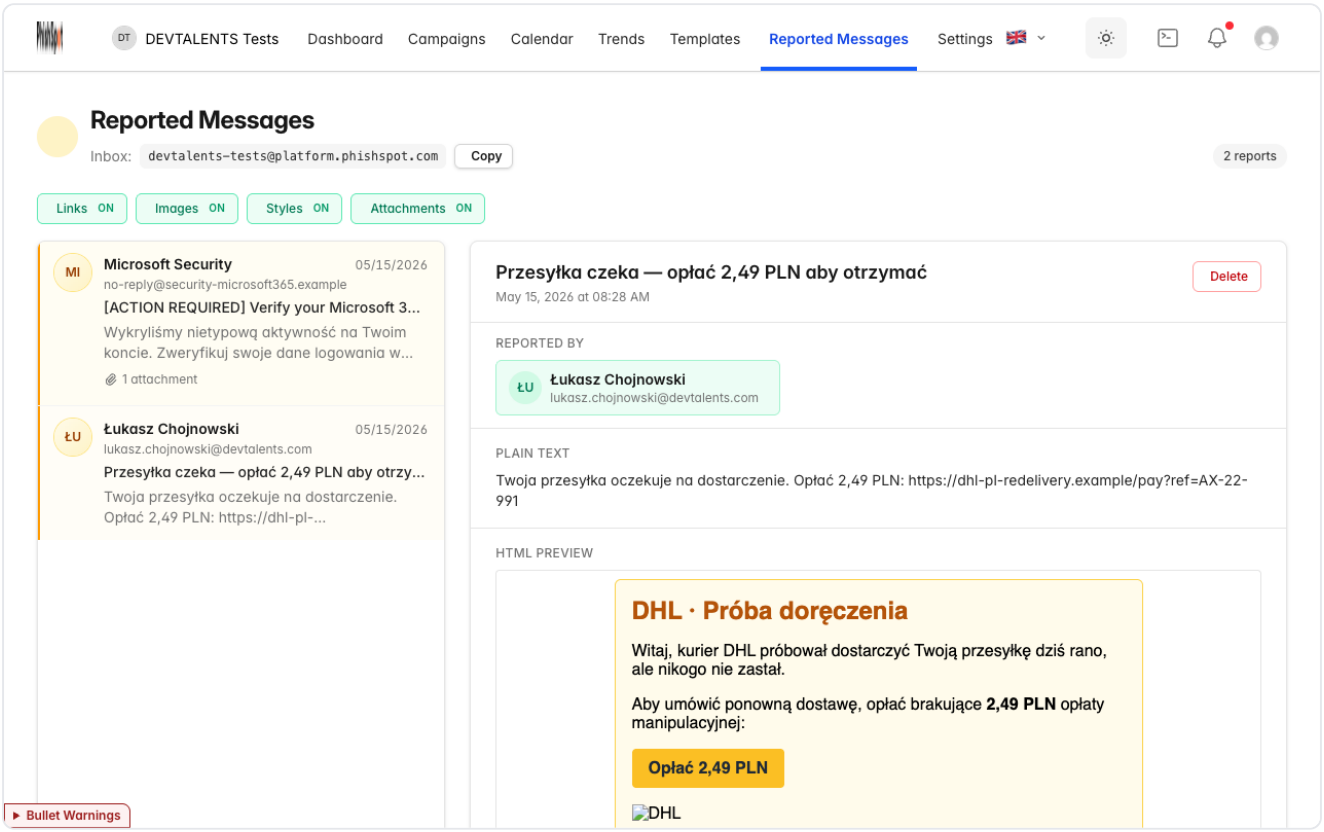
- **Left** — list of reports, newest first. Each item shows the reporter's name (or email), the email's subject and excerpt, an attachment count, and the receive date.
- **Right** — the detail panel for the selected report. Click a different item on the left and the right pane updates instantly.
- **Counter** — the pill in the top-right shows how many reports your account has.
- **Inbox** — the receiving address is repeated under the title with a **Copy** button.

19.5 Who reported it

Right under the subject, every report shows a **Reported by** panel that tells you whether the sender is known to your account.

Known reporter

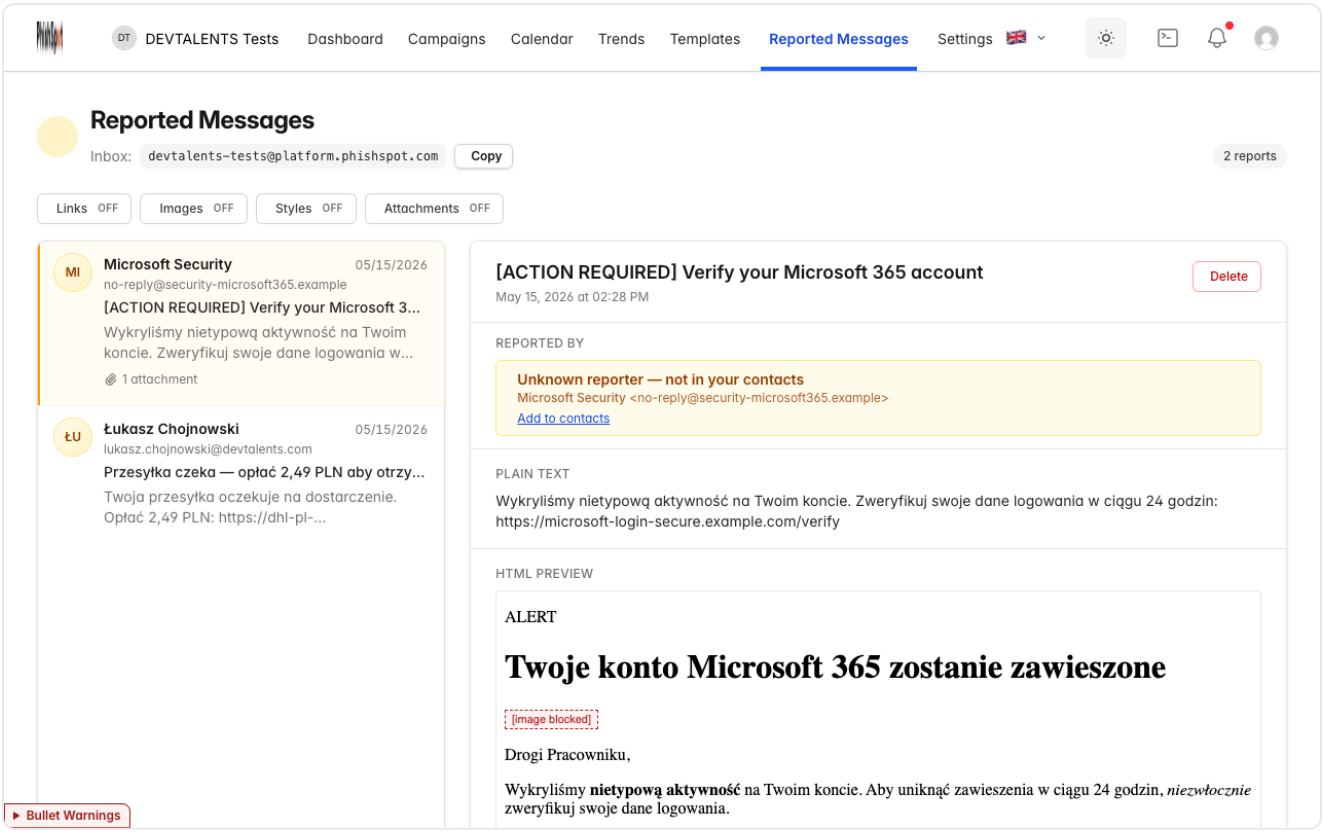
If the sender's email matches a **Contact** in your account, the panel is green and acts as a link to that contact's profile:



Clicking the panel takes you straight to the contact, where you can review their group memberships, prior campaign results, and contact history.

Unknown reporter

If no contact matches, the panel is amber and shows the sender details plus a quick-add link:



The **Add to contacts** link opens the new-contact form with the email already filled in — just add the first/last name and save.

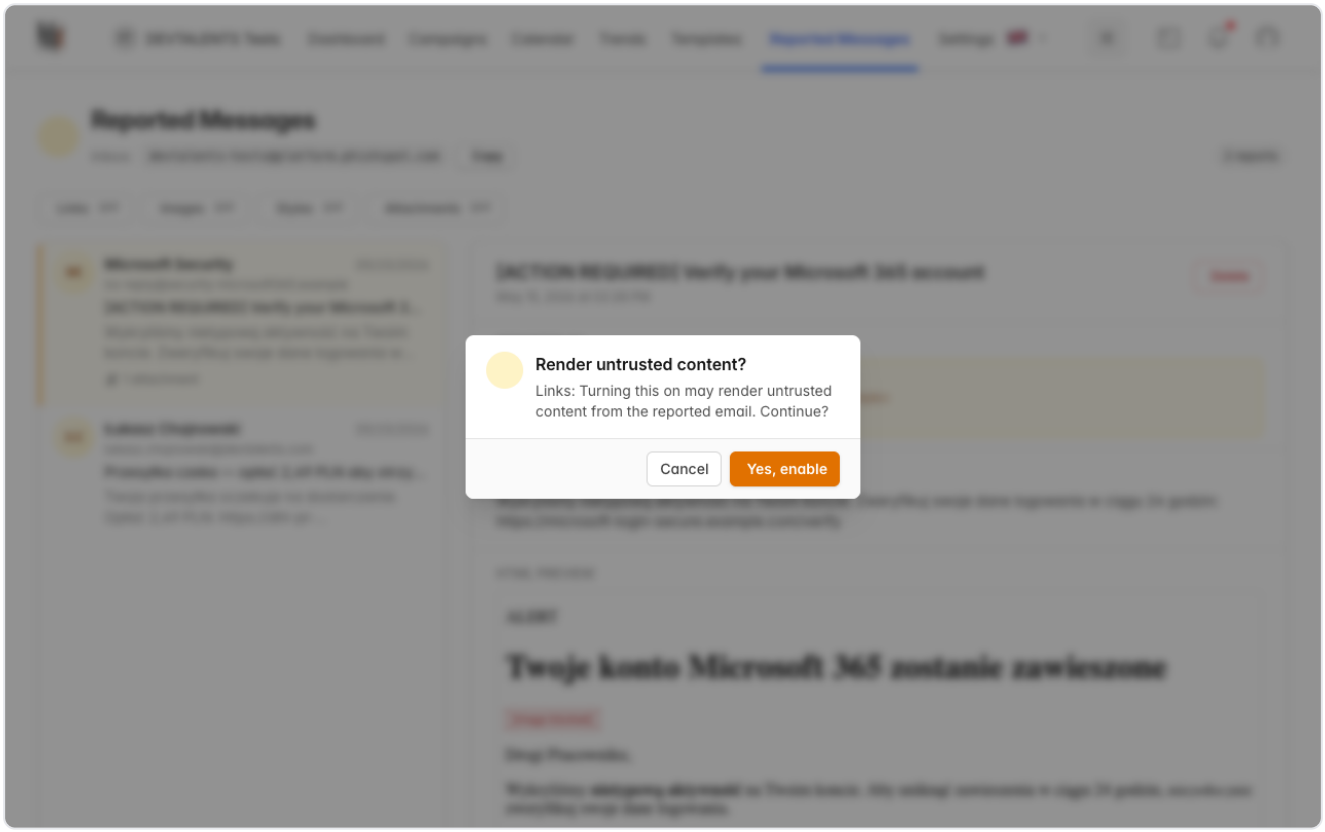
A report from an address you have never onboarded deserves extra scrutiny. The amber warning is your first signal that something is off.

19.6 Safe-preview controls

Phishing emails are by definition untrusted. The detail view keeps potentially dangerous parts of the email switched OFF by default. Four controls let you progressively allow more of the original content:

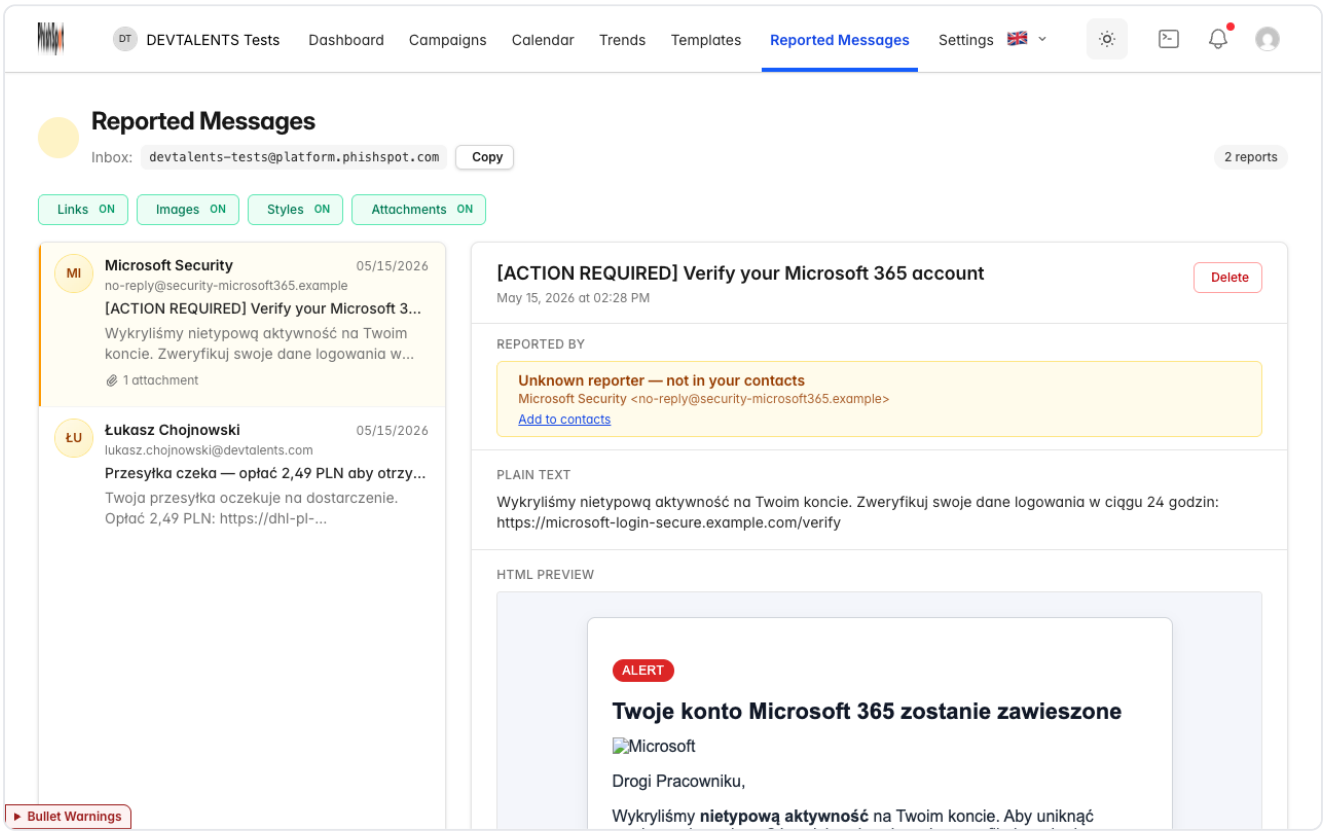
Control	When OFF (default)	When ON
Links	Links are shown as red strike-through text. Hover to see the destination URL in a tooltip. Not clickable.	Links are clickable and open in a **new browser tab** (never inside PhishSpot).
Images	Each image is replaced with a `[image blocked]` placeholder.	Images load from their original sources.
Styles	All custom styling is removed — plain text only.	The email is rendered with its original styling.
Attachments	Filenames are listed, but no download buttons are shown.	Each attachment has a **Download** button.

Turning any control ON opens a confirmation dialog:

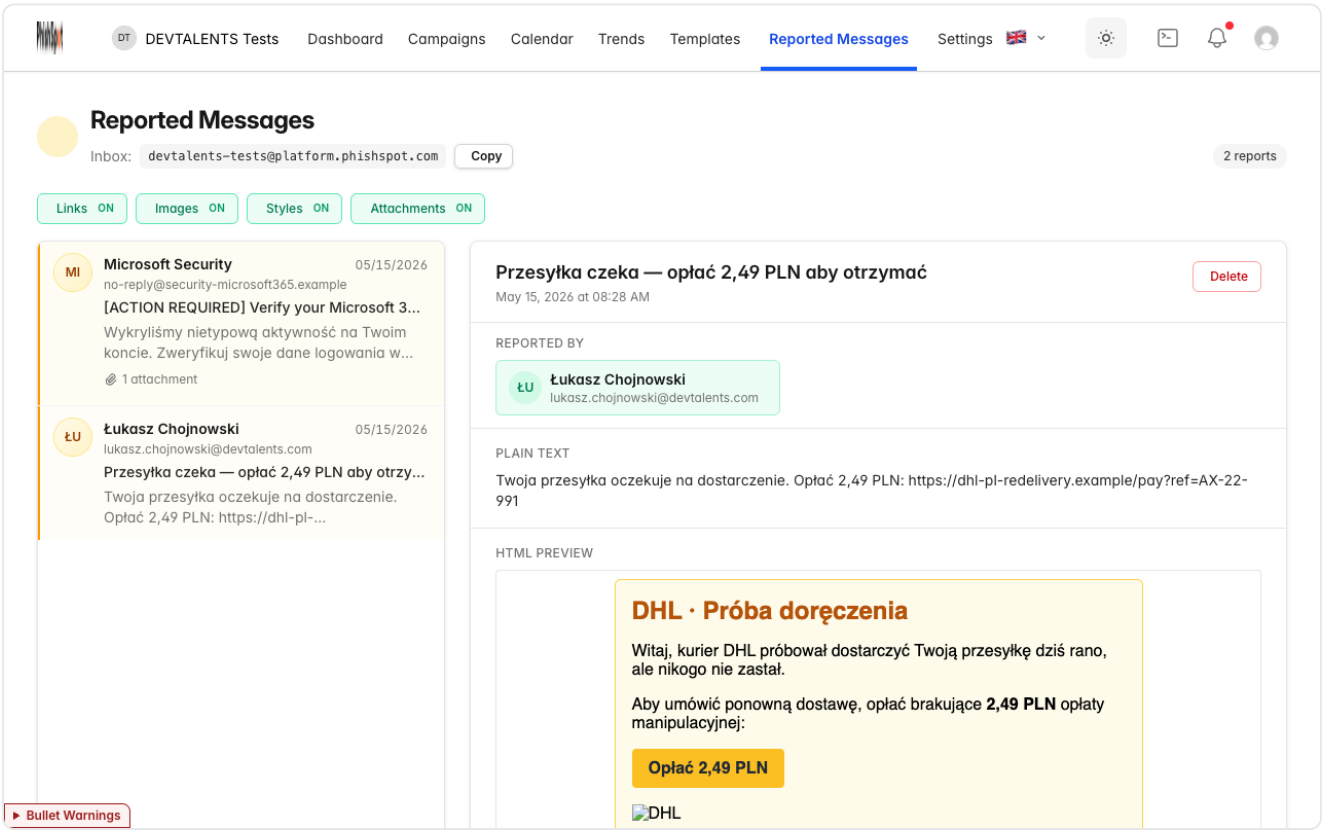


Turning a control OFF requires no confirmation.

Once styles and images are both enabled, the email renders the way the attacker designed it — useful when investigating a click-through campaign:



With Links ON, the buttons inside the email become clickable. They always open in a **new browser tab**, so you can inspect the destination URL without leaving PhishSpot:



> Controls are **account-level**, not per-report. Flipping one affects every report in the inbox. The default is **everything OFF** so new accounts are safe from the start.

19.7 Deleting a report

The **Delete** button in the top-right of the detail panel removes the report (after a confirmation). Only **admins** and **editors** can delete; **members** can only view.

Outlook Add-in

The PhishSpot Outlook add-in puts a **Report Phishing** button on every email you read. One click sends the message (body, headers, attachments) to your PhishSpot account's Reported Messages list. No forwarding to a special address, no manual copy-paste.

This page is for end users. If you're an admin rolling the add-in out to a whole organisation, see Outlook Add-in: Central Deployment.

20.1 What you need

- Outlook on the web, Outlook for Windows or Mac, or the **new** Outlook for Windows.
- A PhishSpot **Contact** account at your organisation (your IT team can create one for you if you don't have one).
- A few minutes to install the add-in and pair it.

The add-in does not work on Outlook for iOS / Android in v1.

20.2 Install the add-in

1. Download the sideload package: phishspot-outlook-sideload-v1.1.0.zip.
2. Unzip it. You'll get a `manifest.xml`, a folder of icons, and a `README.md` with click-by-click instructions for each Outlook variant.
3. In Outlook, open **Get Add-ins** → **My add-ins** → **Add a custom add-in** → **Add from File…** and pick `manifest.xml`.
4. Confirm the install dialog. The **Report Phishing** button appears on the message-read ribbon.

If your IT team has already deployed the add-in centrally for everyone, **skip the install** — you'll see the button automatically.

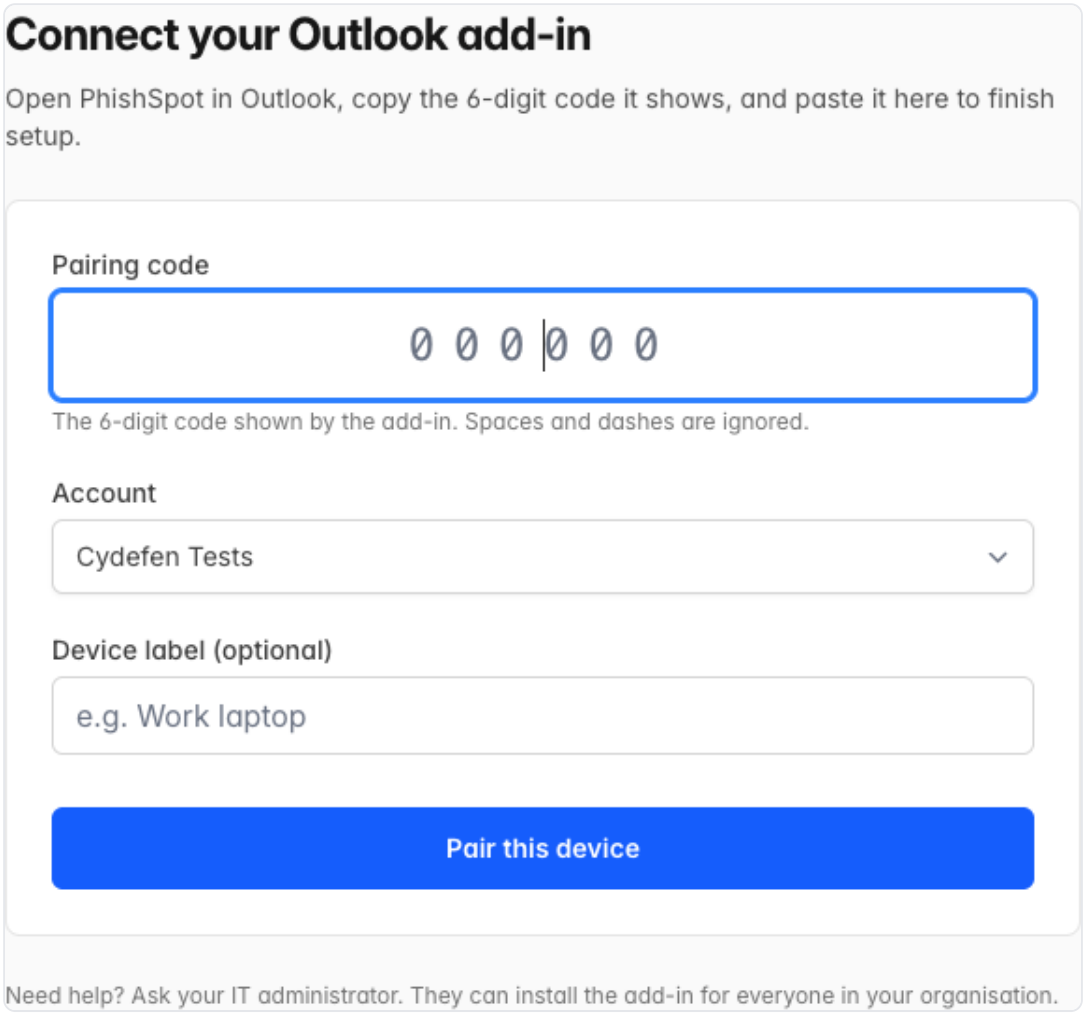
20.3 Pair the add-in (one-time)

The first time you click **Report Phishing**, the add-in shows a 6-digit code in the PhishSpot task pane on the right side of Outlook:

Outlook taskpane showing a 6-digit pairing code, copy button, and waiting-for-activation status

Outlook taskpane showing a 6-digit pairing code, copy button, and waiting-for-activation status

1. Open `https://platform.phishspot.com/guest/activation/new` in a browser (the **here** link in the task pane will take you straight there). Once signed in, you'll see the **Connect your Outlook add-in** page:



Connect your Outlook add-in page in the browser, with the pairing-code input, account selector, optional device label, and Pair this device button

2. Sign in with the same email your IT team registered for PhishSpot.
3. Type or paste the 6-digit code shown in the task pane.
4. Pick which account you're pairing with (if you belong to more than one), optionally give the device a name, then click **Pair this device**.

The add-in detects the pairing within a few seconds and switches to its normal view, with a big **Report suspicious message** button and your organisation name underneath:

Outlook taskpane in the paired state — Report suspicious message button, organisation name, watermark logo, and theme/language controls at the bottom
Outlook taskpane in the paired state — Report suspicious message button, organisation name, watermark logo, and theme/language controls at the bottom

The pairing is per-device — if you have Outlook on two computers, you'll pair each one separately. Optionally name the device (e.g. "Work laptop", "Home iMac") so admins can tell them apart on the API tokens screen.

20.4 Report a suspicious email

1. Open the email you suspect is phishing.
2. Click **Report suspicious message** in the task pane.
3. The task pane briefly shows “Reporting…” while the message is sent.
4. A thank-you screen confirms the report:

   Outlook taskpane showing the green-check Thank you confirmation after a successful report
   Outlook taskpane showing the green-check Thank you confirmation after a successful report

Click **Close** to dismiss the task pane.

The report appears in your organisation’s **Reported Messages** list. Your security team will review it.

20.5 What gets sent

- The sender’s email address and display name
- The subject and the message body (HTML + plain text)
- The full internet headers
- All file attachments
- A timestamp and the message’s Internet Message ID (for deduplication)

The bearer token the add-in uses is scoped to **reported_messages:create** only. The add-in cannot read, modify, or send any of your other mail.

20.6 The “Update available” banner

Each time you click the button the add-in checks its version against the server. Two outcomes:

- **An update is available** — soft banner; you can still report. Ask your IT team to push the new version when convenient.
- **Update required** — hard block; the button is hidden until the add-in is updated. This happens only when an old version is incompatible with a server change (rare).

20.7 Unpair / sign out

In the paired card, click **Unpair this device**. The token is removed from your Outlook. Your IT team can additionally revoke the token from the PhishSpot admin under **API Tokens**.

Outlook Add-in: Central Deployment

This page is for IT administrators who want the **Report Phishing** button to appear automatically for every user in their M365 tenant. For end-user install steps, see Outlook Add-in.

21.1 What gets installed

A small XML manifest (~5 KB). The manifest's `SourceLocation` points at `https://platform.phishspot.com/outlook/taskpane`, which loads the live UI bundle. Result: **shipping a new feature does not require re-distributing the add-in** — the manifest itself only changes when the button label, permissions, or icons change.

21.2 Download the artifact

Download the manifest file for direct upload:

phishspot-outlook-manifest-v1.0.0.xml

Or the full sideload package (zip with icons + README):

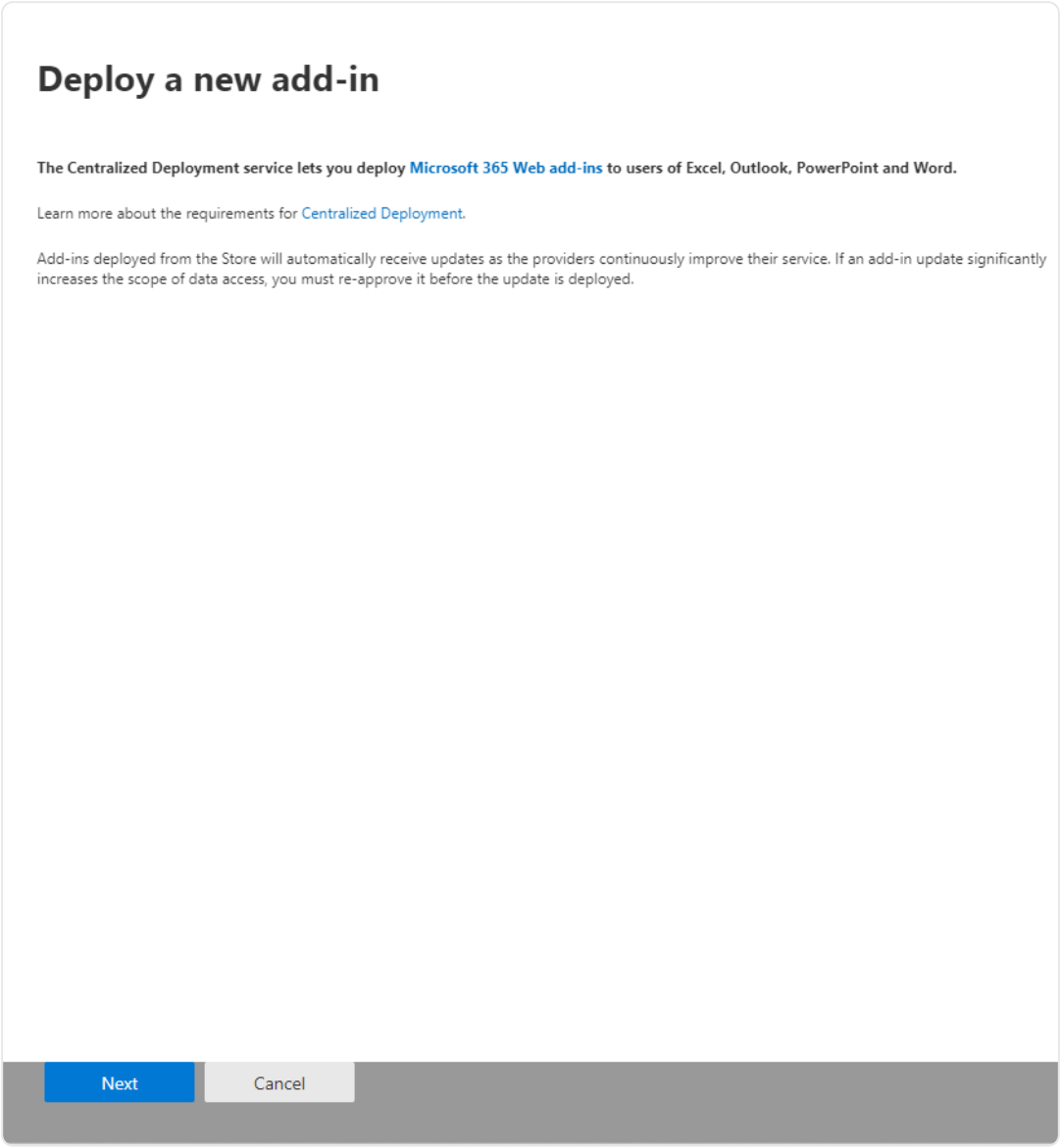
phishspot-outlook-sideload-v1.0.0.zip

21.3 Deploy via Microsoft 365 Admin Center

PhishSpot uses the **add-in only manifest** format, so deploy it from the **Integrated apps** portal (Microsoft's recommended path). The classic admin-center *Add-ins* portal works too but only supports this manifest type — the unified Microsoft 365 manifest is not used here, so either portal is fine.

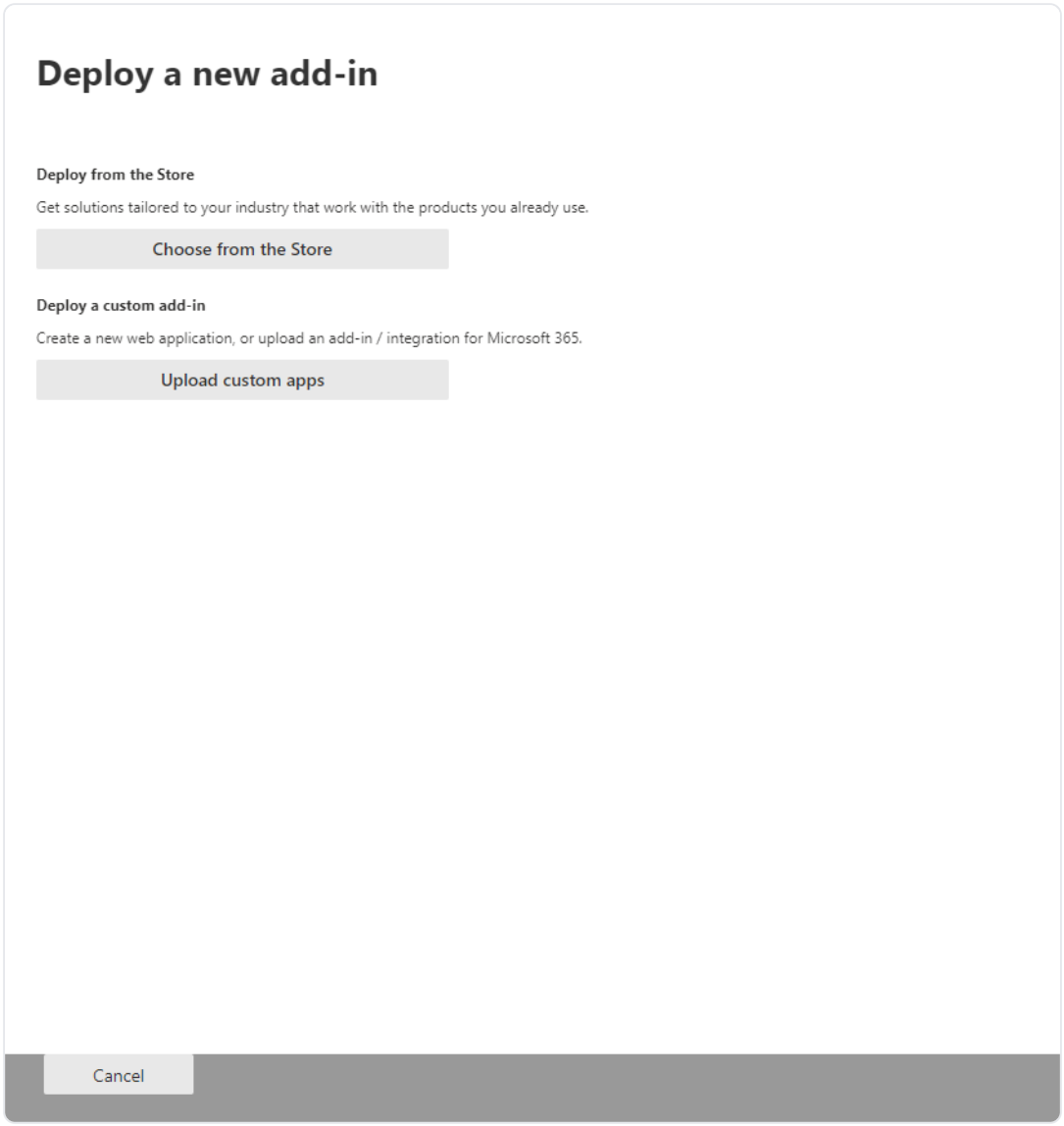
Walk-through

1. Sign in to admin.microsoft.com as a Global Admin.
2. From the left navigation, expand **… Show all**, then choose **Settings** → **Integrated apps**.
3. Click the **Add-ins** link near the top of the Integrated apps page, then **Deploy Add-in**.



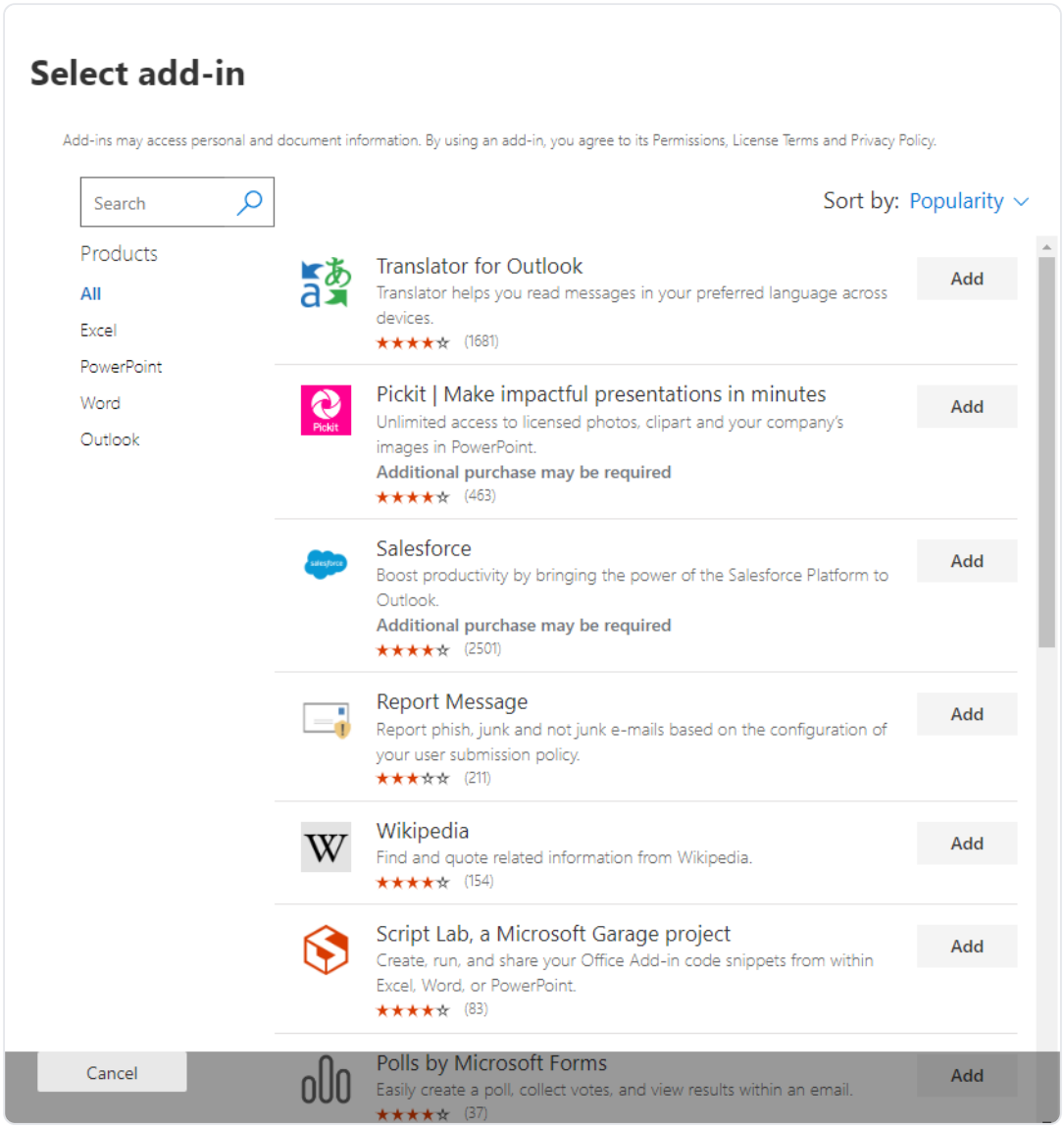
Microsoft 365 admin center showing the Deploy Add-in button at the top of the Integrated apps page

4. In the source picker, choose **Upload custom apps → Upload manifest file (.xml) from device** and pick `phishspot-outlook-manifest-v1.0.0.xml`.



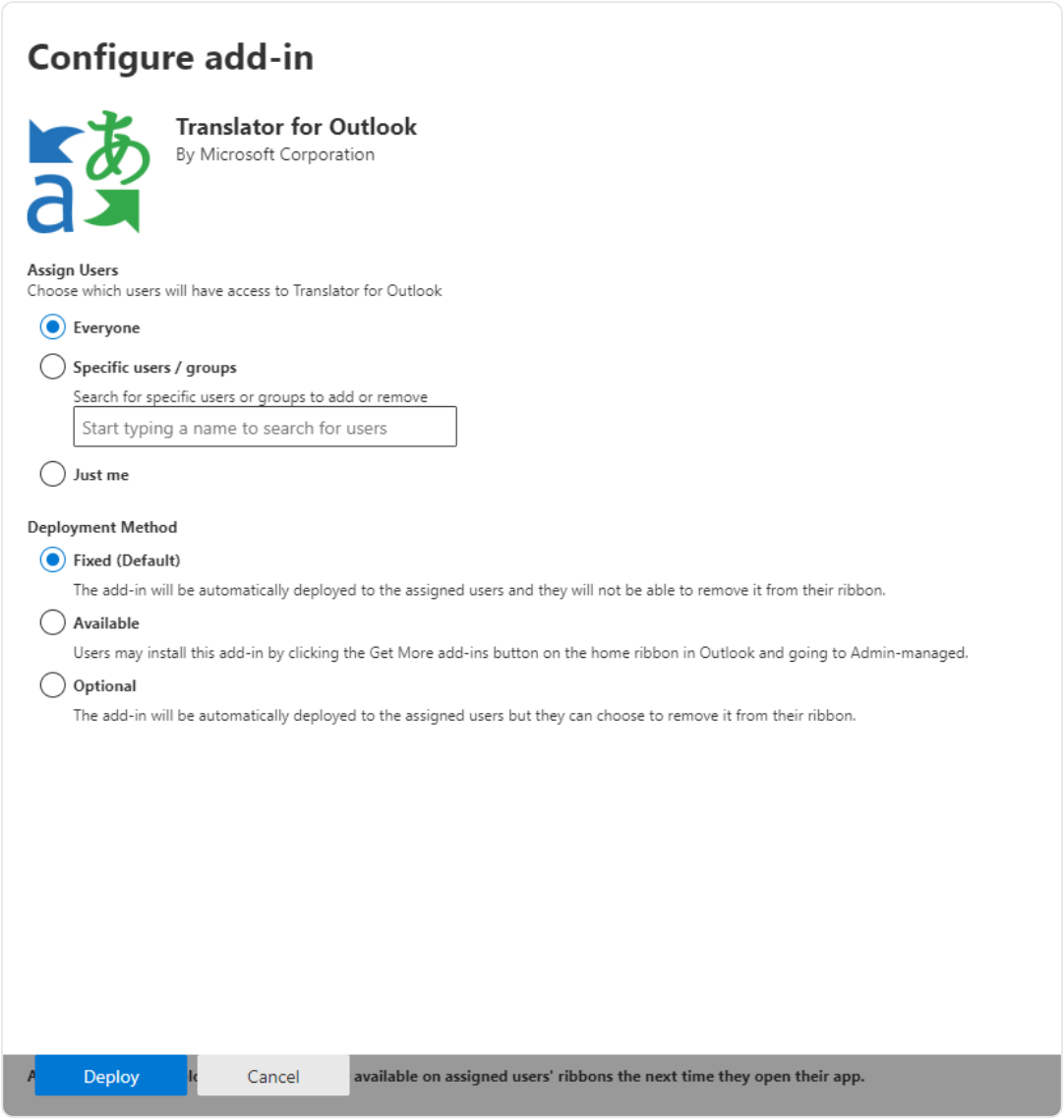
Deploy wizard source picker offering Microsoft Marketplace or uploading a custom add-in by file or URL

(The wizard also lists Microsoft Marketplace add-ins — those are unrelated, ignore them. PhishSpot is a line-of-business (LOB) add-in delivered by manifest file.)



Deploy wizard browsing Microsoft Marketplace add-ins by category

5. On the **Assign users** step, pick the scope and click **Deploy**.



Deploy wizard user/group assignment step with the Everyone, Specific users/groups, and Just me options

6. Review permissions on the next pane — PhishSpot requests only the **ReadItem** scope. It cannot send mail, modify mail, or read folders other than the currently open message. These permissions are declared in the manifest and never change across manifest updates.

7. Confirm. The wizard's final step prompts you to **announce the deployment** to users — see *Tips for getting users started* below.

Assignment scope: pick groups, not individuals

Scope	When to use
Everyone	"Use sparingly — only for add-ins that are truly universal." Reporting phishing is a good fit for *everyone* in most organisations, so this is usually correct.

Scope	When to use
Specific users / groups ⭐	**Recommended.** Assigning by group means new joiners get the add-in automatically when they're added to the group, and leavers lose it when removed. No admin action needed on either event. Assigning to *individual users* is fragile — every new hire requires a manual add.
Just me	Ideal for testing. After verifying the button works in your own mailbox, return to the deployed add-in and click **Change who has access to add-in** to widen the rollout.

Propagation timing

Microsoft's documented expectation is that **add-ins can take 24–72 hours to appear on the ribbon** after deployment, though most users see it within 1–6 hours. Users may need to relaunch Outlook (close every window, then reopen) before the button shows up. Don't escalate too eagerly — the propagation is normal.

21.4 Recommended rollout strategy

Microsoft's published guidance is to **roll out in waves**:

1. **Wave 1 — IT + stakeholders.** Deploy to your IT team and a handful of business stakeholders. Verify the **Report Phishing** button appears in their Outlook, that pairing works end-to-end, and that a reported test message lands in PhishSpot's **Reported Messages** list under the right account. Resolve any tenant-specific surprises here (proxy / firewall / Contact provisioning gaps).
2. **Wave 2 — a department or two.** Expand to one or two departments. Re-evaluate adoption and incident-response load. Tweak your user comms based on Wave 1 feedback.
3. **Wave 3 — full rollout.** Once Wave 2 looks healthy, switch the assignment to the org-wide group (or **Everyone**) and announce broadly.

For a tenant with under ~50 mailboxes you can collapse Waves 1 and 2 into a single pilot. For tenants over a few thousand mailboxes, add a fourth wave that splits Wave 3 by region or job function.

21.5 Tips for getting users started

Microsoft explicitly calls this out as good practice, and it materially boosts reporting rates:

- **Email everyone the day the add-in goes live.** Include a one-paragraph explanation of what the button does, a screenshot of the ribbon, and a single sentence on what *not* to do (e.g. "if in doubt, click Report — false reports are fine; clicking the link inside the email is not.").
- **Link to your help-desk runbook.** A short FAQ that covers: "I don't see the button yet" (24–72 h propagation), "It asks for a 6-digit code" (pair-once flow), "I got a thank-you message — what happens next?" (security team triage SLA).
- **Onboarding integration.** Add a step to your new-hire IT onboarding that confirms the user can see the button and has paired their device.

- **Reinforce on Phishing Awareness Month.** Bump the comms in October — most orgs see a spike in reports during that month.

21.6 Provision your contacts in PhishSpot

The add-in pairs a user to a single PhishSpot **Contact**. Make sure every user who'll use the add-in has a corresponding Contact in your PhishSpot account before they try to pair, otherwise pairing will fail with "We could not find an account for your sign-in."

You can bulk-create contacts from:

- A CSV import (see Contacts)
- Microsoft Entra (Azure AD) directory sync — automatic
- Manual creation

21.7 First-pair user journey

Each user pairs once per device. Their journey:

1. Outlook → click **Report Phishing** in any read message.
2. The taskpane shows a 6-digit code.
3. User opens `https://platform.phishspot.com/guest/activation/new`, signs in, pastes the code.
4. The taskpane flips to the **Paired** state automatically.

Each successful pair creates an **API token** in PhishSpot, scoped to `reported_messages:create` for one specific account. You can list and revoke these tokens from **Settings** → **API Tokens**.

21.8 Rolling updates

We release new versions of the JS bundle every few weeks. **You don't need to re-upload the manifest** for those releases — the version pointer at `https://platform.phishspot.com/api/v1/outlook/version` is the single source of truth, and every Outlook client picks up the new bundle on next open.

When the manifest itself changes (new permission, new button surface), you'll get a release note that says "manifest update required" and a new `phishspot-outlook-manifest-vX.Y.Z.xml`. Upload that the same way you uploaded v1.0.0 — M365 Admin Center recognises it as an upgrade of the existing app (same `Id` GUID). To force an update from the LOB add-in's pane, select the deployed add-in and click the **Update Button** at the bottom-right of its details panel; the change applies the next time each user launches Outlook.

21.9 Updates vs. blocked clients

The add-in's bootstrap checks the version endpoint on every open. Two outcomes:

- `latest > bundled` — soft banner shown to the user. They can still report.
- `min_supported > bundled` — hard block. Reporting is disabled until the manifest is re-uploaded.

We only bump `min_supported` when an old version is incompatible with a security or data-model change. This is rare; expect one or two events per year at most.

21.10 Decommissioning

To remove the add-in:

1. **M365 Admin Center → Integrated apps → PhishSpot Report Phishing → Remove**. This unlinks it from all user mailboxes within a few hours.
2. **PhishSpot → Settings → API Tokens** — revoke all tokens with source = `outlook_addin`. Users who somehow still have an installed copy lose their ability to submit reports.

21.11 Troubleshooting

Symptom	Likely cause	Fix
Button doesn't appear for any user	Propagation pending	Microsoft says 24–72 h is normal; force-restart Outlook to speed it up
Button appears, taskpane shows blank	Browser can't reach `platform.phishspot.com`	Check corporate proxy / firewall
Pairing always says "no account"	User has no Contact record in PhishSpot	Provision the Contact, retry
Reports fail with 403	Token's pinned account doesn't match	Unpair + re-pair the device
New Outlook for Windows: stuck on old version	M365 caches add-in metadata aggressively	Run `outlook.exe /resetnavpane` or clear the Wef folder

21.12 Compliance notes

- Reports are stored under your PhishSpot account, subject to your data residency settings.
- The bearer token never leaves the user's mailbox (stored in `Office.roamingSettings`).
- The add-in's source code lives in the same Git repo as the PhishSpot platform under `plugins/office/`. It's reviewed under the same change-control as the rest of the product.

Spam Filter Whitelist

PhishSpot phishing simulations look like real attacks — that's the point. Corporate spam filters (Microsoft 365, Google Workspace, Mimecast, Proofpoint, on-prem Postfix/SpamAssassin) will routinely block them unless they're explicitly whitelisted. This chapter explains how to give your mail-server admin a single URL they can plug into their filter, and have it stay current automatically.

22.1 Why a whitelist?

SPF, DKIM and DMARC tell receiving servers "this email is genuinely from the domain it claims to be from." For real phishing they often pass — that's exactly why phishing is hard. But the same checks pass for **our** simulations too, and modern spam filters use much more than SPF/DKIM: they look at content patterns, link reputation, sender behaviour history, and dozens of other signals. Several of those signals will (correctly!) classify a phishing simulation as suspicious.

The right fix is for the receiving filter to **bypass** spam scanning for traffic that comes from PhishSpot. That requires the admin to tell their filter:

- which **IP addresses** we send from,
- which **sending domains** we use, and
- (optionally) which exact **sender addresses** appear on the From: header.

PhishSpot generates that list per account and exposes it at a stable URL. Configure your filter to pull it on a schedule (or react to our webhook when it changes) and you're done.

22.2 Your whitelist URL

Open **Account settings** → **Integrations** → **Spam Filter Whitelist**. You'll see a panel with:

- Your **unique URL** containing a 64-character secret token,
- A **format picker** (txt / json / csv / md / Microsoft 365 / Google Workspace / Mimecast / Proofpoint / Postfix / SpamAssassin),
- A **status badge** showing when the URL was last fetched and from which IP,
- A **rotate button** to invalidate the current URL,
- A **disable toggle** that returns 410 Gone until re-enabled,
- A live **preview** of what's currently allowed,
- A **download history** of the last 50 fetches.

The URL is a single line you copy-paste into your spam filter or a small refresher script. There's no API token, no Authorization header — the secret is in the path, and HTTPS encrypts it in transit. We rate-limit each token to 60 requests per minute, log every fetch (IP + UA) in the download history, and run on HTTPS only.

> Treat the URL like a password. Anyone with it can read the full list of sending IPs and domains for your account. If you suspect a leak — **rotate** in the panel; the old URL keeps working for 24 hours so your spam filter has time to switch over.

22.3 Picking the right format

Format	When to use
`txt`	Plain text. Default. Easy to grep and pipe into scripts.
`json`	Structured payload. Best for custom integrations.
`csv`	Generic CSV — good as a fallback.
`md`	Human-readable Markdown — for documentation and review.
`microsoft365`	PowerShell snippet + Tenant Allow/Block List commands for Exchange Online.
`google-workspace`	CSV laid out for the Google Admin email allowlist import.
`mimecast`	CSV in Mimecast's Permitted Senders policy shape.
`proofpoint`	CSV in Proofpoint PPS Safelist shape.
`postfix`	`access` table snippet for on-prem Postfix.
`spamassassin`	`whitelist_from` lines for `local.cf`.

URL pattern: `https://platform.phishspot.com/api/v1/integrations/spam/<TOKEN>/<format>` — leave the format off and you get plain text.

22.4 Setup guides per provider

22.4.1 Microsoft 365 / Exchange Online

1. In the PhishSpot panel, pick **Microsoft 365 (PowerShell)** and copy the URL.
2. Save it to `phishspot-whitelist.ps1` on a workstation that has Exchange Online PowerShell installed.
3. Run `Connect-ExchangeOnline` (you'll need Exchange Administrator rights).
4. Execute the script. It does two things:
    - Adds each sending domain and address to the **Tenant Allow/Block List** with `New-TenantAllowBlockListItems`,
    - Merges the gateway IPs into the **Hosted Connection Filter Policy** via `Set-HostedConnectionFilterPolicy`.
5. Optionally create a **Mail Flow rule** with "skip spam filtering" for senders matching `@<your-phishspot-domain>`. Re-pull the URL weekly to keep the list current.

22.4.2 Google Workspace

1. Pick the **Google Workspace (CSV)** format and download the file.
2. In Google Admin Console, go to **Apps** → **Google Workspace** → **Gmail** → **Spam, phishing and malware**.
3. Open the **Email allowlist** for your top-level OU and paste the IP entries from the CSV (one per line).
4. Open **Inbound gateway** (also in Spam settings) and add the same IPs. This is what makes Gmail bypass spam scoring for those connections.
5. To allow by domain instead of IP, add the domain entries from the CSV to the **Approved senders** list (same section).

22.4.3 Mimecast

1. Pick **Mimecast (CSV)** in the panel.
2. In Mimecast Administration, go to **Gateway** → **Policies** → **Permitted Senders**.
3. Click **Import** and upload the CSV. Mimecast picks up sender IPs from the `Sender IP` column and senders/domains from the `Sender` column.
4. Either schedule a `curl` job (`curl -fSL '<URL>' > whitelist.csv` then re-import) or use the Mimecast API for automation.

22.4.4 Proofpoint Protection Server (PPS)

1. Pick **Proofpoint PPS (CSV)**.
2. Either upload via **System** → **User Management** → **Safelists** → **Import** in the PPS UI, or push via the PPS REST API (`/api/v1/safelist/import`).
3. PPS treats sender, domain and IP entries differently — the CSV's `type` column tells PPS which list to put each row in.

22.4.5 Postfix (on-prem)

1. Pick **Postfix access table**.
2. Save to `/etc/postfix/phishspot_whitelist`, then run `postmap /etc/postfix/phishspot_whitelist` to compile the lookup table.
3. Reference it from `main.cf`:

```
smtpd_sender_restrictions =
  check_sender_access hash:/etc/postfix/phishspot_whitelist,
  ...
```

4. Run `postfix reload`.
5. For IP-based bypass, copy the IPs into a separate CIDR file and add `check_client_access cidr:/etc/postfix/phishspot_ips` to `smtpd_client_restrictions`.

22.4.6 SpamAssassin

1. Pick **SpamAssassin local.cf**.
2. Append the snippet to `/etc/spamassassin/local.cf`.
3. Validate with `spamassassin -D --lint`.
4. Restart `spamd`.
5. The snippet uses `whitelist_from *@<domain>` and `trusted_networks <ips>` — the latter raises the trust score for relayed mail.

22.5 Auto-refresh via webhook

The whitelist changes when you (or PhishSpot) add a sending domain, when a campaign uses a new From: address, or when our infrastructure team rotates gateway IPs. To keep the customer side current automatically:

1. In **Account settings** → **Webhooks** → **Endpoints**, add a new endpoint pointing at a URL on your side.
2. Subscribe to the event type `spam_whitelist.updated`.
3. When the list changes we POST to that URL with a signed payload (HMAC-SHA256 in `X-Webhook-Signature` using the endpoint's signing secret). The payload includes the new snapshot digest and the full set of whitelist URLs across formats.
4. Your handler verifies the signature, then triggers your platform-specific import (the PowerShell job above, the Google Admin API call, Mimecast / Proofpoint API, etc.).

We retry failed deliveries 5 times with exponential backoff. After 5 consecutive failures we email account admins so the integration doesn't silently rot.

22.6 Stale-fetch alerts

We track when each URL was last fetched. If 24 hours pass without a successful fetch — meaning your spam filter has stopped pulling the list — we email **every admin** on the account. The most common causes:

- The cron / scheduled task that pulls the URL stopped.
- Your firewall is now blocking outbound HTTPS to `platform.phishspot.com`.
- The URL was rotated and the old one expired before anyone updated the filter.
- The integration was removed from the spam filter accidentally.

To silence the warning, re-trigger the fetch from your side (one `curl` is enough — we reset the counter on every successful request).

22.7 Best practices

- **Schedule the pull** at least daily, ideally hourly. The endpoint is cheap to hit.

- **Verify the snapshot digest** (`X-PhishSpot-Snapshot-Digest` header) — if it matches what you have, skip the re-import to avoid noise in your downstream system.
- **Rotate quarterly**. Even with no leak, regular rotation limits blast radius if a script ever logs the URL.
- **Monitor your side too**. Alert if the cron job hasn't run successfully in N hours. Don't rely only on our stale-warning email.
- **Use the webhook in addition to the cron**, not instead of it. The cron is the safety net; the webhook is the fast path.
- **Test your bypass** with PhishSpot's "Send test email" feature in a campaign before going live. If the test doesn't land in the recipient inbox, your bypass isn't working.

22.8 FAQ & troubleshooting

"Our phishing simulations still go to spam." Confirm the spam filter is actually pulling the URL: check the download history in the PhishSpot panel — does the IP and timestamp match your filter's egress IP and pull schedule? If yes, check the bypass rule is on the correct policy (often filters have separate inbound vs. transport policies). If no, the URL isn't reaching the filter.

"The CSV format my filter expects is different." Use the plain `csv` format as a template and transform it server-side. The `json` format is the most flexible source — easy to map to any target schema with `jq` or a 20-line script.

"Our webhook isn't receiving deliveries." Check the endpoint URL in **Webhooks** → **Endpoints** — make sure it's HTTPS, publicly reachable, and not behind an authentication wall. Open the endpoint detail page in PhishSpot to see the delivery log with response codes and response bodies. Verify the HMAC signature handling on your side matches `OpenSSL::HMAC.hexdigest("SHA256", signing_secret, raw_body)`.

"What if you change your gateway IPs?" You'll get a `spam_whitelist.updated` event the moment we make the change, and the URL response includes the new IPs immediately. If your filter has a fresh pull within the change window, you'll never notice.

"Can I have multiple URLs for different filters?" Not in the MVP — there's one active URL per account. If you need separate URLs (e.g., for a phased rollout), use the rotation flow: rotate, wait 24h, rotate again. Each rotation gives you a fresh URL with a 24-hour grace period.

Autopilots

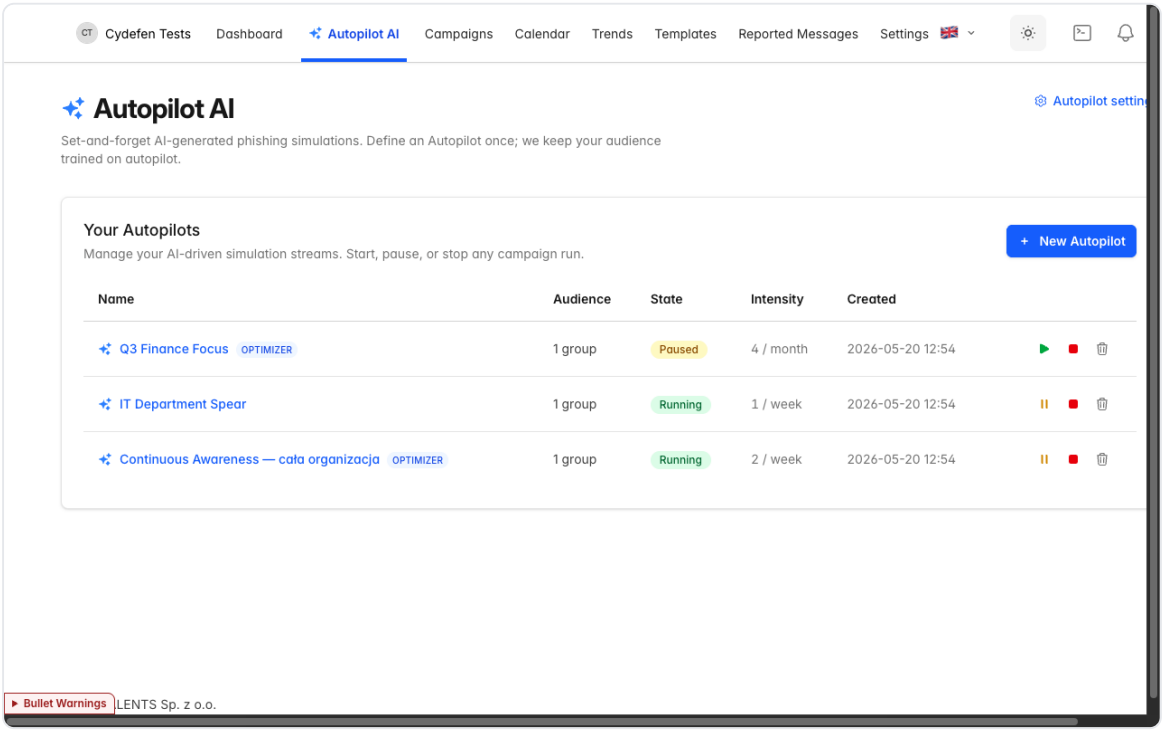
A campaign is a one-off send. An **autopilot** is the configuration that produces campaigns automatically, on a cadence you set, for as long as you want it to run. You define the audience, the intensity (how often), the post-click outcome and the targeting context (language, country, industry) — and PhishSpot launches simulations against the matching contacts until you pause or stop the autopilot.

The model is “set it once, then let it work.” If a new contact joins one of the autopilot’s groups (or arrives from a directory sync), they get picked up automatically on the next iteration. If you change the outcome or course mid-flight, the next launched campaign uses the new setting.

23.1 What an autopilot is — and isn’t

An autopilot is **not** a single long-running campaign. It’s a recipe. Each time the autopilot fires it creates a fresh `Campaign` record, picks a phishing template suited to the autopilot’s language and industry, snapshots recipients from the configured groups, and dispatches the send. Reports for those campaigns live in **Reports & Analytics** like any other.

Use autopilots when you want a continuous, low-touch awareness program. Use one-off campaigns (Chapter 4) when you want full control over timing, copy, and recipients for a single send.



Autopilots index showing three configured autopilots

23.2 Creating an autopilot

Open **Autopilots** in the left sidebar and click **New autopilot**. The form has two visible sections:

23.2.1 Name and audience

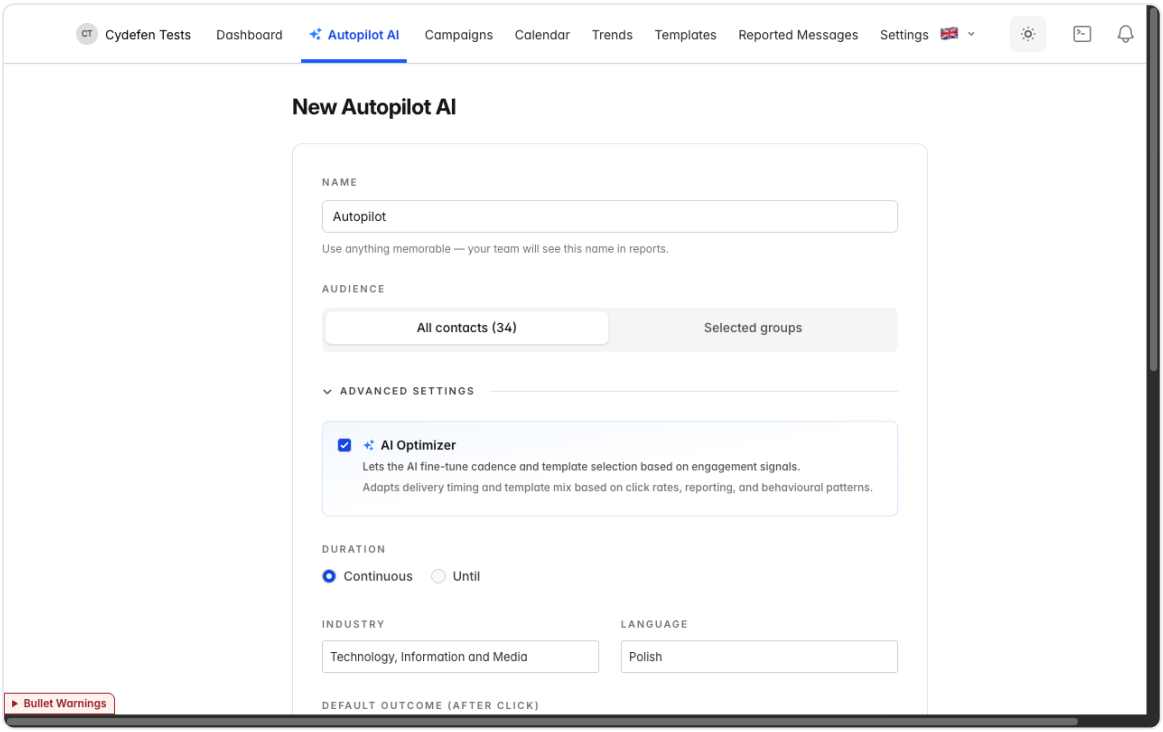
- **Name** — what you'll see in the autopilot list. Max 80 characters. Examples from a typical setup: "Continuous Awareness — cała organizacja", "IT Department Spear", "Q3 Finance Focus".
- **Audience** — pick **All contacts** to target every contact on the account, or **Selected groups** to scope the autopilot to one or more groups. When the autopilot fires, recipients are sampled from the audience at that moment — so groups that grow over time grow the autopilot's reach.

23.2.2 Advanced settings

This section is expanded by default when you're editing an existing autopilot. It contains:

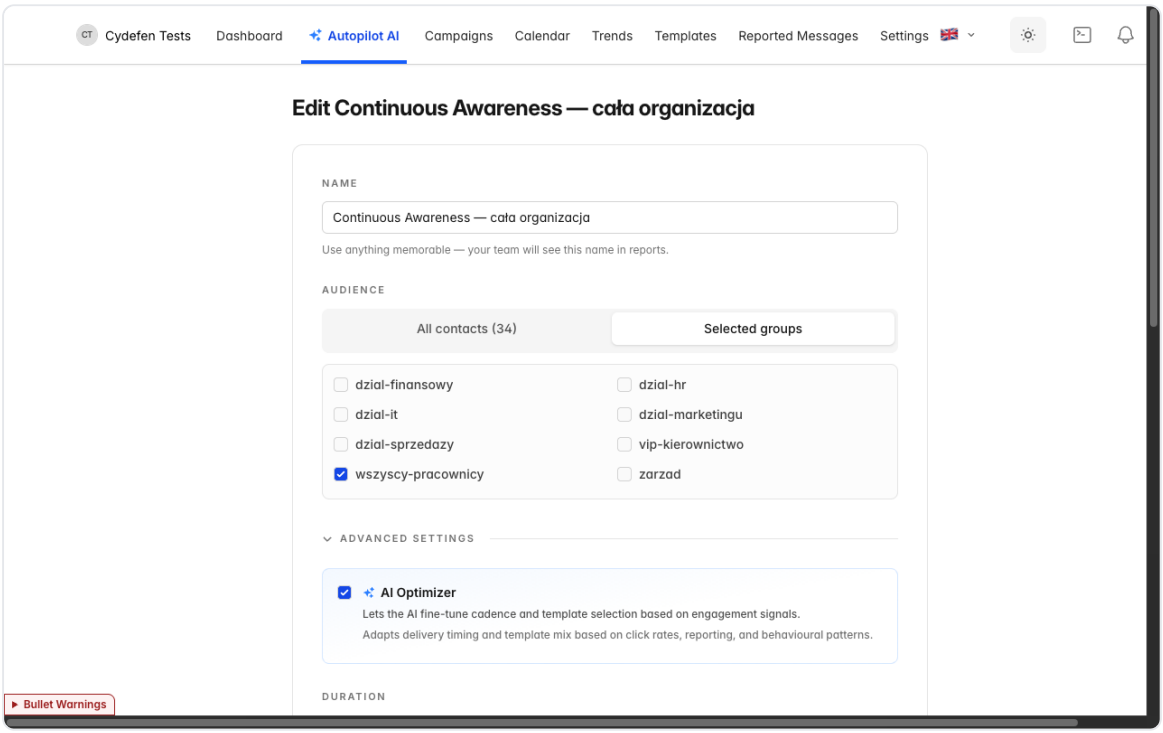
- **AI Optimizer** — when on, PhishSpot fine-tunes which templates are sent to whom based on past interactions. New autopilots default to ON.
- **Duration** — **Continuous** (runs until you stop it) or **Until** (stops automatically on the chosen day).
- **Industry** — the industry of the target organization (NAICS + LinkedIn taxonomy). Used to bias template selection toward themes that look plausible for that vertical. Leave blank to inherit from autopilot settings (§23.6).
- **Language** — the language the simulation copy will be authored in. Leave blank to inherit.
- **Default outcome (after click)** — what to show the recipient after they click the simulated phishing link:
  - **Do nothing** — no landing page; the click is just logged.
  - **Redirect to training course** — opens the course you select.
  - **Show awareness page (recommended)** — renders an in-context "this was a phishing simulation" page.
  - **Redirect to URL** — sends the user to an external URL of your choice.
- **Automatically include new members of groups and contacts** — when on, contacts added to the autopilot's groups after the autopilot starts will be included from the next iteration onward. Default ON.
- **Campaign intensity** — see §23.3.

Save and the autopilot is created in **Draft** state. Click **Start** to begin.



New autopilot form with the Advanced settings section expanded

The form below shows an existing autopilot in edit mode — every advanced setting is visible: AI Optimizer, duration, industry, language, outcome, auto-include new members, and intensity.



Editing a running autopilot — full settings panel

23.3 Intensity and the daily cap

Intensity is two values: a **count** and a **period** — `2 per week`, `1 per month`, `4 per year`, etc. Periods are **day**, **week**, **month**, **year**.

PhishSpot enforces a hard ceiling: **no single contact will be targeted by an autopilot more than twice per day**, regardless of intensity setting. The intensity field in the form refuses values that would breach this:

- `1/day` and `2/day` are allowed.
- `3/day` and above are rejected — the form shows an error.
- Weekly/monthly/yearly are converted internally to a per-day rate (`PERIOD_DAILY_RATE` of `1`, `7`, `30`, `365` respectively) and checked against the same cap.

The cap is per-autopilot. If a contact sits in multiple autopilots, each autopilot honours its own limit independently — keep that in mind when running parallel programs against overlapping groups.

23.4 Lifecycle states

Every autopilot is in exactly one state:

State	Meaning	Editable?
Draft	Created but not yet launched. No campaigns fired.	Yes
Running	Active. Campaigns fire on the configured cadence.	Yes
Paused	Temporarily halted. No new campaigns until resumed.	Yes
Stopped	Permanently terminated. **Read-only.** Remove the autopilot to start fresh.	No

Transitions are explicit buttons on the autopilot row:

- **Start** — `Draft` or `Paused` → `Running`.
- **Pause** — `Running` → `Paused`.
- **Stop** — any state → `Stopped`. Cannot be undone; you'll need to delete the autopilot and re-create it.

A **Stopped** autopilot is a tombstone — it retains its history (which campaigns it fired, when) but no fields can be changed. The intent is to give you an auditable trail of past programs without cluttering active ones.

23.5 The AI Optimizer

When enabled, the AI Optimizer adapts which templates the autopilot picks for each recipient based on past behaviour: people who consistently fall for invoice-themed lures see more of those (and the training that follows); people who never miss them get harder, less obvious variants. The optimizer is on by default for new autopilots and can be toggled per-autopilot in the **Advanced settings** section.

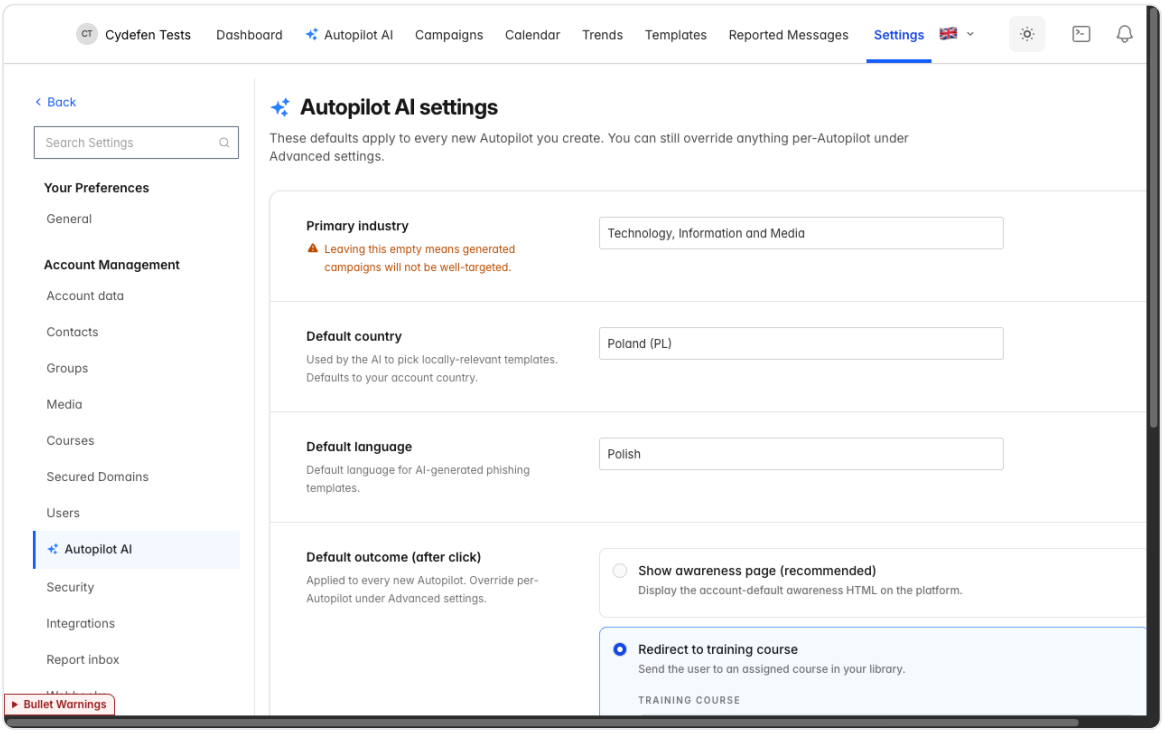
The optimizer's adaptive logic ships in phases. The toggle and the per-template scoring are live today; the full multi-armed-bandit selection layer rolls out during the implementation engagement and is included in the SaaS subscription — no separate configuration is required when it lands.

23.6 Default settings

Click the **gear icon** → **Autopilot settings** on the autopilots list to open account-level defaults. Settings here pre-fill the new-autopilot form so you don't repeat yourself across autopilots:

- **Primary industry** — your organization's industry. Templates pick it up automatically.
- **Default country** — used to bias template selection (sender names, brand spoofing targets).
- **Default language** — language of simulation copy.
- **Default outcome (after click)** — same four options as on the autopilot form, used as the starting value.
- **Default campaign intensity** — count + period used as the starting value.

Changing settings here does **not** retroactively edit existing autopilots — it only changes the defaults for future ones. Per-autopilot fields override these defaults when set.



Autopilot settings — account-level defaults

23.7 Real-world examples

Below are three autopilot configurations from a working setup, illustrating the spread of typical use cases.

Continuous Awareness — full organisation

A baseline program for everyone on the account.

- Audience: the “Wszyscy pracownicy” group (all employees synced from Entra AD).
- Intensity: 2/week. With the daily cap that’s still no more than 2 emails on any single day per contact — but spread thinly across the week.
- AI Optimizer: ON.
- Language: pl. Industry: Technology, Information and Media.
- Outcome: Redirect to course “Świadomość phishingowa 101”.
- Auto-include new members: ON — directory sync changes flow straight through.

IT Department Spear

Higher-difficulty simulations aimed at the IT team — the group most likely to be targeted by real attackers.

- Audience: just the “Dział IT” group.
- Intensity: 1/week — lower than the org-wide program because the templates used are harder and the cohort is small.
- AI Optimizer: OFF — the admin wants deterministic, manually-curated targeting for this group during initial calibration.
- Outcome: Redirect to URL — a custom internal security wiki page.

Q3 Finance Focus (paused)

A time-boxed campaign for the finance team.

- Audience: “Dział Finansowy”.
- Intensity: 4/month.
- State: Paused — kept around between quarters; resumed when the next quarterly push starts.

Each of these is created once and then left alone. The reporting per autopilot shows up under the matching campaigns in Reports & Analytics.

23.8 Cross-references

- Account-level defaults: see §23.6 above.
- The contacts and groups autopilots target: Chapter 5 Contacts and Chapter 6 Groups.
- The directory sync that grows the audience automatically: Chapter 25 Directory Sync.
- Reporting for autopilot-fired campaigns: Chapter 11 Reports & Analytics.
- Course used as the post-click outcome: Chapter 8 Courses.

Sign-in with Microsoft 365

PhishSpot integrates with Microsoft 365 (Entra ID) for end-user sign-in. Employees imported from your directory log in with their corporate Microsoft account — no separate password to manage, no separate identity to onboard. The first time they sign in, PhishSpot links their authenticated user to the existing contact record imported from Entra, and lands them on their personal training portal at `/guest/dashboard`.

This chapter covers the sign-in flow as it appears to employees, the personal portal they reach after signing in, and the dual-role picker shown to admins who are also end users.

24.1 Why Microsoft 365 SSO?

Three reasons:

- **No extra password to lose.** Employees use the same Entra account they already use for Outlook, Teams, SharePoint and the rest of Microsoft 365. There's nothing new to forget.
- **Automatic onboarding.** Once Entra directory sync (Chapter 25) imports an employee, their first Microsoft sign-in turns the contact into a fully-formed user account — no admin intervention required.
- **Inherited security posture.** Conditional access, MFA, device compliance — every Entra policy you have already applies. PhishSpot doesn't run its own MFA in front of an Entra-authenticated session because Microsoft already enforces it upstream.

24.2 Admin setup

The platform-wide OAuth app is already registered in PhishSpot's tenant. To enable end-user sign-in from your tenant you only need to grant admin consent for the PhishSpot enterprise application and (optionally) connect directory sync:

1. Open **Account settings** → **Integrations** → **Microsoft 365**.
2. Click **Connect Microsoft 365**. You'll be redirected to the Microsoft admin consent screen.
3. Grant the requested scopes (`User.Read.All`, `Group.Read.All`, `Directory.Read.All` for sync; `openid`, `profile`, `email` for sign-in).
4. Microsoft redirects you back. PhishSpot stores your tenant ID and a tenant-scoped app token.
5. (Optional but recommended) configure sync schedule — see Chapter 25 §25.3.

After admin consent, any user in your tenant whose email matches an imported Contact on your account can sign in. Users who don't match a Contact (or whose tenant ID doesn't match a configured integration) hit a friendly "no access" page — they cannot create accounts on the fly.

24.3 End-user sign-in flow

What an employee sees:

1. They open `https://platform.phishspot.com/users/sign_in`.
2. Below the email/password form, separated by an “OR” divider, they see a **Continue with Microsoft** button (with the Microsoft logo). Clicking it redirects to the Microsoft sign-in screen.
3. Microsoft authenticates the user — including MFA if your tenant requires it. The user grants the PhishSpot app permission on first sign-in (a one-time consent per user, unless you’ve granted admin consent on their behalf, which suppresses the prompt entirely).
4. Microsoft redirects back to PhishSpot. Behind the scenes:
    - If a User row exists with the same email, it’s reused.
    - If not, a new User row is created with that email, marked confirmed.
    - PhishSpot caches the user’s Entra **tenant ID** for fast lookups.
    - Any unlinked Contact row matching the email is bridged to this user — they become “your contact, your user”.
5. The user lands on `/guest/dashboard`. No password was set. No invite email was sent. They’re in.

The first sign-in completes in one round-trip; subsequent sign-ins are even faster — Microsoft remembers consent, and the bridging step is a no-op after the first time.

24.4 The Guest Dashboard

`/guest/dashboard` is the employee-facing portal. It shows everything the employee is expected to engage with personally — and nothing else. They do not see other people’s results, the campaign list, account settings or any admin pages.

The dashboard has three sections:

24.4.1 Your training

The list of training assignments the employee has — typically one entry per phishing campaign that the employee clicked into and that has a course attached as the post-click outcome. Each row shows:

- The course name (e.g., “Świadomość phishingowa 101”).
- Completion status: not started / in progress (with % progress) / completed.
- A button to open the course player in the same tab.

Training assignments are derived from `Deliverable` records — when a contact reaches the `clicked` or later state on a campaign with a course, an obligation appears here.

24.4.2 Email history

The last 50 phishing simulation emails the user has interacted with. For each, the dashboard shows:

- The campaign name (the company-facing label, not the simulation-from address).
- Which actions the user took (opened / clicked / submitted form / reported).

- A timestamp.

This is the personal-history mirror of what admins see on the campaign dashboard. It's deliberately short — long retention is an admin concern, not an end-user one.

24.4.3 Reported emails

If the user has reported phishing through the Outlook add-in (Chapter 20), each report appears here with subject, sender and the time it was reported. The section also displays the per-account **reporting inbox address** — useful for users on devices without the Outlook add-in installed, who can forward the suspicious mail manually instead.

24.5 The dual-role picker

Some users are both admins and employees: a security manager who runs phishing campaigns is also a phishing target themselves. PhishSpot handles this with an explicit picker.

When a user with both an `account_user` (admin role on at least one account) and a `contact_membership` (contact row on at least one account) signs in, the resolver routes them to `/guest/role` instead of straight to a dashboard. The picker shows two large card buttons:

- 🛡️ **Admin panel** — opens the account-scoped admin UI (`/accounts/:account_id`).
- 🎓 **Training portal** — opens `/guest/dashboard`.

The user picks once per session. The picker is also reachable from the user menu so admins can switch contexts mid-session.

A user with only an `account_user` (pure admin) skips the picker and lands directly on the admin dashboard. A user with only a `contact_membership` (pure employee) skips the picker and lands directly on `/guest/dashboard`. The picker only appears when both apply.

24.6 Security model

PhishSpot delegates authentication to Microsoft for SSO sessions. That means:

- **MFA is enforced upstream.** If your tenant requires MFA, every PhishSpot sign-in goes through it. If you don't require MFA, PhishSpot does not silently re-impose it on the SSO path.
- **Conditional access applies.** Tenant policies (device compliance, geographic restrictions, app protection) gate the PhishSpot sign-in like any other Entra app.
- **Tenant scoping (optional).** PhishSpot can be configured to reject sign-in attempts whose Entra tenant ID doesn't match any connected `AccountOauthIntegration` on the platform. This is recommended for tenants that don't want any random Microsoft user to attempt sign-in.
- **Local 2FA still available for non-SSO accounts.** Admins who don't sign in via Microsoft (e.g., service accounts) can still enable TOTP-based 2FA — see Chapter 15 User Profile & Preferences.

24.7 Troubleshooting

The Microsoft button takes me to "no access". Either no Contact row matches your email on any account, or the tenant ID isn't recognized. Ask your admin to confirm: (1) directory sync ran and your account was imported; (2) the integration is connected with admin consent (Chapter 25 §25.2).

I clicked Continue with Microsoft but Microsoft never asked me to consent. Admin consent is already granted in your tenant — that's expected and faster.

I'm an admin and I keep landing on the role picker. That's expected: you're both an admin and a contact on the same account. Pick the role you want; the choice lasts for the current sign-in session.

I expected to see other employees' results. You won't — the Guest Dashboard only shows your own data. Aggregate dashboards are admin-only and live under the account admin UI.

24.8 Cross-references

- The directory sync that creates Contact rows from Entra: Chapter 25 Directory Sync.
- The Outlook add-in that feeds the "Reported emails" section: Chapter 20 Outlook Add-in.
- The campaign reports that admins use to monitor what employees see on this portal: Chapter 11 Reports & Analytics.
- Local TOTP-based 2FA for non-SSO accounts: Chapter 15 User Profile & Preferences.

Directory Sync with Entra AD

When a company runs phishing simulations against 50, 500 or 5,000 employees, keeping the contact list current by hand isn't viable. PhishSpot connects to Microsoft Entra ID (formerly Azure AD) and pulls users and groups directly from your directory. New hires show up, leavers get disabled, group membership reflects today's org chart — without anyone editing a spreadsheet.

This chapter covers the integration setup, the sync schedule, what gets imported and how, manual syncs, and the history log you check when something looks off.

> Before you connect Entra ID, read Chapter 28 — Entra ID: tradeoffs to consider. For most organisations, PhishSpot recommends manual CSV imports over directory sync. This chapter is the technical reference for when you've decided to connect anyway.

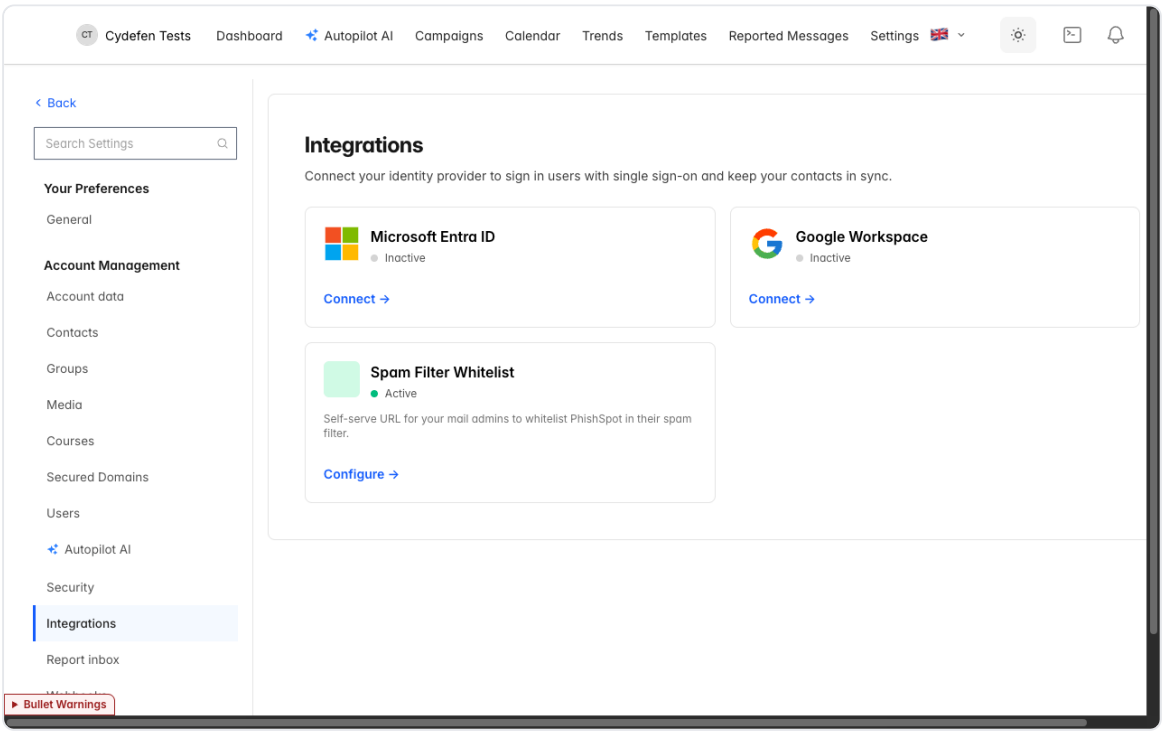
25.1 Why directory sync?

The alternative is CSV imports (Chapter 5 Contacts covers the manual paths). Those are fine for proof-of-concept or one-shot pilots, but for production they get stale fast. Directory sync gives you:

- **Authoritative source.** Your IT team already maintains Entra. PhishSpot inherits that work — no parallel list to keep in sync.
- **Automatic onboarding.** A new hire whose Entra account is created today shows up in PhishSpot tomorrow morning (or sooner, depending on schedule), and immediately becomes a valid target for autopilot campaigns.
- **Leavers handled cleanly.** When IT disables an Entra account, the matching PhishSpot Contact is marked `disabled` — it stays in the database for historical reporting, but autopilots stop targeting it.
- **Groups follow.** Membership in "Engineering", "Finance", "Zarząd" — whatever you've defined in Entra — is mirrored into PhishSpot Groups, so autopilot audiences track org structure automatically.

If you also want users to sign in to their personal training portal with Microsoft, the same integration backs that flow — see Chapter 24 SSO with Microsoft 365.

25.2 Connecting Entra



Integrations index showing the Microsoft Entra ID card alongside Google Workspace and the Spam Filter Whitelist

1. Open **Account settings** → **Integrations**. You'll see a grid of integration cards. Find the **Microsoft Entra ID** card; it shows a gray status dot ("inactive") until connected.
2. Click **Connect to Microsoft**. PhishSpot prompts you for your Entra **tenant ID** (a GUID, e.g., `1f3a8d2e-...`). You can find this in the Entra admin centre under "Properties".
3. Submit. PhishSpot generates an HMAC-signed state token and redirects you to the Microsoft admin-consent screen at `login.microsoftonline.com/<tenant>/v2.0/adminconsent`. **You must be signed in as a Global Administrator** in the target tenant — admin consent grants the PhishSpot enterprise application directory-read scopes on behalf of all users.
4. The consent screen lists the requested scopes:
   - `User.Read.All` — read user profiles (for the contact list).
   - `Group.Read.All` — read group definitions.
   - `Directory.Read.All` — read group memberships.
   - `openid` / `profile` / `email` — needed for the SSO sign-in flow.
5. Grant. Microsoft redirects back to PhishSpot's callback endpoint with `admin_consent=True&tenant=<ID>&state=<HMAC>`. PhishSpot verifies the HMAC, stores the tenant-scoped app token and turns the integration status to **active**.

You're now connected. The Microsoft card shows an emerald status dot, the tenant ID in monospace, and the consent timestamp.

25.3 Sync schedule

Click **Manage** on the active Microsoft card to open the integration settings. Two checkboxes and one dropdown:

- **Sync users to contacts** — on by default. Pulls every Entra user (excluding guests and disabled accounts) and upserts them as PhishSpot `Contact` rows.
- **Sync groups** — on by default. Pulls every Entra group (security and Microsoft 365 groups) and upserts them as PhishSpot `Group` rows. Memberships are reconciled in the same pass.
- **Schedule** — one of:
  - **Off** — no automatic sync. You can still trigger sync manually (§25.5).
  - **Hourly** — runs every hour on the hour.
  - **Daily** — runs once a day at 02:00 UTC. **Default for production tenants.**
  - **Weekly** — runs once a week, on Mondays at 02:00 UTC.

Save settings. The platform's scheduler (`ScheduledDirectorySyncsJob`) fan-outs at each interval, queuing one `DirectorySyncJob` per active integration matching that schedule.

The first sync after a fresh connection is usually the largest — it imports everything. Subsequent syncs only touch what changed (a handful of users updated, one group created, one membership removed), so they're fast — typically under a minute even for thousands of users.

25.4 What gets imported

For each Entra **user** the importer creates or updates a `Contact` row keyed by the Entra object ID (`oid`). The mapping is:

PhishSpot Contact field	Entra source
`email`	`userPrincipalName` (falls back to `mail`)
`first_name`, `last_name`	`givenName`, `surname`
`title`	`jobTitle`
`department`	`department`
`location`	`officeLocation` (city + country fallback)
`telephone`	`mobilePhone` (falls back to `businessPhones`)
`external_id`	`id` (the Entra OID)
`external_state`	`active` if `accountEnabled=true`, else `disabled`
`source`	always `:entra`
`synced_at`	sync timestamp

For each Entra **group** the importer creates a `Group` row keyed by group OID:

PhishSpot Group field	Entra source
`name`	slugified `displayName` (e.g., "Dział IT" → `dzial-it`)
`display_name`	`displayName` (preserves casing and Polish characters)
`external_id`	group `id`
`source`	`:entra`

Membership is reconciled per sync: PhishSpot lists all current members of each Entra group, removes any local `ContactGroup` rows whose contacts are no longer in the Entra group, and creates new ones for additions.

Important: Contacts whose Entra account is **deleted** (not just disabled) are not removed from PhishSpot — they're marked `external_state: disabled` so historical reports remain intact. To purge a contact entirely, delete it from the PhishSpot UI manually.

25.5 Manual sync ("Sync now")

The integration management page has a **Sync now** button. Click it to enqueue a `DirectorySyncJob` immediately, regardless of the configured schedule. Use this for:

- **Initial connection** — most admins click Sync now once right after consent so the first import doesn't wait for tomorrow's 02:00 UTC cron.
- **Onboarding push** — after IT marks 20 new hires in Entra, you want them in PhishSpot before your next scheduled run.
- **Troubleshooting** — to retry after a transient Microsoft Graph error visible in the activity log.

A manual sync writes a `DirectorySyncLog` entry with `trigger: manual` — useful for distinguishing intentional admin actions from cron-driven runs when auditing.

25.6 Sync history

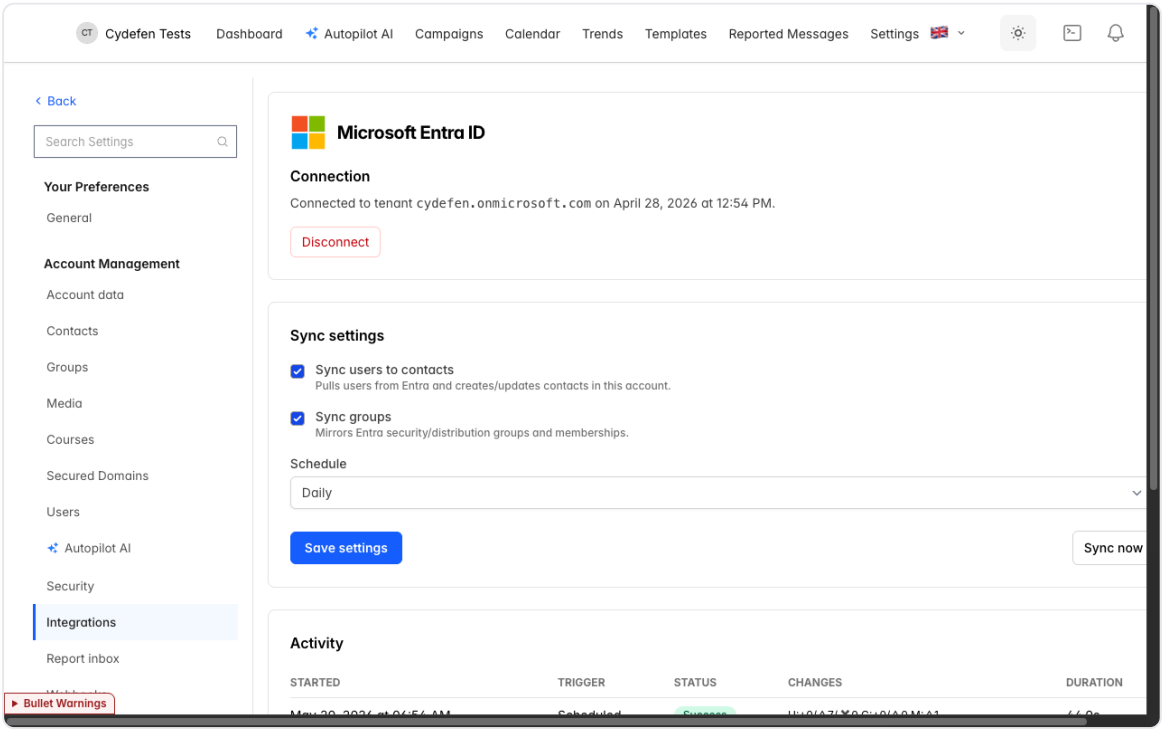
Below the settings, the activity table shows the last 50 sync runs. Columns:

Column	What it shows
Started	When the run kicked off
Trigger	`Manual` (admin clicked Sync now), `Scheduled` (cron), or `OAuth callback` (auto-sync right after admin consent)
Status	`Running`, `Success`, `Failed`, or `Partial` (with color-coded badge)
Changes	Users created / updated / disabled, groups created / updated, memberships changed
Duration	Wall-clock time for the run

A typical activity table for a small organisation looks like (real example from a working setup):

Started	Trigger	Status	Changes	Duration
6 hours ago	Scheduled	Success	0c · 7u · 0d / 0c · 0u / 1m	22 s
2 days ago	Scheduled	Success	1c · 2u · 1d / 0c · 0u / 2m	19 s
7 days ago	Manual	Success	0c · 4u · 0d / 0c · 0u / 0m	14 s
8 days ago	Scheduled	Failed	—	1 s
10 days ago	Scheduled	Success	1c · 3u · 0d / 0c · 0u / 1m	16 s

Failed runs expand to show the Microsoft Graph error message — most are transient rate-limit responses (HTTP 429) that the next scheduled run absorbs cleanly.



Entra integration management page — sync settings + activity history

25.7 Troubleshooting

Connect button takes me to a "consent denied" screen. You signed in as a non-admin user in your tenant. Sign out of all Microsoft accounts, sign in fresh as a Global Administrator, and retry.

Sync runs but no contacts appear. Open the activity log. If status is **Success** and changes are all zero, your Entra tenant has no users matching the filter (guests and disabled accounts are skipped). Verify in Entra that the users you expect have `accountEnabled=true`.

“Tenant mismatch” on callback. The Entra tenant ID you entered doesn’t match the tenant whose admin granted consent. Disconnect and reconnect with the correct tenant ID.

Partial syncs. If a sync finishes with status **Partial**, some upserts succeeded and others failed — usually because one user record violated a uniqueness constraint (e.g., two Entra users with the same email). Check the activity log entry; the error message includes the affected emails.

Schedule is “Off” but I expected Daily. Schedule defaults to the value you saved last — there’s no platform-wide default. New connections are created with `Off` so you can review settings before automation starts.

25.8 Cross-references

- The Microsoft SSO sign-in that uses the same integration: Chapter 24 SSO with Microsoft 365.
- The Contacts list that imported contacts join: Chapter 5 Contacts.
- The Groups feature mirroring Entra groups: Chapter 6 Groups.
- The autopilots that automatically include newly-synced contacts via the **Auto-include new members** flag: Chapter 23 Autopilots.
- The Webhooks that can notify external systems when contacts or groups change: Chapter 26 Webhooks.

Webhooks

The PhishSpot REST API (Chapter 27) lets you pull data on demand. Webhooks flip the direction: instead of you polling us, we POST to your URL the moment something happens. Wire a webhook into your SIEM and an `opened` campaign event becomes a security alert seconds after the user clicks. Wire one into your LMS and a finished course updates the learner record without you running a nightly sync.

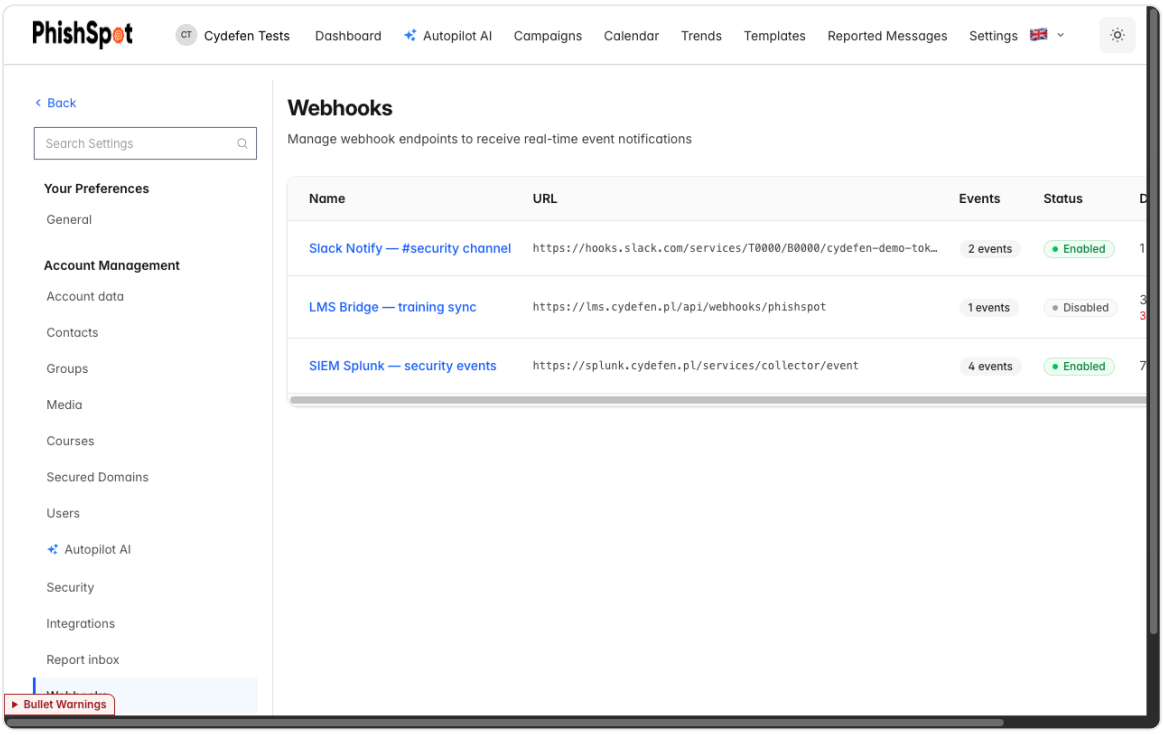
This chapter covers what events are available, how to register an endpoint, what arrives on the wire, and how retries and signing work.

26.1 Why webhooks vs polling

Polling means asking “anything new?” on a timer and ignoring the answer most of the time. It wastes API calls, has built-in latency (you find out about events on your next poll, not when they happen), and gets clunky at scale (longer polls miss events, shorter polls hammer the API).

Webhooks invert it. You register a URL once; we deliver each event there exactly when it occurs. Same data, lower latency, fewer requests. The downside: you need a reachable HTTPS endpoint to receive deliveries — but for an integration target (SIEM, SOAR, LMS, chat bot, internal tool) that’s usually trivial.

26.2 Creating an endpoint



Webhooks endpoints index showing three configured webhooks

1. Open **Account settings** → **Webhooks**. The endpoints page lists existing webhooks with columns for **Name**, **URL**, **Events** (count of subscribed types), **Status** (Enabled / Disabled), **Deliveries** (total + failed count), and **Actions**.
2. Click **New webhook**. The form has four fields:
    - **Name** — a friendly label. Examples from a working setup: “SIEM Splunk — security events”, “LMS Bridge — training sync”, “Slack Notify — #security channel”.
    - **Webhook URL** — where deliveries POST. Must be HTTPS. The platform rejects URLs that resolve to localhost, link-local addresses (169.254/16), or any RFC1918 private range (10/8, 172.16/12, 192.168/16). It also blocks `*.phishspot.com`. The goal is to keep the webhook system from being used as an internal-network probe.
    - **Subscribe to Events** — checkboxes for each event type (see §26.3). Pick at least one.
    - **Enable webhook endpoint** — toggle. Off means we keep the record but don’t deliver anything.
3. Save. PhishSpot generates a **signing secret** (64-char hex from `SecureRandom.hex(32)`) and displays it in full on the endpoint detail page with a copy-to-clipboard button. Store it somewhere safe; we never re-display it elsewhere.

The endpoint is live immediately. Any event you subscribed to that fires from now on will be POSTed to your URL.

26.3 Available event types

Nine event types are available today, grouped by subject:

Event type	When it fires
`campaign.created`	A new campaign is created (manual or from an autopilot iteration).
`campaign.updated`	A campaign’s state, recipients or content changes.
`campaign.deleted`	A campaign is deleted.
`contact.created`	A contact is added (CSV, manual, or directory sync).
`contact.updated`	A contact’s email, department, title, group membership or external state changes.
`contact.deleted`	A contact is removed from the account.
`deliverable.created`	A campaign send produces a deliverable row (one per recipient).
`deliverable.updated`	A recipient’s state changes (sent → opened → clicked → submitted → educated, or bounced).
`spam_whitelist.updated`	The sending-IP / sending-domain list for the account changes — see Chapter 22 §22.5.

An endpoint can subscribe to any combination. A typical SIEM subscribes to `contact.*` and `deliverable.*` so it sees both who's targeted and how they react. A typical LMS bridge subscribes only to `deliverable.updated` because it only cares when someone progresses through a training assignment.

26.4 The delivery: payload + signature

Each delivery is an HTTP POST with a JSON body. The body is the **event** — the same record you can fetch via the API. Shape:

```
{
  "id": "550e8400-e29b-41d4-a716-446655440000",
  "type": "contact.created",
  "created_at": "2026-05-20T14:22:33.000Z",
  "api_version": 1,
  "data": {
    "id": 42,
    "email": "anna.kowalska@cydefen.pl"
  }
}
```

- `id` — UUID identifying this event; idempotency key. Replays of the same event use the same `id`.
- `type` — the event type name (one of the nine in §26.3).
- `created_at` — ISO-8601 UTC timestamp of when the event occurred.
- `api_version` — the integer schema version. `1` for the format above. Future-breaking shape changes will bump this and we'll notify you in advance.
- `data` — subject-specific fields. For now `data` includes the subject's `id` and key identifying attributes; expand the payload by fetching the full record via the REST API using the included `id`.

Signing. Every POST carries an `X-Webhook-Signature` header containing the HMAC-SHA256 of the JSON body, computed with the endpoint's signing secret. Verification on your side:

```ruby
expected = OpenSSL::HMAC.hexdigest("SHA256", signing_secret, request.raw_post)
signature = request.headers["X-Webhook-Signature"]
ActiveSupport::SecurityUtils.secure_compare(expected, signature) or render status: 401
```

```python
import hmac, hashlib
expected = hmac.new(secret.encode(), request.body, hashlib.sha256).hexdigest()
if not hmac.compare_digest(expected, request.headers["X-Webhook-Signature"]):
    abort(401)
```

Use the secret-protected POSTs only — never trust an unsigned request claiming to be from PhishSpot.

26.5 Retries

If your endpoint responds with anything other than HTTP `2xx`, the delivery is queued for retry. Retries follow a fixed schedule:

Attempt	Delay from previous
1	(immediate)
2	+15 seconds
3	+1 minute
4	+5 minutes
5	+15 minutes
6	+1 hour

After **5 retries** (6 total attempts) the delivery is marked **Failed** and not retried again. The endpoint's `consecutive_failures` counter increments on each fully-failed delivery. After it crosses 5 consecutive failures the account admins are emailed once (with a 7-day cooldown to avoid spamming when an endpoint is misconfigured long-term). The endpoint itself stays enabled — we don't auto-disable it, because most outages are transient and self-resolving deliveries should be allowed to succeed.

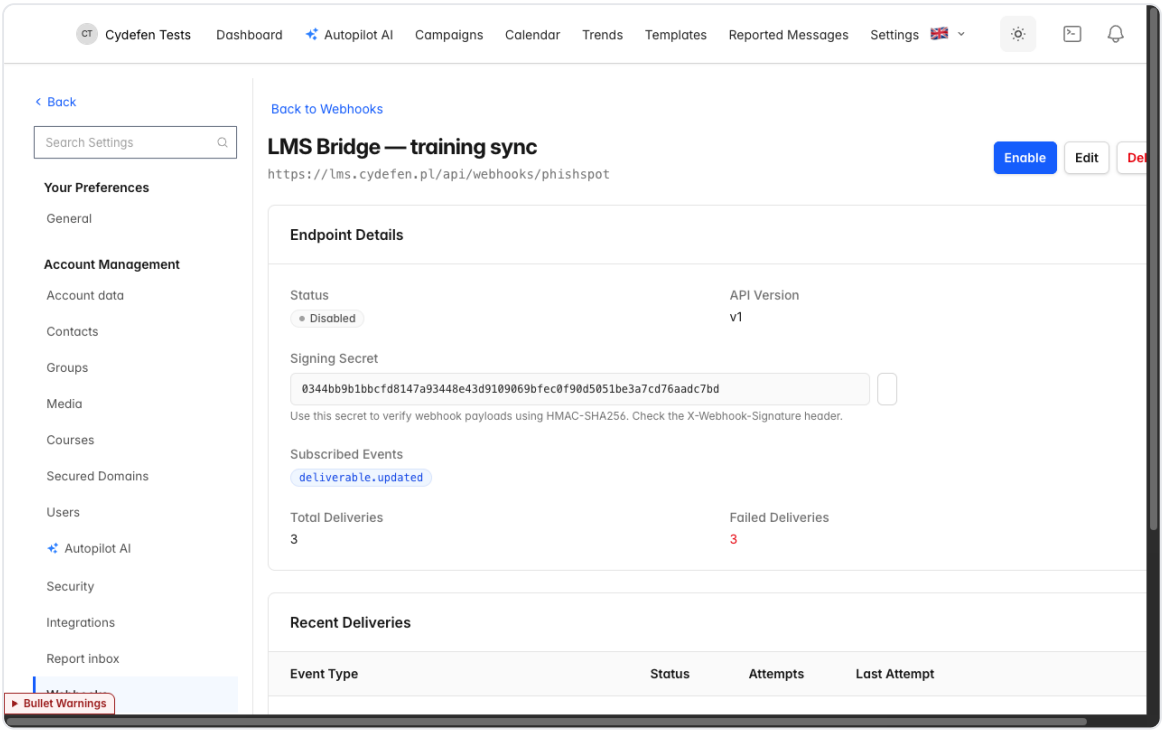
If you want to retry a single failed delivery manually after fixing your side, open the delivery detail page (§26.6) and click **Retry**. That creates a fresh delivery for the same event, with attempt counters reset.

26.6 Delivery history

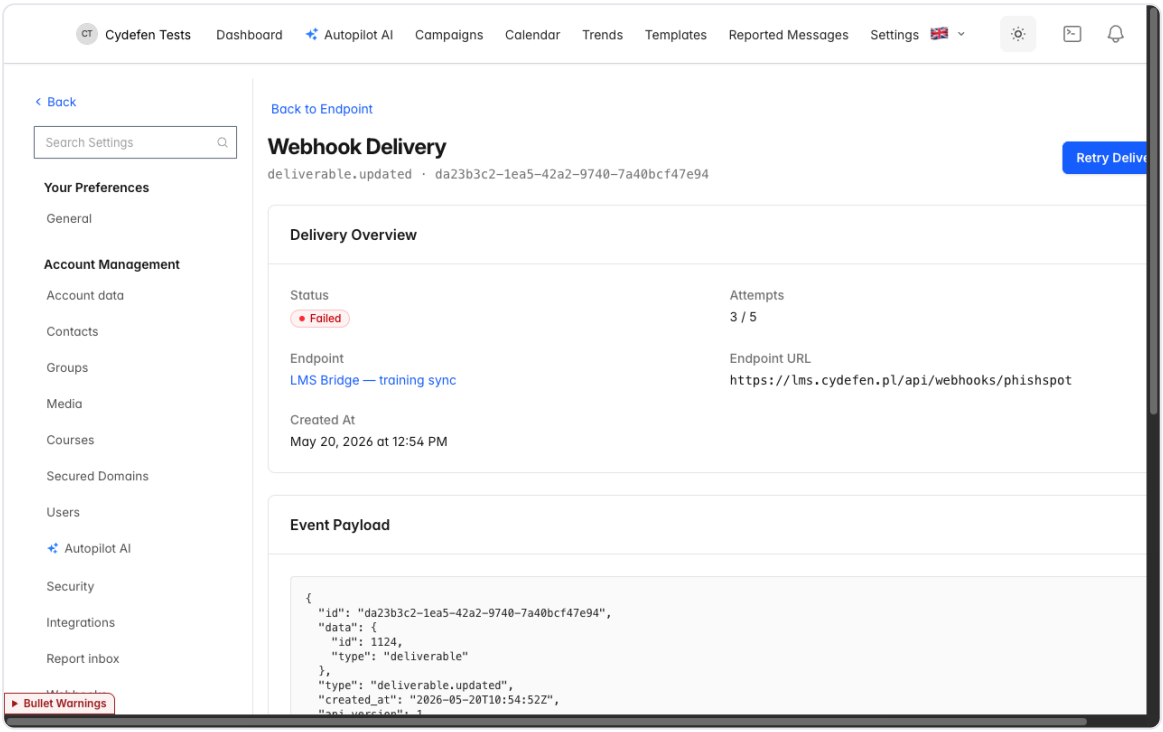
The endpoint detail page (click any endpoint name in the index) shows everything we know about that endpoint:

- **Endpoint Details** — name, URL, API version, signing secret (with copy button), subscribed events.
- **Status** — Enabled / Disabled toggle.
- **Recent Deliveries** — table of the last 50 deliveries with columns:
  - **Event Type** (e.g., `contact.created`)
  - **Status** — Pending / Delivering / Delivered / Failed
  - **Attempts** — current attempt count / max (e.g., `1 / 5`, `5 / 5`)
  - **Last Attempt** — timestamp
  - **Actions** — View Details, Retry Delivery

Click View Details on any delivery row to open the delivery detail page. It shows the full payload (pretty-printed JSON), the destination URL we POSTed to, and a per-attempt log: attempt number, HTTP status code, response duration in ms, and response body (or the error message if the request didn't complete). This is your primary debug surface when an integration breaks.



Webhook endpoint detail — signing secret, subscribed events, recent deliveries with attempt counts



Webhook delivery detail — event payload + per-attempt HTTP log

26.7 Operational guidance

- **Acknowledge fast.** Your handler should accept the delivery (return 2xx) and process asynchronously. We give up on slow responders — and slow responders cascade into retries, which cascade into “consecutive failures” emails.

- **Handle duplicates.** Network issues can cause the same event to arrive twice. Dedupe on the `id` field — it's stable across retries.
- **Validate the signature.** Don't act on a webhook whose signature doesn't match. The secret-protected POST is the only authentication; without it the integration is replayable by anyone who guesses the URL.
- **Expect bursts.** A campaign with 1,000 recipients produces 1,000 `deliverable.created` events in a short window. Make sure your handler scales.
- **Rotate the secret if it leaks.** Delete and recreate the endpoint — there's no in-place rotation in the UI today.

26.8 Cross-references

- The REST API for pulling the same data on demand: Chapter 27 REST API Reference.
- The auto-whitelist refresh integration that uses the same delivery pipeline: Chapter 22 Spam Filter Whitelist.
- The Contacts the `contact.*` events refer to: Chapter 5 Contacts.
- The Campaigns the `campaign.*` events refer to: Chapter 4 Campaigns.
- Directory sync events that produce most `contact.*` traffic in production: Chapter 25 Directory Sync.

REST API Reference

PhishSpot exposes a REST API over JSON at `https://platform.phishspot.com/api/v1`. It covers essentially everything you can do in the admin app: build, schedule and analyze campaigns, manage contacts/groups/templates/courses/media/domains/autopilots, and stream results into your own tooling.

This chapter documents every endpoint in detail — parameters, request bodies, response fields and status codes — so you can integrate without reading the source. For a push-based event model see Chapter 26 Webhooks; for a natural-language AI interface over the same capabilities see Chapter 29 MCP Server.

:::tip[How to read this chapter] §27.1 and §27.2 describe authentication, ID formats, pagination and the **error responses every endpoint shares** — endpoint sections below only list their *additional* status codes. Each endpoint lists its parameters (path / query / body), a runnable `curl` example, the response fields, and a sample response. :::

27.1 Authentication

Every authenticated request must send an API token as a bearer header:

```
Authorization: Bearer <token>
```

Get a token one of two ways:

From the admin UI (recommended). Account settings → API Tokens → New token (see Chapter 14). Copy the value — it is shown once. Store it in a secrets manager.

From the API. POST `email` + `password` (plus `otp_attempt` if 2FA is on) to `/auth`:

```
curl -X POST https://platform.phishspot.com/api/v1/auth \
  -H 'Content-Type: application/json' \
  -d '{"email":"admin@example.com","password":"secret"}'
```

```
{ "token": "abc123…", "user": { "id": 2, "email": "admin@example.com" } }
```

Field	Type	Description
`email`	string	**Required.** The user's email.
`password`	string	**Required.** The user's password.
`otp_attempt`	string	Required only if the user has two-factor auth enabled.

A token belongs to a single user and inherits that user's account memberships. Treat it like a password. All examples below assume `$TOKEN` holds a valid token.

27.2 Conventions

- **Base URL:** `https://platform.phishspot.com/api/v1`. All paths below are relative to it.
- **Content type:** send `Content-Type: application/json`; bodies and responses are JSON.
- **Times:** ISO-8601 (`2026-05-20T14:22:33.000Z`), UTC unless stated. The campaign `scheduled_at` input is interpreted in the **account's timezone**.
- **IDs in paths:** wherever a path takes `:id`, you may pass **either** the integer primary key (`/campaigns/42`) **or** the record's prefixed id (`/campaigns/camp_0u1k…`). Responses always expose the integer `id`; some also expose the prefixed id.
- **`account_id`:** nested routes take `account_id` in the path; it accepts the integer id or the `acct_…` prefixed id. Discover yours with `GET /accounts`.
- **Account scoping:** a token can act only on accounts its user belongs to. Requesting a record in another account returns **404** — the API never confirms another tenant's data exists.
- **Roles:** read endpoints require any role (incl. `member`). Write endpoints (`POST` / `PATCH` / `PUT` / `DELETE` and state actions) require **admin** or **editor**; a `member` token gets `403`. Team/billing and platform-domain admin actions require **admin**.
- **Pagination:** endpoints that paginate accept `?page=N` (1-based) and sometimes `?per_page=M` or `?limit=M`; defaults are noted per endpoint. Non-paginated lists return the full ordered set.

Shared error responses

Unless an endpoint says otherwise, these apply to every call (only endpoint-specific codes are repeated below):

Code	Body	When
`401 Unauthorized`	*(empty)*	Missing or invalid `Authorization` token.
`403 Forbidden`	`{"error":"You are not authorized to perform this action"}`	Token valid but the user's role is insufficient for this action.
`404 Not Found`	`{"error":"Resource not found"}`	No such record, **or** the record belongs to an account the token can't access.
`422 Unprocessable Content`	`{"errors":{"field":["message"]}}` *(or* `{"errors":["message"]}` *for action endpoints)*	Validation failed; inspect `errors`.
`429 Too Many Requests`	*(varies)*	Rate limit exceeded; see §27.17. The `Retry-After` header says when to retry.

27.3 Identity & accounts

`GET /me`

Returns the user behind the token.

Parameters: none (bearer token only).

```
curl -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/me
```

Response `200 OK`

Field	Type	Description
`id`	integer	User id.
`email`	string	User email.
`name`	string	Display name.
`locale`	string	UI locale (`en` / `pl`).
`accounts`	array	Accounts the token can act on (see `GET /accounts`).

`GET /accounts`

Lists every account the token's user can access. Use it to find the `account_id` for nested routes.

Parameters: none.

```
curl -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/accounts
```

Response `200 OK` — array of:

Field	Type	Description
`id`	integer	Account id (use in nested paths).
`prefix_id`	string	Prefixed id (`acct_…`).
`name`	string	Account name.
`locale`	string	Account default locale.

```
[{ "id": 11, "name": "Cydefen Tests", "locale": "pl", "prefix_id": "acct_3kf…" }]
```

27.4 Campaigns

Manage phishing-simulation campaigns: create and edit drafts, drive the campaign through its lifecycle (start, pause, stop, cancel), schedule a future send, duplicate, and read results, recipient progress, replies, and a per-contact event timeline.

Authorization is enforced per account: a campaign is only reachable if it belongs to one of the accounts the bearer token's user is a member of (any membership role can read and write — there are no admin-only campaign actions). State-transition endpoints additionally require the campaign to be in a compatible state (e.g. you can only pause an in-progress campaign), returning `403` otherwise.

> `POST /campaigns/:id/start` and `POST /campaigns/:id/schedule` send **real phishing emails** to the campaign's recipients (immediately, or at the scheduled time). They are not dry-runs. Treat them as destructive, irreversible sends.

The campaign object emitted by `show`, `create`, `update`, and all state-transition endpoints is identical and described once under `GET /campaigns/:id`.

`GET /accounts/:account_id/campaigns`

Lists every campaign in the account, newest first. Use it to enumerate campaigns before drilling into one. **Auth:** Bearer; **role:** read (any role).

Parameters

Name	In	Type	Required	Description
account_id	path	string	yes	Account id (`acct_…` or integer). Must be an account the token's user belongs to.

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/accounts/11/campaigns
```

Response `200 OK` — JSON array of campaign objects (see `GET /campaigns/:id` for the per-object field list), ordered by `created_at` descending.

```
[
  {
    "id": 42,
    "account_id": 11,
    "name": "Q2 Invoice Lure",
    "state": "in_progress",
    "delivery_mode": "immediate",
    "delivery_schedule": null,
    "created_at": "2026-05-01T09:00:00.000Z",
    "updated_at": "2026-05-02T14:12:00.000Z",
    "email_subject": "Your April invoice is ready",
    "email_content": "<p>Hello {{first_name}}…</p>",
    "landing_html": "<form>…</form>",
    "domain": "officelogin.in",
    "course_id": 7,
    "groups": [{ "id": 3, "name": "Finance" }],
    "statistics": {
      "total_contacts": 120,
      "total_deliverables": 120,
      "completion_percentage": 100.0
    },
    "can_start": false,
    "can_pause": true,
    "can_cancel": true
  }
]
```


Status codes

Code	When
200	Campaigns listed (empty array if the account has none).
403	Token's user is not authorized to view the account.
404	`account_id` is not an account the token's user belongs to.

`POST /accounts/:account_id/campaigns`

Creates a new draft campaign in the account. All content fields are optional at creation — only `name` is required — so you can create a bare draft and fill it in with `PATCH`. **Auth:** Bearer; **role:** any role.

Parameters

All body params are wrapped in a `campaign` object.

Name	In	Type	Required	Description
account_id	path	string	yes	Account id (`acct_…` or integer).
campaign[name]	body	string	yes	Campaign name. Must be unique within the account (case-insensitive).

Name	In	Type	Required	Description
campaign[delivery_mode]	body	string	no	One of `immediate`, `scheduled`, `staggered`. Defaults to `immediate`.
campaign[delivery_schedule]	body	string	no	Free-form schedule string used only when `delivery_mode` is `scheduled` (ISO8601 datetime, or 5-field cron). Prefer the `/schedule` endpoint instead.
campaign[email_subject]	body	string	no	Subject line. May contain email merge tags (e.g. `{{first_name}}`); unknown tags fail validation.
campaign[email_content]	body	string	no	HTML email body. Must be well-formed HTML and use only allowed email merge tags.
campaign[landing_html]	body	string	no	Landing-page HTML. Must be well-formed HTML and use only allowed landing merge tags.
campaign[landing_css]	body	string	no	Landing-page CSS. Must be well-formed CSS.
campaign[landing_page_enabled]	body	boolean	no	Whether the landing page is served. Defaults to `false`.
campaign[platform_domain_id]	body	integer	no	Id of the `PlatformDomain` (attacker domain) used for sending and landing. Required before the campaign can start.
campaign[course_id]	body	integer	no	Id of the e-learning course to redirect victims to (used when `end_action_type` is `redirect_to_course`).
campaign[from_email]	body	string	no	Sender email address. Required before the campaign can start.
campaign[from_name]	body	string	no	Sender display name.
campaign[end_action_type]	body	string	no	What happens after a victim acts. One of `nothing`, `redirect_to_course`, `message_page`, `redirect_to_url`. Defaults to `message_page`.

Name	In	Type	Required	Description
campaign[end_action_url]	body	string	no	External URL to redirect to. Required (and must be `http` / `https`, not loop back to a platform domain) when `end_action_type` is `redirect_to_url`.
campaign[end_action_html]	body	string	no	Custom HTML message page. Required when `end_action_type` is `message_page` (auto-seeded with a default if omitted).
campaign[group_ids]	body	array of integer	no	Ids of contact groups to target.

Request

```
curl -X POST -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{
        "campaign": {
          "name": "Q2 Invoice Lure",
          "delivery_mode": "immediate",
          "email_subject": "Your April invoice is ready",
          "email_content": "<p>Hello {{first_name}}…</p>",
          "from_email": "billing@officelogin.in",
          "from_name": "Accounts Payable",
          "platform_domain_id": 5,
          "end_action_type": "redirect_to_course",
          "course_id": 7,
          "group_ids": [3]
        }
      }' \
  https://platform.phishspot.com/api/v1/accounts/11/campaigns
```

Response `201 Created` — the newly created campaign object (same shape as `GET /campaigns/:id`).

Status codes

Code	When
201	Campaign created.
400	The `campaign` object is missing from the body (`ParameterMissing`).
403	Token's user is not authorized to create campaigns in the account.
404	`account_id` is not an account the token's user belongs to.

Code	When
422	Validation failed — e.g. blank/duplicate `name`, invalid `delivery_mode` / `end_action_type` enum, malformed HTML/CSS, disallowed merge tag, or missing `end_action_url` / `end_action_html` for the chosen `end_action_type`. Body: `{ "errors": { "<field>": ["…"] } }`.

`GET /campaigns/:id`

Fetches a single campaign by id (shallow route, not nested under account). Use it to read full campaign content and the action flags that tell you which transitions are currently allowed. **Auth:** Bearer; **role:** any role.

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Campaign id (`camp_…` or integer). Must belong to an account the token's user is a member of.

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/campaigns/42
```

Response `200 OK` — the campaign object.

Field	Type	Description
id	integer	Campaign id.
account_id	integer	Owning account id.
name	string	Campaign name.
state	string	Lifecycle state: `draft`, `in_progress`, `paused`, `cancelled`, `done`, or `scheduled`.
delivery_mode	string	`immediate`, `scheduled`, or `staggered`.
delivery_schedule	string | null	Raw delivery-schedule string (only meaningful for `scheduled` mode).
created_at	string	ISO8601 timestamp.
updated_at	string	ISO8601 timestamp.
email_subject	string | null	Email subject.
email_content	string | null	HTML email body.
landing_html	string | null	Landing-page HTML.

Field	Type	Description
domain	string | null	Name of the associated `PlatformDomain` (e.g. `officelogin.in`), or null if none set.
course_id	integer | null	Associated course id, or null.
groups	array	Targeted groups, each `{ "id": integer, "name": string }`.
statistics	object	**Present only when** `state` is `in_progress`, `paused`, or `done`. Object with `total_contacts` (integer), `total_deliverables` (integer), `completion_percentage` (float).
can_start	boolean	Whether `start` / `schedule` is allowed now (true for `draft` / `scheduled`).
can_pause	boolean	Whether `pause` is allowed now (true only when `in_progress`).
can_cancel	boolean	Whether `cancel` is allowed now (true for `in_progress` / `paused` / `scheduled`).


```
{
  "id": 42,
  "account_id": 11,
  "name": "Q2 Invoice Lure",
  "state": "draft",
  "delivery_mode": "immediate",
  "delivery_schedule": null,
  "created_at": "2026-05-01T09:00:00.000Z",
  "updated_at": "2026-05-01T09:00:00.000Z",
  "email_subject": "Your April invoice is ready",
  "email_content": "<p>Hello {{first_name}}…</p>",
  "landing_html": "<form>…</form>",
  "domain": "officelogin.in",
  "course_id": 7,
  "groups": [{ "id": 3, "name": "Finance" }],
  "can_start": true,
  "can_pause": false,
  "can_cancel": false
}
```


Status codes

Code	When
200	Campaign returned.
404	No campaign with that id in any account the token's user belongs to (includes cross-account access attempts).

`PATCH /campaigns/:id`

Updates an existing campaign. Editing is only permitted while the campaign is in `draft` or `scheduled` state (the authorization policy rejects edits to running/finished campaigns). **Auth:** Bearer; **role:** any role.

Parameters

Body params are wrapped in a `campaign` object; the same permitted keys as `POST` apply (all optional on update — send only the fields you want to change).

Name	In	Type	Required	Description
id	path	string	yes	Campaign id (`camp_…` or integer).
campaign[…]	body	—	no	Any subset of the keys listed under `POST /accounts/:account_id/campaigns` (`name`, `delivery_mode`, `delivery_schedule`, `email_subject`, `email_content`, `landing_html`, `landing_css`, `landing_page_enabled`, `platform_domain_id`, `course_id`, `from_email`, `from_name`, `end_action_type`, `end_action_url`, `end_action_html`, `group_ids[]`).

Request

```
curl -X PATCH -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{ "campaign": { "email_subject": "Action required: invoice overdue" } }' \
  https://platform.phishspot.com/api/v1/campaigns/42
```

Response `200 OK` — the updated campaign object (same shape as `GET /campaigns/:id`).

Status codes

Code	When
200	Campaign updated.
400	The `campaign` object is missing from the body (`ParameterMissing`).
403	Campaign is not in `draft` / `scheduled` state (editing locked), or user not authorized.
404	Campaign not found in the user's accounts.
422	Validation failed (same validations as `POST`). Body: `{ "errors": { … } }`.

`DELETE /campaigns/:id`

Permanently deletes a campaign and its dependent records (recipients, deliverables, events, replies). Allowed in any state. **Auth:** Bearer; **role:** any role.

No parameters beyond the bearer token and the path id.

Name	In	Type	Required	Description
id	path	string	yes	Campaign id (`camp_…` or integer).

Request

```
curl -X DELETE -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/campaigns/42
```

Response `204 No Content` — empty body.

Status codes

Code	When
204	Campaign deleted.
403	User not authorized to delete the campaign.
404	Campaign not found in the user's accounts.

`POST /campaigns/:id/start`

> Starts the campaign **and sends real phishing emails** to all targeted recipients (in batches for `immediate` mode, or per the delivery schedule for other modes). This is irreversible — once started, emails go out.

Transitions a `draft` or `scheduled` campaign to `in_progress` and enqueues the send jobs. Before sending, the server runs a readiness preflight and rejects the request if the campaign is incomplete. **Auth:** Bearer; **role:** any role.

No parameters beyond the bearer token and the path id.

Name	In	Type	Required	Description
id	path	string	yes	Campaign id (`camp_…` or integer).

Request

```
curl -X POST -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/campaigns/42/start
```

Response `200 OK` — the campaign object with `state: "in_progress"`.

Status codes

Code	When
200	Campaign started; sends enqueued.
403	Campaign is not in a startable state (`draft` / `paused` / `scheduled`).
404	Campaign not found in the user's accounts.
422	Readiness preflight failed. Body: `{ "errors": ["…", "…"] }` (a flat array of human-readable messages). Triggers include: missing email subject, missing email content, missing sender email, no platform domain set, platform domain not active or sending-blocked, no recipients targeted, and end-action gaps (missing course for `redirect_to_course`, missing URL for `redirect_to_url`, missing HTML for `message_page`).

`POST /campaigns/:id/stop`

Marks an in-progress campaign as `done` (completed). Use it to end a running campaign early; pending sends are not re-enqueued. **Auth:** Bearer; **role:** any role.

No parameters beyond the bearer token and the path id.

Name	In	Type	Required	Description
id	path	string	yes	Campaign id (`camp_…` or integer).

Request

```
curl -X POST -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/campaigns/42/stop
```

Response `200 OK` — the campaign object with `state: "done"`.

Status codes

Code	When
200	Campaign marked done.
403	Campaign is not `in_progress`.
404	Campaign not found in the user's accounts.
422	State transition rejected by the model. Body: `{ "errors": { … } }`.

`POST /campaigns/:id/pause`

Pauses an in-progress campaign (state → `paused`), halting further scheduled sends. Resume by calling `start` again. **Auth:** Bearer; **role:** any role.

No parameters beyond the bearer token and the path id.

Name	In	Type	Required	Description
id	path	string	yes	Campaign id (`camp_…` or integer).

Request

```
curl -X POST -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/campaigns/42/pause
```

Response `200 OK` — the campaign object with `state: "paused"`.

Status codes

Code	When
200	Campaign paused.
403	Campaign is not `in_progress`.
404	Campaign not found in the user's accounts.
422	State transition rejected by the model. Body: `{ "errors": { … } }`.

`POST /campaigns/:id/cancel`

Cancels a campaign (state → `cancelled`). Allowed from `in_progress`, `paused`, or `scheduled` (not from `draft` or already-finished campaigns). **Auth:** Bearer; **role:** any role.

No parameters beyond the bearer token and the path id.

Name	In	Type	Required	Description
id	path	string	yes	Campaign id (`camp_…` or integer).

Request

```
curl -X POST -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/campaigns/42/cancel
```

Response `200 OK` — the campaign object with `state: "cancelled"`.

Status codes

Code	When
200	Campaign cancelled.
403	Campaign is not in a cancellable state (`in_progress` / `paused` / `scheduled`).
404	Campaign not found in the user's accounts.
422	State transition rejected by the model. Body: `{ "errors": { … } }`.

`POST /campaigns/:id/duplicate`

Clones the campaign into a fresh `draft` (with a numbered-suffix name like `"Q2 Invoice Lure (1)"`), copying content, targeted groups, and recipients but resetting state, schedule, and snapshot. Use it to re-run or branch a campaign. **Auth:** Bearer; **role:** any role.

No parameters beyond the bearer token and the path id.

Name	In	Type	Required	Description
id	path	string	yes	Campaign id (`camp_…` or integer) of the source campaign.

Request

```
curl -X POST -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/campaigns/42/duplicate
```

Response `201 Created` — the new draft campaign object (same shape as `GET /campaigns/:id`), with a new `id` and `state: "draft"`.

Status codes

Code	When
201	Duplicate created.
403	User not authorized.
404	Source campaign not found in the user's accounts.
422	The duplicate failed to save (validation). Body: `{ "errors": { … } }`.

`POST /campaigns/:id/schedule`

> Schedules the campaign to **send real phishing emails** at the given future time. When the scheduled time arrives, emails go out automatically.

Schedules a `draft` campaign for a future send (state → `scheduled`). Runs the same readiness preflight as `start`, plus time-validity checks. **Auth:** Bearer; **role:** any role.

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Campaign id (`camp_…` or integer).
scheduled_at	body	string	yes	Target send time as a local (account-timezone) datetime, no offset — e.g. `2026-06-10T09:00` (the value a `datetime-local` input produces). It is interpreted in the account's timezone (falling back to UTC if the account has none) and converted to UTC server-side. Must be in the future and at least 5 minutes from now. **Not** wrapped in a `campaign` object — sent as a top-level body key.

Request

```
curl -X POST -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{ "scheduled_at": "2026-06-10T09:00" }' \
  https://platform.phishspot.com/api/v1/campaigns/42/schedule
```

Response `200 OK` — the campaign object with `state: "scheduled"`.

Status codes

Code	When
200	Campaign scheduled.
403	Campaign is not in a startable state (must be `draft`).
404	Campaign not found in the user's accounts.
422	Scheduling failed. Body: `{ "errors": ["…"] }` (flat array). Triggers: `scheduled_at` blank, unparseable datetime, time in the past, time less than 5 minutes from now, or the start-readiness preflight failing (same content/sender/domain/recipient/end-action checks as `start`).

`POST /campaigns/:id/cancel_schedule`

Cancels a pending schedule, returning the campaign to `draft` and removing its queued send job. Only valid for a `scheduled` campaign. **Auth:** Bearer; **role:** any role.

No parameters beyond the bearer token and the path id.

Name	In	Type	Required	Description
id	path	string	yes	Campaign id (`camp_…` or integer).

Request

```
curl -X POST -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/campaigns/42/cancel_schedule
```

Response `200 OK` — the campaign object (state returned to `draft`).

Status codes

Code	When
200	Schedule cancelled.
403	User not authorized (policy requires `scheduled` state).
404	Campaign not found in the user's accounts.
422	Campaign is not currently `scheduled`. Body: `{ "error": "Campaign is not scheduled (state: <state>); nothing to cancel." }` (note the singular `error` key here).

`GET /campaigns/:id/results`

Returns aggregated campaign statistics: the overall engagement funnel plus per-group and per-department breakdowns. Use it to render dashboards and reports. **Auth:** Bearer; **role:** any role.

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Campaign id (`camp_…` or integer).

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/campaigns/42/results
```

Response `200 OK` — statistics object.

Field	Type	Description
campaign_id	integer	Campaign id.
name	string	Campaign name.
funnel	object	Overall engagement funnel (counts + rates). See sub-fields below.

Field	Type	Description
groups	array	Per-group breakdown. Empty array if the campaign targets no groups. See sub-fields below.
departments	array	Per-department breakdown (contacts grouped by their `department`). Empty if no departments. Same numeric sub-fields as a group entry, with `name` but no `id`.

`funnel` sub-fields:

Field	Type	Description
sent	integer	Distinct contacts the email was successfully sent to.
opened	integer	Distinct contacts who opened.
clicked	integer	Distinct contacts who clicked.
submitted	integer	Distinct contacts who submitted data on the landing page.
trained	integer	Deliverables that reached the `educated` (training-completed) state.
replied	integer	Distinct contacts who replied to the phishing email (side-channel signal, not part of the click/submit funnel).
open_rate	float	`opened / sent` as a percentage (1 decimal).
click_rate	float	`clicked / sent` percentage.
submit_rate	float	`submitted / sent` percentage.
train_rate	float	`trained / sent` percentage.
reply_rate	float	`replied / sent` percentage.

Each `groups[]` entry: `name` (string), `id` (integer), `total_contacts` (integer), `sent`, `opened`, `clicked`, `submitted`, `trained` (integers), and `open_rate`, `click_rate`, `submit_rate`, `train_rate` (floats).

```
{
  "campaign_id": 42,
  "name": "Q2 Invoice Lure",
  "funnel": {
    "sent": 120,
    "opened": 84,
    "clicked": 37,
    "submitted": 12,
    "trained": 9,
    "replied": 3,
    "open_rate": 70.0,
    "click_rate": 30.8,
    "submit_rate": 10.0,
    "train_rate": 7.5,
    "reply_rate": 2.5
  },
  "groups": [
    {
      "name": "Finance",
      "id": 3,
      "total_contacts": 60,
      "sent": 60,
      "opened": 45,
      "clicked": 22,
      "submitted": 8,
      "trained": 6,
      "open_rate": 75.0,
      "click_rate": 36.7,
      "submit_rate": 13.3,
      "train_rate": 10.0
    }
  ],
  "departments": [
    {
      "name": "Accounting",
      "total_contacts": 40,
      "sent": 40,
      "opened": 30,
      "clicked": 15,
      "submitted": 5,
      "trained": 4,
      "open_rate": 75.0,
      "click_rate": 37.5,
      "submit_rate": 12.5,
      "train_rate": 10.0
    }
  ]
}
```


Status codes

Code	When
200	Statistics returned.
404	Campaign not found in the user's accounts.

`GET /campaigns/:id/recipients`

Returns a paginated, filterable list of campaign recipients with their per-contact delivery stage, training status, and reply flag. Recipients are ordered by contact last name then first name. **Auth:** Bearer; **role:** any role.

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Campaign id (`camp_…` or integer).
page	query	integer	no	1-based page number; values below 1 are clamped to 1. Defaults to 1. Page size is fixed at 25.
stage	query	string	no	Filter by funnel stage: `sent`, `opened`, `clicked`, `submitted`, or `trained`. `all` (or omitted) returns everyone. Filters are cumulative-by-stage (e.g. `opened` includes those who later clicked/ submitted/were educated).
replied	query	boolean	no	When truthy (`true` / `1`), restrict to contacts who replied to the email.
group_id	query	integer	no	Restrict to contacts in this group (must belong to the campaign's account).
department	query	string	no	Restrict to contacts whose `department` matches this exact value.

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  "https://platform.phishspot.com/api/v1/campaigns/42/recipients?stage=clicked&page=1"
```

Response `200 OK` — paginated recipients.

Field	Type	Description
campaign_id	integer	Campaign id.
page	integer	Current page number.
per_page	integer	Page size (always 25).

Field	Type	Description
total	integer	Total recipients matching the filters (across all pages).
recipients	array	Recipient rows (see sub-fields).

Each `recipients[]` entry:

Field	Type	Description
id	integer	Contact id.
contact_id	integer	Contact id (same value as `id`).
email	string	Contact email.
full_name	string	Contact full name.
status	string	Delivery state: `pending`, `sent`, `delivered`, `bounced`, `opened`, `clicked`, `submitted`, or `educated`.
stage	string	Same value as `status` (alias).
training_status	string	`not_started`, `in_progress`, or `completed`.
replied	boolean	Whether the contact replied to the phishing email.


```
{
  "campaign_id": 42,
  "page": 1,
  "per_page": 25,
  "total": 37,
  "recipients": [
    {
      "id": 901,
      "contact_id": 901,
      "email": "jane.doe@victimco.com",
      "full_name": "Jane Doe",
      "status": "clicked",
      "stage": "clicked",
      "training_status": "in_progress",
      "replied": false
    }
  ]
}
```


Status codes

Code	When
200	Recipients returned.
404	Campaign not found in the user's accounts, or a `group_id` / `department` lookup references a record outside the campaign's account.

`GET /campaigns/:id/replies`

Returns a paginated list of inbound replies recipients sent back to the phishing email, newest first. Use it to surface engaged targets and review reply content. **Auth:** Bearer; **role:** any role.

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Campaign id (`camp_…` or integer).
page	query	integer	no	1-based page number; clamped to a minimum of 1. Defaults to 1. Page size is fixed at 25.

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  "https://platform.phishspot.com/api/v1/campaigns/42/replies?page=1"
```

Response `200 OK` — paginated replies.

Field	Type	Description
campaign_id	integer	Campaign id.
page	integer	Current page number.
per_page	integer	Page size (always 25).
total	integer	Total replies for the campaign.
replies	array	Reply rows (see sub-fields).

Each `replies[]` entry:

Field	Type	Description
id	integer	Reply id.
from_email	string	Sender (recipient) email address.
received_at	string	ISO8601 timestamp of when the reply was received.
subject	string	Reply subject line.
excerpt	string	Plain-text excerpt of the reply body (truncated).
attachments_count	integer	Number of attachments on the reply.

```
{
  "campaign_id": 42,
  "page": 1,
  "per_page": 25,
  "total": 3,
  "replies": [
    {
      "id": 5501,
      "from_email": "jane.doe@victimco.com",
      "received_at": "2026-05-02T11:24:00Z",
      "subject": "Re: Your April invoice is ready",
      "excerpt": "Is this really from accounting? I don't recognize…",
      "attachments_count": 0
    }
  ]
}
```


Status codes

Code	When
200	Replies returned.
404	Campaign not found in the user's accounts.

`GET /campaigns/:id/timeline`

Returns the chronological event timeline for a single contact within the campaign (sent → opened → clicked → submitted → training, etc.). Use it to inspect one victim's full interaction history. **Auth:** Bearer; **role:** any role.

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Campaign id (`camp_…` or integer).
contact_id	query	integer	yes	Id of the contact whose timeline to return. Must belong to the campaign's account.

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  "https://platform.phishspot.com/api/v1/campaigns/42/timeline?contact_id=901"
```

Response `200 OK` — the contact's event list, ordered oldest-first.

Field	Type	Description
campaign_id	integer	Campaign id.

Field	Type	Description
contact_id	integer	Contact id.
events	array	Events for this contact (see sub-fields).

Each `events[]` entry:

Field	Type	Description
genre	string	Event type: `sent`, `delivered`, `bounced`, `opened`, `clicked`, `submitted_data`, `started_training`, `completed_training`, `failed_quiz`, `passed_quiz`, or `replied`.
created_at	string	ISO8601 timestamp of the event.
metadata	object | null	Arbitrary event metadata (e.g. user agent, IP, quiz details), as stored.

```
{
  "campaign_id": 42,
  "contact_id": 901,
  "events": [
    { "genre": "sent", "created_at": "2026-05-01T09:05:00Z", "metadata": {} },
    { "genre": "opened", "created_at": "2026-05-01T09:41:00Z", "metadata": { "ua":
        "Mozilla/5.0" } },
    { "genre": "clicked", "created_at": "2026-05-01T09:42:00Z", "metadata": { "ip":
        "203.0.113.7" } }
  ]
}
```

Status codes

Code	When
200	Timeline returned.
404	Campaign not found in the user's accounts, or `contact_id` does not reference a contact in the campaign's account (a missing/invalid `contact_id` raises a not-found).

27.5 Phishing templates

The phishing-template library is the catalog of ready-made phishing scenarios (email + landing page + post-click action) that an account can deploy into a real campaign. Templates come in two flavors: **curated** (platform-provided, shared with every account, read-only) and **custom** (created by an account, visible only to that account). Templates are organized into a tree of categories (up to three levels deep). Deploying a template creates a new draft campaign pre-filled with the template's content — it never sends any email.

`GET /accounts/:account_id/phishing_templates`

Lists templates visible to an account, paginated 12 per page. Use the `tab` parameter to switch between the shared curated library and the account's own custom templates, and `category` / `search` to narrow the results. Returns metadata only (no HTML blobs) — call the show endpoint for full content. **Auth:** Bearer; **role:** read (any role).

Parameters

Name	In	Type	Required	Description
account_id	path	string	yes	Account id (`acct_…` or integer). Must be an account the token's user belongs to.
tab	query	string	no	Which library to list. `custom` returns this account's own templates; any other value (or omitted) returns the shared `curated` library. Default: curated.
category	query	string or array	no	One or more category ids (`tcat_…` prefix ids or integers) to filter by. Matches templates assigned to the given category **or any of its descendants**. Pass multiple as repeated `category[]` params. Unrecognized ids are ignored.
search	query	string	no	Case-insensitive substring match against template `name` and `description`.
page	query	integer	no	1-based page number. Values below 1 are clamped to 1. Default: 1. Page size is fixed at 12.

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  "https://platform.phishspot.com/api/v1/accounts/11/phishing_templates?
        tab=curated&category=tcat_abc123&search=invoice&page=1"
```

Response `200 OK` — pagination envelope plus a `templates` array of template metadata.

Field	Type	Description
tab	string	The resolved tab: `"curated"` or `"custom"`.
page	integer	Current page number.
per_page	integer	Page size (always 12).
total	integer	Total number of templates matching the filters (across all pages).

Field	Type	Description
templates	array	Array of template objects (see fields below).
templates[].id	integer	Raw numeric template id.
templates[].name	string	Template name.
templates[].description	string | null	Free-text description.
templates[].curated	boolean	`true` for platform-provided templates, `false` for account-owned.
templates[].draft	boolean	`true` if the template is an unpublished draft (missing required content). Drafts cannot be deployed.
templates[].email_subject	string | null	The phishing email subject line.
templates[].landing_page_enabled	boolean	Whether the template includes a hosted landing page.
templates[].created_at	string (ISO 8601)	Creation timestamp.
templates[].updated_at	string (ISO 8601)	Last-update timestamp.
templates[].template_id	string	Prefixed id (`tmpl_…`). Use this in the show/deploy paths.
templates[].categories	array	Categories this template is assigned to.
templates[].categories[].id	integer	Raw numeric category id.
templates[].categories[].category_id	string	Prefixed category id (`tcat_…`).
templates[].categories[].name	string	Localized category name (current request locale, falling back to English).

```
{
  "tab": "curated",
  "page": 1,
  "per_page": 12,
  "total": 37,
  "templates": [
    {
      "id": 84,
      "name": "Unpaid Invoice Reminder",
      "description": "Spoofed accounts-payable invoice with a credential-harvesting
        login page.",
      "curated": true,
      "draft": false,
      "email_subject": "Action required: invoice #44021 is overdue",
      "landing_page_enabled": true,
      "created_at": "2026-01-14T09:12:00.000Z",
      "updated_at": "2026-03-02T16:40:11.000Z",
      "template_id": "tmpl_8x2k9q",
      "categories": [
        { "id": 5, "category_id": "tcat_abc123", "name": "Finance" }
      ]
    }
  ]
}
```


Status codes

Code	When
200	Templates listed (the array may be empty).
404	`account_id` is not an account the token's user belongs to. Body: `{"error":"Account not found"}`.

`GET /accounts/:account_id/phishing_template_categories`

Returns the full category tree (roots with nested children, up to three levels) for the template-library filter UI. Useful for building a category picker before calling the templates list with `category=`. **Auth:** Bearer; **role:** read (any role).

Parameters

Name	In	Type	Required	Description
account_id	path	string	yes	Account id (`acct_…` or integer). Must be an account the token's user belongs to.

(Categories are global, not account-scoped — the `account_id` only gates access. No other parameters.)

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/accounts/11/phishing_template_categories
```

Response `200 OK` — a `categories` array of root categories, each recursively embedding its children.

Field	Type	Description
categories	array	Root categories, ordered by `position`.
categories[].id	integer	Raw numeric category id.
categories[].category_id	string	Prefixed category id (`tcat_…`).
categories[].name	string	Localized category name (request locale, falling back to English).
categories[].slug	string	URL-safe slug (unique, derived from the English name).
categories[].depth	integer	Tree depth: `0` for roots, `1` for children, `2` for grandchildren.
categories[].is_leaf	boolean	`true` when the category has no children.
categories[].children	array	Nested child categories with the same shape (empty array for leaves).

```
{
  "categories": [
    {
      "id": 1,
      "category_id": "tcat_root01",
      "name": "Finance",
      "slug": "finance",
      "depth": 0,
      "is_leaf": false,
      "children": [
        {
          "id": 5,
          "category_id": "tcat_abc123",
          "name": "Invoices",
          "slug": "invoices",
          "depth": 1,
          "is_leaf": true,
          "children": []
        }
      ]
    }
  ]
}
```

Status codes

Code	When
200	Category tree returned (may be an empty array).
404	`account_id` is not an account the token's user belongs to. Body: `{"error":"Account not found"}`.

`GET /phishing_templates/:id`

Returns a single template with its **full content** — email body, landing-page HTML/CSS, and post-click (end action) configuration. Use this to render a preview or to inspect what a deploy will copy into a campaign. This route is shallow (no `account_id` in the path); the token's user must be able to see the template (their own custom templates plus all curated ones). **Auth:** Bearer; **role:** read (any role).

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Template id (`tmpl_…` or integer).

No parameters beyond the bearer token.

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/phishing_templates/tmpl_8x2k9q
```

Response `200 OK` — the full template object.

Field	Type	Description
id	integer	Raw numeric template id.
name	string	Template name.
description	string | null	Free-text description.
curated	boolean	`true` for platform-provided templates, `false` for account-owned.
draft	boolean	`true` if unpublished. Drafts cannot be deployed.
email_subject	string | null	Phishing email subject line (may contain merge tags).
email_content	string | null	Full phishing email HTML body.
landing_html	string | null	Landing-page HTML.
landing_css	string | null	Landing-page CSS.
landing_page_enabled	boolean	Whether a hosted landing page is included.
end_action_type	string	What happens after a victim submits the landing page. One of `nothing`, `redirect_to_course`, `message_page`, `redirect_to_url`.

Field	Type	Description
end_action_url	string | null	Target URL when `end_action_type` is `redirect_to_url` (must be http/https).
end_action_html	string | null	HTML shown when `end_action_type` is `message_page` (e.g. the awareness page).
created_at	string (ISO 8601)	Creation timestamp.
updated_at	string (ISO 8601)	Last-update timestamp.
template_id	string	Prefixed id (`tmpl_…`).
course_id	integer | null	Linked e-learning course id (used when `end_action_type` is `redirect_to_course`).
publishable	boolean	`true` when all required fields (name, subject, email body, landing HTML) are present.
categories	array	Assigned categories: each with `id` (integer), `category_id` (`tcat_…`), and `name` .


```
{
  "id": 84,
  "name": "Unpaid Invoice Reminder",
  "description": "Spoofed accounts-payable invoice with a credential-harvesting login
      page.",
  "curated": true,
  "draft": false,
  "email_subject": "Action required: invoice #44021 is overdue",
  "email_content": "<html><body><p>Dear {{first_name}}, your invoice is overdue…</p></
      body></html>",
  "landing_html": "<form action=\"#\">…</form>",
  "landing_css": "body { font-family: sans-serif; }",
  "landing_page_enabled": true,
  "end_action_type": "message_page",
  "end_action_url": null,
  "end_action_html": "<h1>You've been phished by a simulation.</h1>",
  "created_at": "2026-01-14T09:12:00.000Z",
  "updated_at": "2026-03-02T16:40:11.000Z",
  "template_id": "tmpl_8x2k9q",
  "course_id": null,
  "publishable": true,
  "categories": [
    { "id": 5, "category_id": "tcat_abc123", "name": "Finance" }
  ]
}
```


Status codes

Code	When
200	Template returned.

Code	When
403	The template is not visible to the token's user (another account's custom template). Body: `{"error":"You are not authorized to perform this action"}`.
404	No template matches `id`. Body: `{"error":"Resource not found"}`.

`POST /phishing_templates/:id/deploy`

Creates a new **draft campaign** in the target account, pre-filled with the template's content (email subject/body, landing HTML/CSS, end action, course). This route is shallow, so the deploying account is passed in the body as `account_id`. With `quick_launch=true` it additionally adds **all** of the account's contacts as recipients and advances the campaign to the review step. **This endpoint never sends any email** — a human launches the campaign from the PhishSpot UI. Drafts cannot be deployed. **Auth:** Bearer; **role:** any member of the target account.

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Template id to deploy (`tmpl_…` or integer). Must not be a draft template.
account_id	body	string	yes	Account to create the campaign in (`acct_…` or integer). Must be an account the token's user belongs to.
quick_launch	body	boolean	no	When truthy (`true`, `"1"`, etc.), bulk-adds every account contact as a recipient and marks wizard steps 1–5 complete so the campaign lands on the review step. Requires an active sending domain and at least one contact. Default: `false`.

Request

```
curl -X POST -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{ "account_id": "acct_4f8a2c", "quick_launch": true }' \
  https://platform.phishspot.com/api/v1/phishing_templates/tmpl_8x2k9q/deploy
```

Response `201 Created` — the newly created campaign (same shape as the campaign show endpoint). A freshly deployed campaign is in the `draft` state, so the `statistics` block is omitted (it only appears once a campaign is in progress, paused, or done).

Field	Type	Description
id	integer	Raw numeric campaign id.

Field	Type	Description
account_id	integer	Owning account id.
name	string	Auto-generated name: `"<Template name> – YYYY-MM-DD HH:MM:SS"` (with a numeric suffix on collision).
state	string	Lifecycle state — `draft` immediately after deploy. One of `draft`, `in_progress`, `paused`, `cancelled`, `done`, `scheduled`.
delivery_mode	string | null	`immediate`, `scheduled`, or `staggered` (not set by deploy).
delivery_schedule	object | null	Delivery schedule config (not set by deploy).
created_at	string (ISO 8601)	Creation timestamp.
updated_at	string (ISO 8601)	Last-update timestamp.
email_subject	string | null	Copied from the template.
email_content	string | null	Copied from the template.
landing_html	string | null	Copied from the template.
domain	string | null	Sending/landing platform domain name (auto-selected from the account's available domains, may be null).
course_id	integer | null	Linked course id (template's course, or the account default).
groups	array	Contact groups on the campaign — empty right after deploy. Each: `id`, `name`.
can_start	boolean	Whether the campaign can transition to start.
can_pause	boolean	Whether the campaign can be paused.
can_cancel	boolean	Whether the campaign can be cancelled.

```
{
  "id": 512,
  "account_id": 11,
  "name": "Unpaid Invoice Reminder – 2026-06-02 14:30:07",
  "state": "draft",
  "delivery_mode": null,
  "delivery_schedule": null,
  "created_at": "2026-06-02T14:30:07.000Z",
  "updated_at": "2026-06-02T14:30:07.000Z",
  "email_subject": "Action required: invoice #44021 is overdue",
  "email_content": "<html><body><p>Dear {{first_name}}…</p></body></html>",
  "landing_html": "<form action=\"#\">…</form>",
  "domain": "officelogin.in",
  "course_id": null,
  "groups": [],
  "can_start": false,
  "can_pause": false,
  "can_cancel": false
}
```


Status codes

Code	When
201	Campaign created from the template.
403	The template is a draft (drafts cannot be deployed), or it is not visible to the token's user. Body: `{"error":"You are not authorized to perform this action"}`.
404	No template matches `id`, **or** the body `account_id` is not an account the token's user belongs to. Body: `{"error":"Resource not found"}` (template) / `{"error":"Account not found"}` (account).
422	`quick_launch=true` but the account has no active sending domain (`"Quick launch needs an active sending domain for this account."`) or no contacts (`"Quick launch needs at least one contact in the account."`). Body: `{"error":"<message>"}`.

27.6 Contacts & groups

Contacts are the employees you target with phishing simulations; groups are named collections used to scope campaigns. Both are account-scoped: collection actions are nested under `/accounts/:account_id/…`, while reads/writes on an individual record use the shallow `/contacts/:id` and `/groups/:id` routes. Membership in the account is required for every endpoint — all policy checks pass for any account member (read and write alike), so there is no admin/editor distinction here. The only

write gate is that a group participating in an active campaign (`in_progress` or `paused`) is **locked** and cannot be updated, deleted, or have its membership changed.

`GET /accounts/:account_id/contacts`

Lists every contact in the account, ordered by last name then first name, with each contact's groups inlined. Use it to page through or sync your roster. **Auth:** Bearer; **role:** read (any role).

Parameters

Name	In	Type	Required	Description
account_id	path	string	yes	Account id (`acct_…` or integer).

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/accounts/11/contacts
```

Response `200 OK` — a JSON array of contact objects (see the contact fields below).

Field	Type	Description
id	integer	Contact primary key.
account_id	integer	Owning account id.
first_name	string	Given name.
last_name	string|null	Surname.
email	string	Email address (unique per account).
telephone	string|null	Phone number.
created_at	string	ISO 8601 timestamp.
updated_at	string	ISO 8601 timestamp.
full_name	string	Convenience: `"first_name last_name"` trimmed.
groups	array	Groups this contact belongs to; each `{ id, name }`.
groups[].id	integer	Group id.
groups[].name	string	Group name (normalized, snake_case for manual groups).

```
[
  {
    "id": 501,
    "account_id": 11,
    "first_name": "Ada",
    "last_name": "Kowalska",
    "email": "ada.kowalska@example.com",
    "telephone": "+48 600 123 456",
    "created_at": "2026-05-01T09:30:00.000Z",
    "updated_at": "2026-05-12T14:02:11.000Z",
    "full_name": "Ada Kowalska",
    "groups": [
      { "id": 90, "name": "finance" }
    ]
  }
]
```

Status codes

Code	When
200	Contacts returned (possibly an empty array).
404	`account_id` is not an account the token's user belongs to.

`POST /accounts/:account_id/contacts`

Creates a single contact in the account. Use the import endpoint instead for bulk loads. **Auth:** Bearer; **role:** read (any role).

Parameters

Body params are wrapped in a `contact` object.

Name	In	Type	Required	Description
account_id	path	string	yes	Account id (`acct_…` or integer).
first_name	body	string	yes	Given name (max 255). Required by the model.
email	body	string	yes	Email address. Must match a standard email format and be unique within the account (case-insensitive). Max 255.
last_name	body	string	no	Surname (max 255).
telephone	body	string	no	Phone number (max 50). Must match `+CC (NNN) NNN-NNNN`-style formats; blank allowed.
group_ids	body	array	no	Array of group ids to attach the contact to on creation.

Request

```
curl -X POST -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{ "contact": { "first_name": "Ada", "last_name": "Kowalska", "email":
        "ada.kowalska@example.com", "telephone": "+48 600 123 456", "group_ids":
        [90] } }' \
  https://platform.phishspot.com/api/v1/accounts/11/contacts
```

Response `201 Created` — the created contact, using the same fields as the list endpoint above.

```
{
  "id": 501,
  "account_id": 11,
  "first_name": "Ada",
  "last_name": "Kowalska",
  "email": "ada.kowalska@example.com",
  "telephone": "+48 600 123 456",
  "created_at": "2026-05-01T09:30:00.000Z",
  "updated_at": "2026-05-01T09:30:00.000Z",
  "full_name": "Ada Kowalska",
  "groups": [
    { "id": 90, "name": "finance" }
  ]
}
```

Status codes

Code	When
201	Contact created.
400	Body is missing the top-level `contact` key.
404	`account_id` is not an account the token's user belongs to.
422	Validation failed (e.g. missing `first_name`, missing/malformed `email`, or duplicate email within the account). Body is `{ "errors": { … } }`.

`POST /accounts/:account_id/contacts/import`

Bulk-imports contacts into the account from CSV. Existing contacts (matched by email) are updated with non-blank values; new ones are created; groups named in the data are created and associations are made. Provide **either** raw CSV text in `csv` **or** a JSON array in `contacts` — the array is converted to CSV server-side using the canonical header order. **Auth:** Bearer; **role:** read (any role).

The canonical CSV header order is: `first_name`, `last_name`, `email`, `telephone`, `groups`, `department`, `title`, `location`. In the `groups` column, multiple groups are separated by `|` (a pipe). When using the JSON `contacts` form, each row's `groups` may be an array (e.g. `["finance","exec"]`) which is joined with `|` automatically.

Parameters

Name	In	Type	Required	Description
account_id	path	string	yes	Account id (`acct_…` or integer).
csv	body	string	conditional	Raw CSV text with the canonical header row. Required if `contacts` is omitted. Takes precedence if both are given.
contacts	body	array	conditional	Array of row objects keyed by the canonical headers. Required if `csv` is omitted. Each row's `groups` may be a string or an array of group names.

> You must supply exactly one of `csv` or `contacts`. If both are blank/absent the request fails with `422`.

Request

```
curl -X POST -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{ "contacts": [
        { "first_name": "Ada", "last_name": "Kowalska", "email":
        "ada.kowalska@example.com", "telephone": "+48600123456", "groups": ["finance"],
        "department": "Finance", "title": "Analyst", "location": "Warsaw" },
        { "first_name": "Jan", "email": "jan.nowak@example.com", "groups": ["finance",
        "exec"] }
      ] }' \
  https://platform.phishspot.com/api/v1/accounts/11/contacts/import
```

Response `200 OK` — a summary of how the rows were processed.

Field	Type	Description
created	integer	Number of new contacts inserted.
updated	integer	Number of existing contacts (matched by email) updated with new non-blank values.
failed	integer	Number of rows rejected as invalid. (A downloadable failed-rows CSV report is attached to the account.)

```
{
  "created": 1,
  "updated": 1,
  "failed": 0
}
```

Status codes

Code	When
200	Import ran; returns the `{created, updated, failed}` summary.
404	`account_id` is not an account the token's user belongs to.
422	Neither `csv` nor `contacts` was provided. Body is `{ "error": "Provide either csv or contacts." }`.

`GET /contacts/:id`

Fetches a single contact by id, scoped to the token user's accounts. Use it to read one contact without listing the whole account. **Auth:** Bearer; **role:** read (any role).

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Contact id (`cont_…` or integer).

No parameters beyond the bearer token and path id.

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/contacts/cont_abc123
```

Response `200 OK` — the contact, using the same fields as the list endpoint.

```
{
  "id": 501,
  "account_id": 11,
  "first_name": "Ada",
  "last_name": "Kowalska",
  "email": "ada.kowalska@example.com",
  "telephone": "+48 600 123 456",
  "created_at": "2026-05-01T09:30:00.000Z",
  "updated_at": "2026-05-12T14:02:11.000Z",
  "full_name": "Ada Kowalska",
  "groups": [
    { "id": 90, "name": "finance" }
  ]
}
```

Status codes

Code	When
200	Contact found.

Code	When
404	No contact with that id in any account the token's user belongs to.

`PATCH /contacts/:id`

Updates a single contact. Send only the fields you want to change, wrapped in a `contact` object. **Auth:** Bearer; **role:** read (any role).

Parameters

Body params are wrapped in a `contact` object.

Name	In	Type	Required	Description
id	path	string	yes	Contact id (`cont_…` or integer).
first_name	body	string	no	Given name (max 255). Cannot be cleared to blank — it is required.
last_name	body	string	no	Surname (max 255).
email	body	string	no	Email address. Must stay valid and unique per account (case-insensitive). Max 255.
telephone	body	string	no	Phone number (max 50, format-validated; blank allowed).
group_ids	body	array	no	Replaces the contact's group membership with this exact set of group ids.

Request

```
curl -X PATCH -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{ "contact": { "title": "Senior Analyst", "group_ids": [90, 91] } }' \
  https://platform.phishspot.com/api/v1/contacts/cont_abc123
```

> `title`, `department`, and `location` exist on the model but are **not** in the API's permitted params, so they are ignored on create/update via this endpoint — set them through CSV import instead. The example above shows `group_ids`, which is honored; `title` would be silently dropped.

Response `200 OK` — the updated contact, using the same fields as the list endpoint.

```
{
  "id": 501,
  "account_id": 11,
  "first_name": "Ada",
  "last_name": "Kowalska",
  "email": "ada.kowalska@example.com",
  "telephone": "+48 600 123 456",
  "created_at": "2026-05-01T09:30:00.000Z",
  "updated_at": "2026-05-20T08:15:00.000Z",
  "full_name": "Ada Kowalska",
  "groups": [
    { "id": 90, "name": "finance" },
    { "id": 91, "name": "exec" }
  ]
}
```

Status codes

Code	When
200	Contact updated.
400	Body is missing the top-level `contact` key.
404	No contact with that id in any account the token's user belongs to.
422	Validation failed (blank `first_name`, invalid/duplicate `email`, bad `telephone` format). Body is `{ "errors": { … } }`.

`DELETE /contacts/:id`

Permanently deletes a contact and its group memberships, deliverables, events, and results. **Auth:** Bearer; **role:** read (any role).

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Contact id (`cont_…` or integer).

No parameters beyond the bearer token and path id.

Request

```
curl -X DELETE -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/contacts/cont_abc123
```

Response `204 No Content` — empty body.

Status codes

Code	When
204	Contact deleted.
404	No contact with that id in any account the token's user belongs to.

`GET /accounts/:account_id/groups`

Lists every group in the account, ordered by name, with each group's contacts inlined. **Auth:** Bearer; **role:** read (any role).

Parameters

Name	In	Type	Required	Description
account_id	path	string	yes	Account id (`acct_…` or integer).

Response `200 OK` — a JSON array of group objects (see the group fields below).

Field	Type	Description
id	integer	Group primary key.
account_id	integer	Owning account id.
name	string	Group name. For manual groups this is normalized to snake_case (spaces → underscores, non-alphanumerics stripped, lowercased).
contact_count	integer	Cached count of contacts in the group.
created_at	string	ISO 8601 timestamp.
updated_at	string	ISO 8601 timestamp.
contacts	array	Members of the group.
contacts[].id	integer	Contact id.
contacts[].email	string	Contact email.
contacts[].first_name	string	Contact given name.
contacts[].last_name	string|null	Contact surname.
contacts[].full_name	string	`"first_name last_name"` trimmed.

```
[
  {
    "id": 90,
    "account_id": 11,
    "name": "finance",
    "contact_count": 2,
    "created_at": "2026-04-10T11:00:00.000Z",
    "updated_at": "2026-05-20T08:15:00.000Z",
    "contacts": [
      { "id": 501, "email": "ada.kowalska@example.com", "first_name": "Ada",
        "last_name": "Kowalska", "full_name": "Ada Kowalska" }
    ]
  }
]
```

Status codes

Code	When
200	Groups returned (possibly an empty array).
404	`account_id` is not an account the token's user belongs to.

`POST /accounts/:account_id/groups`

Creates a group in the account. Manual group names are normalized to snake_case server-side. **Auth:** Bearer; **role:** read (any role).

Parameters

Body params are wrapped in a `group` object.

Name	In	Type	Required	Description
account_id	path	string	yes	Account id (`acct_…` or integer).
name	body	string	yes	Group name (max 255). Normalized to snake_case; must be unique within the account (case-insensitive, compared after normalization).
description	body	string	no	Free-text description. Permitted by the controller (the model has no `description` column, so it is accepted but not persisted/returned).
contact_ids	body	array	no	Array of contact ids to add as members on creation.

Request

```
curl -X POST -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{ "group": { "name": "Finance Team", "contact_ids": [501, 502] } }' \
  https://platform.phishspot.com/api/v1/accounts/11/groups
```

Response `201 Created` — the created group, using the same fields as the list endpoint above. Note `"Finance Team"` is stored and returned as `"finance_team"`.

```
{
  "id": 90,
  "account_id": 11,
  "name": "finance_team",
  "contact_count": 2,
  "created_at": "2026-04-10T11:00:00.000Z",
  "updated_at": "2026-04-10T11:00:00.000Z",
  "contacts": [
    { "id": 501, "email": "ada.kowalska@example.com", "first_name": "Ada", "last_name":
        "Kowalska", "full_name": "Ada Kowalska" },
    { "id": 502, "email": "jan.nowak@example.com", "first_name": "Jan", "last_name":
        "Nowak", "full_name": "Jan Nowak" }
  ]
}
```

Status codes

Code	When
201	Group created.
400	Body is missing the top-level `group` key.
404	`account_id` is not an account the token's user belongs to.
422	Validation failed (blank `name`, or a name that normalizes to a duplicate within the account). Body is `{ "errors": { … } }`.

`GET /groups/:id`

Fetches a single group by id, scoped to the token user's accounts. **Auth:** Bearer; **role:** read (any role).

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Group id (`grp_…` or integer).

No parameters beyond the bearer token and path id.

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/groups/grp_xyz789
```

Response `200 OK` — the group, using the same fields as the list endpoint.


```
{
  "id": 90,
  "account_id": 11,
  "name": "finance",
  "contact_count": 1,
  "created_at": "2026-04-10T11:00:00.000Z",
  "updated_at": "2026-05-20T08:15:00.000Z",
  "contacts": [
    { "id": 501, "email": "ada.kowalska@example.com", "first_name": "Ada", "last_name":
        "Kowalska", "full_name": "Ada Kowalska" }
  ]
}
```


Status codes

Code	When
200	Group found.
404	No group with that id in any account the token's user belongs to.

`PATCH /groups/:id`

Updates a group's name (and, optionally, replaces its membership). Blocked if the group is locked by an active campaign. **Auth:** Bearer; **role:** read (any role).

Parameters

Body params are wrapped in a `group` object.

Name	In	Type	Required	Description
id	path	string	yes	Group id (`grp_…` or integer).
name	body	string	no	New group name (max 255). Normalized to snake_case; must remain unique within the account.
description	body	string	no	Accepted but not persisted (no model column).
contact_ids	body	array	no	Replaces the group's membership with this exact set of contact ids.

Request

```
curl -X PATCH -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{ "group": { "name": "Finance and Ops", "contact_ids": [501, 503] } }' \
  https://platform.phishspot.com/api/v1/groups/grp_xyz789
```

Response `200 OK` — the updated group, using the same fields as the list endpoint.

```
{
  "id": 90,
  "account_id": 11,
  "name": "finance_and_ops",
  "contact_count": 2,
  "created_at": "2026-04-10T11:00:00.000Z",
  "updated_at": "2026-05-21T10:00:00.000Z",
  "contacts": [
    { "id": 501, "email": "ada.kowalska@example.com", "first_name": "Ada", "last_name":
        "Kowalska", "full_name": "Ada Kowalska" },
    { "id": 503, "email": "ola.wisniewska@example.com", "first_name": "Ola",
        "last_name": "Wisniewska", "full_name": "Ola Wisniewska" }
  ]
}
```

Status codes

Code	When
200	Group updated.
400	Body is missing the top-level `group` key.
403	The group is locked (used in an `in_progress` / `paused` campaign) so it cannot be modified.
404	No group with that id in any account the token's user belongs to.
422	Validation failed (blank `name` or a name that normalizes to a duplicate). Body is `{ "errors": { … } }`.

`DELETE /groups/:id`

Permanently deletes a group and its contact-group / campaign-group associations (the contacts themselves are not deleted). Blocked if the group is locked by an active campaign. **Auth:** Bearer; **role:** read (any role).

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Group id (`grp_…` or integer).

No parameters beyond the bearer token and path id.

Request

```
curl -X DELETE -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/groups/grp_xyz789
```

Response `204 No Content` — empty body.

Status codes

Code	When
204	Group deleted.
403	The group is locked (used in an `in_progress` / `paused` campaign) so it cannot be deleted.
404	No group with that id in any account the token's user belongs to.

`POST /groups/:id/add_contacts`

Adds one or more contacts to a group. Contact ids are resolved against the group's own account — ids that belong to another account, or that don't exist, are silently dropped. Contacts already in the group are skipped. Blocked if the group is locked. **Auth:** Bearer; **role:** read (any role).

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Group id (`grp_…` or integer).
contact_ids	body	array	yes	Contact ids (`cont_…` or integers) to add. Ids outside the group's account or non-existent are ignored.

Request

```
curl -X POST -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{ "contact_ids": [501, "cont_def456"] }' \
  https://platform.phishspot.com/api/v1/groups/grp_xyz789/add_contacts
```

Response `200 OK` — the updated group (after reload), using the same fields as the list endpoint. The response does not include a separate count of how many were added; compare `contact_count` / `contacts` before and after.

```
{
  "id": 90,
  "account_id": 11,
  "name": "finance",
  "contact_count": 2,
  "created_at": "2026-04-10T11:00:00.000Z",
  "updated_at": "2026-05-22T09:00:00.000Z",
  "contacts": [
    { "id": 501, "email": "ada.kowalska@example.com", "first_name": "Ada", "last_name":
        "Kowalska", "full_name": "Ada Kowalska" },
    { "id": 540, "email": "marek.zielinski@example.com", "first_name": "Marek",
        "last_name": "Zielinski", "full_name": "Marek Zielinski" }
  ]
}
```


Status codes

Code	When
200	Returns the updated group (even if every supplied id was dropped/already present — it just won't change).
403	The group is locked (used in an `in_progress` / `paused` campaign).
404	No group with that id in any account the token's user belongs to.

`DELETE /groups/:id/remove_contacts`

Removes one or more contacts from a group. Contact ids are resolved against the group's account; ids not in the group (or outside the account) are ignored. Blocked if the group is locked. **Auth:** Bearer; **role:** read (any role).

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Group id (`grp_…` or integer).
contact_ids	body	array	yes	Contact ids (`cont_…` or integers) to remove from the group.

Request

```
curl -X DELETE -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{ "contact_ids": [540] }' \
  https://platform.phishspot.com/api/v1/groups/grp_xyz789/remove_contacts
```

Response `200 OK` — the updated group (after reload), using the same fields as the list endpoint.

```
{
  "id": 90,
  "account_id": 11,
  "name": "finance",
  "contact_count": 1,
  "created_at": "2026-04-10T11:00:00.000Z",
  "updated_at": "2026-05-22T09:10:00.000Z",
  "contacts": [
    { "id": 501, "email": "ada.kowalska@example.com", "first_name": "Ada", "last_name":
        "Kowalska", "full_name": "Ada Kowalska" }
  ]
}
```


Status codes

Code	When
200	Returns the updated group (ids not in the group are simply ignored).
403	The group is locked (used in an `in_progress` / `paused` campaign).
404	No group with that id in any account the token's user belongs to.

27.7 Deliverables, events & results

These three resources record the per-recipient telemetry of a campaign. A **deliverable** is the join between a campaign and a contact (one row per recipient) and tracks its position in the engagement funnel via `state`. An **event** is an immutable-ish timeline entry (sent, opened, clicked, …) keyed by `genre`. A **result** stores a contact's answer/score for a single e-learning `block`.

All endpoints in this section authorize with the resource's Pundit policy, which permits **any team member** (any role) to read, create, update, and destroy. There is no admin/editor gate. Listing and creation are nested under an account (`/accounts/:account_id/…`); show/update/destroy are shallow (`/deliverables/:id` etc.) and are scoped to accounts the token's user belongs to — requesting a record outside those accounts returns `404`, never another tenant's data.

`GET /api/v1/accounts/:account_id/deliverables`

Lists every deliverable for the account, newest first, optionally filtered to one campaign. Use it to pull the recipient roster and funnel state of a campaign. **Auth:** Bearer; **role:** read (any role).

Parameters

Name	In	Type	Required	Description
account_id	path	string	yes	Account id (`acct_…` or integer).

Name	In	Type	Required	Description
campaign_id	query	string	no	Restrict to one campaign (`camp_…` or integer). When omitted, all deliverables for the account are returned.

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  "https://platform.phishspot.com/api/v1/accounts/11/deliverables?campaign_id=42"
```

Response `200 OK` — JSON array of deliverable objects.

Field	Type	Description
id	integer	Deliverable id.
campaign_id	integer	Owning campaign.
contact_id	integer	Targeted contact.
state	string	Funnel state (see enum below).
user_agent	string | null	User-agent captured on open/click, if any.
ip_address	string | null	IP captured on open/click, if any.
created_at	string	ISO 8601 timestamp.
updated_at	string	ISO 8601 timestamp.
campaign	object | null	Present when the campaign loads: `{ id, name, account_id }`.
contact	object | null	Present when the contact loads: `{ id, email, first_name, last_name, full_name }`.
events	array	Present only when the contact has events in this campaign; each item is `{ id, genre, created_at }`.

`state` is one of: `pending` (not yet sent), `sent`, `delivered`, `bounced`, `opened`, `clicked`, `submitted` (data entered on landing page), `educated` (completed training).

```
[
  {
    "id": 5012,
    "campaign_id": 42,
    "contact_id": 880,
    "state": "clicked",
    "user_agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64)",
    "ip_address": "203.0.113.7",
    "created_at": "2026-05-30T09:12:44.000Z",
    "updated_at": "2026-05-30T10:01:08.000Z",
    "campaign": { "id": 42, "name": "Q2 Invoice Lure", "account_id": 11 },
    "contact": { "id": 880, "email": "jan.kowalski@acme.test", "first_name": "Jan",
        "last_name": "Kowalski", "full_name": "Jan Kowalski" },
    "events": [
      { "id": 9001, "genre": "sent", "created_at": "2026-05-30T09:12:44.000Z" },
      { "id": 9044, "genre": "opened", "created_at": "2026-05-30T09:58:21.000Z" },
      { "id": 9051, "genre": "clicked", "created_at": "2026-05-30T10:01:08.000Z" }
    ]
  }
]
```


Status codes

Code	When
200	Deliverables returned.
403	Token user is not authorized to view the account (`account.show?` denied).
404	`account_id` does not belong to the token user.

`POST /api/v1/accounts/:account_id/deliverables`

Creates a deliverable, linking a contact to a campaign. The `account_id` is derived from the campaign automatically (the supplied `account_id` path segment selects the account context). **Auth:** Bearer; **role:** read (any role).

Parameters

All body params are wrapped in a `deliverable` object.

Name	In	Type	Required	Description
account_id	path	string	yes	Account id (`acct_…` or integer).
deliverable.campaign_id	body	integer	yes	Campaign to attach to. Validated `presence`.
deliverable.contact_id	body	integer	yes	Contact being targeted. Validated `presence`.

Name	In	Type	Required	Description
deliverable.state	body	string	no	Funnel state; defaults to `pending`. One of the `state` enum values. Validated `presence`.
deliverable.name	body	string	no	Accepted by strong params but not persisted (no `name` column).

Request

```
curl -X POST -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{ "deliverable": { "campaign_id": 42, "contact_id": 880, "state": "pending" } }' \
  https://platform.phishspot.com/api/v1/accounts/11/deliverables
```

Response `201 Created` — the created deliverable (same shape as the show/index object above).

```
{
  "id": 5099,
  "campaign_id": 42,
  "contact_id": 880,
  "state": "pending",
  "user_agent": null,
  "ip_address": null,
  "created_at": "2026-06-02T08:00:00.000Z",
  "updated_at": "2026-06-02T08:00:00.000Z",
  "campaign": { "id": 42, "name": "Q2 Invoice Lure", "account_id": 11 },
  "contact": { "id": 880, "email": "jan.kowalski@acme.test", "first_name": "Jan",
        "last_name": "Kowalski", "full_name": "Jan Kowalski" }
}
```

Status codes

Code	When
201	Deliverable created.
404	`account_id` does not belong to the token user.
422	Validation failed (missing `campaign_id` / `contact_id` / `state`, or an invalid `state` value). Body: `{ "errors": { … } }`.

`GET /api/v1/deliverables/:id`

Fetches a single deliverable, including its campaign, contact, and event timeline. **Auth:** Bearer; **role:** read (any role).

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Deliverable id (`delv_…` or integer).

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/deliverables/5012
```

Response `200 OK` — single deliverable object (same fields as the index item above).

```
{
  "id": 5012,
  "campaign_id": 42,
  "contact_id": 880,
  "state": "clicked",
  "user_agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64)",
  "ip_address": "203.0.113.7",
  "created_at": "2026-05-30T09:12:44.000Z",
  "updated_at": "2026-05-30T10:01:08.000Z",
  "campaign": { "id": 42, "name": "Q2 Invoice Lure", "account_id": 11 },
  "contact": { "id": 880, "email": "jan.kowalski@acme.test", "first_name": "Jan",
       "last_name": "Kowalski", "full_name": "Jan Kowalski" },
  "events": [
    { "id": 9001, "genre": "sent", "created_at": "2026-05-30T09:12:44.000Z" }
  ]
}
```

Status codes

Code	When
200	Deliverable returned.
404	No deliverable with that id in an account the token user belongs to.

`PATCH /api/v1/deliverables/:id`

Updates a deliverable — typically to advance its `state` or re-link campaign/contact. **Auth:** Bearer; **role:** read (any role).

Parameters

Body params are wrapped in a `deliverable` object; send only the fields you want to change.

Name	In	Type	Required	Description
id	path	string	yes	Deliverable id (`delv_…` or integer).

Name	In	Type	Required	Description
deliverable.state	body	string	no	New funnel state (one of the `state` enum values).
deliverable.campaign_id	body	integer	no	Reassign campaign (`presence` still enforced — cannot be blanked).
deliverable.contact_id	body	integer	no	Reassign contact (`presence` still enforced).
deliverable.name	body	string	no	Accepted but not persisted (no column).

Request

```
curl -X PATCH -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{ "deliverable": { "state": "submitted" } }' \
  https://platform.phishspot.com/api/v1/deliverables/5012
```

Response `200 OK` — the updated deliverable object (same shape as show).

Status codes

Code	When
200	Deliverable updated.
404	No deliverable with that id in an account the token user belongs to.
422	Validation failed (e.g. invalid `state`, or `campaign_id` / `contact_id` blanked). Body: `{ "errors": { … } }`.

`DELETE /api/v1/deliverables/:id`

Permanently deletes a deliverable (and its dependent `campaign_replies`). **Auth:** Bearer; **role:** read (any role).

No parameters beyond the bearer token and the path id.

Request

```
curl -X DELETE -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/deliverables/5012
```

Response `204 No Content` — empty body on success.

Status codes

Code	When
204	Deliverable deleted.
404	No deliverable with that id in an account the token user belongs to.

`GET /api/v1/accounts/:account_id/events`

Lists the account's events newest-first, with optional filtering by campaign, contact, and genre. Use it to reconstruct an engagement timeline. **Auth:** Bearer; **role:** read (any role).

Parameters

Name	In	Type	Required	Description
account_id	path	string	yes	Account id (`acct_…` or integer).
campaign_id	query	string	no	Restrict to one campaign (`camp_…` or integer).
contact_id	query	string	no	Restrict to one contact (`cont_…` or integer).
genre	query	string	no	Restrict to one genre (see enum below).

`genre` is one of: `sent`, `delivered`, `bounced`, `opened`, `clicked`, `submitted_data` (data submitted on landing page), `started_training`, `completed_training`, `failed_quiz`, `passed_quiz`, `replied` (recipient replied to the phishing email).

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  "https://platform.phishspot.com/api/v1/accounts/11/events?
        campaign_id=42&genre=clicked"
```

Response `200 OK` — JSON array of event objects.

Field	Type	Description
id	integer	Event id.
account_id	integer	Owning account.
campaign_id	integer	Campaign the event belongs to.
contact_id	integer	Contact the event belongs to.
genre	string	Event genre (see enum above).
metadata	object	Free-form JSON (e.g. `ip_address`, `user_agent`, submitted fields, quiz data). Defaults to `{}`.
created_at	string	ISO 8601 timestamp.

Field	Type	Description
updated_at	string	ISO 8601 timestamp.
genre_display_name	string	Humanized genre (e.g. `"Submitted data"`); present only when `genre` is set.
ip_address	string	Convenience copy of `metadata.ip_address`; present only when set.
user_agent	string	Convenience copy of `metadata.user_agent`; present only when set.


```
[
  {
    "id": 9051,
    "account_id": 11,
    "campaign_id": 42,
    "contact_id": 880,
    "genre": "clicked",
    "metadata": { "ip_address": "203.0.113.7", "user_agent": "Mozilla/5.0" },
    "created_at": "2026-05-30T10:01:08.000Z",
    "updated_at": "2026-05-30T10:01:08.000Z",
    "genre_display_name": "Clicked",
    "ip_address": "203.0.113.7",
    "user_agent": "Mozilla/5.0"
  }
]
```


Status codes

Code	When
200	Events returned.
404	`account_id` does not belong to the token user.

`POST /api/v1/accounts/:account_id/events`

Records a new event. The event's `account` is set from the path account, and on save the model overrides it to match the campaign's account. **Auth:** Bearer; **role:** read (any role).

Parameters

Body params are wrapped in an `event` object.

Name	In	Type	Required	Description
account_id	path	string	yes	Account id (`acct_…` or integer).
event.campaign_id	body	integer	yes	Campaign for this event. Validated `presence`.

Name	In	Type	Required	Description
event.contact_id	body	integer	yes	Contact for this event. Validated `presence`.
event.genre	body	string	no	Event genre; defaults to `sent`. One of the `genre` enum values. Validated `presence`.
event.metadata	body	object	no	Arbitrary JSON hash (permitted as `metadata: {}`). Defaults to `{}`.
event.name	body	string	no	Accepted by strong params but not persisted (no `name` column).

Request

```
curl -X POST -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{ "event": { "campaign_id": 42, "contact_id": 880, "genre": "opened", "metadata":
        { "ip_address": "203.0.113.7", "user_agent": "Mozilla/5.0" } } }' \
  https://platform.phishspot.com/api/v1/accounts/11/events
```

Response `201 Created` — the created event object (same shape as the index item above).

```
{
  "id": 9044,
  "account_id": 11,
  "campaign_id": 42,
  "contact_id": 880,
  "genre": "opened",
  "metadata": { "ip_address": "203.0.113.7", "user_agent": "Mozilla/5.0" },
  "created_at": "2026-05-30T09:58:21.000Z",
  "updated_at": "2026-05-30T09:58:21.000Z",
  "genre_display_name": "Opened",
  "ip_address": "203.0.113.7",
  "user_agent": "Mozilla/5.0"
}
```

Status codes

Code	When
201	Event created.
404	`account_id` does not belong to the token user.
422	Validation failed (missing `campaign_id` / `contact_id` / `genre`, or an invalid `genre`). Body: `{ "errors": { … } }`.

`GET /api/v1/events/:id`

Fetches a single event. **Auth:** Bearer; **role:** read (any role).

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Event id (`evt_…` or integer).

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/events/9044
```

Response `200 OK` — single event object (same fields as the index item above).

```
{
  "id": 9044,
  "account_id": 11,
  "campaign_id": 42,
  "contact_id": 880,
  "genre": "opened",
  "metadata": { "ip_address": "203.0.113.7" },
  "created_at": "2026-05-30T09:58:21.000Z",
  "updated_at": "2026-05-30T09:58:21.000Z",
  "genre_display_name": "Opened",
  "ip_address": "203.0.113.7"
}
```

Status codes

Code	When
200	Event returned.
404	No event with that id in an account the token user belongs to. Body: `{ "error": "Event not found" }`.

`PATCH /api/v1/events/:id`

Updates an event's genre or metadata. **Auth:** Bearer; **role:** read (any role).

Parameters

Body params are wrapped in an `event` object; send only the fields you want to change.

Name	In	Type	Required	Description
id	path	string	yes	Event id (`evt_…` or integer).
event.genre	body	string	no	New genre (one of the `genre` enum values; `presence` still enforced).
event.metadata	body	object	no	Replacement metadata hash.

Name	In	Type	Required	Description
event.campaign_id	body	integer	no	Reassign campaign (`presence` enforced).
event.contact_id	body	integer	no	Reassign contact (`presence` enforced).
event.name	body	string	no	Accepted but not persisted.

Request

```
curl -X PATCH -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{ "event": { "metadata": { "note": "manual correction" } } }' \
  https://platform.phishspot.com/api/v1/events/9044
```

Response `200 OK` — the updated event object (same shape as show).

Status codes

Code	When
200	Event updated.
404	No event with that id in an account the token user belongs to. Body: `{ "error": "Event not found" }`.
422	Validation failed (e.g. invalid/blank `genre`). Body: `{ "errors": { … } }`.

`DELETE /api/v1/events/:id`

Permanently deletes an event. **Auth:** Bearer; **role:** read (any role).

No parameters beyond the bearer token and the path id.

Request

```
curl -X DELETE -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/events/9044
```

Response `204 No Content` — empty body on success.

Status codes

Code	When
204	Event deleted.
404	No event with that id in an account the token user belongs to. Body: `{ "error": "Event not found" }`.

`GET /api/v1/accounts/:account_id/results`

Lists the account's e-learning results newest-first. There are no query filters on this endpoint. **Auth:** Bearer; **role:** read (any role).

Parameters

Name	In	Type	Required	Description
account_id	path	string	yes	Account id (`acct_…` or integer).

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/accounts/11/results
```

Response `200 OK` — JSON array of result objects.

Field	Type	Description
id	integer	Result id.
block_id	integer	Course block this result is for.
contact_id	integer	Contact who produced the result.
account_id	integer	Owning account.
metadata	object	Free-form JSON (e.g. `answer`, `correct`, `score`, `time_spent`, `completed`). Defaults to `{}`.
created_at	string	ISO 8601 timestamp.
updated_at	string	ISO 8601 timestamp.
block	object | null	Present when the block loads: `{ id, name }`.
contact	object | null	Present when the contact loads: `{ id, email, first_name, last_name, full_name }`.

```
[
  {
    "id": 7100,
    "block_id": 320,
    "contact_id": 880,
    "account_id": 11,
    "metadata": { "answer": "B", "correct": true, "score": 100, "completed": true },
    "created_at": "2026-05-31T14:20:00.000Z",
    "updated_at": "2026-05-31T14:20:00.000Z",
    "block": { "id": 320, "name": "Spot the Lookalike Domain" },
    "contact": { "id": 880, "email": "jan.kowalski@acme.test", "first_name": "Jan",
        "last_name": "Kowalski", "full_name": "Jan Kowalski" }
  }
]
```

Status codes

Code	When
200	Results returned.
403	Token user is not authorized to view the account (`account.show?` denied).
404	`account_id` does not belong to the token user.

`POST /api/v1/accounts/:account_id/results`

Records a contact's result for a course block. The `account` is derived from the block automatically. **Auth:** Bearer; **role:** read (any role).

Parameters

Body params are wrapped in a `result` object.

Name	In	Type	Required	Description
account_id	path	string	yes	Account id (`acct_…` or integer).
result.block_id	body	integer	yes	Course block this result is for. Validated `presence`.
result.contact_id	body	integer	yes	Contact producing the result. Validated `presence`.
result.metadata	body	object	no	JSON hash of answer/score/completion data. Defaults to `{}`. Permitted as a scalar param (`:metadata`), so send it as a JSON object value.
result.name	body	string	no	Accepted by strong params but not persisted (no `name` column).
result.state	body	string	no	Accepted by strong params but not persisted (Result has no `state` column/enum).

Request

```
curl -X POST -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{ "result": { "block_id": 320, "contact_id": 880, "metadata": { "answer": "B",
        "correct": true, "score": 100, "completed": true } } }' \
  https://platform.phishspot.com/api/v1/accounts/11/results
```

Response `201 Created` — the created result object (same shape as the index item above).

```
{
  "id": 7150,
  "block_id": 320,
  "contact_id": 880,
  "account_id": 11,
  "metadata": { "answer": "B", "correct": true, "score": 100, "completed": true },
  "created_at": "2026-06-02T08:30:00.000Z",
  "updated_at": "2026-06-02T08:30:00.000Z",
  "block": { "id": 320, "name": "Spot the Lookalike Domain" },
  "contact": { "id": 880, "email": "jan.kowalski@acme.test", "first_name": "Jan",
        "last_name": "Kowalski", "full_name": "Jan Kowalski" }
}
```


Status codes

Code	When
201	Result created.
404	`account_id` does not belong to the token user.
422	Validation failed (missing `block_id` / `contact_id`). Body: `{ "errors": { … } }`.

`GET /api/v1/results/:id`

Fetches a single result. **Auth:** Bearer; **role:** read (any role).

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Result id (`res_…` or integer).

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/results/7100
```

Response `200 OK` — single result object (same fields as the index item above).

```
{
  "id": 7100,
  "block_id": 320,
  "contact_id": 880,
  "account_id": 11,
  "metadata": { "answer": "B", "correct": true, "score": 100 },
  "created_at": "2026-05-31T14:20:00.000Z",
  "updated_at": "2026-05-31T14:20:00.000Z",
  "block": { "id": 320, "name": "Spot the Lookalike Domain" },
  "contact": { "id": 880, "email": "jan.kowalski@acme.test", "first_name": "Jan",
        "last_name": "Kowalski", "full_name": "Jan Kowalski" }
}
```


Status codes

Code	When
200	Result returned.
404	No result with that id in an account the token user belongs to.

`PATCH /api/v1/results/:id`

Updates a result, typically to revise its metadata (answer, score, completion). **Auth:** Bearer; **role:** read (any role).

Parameters

Body params are wrapped in a `result` object; send only the fields you want to change.

Name	In	Type	Required	Description
id	path	string	yes	Result id (`res_…` or integer).
result.metadata	body	object	no	Replacement metadata hash.
result.block_id	body	integer	no	Reassign block (`presence` enforced).
result.contact_id	body	integer	no	Reassign contact (`presence` enforced).
result.name	body	string	no	Accepted but not persisted.
result.state	body	string	no	Accepted but not persisted.

Request

```
curl -X PATCH -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{ "result": { "metadata": { "answer": "C", "correct": false, "score": 0 } } }' \
  https://platform.phishspot.com/api/v1/results/7100
```

Response `200 OK` — the updated result object (same shape as show).

Status codes

Code	When
200	Result updated.
404	No result with that id in an account the token user belongs to.
422	Validation failed (e.g. `block_id` / `contact_id` blanked). Body: `{ "errors": { … } }`.

`DELETE /api/v1/results/:id`

Permanently deletes a result. **Auth:** Bearer; **role:** read (any role).

No parameters beyond the bearer token and the path id.

Request

```
curl -X DELETE -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/results/7100
```

Response `204 No Content` — empty body on success.

Status codes

Code	When
204	Result deleted.
404	No result with that id in an account the token user belongs to.

27.8 Account trends

Aggregated phishing-susceptibility analytics for an account: one data point per delivered campaign over a date range, plus a roll-up summary. Backed by `Campaigns::TrendService`, which only counts campaigns whose state is `in_progress` or `done` (the `delivered` scope) created within the selected range.

`GET /accounts/:account_id/trends`

Returns susceptibility metrics for an account's delivered campaigns, ordered oldest-first, together with a summary (campaign count, average click rate, trend direction, and the most-clicked recipient group). Use it to power a trend dashboard or to track whether employees are getting better or worse at spotting phishing over time. **Auth:** Bearer; **role:** read (any role — any member of the account).

Parameters

Name	In	Type	Required	Description
account_id	path	string	yes	Account id (`acct_…` or integer). Must be an account the token's user belongs to.
start_date	query	string	no	Start of a custom range, `YYYY-MM-DD`. Parsed to the beginning of that day. If present (with or without `end_date`), the custom range takes precedence over `range`. When omitted but `end_date` is given, defaults to 1 year ago.
end_date	query	string	no	End of a custom range, `YYYY-MM-DD`. Parsed to the end of that day. When omitted but `start_date` is given, defaults to now.
range	query	string	no	Preset range, used only when neither `start_date` nor `end_date` is present. One of `30d`, `90d`, `6m`, `all`. Any other/absent value (including the default) yields the last 1 year. `all` spans from the Unix epoch to now.

> `start_date` / `end_date` form a custom range that **overrides** `range`. Dates are inclusive (start = beginning of day, end = end of day). Invalid dates return `422` (see below).

Request

```
curl -X GET -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  "https://platform.phishspot.com/api/v1/accounts/11/trends?
        start_date=2026-01-01&end_date=2026-06-01"
```

Response `200 OK` — an object with a `campaigns` array (one entry per delivered campaign in range) and a `summary` object.

Field	Type	Description
campaigns	array	Delivered campaigns in the range, ordered by `created_at` ascending. Each element is a data point (fields below). Empty array if none.
campaigns[].id	integer	Campaign id (raw integer).
campaigns[].name	string	Campaign name.
campaigns[].date	string	Date the campaign ran, `YYYY-MM-DD` (ISO 8601 date). Uses `scheduled_at` if set, otherwise `created_at`.

Field	Type	Description
campaigns[].open_rate	number (float)	Percent of recipients who opened the email (0.0 when no data).
campaigns[].click_rate	number (float)	Percent of recipients who clicked a link (0.0 when no data).
campaigns[].submit_rate	number (float)	Percent of recipients who submitted data on the landing page (0.0 when no data).
campaigns[].total_sent	integer	Number of recipients the campaign was sent to.
summary	object	Roll-up across the campaigns above (fields below).
summary.total_campaigns	integer	Count of delivered campaigns in the range (`0` when none).
summary.avg_click_rate	number (float)	Mean of the per-campaign `click_rate` values, rounded to 1 decimal (`0.0` when none).
summary.trend_direction	string	One of `improving`, `worsening`, `stable`, or `neutral`. Compares the average click rate of the most recent up-to-3 campaigns against the earliest up-to-3; `neutral` when fewer than 2 campaigns, `stable` when the change is within ±1.0 percentage point.
summary.most_vulnerable_group	string | null	Name of the recipient group with the highest click-to-sent ratio across the range; `null` when no group data is available.

```
{
  "campaigns": [
    {
      "id": 42,
      "name": "Q1 Invoice Lure",
      "date": "2026-01-14",
      "open_rate": 61.5,
      "click_rate": 23.1,
      "submit_rate": 7.7,
      "total_sent": 130
    },
    {
      "id": 57,
      "name": "Password Expiry Notice",
      "date": "2026-04-02",
      "open_rate": 54.0,
      "click_rate": 12.0,
      "submit_rate": 4.0,
      "total_sent": 150
    }
  ],
  "summary": {
    "total_campaigns": 2,
    "avg_click_rate": 17.6,
    "trend_direction": "improving",
    "most_vulnerable_group": "Finance"
  }
}
```


Status codes

Code	When
200	Trend data returned (the `campaigns` array and `summary` are present even when there are no matching campaigns).
404	`account_id` does not belong to the token's user (`{"error":"Account not found"}`).
422	`start_date` or `end_date` is not a parseable `YYYY-MM-DD` date (`{"error":"Invalid date; use YYYY-MM-DD."}`).

27.9 Courses & blocks

E-learning courses delivered to employees who fall for a phishing simulation. A **course** is an ordered collection of **blocks** (text, HTML, video, quiz, etc.). Courses are either owned by your account or **global** (curated, shared library). Blocks are nested under a course for listing/creation but addressable by their own shallow id for show/update/destroy.

> All endpoints in this section authorize through the per-record policy, which grants read **and** write to **any account role** (member, editor, admin). The only write restrictions are: you cannot modify a **global** course/block that your account does not own (`403`), and you cannot update/delete a course or block that is **locked** by an in-progress or paused campaign.

`GET /api/v1/accounts/:account_id/courses`

Lists every course available to the account — courses your account owns plus all `global: true` courses — ordered by name, with a lightweight block summary inlined. **Auth:** Bearer; **role:** read (any role).

Parameters

Name	In	Type	Required	Description
account_id	path	string	yes	Account id (`acct_…` or integer). Must be an account the token user belongs to.

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/accounts/11/courses
```

Response `200 OK` — JSON array of course objects (see the course fields below).

Field	Type	Description
id	integer	Course id.
account_id	integer	Owning account id.
name	string	Course name.
description	string	Course description.
global	boolean	`true` for shared/curated library courses, `false` for account-owned.
created_at	string	ISO 8601 timestamp.
updated_at	string	ISO 8601 timestamp.
blocks	array	Inlined block summaries (see fields below). Ordered as stored on the association.
blocks[].id	integer	Block id.
blocks[].name	string	Block name.
blocks[].order	integer	Zero-based position within the course.
blocks[].genre	string	Block genre (see enum under block endpoints).

```
[
  {
    "id": 7,
    "account_id": 11,
    "name": "Spotting Spoofed Senders",
    "description": "A short course on recognising display-name and domain spoofing.",
    "global": false,
    "created_at": "2026-05-01T09:12:00.000Z",
    "updated_at": "2026-05-14T16:40:11.000Z",
    "blocks": [
      { "id": 31, "name": "Intro", "order": 0, "genre": "html" },
      { "id": 32, "name": "Quick check", "order": 1, "genre": "quiz" }
    ]
  }
]
```


Status codes

Code	When
200	Courses returned.
404	`account_id` is not an account the token user belongs to.

`POST /api/v1/accounts/:account_id/courses`

Creates a new account-owned course. The course is always created under the path account (`global` cannot be set — new courses are private). Both `name` and `description` are required by the model. **Auth:** Bearer; **role:** read (any role — all team members can create courses).

Parameters

Body params are wrapped in a `course` object.

Name	In	Type	Required	Description
account_id	path	string	yes	Account id (`acct_…` or integer).
course	body	object	yes	Wrapper object holding the fields below.
course.name	body	string	yes	Course name. Presence-validated.
course.description	body	string	yes	Course description. Presence-validated.

Request

```
curl -X POST -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{ "course": { "name": "Invoice Fraud 101", "description": "Recognising fake
        supplier invoices." } }' \
  https://platform.phishspot.com/api/v1/accounts/11/courses
```

Response `201 Created` — the created course, same shape as a list item (with an empty `blocks` array).

Field	Type	Description
id	integer	New course id.
account_id	integer	Owning account id (the path account).
name	string	Course name.
description	string	Course description.
global	boolean	Always `false` for newly created courses.
created_at	string	ISO 8601 timestamp.
updated_at	string	ISO 8601 timestamp.
blocks	array	Empty on creation.


```
{
  "id": 19,
  "account_id": 11,
  "name": "Invoice Fraud 101",
  "description": "Recognising fake supplier invoices.",
  "global": false,
  "created_at": "2026-06-02T10:00:00.000Z",
  "updated_at": "2026-06-02T10:00:00.000Z",
  "blocks": []
}
```


Status codes

Code	When
201	Course created.
404	`account_id` is not an account the token user belongs to.
422	Validation failed — e.g. `name` or `description` blank. Body: `{"errors": {"name": ["can't be blank"]}}`.

`GET /api/v1/courses/:id`

Fetches a single course by its shallow id. Resolvable for courses your account owns or any `global` course; a course belonging to another tenant returns `404`. **Auth:** Bearer; **role:** read (any role).

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Course id (`course_…` or integer).

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/courses/7
```

Response `200 OK` — one course object, same fields as a list item (including the inlined `blocks` summary).

Field	Type	Description
id	integer	Course id.
account_id	integer	Owning account id.
name	string	Course name.
description	string	Course description.
global	boolean	Whether the course is from the shared library.
created_at	string	ISO 8601 timestamp.
updated_at	string	ISO 8601 timestamp.
blocks	array	Inlined block summaries: `id`, `name`, `order`, `genre`.

```
{
  "id": 7,
  "account_id": 11,
  "name": "Spotting Spoofed Senders",
  "description": "A short course on recognising display-name and domain spoofing.",
  "global": false,
  "created_at": "2026-05-01T09:12:00.000Z",
  "updated_at": "2026-05-14T16:40:11.000Z",
  "blocks": [
    { "id": 31, "name": "Intro", "order": 0, "genre": "html" },
    { "id": 32, "name": "Quick check", "order": 1, "genre": "quiz" }
  ]
}
```

Status codes

Code	When
200	Course returned.
404	No course with that id is owned by your account and it is not global.

`PATCH /api/v1/courses/:id`

Updates a course's `name` and/or `description`. **Auth:** Bearer; **role:** read (any role), subject to the global-ownership and lock checks below.

Parameters

Body params are wrapped in a `course` object. Only `name` and `description` are permitted.

Name	In	Type	Required	Description
id	path	string	yes	Course id (`course_…` or integer).
course	body	object	yes	Wrapper object holding the fields below.
course.name	body	string	no	New name. Cannot be blank if supplied (presence-validated).
course.description	body	string	no	New description. Cannot be blank if supplied (presence-validated).

Request

```
curl -X PATCH -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{ "course": { "description": "Updated for the 2026 supplier-fraud wave." } }' \
  https://platform.phishspot.com/api/v1/courses/7
```

Response `200 OK` — the updated course (same shape as `GET /courses/:id`).

```
{
  "id": 7,
  "account_id": 11,
  "name": "Spotting Spoofed Senders",
  "description": "Updated for the 2026 supplier-fraud wave.",
  "global": false,
  "created_at": "2026-05-01T09:12:00.000Z",
  "updated_at": "2026-06-02T10:05:00.000Z",
  "blocks": [
    { "id": 31, "name": "Intro", "order": 0, "genre": "html" }
  ]
}
```

Status codes

Code	When
200	Course updated.
403	The course is `global` and not owned by your account, **or** it is locked by an in-progress/paused campaign.
404	Course not reachable by your account (not owned and not global).
422	Validation failed — e.g. `name` / `description` set to blank. Body: `{"errors": {...}}`.

`DELETE /api/v1/courses/:id`

Deletes a course and (via `dependent: :destroy`) all of its blocks. **Auth:** Bearer; **role:** read (any role), subject to the global-ownership and lock checks below.

Parameters

No parameters beyond the bearer token and the path id.

Name	In	Type	Required	Description
id	path	string	yes	Course id (`course_…` or integer).

Request

```
curl -X DELETE -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/courses/19
```

Response `204 No Content` — empty body on success.

Status codes

Code	When
204	Course deleted.
403	The course is `global` and not owned by your account, **or** it is locked by an in-progress/paused campaign.
404	Course not reachable by your account.

`GET /api/v1/courses/:course_id/blocks`

Lists the blocks of a course, ordered by `order` (ascending), scoped through Pundit to blocks of accessible (owned or global) courses. **Auth:** Bearer; **role:** read (any role).

Parameters

Name	In	Type	Required	Description
course_id	path	string	yes	Course id (`course_…` or integer). Must be owned by your account or global.

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/courses/7/blocks
```

Response `200 OK` — JSON array of full block objects (fields below).

Field	Type	Description
id	integer	Block id.
name	string	Block name.
course_id	integer	Parent course id.
order	integer	Zero-based position within the course.
genre	string	One of: `text`, `html`, `image`, `video`, `quiz`, `interactive`, `code`, `file_download`.
metadata	object	Free-form JSON. For quiz blocks this holds the question/answers payload.
created_at	string	ISO 8601 timestamp.
updated_at	string	ISO 8601 timestamp.
html_data	string	Rendered rich-text HTML. **Present only when** the block has ActionText content.
locked	boolean	`true` when an associated campaign is in progress (block cannot be updated/deleted).
quiz_question	string	**Quiz blocks only.** The parsed question text.
quiz_answers	array	**Quiz blocks only.** Array of answer hashes parsed from `metadata`.
url	string	Canonical API URL for this block (`/api/v1/blocks/:id`).

```
[
  {
    "id": 31,
    "name": "Intro",
    "course_id": 7,
    "order": 0,
    "genre": "html",
    "metadata": {},
    "created_at": "2026-05-01T09:12:00.000Z",
    "updated_at": "2026-05-01T09:12:00.000Z",
    "html_data": "<div>Welcome to the course.</div>",
    "locked": false,
    "url": "https://platform.phishspot.com/api/v1/blocks/31"
  },
  {
    "id": 32,
    "name": "Quick check",
    "course_id": 7,
    "order": 1,
    "genre": "quiz",
    "metadata": [
      { "question_text": "Which sender is spoofed?" },
      { "answer_text": "no-reply@paypa1.com", "right_answer": "on" },
      { "answer_text": "no-reply@paypal.com" }
    ],
    "created_at": "2026-05-01T09:13:00.000Z",
    "updated_at": "2026-05-01T09:13:00.000Z",
    "locked": false,
    "quiz_question": "Which sender is spoofed?",
    "quiz_answers": [
      { "answer_text": "no-reply@paypa1.com", "right_answer": "on" },
      { "answer_text": "no-reply@paypal.com" }
    ],
    "url": "https://platform.phishspot.com/api/v1/blocks/32"
  }
]
```


Status codes

Code	When
200	Blocks returned (empty array if the course has none).
404	`course_id` not reachable by your account (not owned and not global).

`POST /api/v1/courses/:course_id/blocks`

Adds a block to a course. The block's `account` is set automatically from the course; if `order` is omitted it is auto-assigned to the end of the course. Content requirements vary by `genre` (see below). **Auth:** Bearer; **role:** read (any role), but you cannot add blocks to a global course you don't own (`403`).

Parameters

Body params are wrapped in a `block` object.

Name	In	Type	Required	Description
course_id	path	string	yes	Course id (`course_…` or integer).
block	body	object	yes	Wrapper object holding the fields below.
block.name	body	string	yes	Block name. Presence-validated.
block.genre	body	string	yes	One of `text` , `html` , `image` , `video` , `quiz` , `interactive` , `code` , `file_download` . Defaults to `text` if omitted.
block.body	body	string	conditional	Plain-text/markdown body. **Required unless** genre is `quiz` , `html` , `video` , or `file_download` . For `video` / `file_download` it serves as an optional description.
block.html_data	body	string	conditional	Rich HTML content (ActionText). For `html` / `text` blocks, supply either `body` or `html_data` .
block.metadata	body	object/array	conditional	Free-form JSON. **Required for `quiz` blocks**, where it carries the question and answers (see quiz constraints).
block.order	body	integer	no	Position within the course (integer ≥ 0). Auto-assigned to the end if omitted.
block.course_id	body	integer	no	Permitted but normally redundant with the path `course_id` .
block.video_file	body	file	conditional	Video attachment. **Required for `video` blocks.** Must be `video/mp4` or `video/webm` , ≤ 300 MB, ≤ 1920×1080, ≤ 600 s, video codec h264/vp8/vp9, audio codec aac/opus. Blank string values are ignored.
block.document_file	body	file	conditional	Document attachment. **Required for `file_download` blocks.** Any format, ≤ 100 MB. Blank string values are ignored.

`quiz` blocks must have between **2 and 6** answers in `metadata`, and **at least one answer must be marked correct** (an answer is correct when its `right_answer` / `correct` value is `true`, `"on"`, `"true"`, `"1"`, or `"yes"`). Otherwise the create fails with `422`.

Request

```
curl -X POST -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{
        "block": {
          "name": "Identify the phish",
          "genre": "quiz",
          "metadata": [
            { "question_text": "Which sender is spoofed?" },
            { "answer_text": "no-reply@paypa1.com", "right_answer": "on" },
            { "answer_text": "no-reply@paypal.com" }
          ]
        }
      }' \
  https://platform.phishspot.com/api/v1/courses/7/blocks
```

Response `201 Created` — the created block (same shape as a list item).

```
{
  "id": 33,
  "name": "Identify the phish",
  "course_id": 7,
  "order": 2,
  "genre": "quiz",
  "metadata": [
    { "question_text": "Which sender is spoofed?" },
    { "answer_text": "no-reply@paypa1.com", "right_answer": "on" },
    { "answer_text": "no-reply@paypal.com" }
  ],
  "created_at": "2026-06-02T10:10:00.000Z",
  "updated_at": "2026-06-02T10:10:00.000Z",
  "locked": false,
  "quiz_question": "Which sender is spoofed?",
  "quiz_answers": [
    { "answer_text": "no-reply@paypa1.com", "right_answer": "on" },
    { "answer_text": "no-reply@paypal.com" }
  ],
  "url": "https://platform.phishspot.com/api/v1/blocks/33"
}
```

Status codes

Code	When
201	Block created.

Code	When
403	The parent course is `global` and not owned by your account.
404	`course_id` not reachable by your account.
422	Validation failed — missing `name` / `genre`, missing required content for the genre (`body`, `html_data`, `metadata`, `video_file`, or `document_file`), too few/too many quiz answers, no correct quiz answer, or an unacceptable video/document file. Body: `{"errors": {...}}`.

`GET /api/v1/blocks/:id`

Fetches a single block by its shallow id, scoped to blocks of courses your account can reach (owned or global). **Auth:** Bearer; **role:** read (any role).

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Block id (`blk_…` or integer).

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/blocks/32
```

Response `200 OK` — one block object (full fields as in the blocks list).

Field	Type	Description
id	integer	Block id.
name	string	Block name.
course_id	integer	Parent course id.
order	integer	Zero-based position.
genre	string	Block genre (see enum above).
metadata	object/array	Free-form JSON; quiz payload for quiz blocks.
created_at	string	ISO 8601 timestamp.
updated_at	string	ISO 8601 timestamp.
html_data	string	Rendered rich text. Present only when set.
locked	boolean	`true` when an associated campaign is in progress.
quiz_question	string	Quiz blocks only.
quiz_answers	array	Quiz blocks only.

Field	Type	Description
url	string	Canonical API URL for this block.


```
{
  "id": 32,
  "name": "Quick check",
  "course_id": 7,
  "order": 1,
  "genre": "quiz",
  "metadata": [
    { "question_text": "Which sender is spoofed?" },
    { "answer_text": "no-reply@paypa1.com", "right_answer": "on" },
    { "answer_text": "no-reply@paypal.com" }
  ],
  "created_at": "2026-05-01T09:13:00.000Z",
  "updated_at": "2026-05-01T09:13:00.000Z",
  "locked": false,
  "quiz_question": "Which sender is spoofed?",
  "quiz_answers": [
    { "answer_text": "no-reply@paypa1.com", "right_answer": "on" },
    { "answer_text": "no-reply@paypal.com" }
  ],
  "url": "https://platform.phishspot.com/api/v1/blocks/32"
}
```


Status codes

Code	When
200	Block returned.
404	Block not reachable by your account (its course is neither owned nor global).

`PATCH /api/v1/blocks/:id`

Updates a block. Same permitted fields and genre-specific content rules as create. **Auth:** Bearer; **role:** read (any role), but you cannot edit a block on a global course you don't own (`403`), and a `locked?` block (campaign in progress) raises a not-destroyed error during the update.

Parameters

Body params are wrapped in a `block` object. Permitted keys: `name` , `body` , `course_id` , `order` , `genre` , `metadata` , `html_data` , `video_file` , `document_file` .

Name	In	Type	Required	Description
id	path	string	yes	Block id (`blk_…` or integer).
block	body	object	yes	Wrapper object holding any of the permitted fields.

Name	In	Type	Required	Description
block.name	body	string	no	New name (presence-validated if supplied).
block.genre	body	string	no	Change genre; same enum as create. Changing genre may make other fields required.
block.body	body	string	no	Body text. Required unless genre is `quiz` / `html` / `video` / `file_download`.
block.html_data	body	string	no	Rich HTML content.
block.metadata	body	object/array	no	Free-form JSON; quiz payload (2–6 answers, ≥1 correct).
block.order	body	integer	no	New position (integer ≥ 0).
block.video_file	body	file	no	Replacement video (same constraints as create). Blank strings ignored.
block.document_file	body	file	no	Replacement document (≤ 100 MB). Blank strings ignored.

Request

```
curl -X PATCH -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{ "block": { "name": "Identify the phishing sender", "order": 1 } }' \
  https://platform.phishspot.com/api/v1/blocks/32
```

Response `200 OK` — the updated block (same shape as `GET /blocks/:id`).

```
{
  "id": 32,
  "name": "Identify the phishing sender",
  "course_id": 7,
  "order": 1,
  "genre": "quiz",
  "metadata": [
    { "question_text": "Which sender is spoofed?" },
    { "answer_text": "no-reply@paypa1.com", "right_answer": "on" },
    { "answer_text": "no-reply@paypal.com" }
  ],
  "created_at": "2026-05-01T09:13:00.000Z",
  "updated_at": "2026-06-02T10:15:00.000Z",
  "locked": false,
  "quiz_question": "Which sender is spoofed?",
  "quiz_answers": [
    { "answer_text": "no-reply@paypa1.com", "right_answer": "on" },
    { "answer_text": "no-reply@paypal.com" }
  ],
  "url": "https://platform.phishspot.com/api/v1/blocks/32"
}
```


Status codes

Code	When
200	Block updated.
403	The block's course is `global` and not owned by your account.
404	Block not reachable by your account.
422	Validation failed — blank `name`, missing genre-required content, invalid quiz answers, unacceptable file, **or** the block is locked by a campaign in progress. Body: `{"errors": {...}}`.

`DELETE /api/v1/blocks/:id`

Deletes a block. The controller first checks whether the block is locked by an in-progress campaign and refuses with `422` if so. **Auth:** Bearer; **role:** read (any role), but you cannot delete a block on a global course you don't own (`403`).

Parameters

No parameters beyond the bearer token and the path id.

Name	In	Type	Required	Description
id	path	string	yes	Block id (`blk_…` or integer).

Request

```
curl -X DELETE -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/blocks/33
```

Response `204 No Content` — empty body on success.

Status codes

Code	When
204	Block deleted.
403	The block's course is `global` and not owned by your account.
404	Block not reachable by your account.
422	The block is locked (a connected campaign is in progress). Body: `{"errors": ["Cannot delete block because connected campaign is in progress"]}`.

27.10 Autopilots

Autopilots are recurring, hands-off phishing programs: you describe an audience and a cadence, and the platform keeps generating and delivering campaigns automatically until you stop it. An autopilot is created as a `draft`, then driven through a small lifecycle (`draft → running ⇄ paused → stopped`) via the dedicated member actions below.

> Starting an autopilot activates a **live, sending program** — the platform will begin generating and delivering real phishing campaigns to the targeted members at the configured cadence. Treat `POST /autopilots/:id/start` as a go-live action, not a dry run.

A few model facts referenced throughout this section:

- **State** (`state`): one of `draft`, `running`, `paused`, `stopped`.
- **Intensity period** (`intensity_period`): one of `day`, `week`, `month`, `year`.
- **Duration kind** (`duration_kind`): `continuous` (runs forever) or `until_date` (stops on `ends_on`).
- **End action type** (`end_action_type`): one of `nothing`, `redirect_to_course`, `message_page`, `redirect_to_url` — controls what a target sees after the simulation.
- **Daily-rate cap**: the effective send rate is `intensity_count / period_in_days` and must not exceed **2 campaigns/day** (`day`=1, `week`=7, `month`=30, `year`=365 days per period). Exceeding it fails validation on `intensity_count`.
- **Editable**: an autopilot is editable unless it is `stopped`. Once `stopped`, only read and delete remain available.

`GET /accounts/:account_id/autopilots`

Lists every autopilot belonging to an account, newest first. Use it to render an autopilot dashboard or to find an autopilot's id before acting on it. **Auth:** Bearer; **role:** read (any role — member, editor, or admin).

Parameters

Name	In	Type	Required	Description
account_id	path	string	yes	Account id (`acct_…` or integer).
state	query	string	no	Filter to a single state. One of `draft`, `running`, `paused`, `stopped`. Any other value returns `422`.

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  "https://platform.phishspot.com/api/v1/accounts/11/autopilots?state=running"
```

Response `200 OK` — a JSON array of autopilot objects (each identical to the single-autopilot response below).

Field	Type	Description
id	integer	Autopilot id.
account_id	integer	Owning account id.
name	string	Display name.
state	string	`draft` | `running` | `paused` | `stopped`.
all_groups	boolean	Whether the autopilot targets all groups (vs. the listed `groups`).
intensity_count	integer	Campaigns per `intensity_period`.
intensity_period	string	`day` | `week` | `month` | `year`.
duration_kind	string	`continuous` | `until_date`.
ai_optimizer_enabled	boolean	Whether AI optimization is on.
auto_include_new_members	boolean	Whether new members are auto-enrolled.
language	string	Target language code (e.g. `en`, `pl`).
end_action_type	string	`nothing` | `redirect_to_course` | `message_page` | `redirect_to_url`.
end_action_url	string | null	Redirect URL (used when `end_action_type` is `redirect_to_url`).
created_at	string	ISO-8601 timestamp.
updated_at	string	ISO-8601 timestamp.
ends_on	string | null	ISO-8601 date the program stops (when `duration_kind` is `until_date`), else `null`.
started_at	string | null	ISO-8601 timestamp of first start, else `null`.

Field	Type	Description
daily_rate	number	Effective campaigns/day, rounded to 2 decimals.
progress_percentage	integer | null	Integer % of expected campaigns delivered this period; `null` for `draft` / `stopped`.
course_id	integer | null	Linked e-learning course id, else `null`.
editable	boolean	`false` only when `state` is `stopped`.
groups	array	Targeted groups. Each: `{ "id": integer, "name": string }`.
recent_campaigns	array	Up to 10 most recent generated campaigns, newest first. Each: `{ "id": integer, "name": string, "state": string }`.


```
[
  {
    "id": 7,
    "account_id": 11,
    "name": "Finance team — quarterly drip",
    "state": "running",
    "all_groups": false,
    "intensity_count": 2,
    "intensity_period": "month",
    "duration_kind": "continuous",
    "ai_optimizer_enabled": true,
    "auto_include_new_members": true,
    "language": "en",
    "end_action_type": "redirect_to_course",
    "end_action_url": null,
    "created_at": "2026-05-01T09:00:00Z",
    "updated_at": "2026-06-01T12:30:00Z",
    "ends_on": null,
    "started_at": "2026-05-02T08:00:00Z",
    "daily_rate": 0.07,
    "progress_percentage": 88,
    "course_id": 14,
    "editable": true,
    "groups": [
      { "id": 3, "name": "Finance" }
    ],
    "recent_campaigns": [
      { "id": 102, "name": "Invoice approval — May", "state": "completed" }
    ]
  }
]
```


Status codes

Code	When
200	List returned (possibly empty).
404	The `account_id` does not exist or the token's user is not a member of it.
422	`state` query param is present but not one of `draft`, `running`, `paused`, `stopped`.

`POST /accounts/:account_id/autopilots`

Creates a new autopilot in the `draft` state. Blank fields are prefilled from the account's autopilot settings and account defaults (industry, language, end-action URL/HTML, default course), so a minimal body still produces a usable draft. The autopilot is **not** started — call `start` afterward to go live. **Auth:** Bearer; **role:** admin/editor.

The body is wrapped in an `autopilot` object.

Parameters

Name	In	Type	Required	Description
account_id	path	string	yes	Account id (`acct_…` or integer).
name	body	string	yes	Display name. Max 80 chars; must be unique within the account (case-insensitive).
all_groups	body	boolean	no	Target all groups. Defaults to `true`.
group_ids	body	array	no	Prefixed group ids (`grp_…`) to target. Unknown/foreign id → `422`. Used when `all_groups` is `false`.
intensity_count	body	integer	no	Campaigns per period. Must be ≥ 1. Defaults to `1`. The resulting daily rate (`intensity_count / period_days`) must be ≤ 2/day.
intensity_period	body	string	no	`day` | `week` | `month` | `year`. Defaults to `month`.
duration_kind	body	string	no	`continuous` | `until_date`. Defaults to `continuous`.
ends_on	body	string (date)	conditional	Required (and must be a future date) when `duration_kind` is `until_date`.
ai_optimizer_enabled	body	boolean	no	Defaults to `true`.
auto_include_new_members	body	boolean	no	Defaults to `true`.

Name	In	Type	Required	Description
language	body	string	no	Target language code (e.g. `en`, `pl`). Prefilled from account settings/locale if blank.
industry_code_id	body	integer	no	Industry code id. Prefilled from account settings if blank.
end_action_type	body	string	no	`nothing` | `redirect_to_course` | `message_page` | `redirect_to_url`. Defaults to `message_page`.
end_action_url	body	string	conditional	Required and must be `http`/`https` when `end_action_type` is `redirect_to_url`.
end_action_html	body	string	conditional	Required when `end_action_type` is `message_page` (prefilled from account defaults if blank).
course_id	body	string	conditional	Prefixed course id (`course_…`); must belong to the account or be a global course (else `422`). Required when `end_action_type` is `redirect_to_course`.

Request

```
curl -X POST -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{
        "autopilot": {
          "name": "Finance team — quarterly drip",
          "all_groups": false,
          "group_ids": ["grp_3a9k"],
          "intensity_count": 2,
          "intensity_period": "month",
          "duration_kind": "continuous",
          "end_action_type": "redirect_to_course",
          "course_id": "course_8h2d"
        }
      }' \
  https://platform.phishspot.com/api/v1/accounts/11/autopilots
```

Response `201 Created` — the newly created autopilot, in the same shape as the list item above (`state` will be `draft`, `started_at` and `progress_percentage` `null`).

```
{
  "id": 9,
  "account_id": 11,
  "name": "Finance team — quarterly drip",
  "state": "draft",
  "all_groups": false,
  "intensity_count": 2,
  "intensity_period": "month",
  "duration_kind": "continuous",
  "ai_optimizer_enabled": true,
  "auto_include_new_members": true,
  "language": "en",
  "end_action_type": "redirect_to_course",
  "end_action_url": null,
  "created_at": "2026-06-02T10:15:00Z",
  "updated_at": "2026-06-02T10:15:00Z",
  "ends_on": null,
  "started_at": null,
  "daily_rate": 0.07,
  "progress_percentage": null,
  "course_id": 14,
  "editable": true,
  "groups": [
    { "id": 3, "name": "Finance" }
  ],
  "recent_campaigns": []
}
```


Status codes

Code	When
201	Autopilot created.
400	The `autopilot` body wrapper is missing entirely.
403	Token's user is a `member` (read-only) on the account — only admins/editors may create.
404	The `account_id` does not exist or the token's user is not a member of it.
422	Validation failed — e.g. blank/duplicate/too-long `name`, daily rate above the 2/day cap, missing `ends_on` for `until_date`, missing `end_action_url` / `end_action_html` / `course_id` for the chosen `end_action_type`, or an unknown `group_ids` / `course_id`.

`GET /autopilots/:id`

Fetches a single autopilot by its id. This is a shallow (non-nested) route — no `account_id` in the path; the account is inferred from the autopilot and the token's membership is verified. **Auth:** Bearer; **role:** admin/editor.

No parameters beyond the bearer token.

Name	In	Type	Required	Description
id	path	string	yes	Autopilot id (`auto_…` or integer).

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/autopilots/9
```

Response `200 OK` — a single autopilot object (identical field set to the list item documented under `GET …/autopilots`).


```
{
  "id": 9,
  "account_id": 11,
  "name": "Finance team — quarterly drip",
  "state": "running",
  "all_groups": false,
  "intensity_count": 2,
  "intensity_period": "month",
  "duration_kind": "continuous",
  "ai_optimizer_enabled": true,
  "auto_include_new_members": true,
  "language": "en",
  "end_action_type": "redirect_to_course",
  "end_action_url": null,
  "created_at": "2026-06-02T10:15:00Z",
  "updated_at": "2026-06-02T10:16:00Z",
  "ends_on": null,
  "started_at": "2026-06-02T10:16:00Z",
  "daily_rate": 0.07,
  "progress_percentage": 100,
  "course_id": 14,
  "editable": true,
  "groups": [
    { "id": 3, "name": "Finance" }
  ],
  "recent_campaigns": []
}
```


Status codes

Code	When
200	Autopilot returned.
403	Token's user is a `member` (read-only) on the autopilot's account — single-autopilot reads require admin/editor.
404	No autopilot with that id, or the token's user has no active membership in its account.

`PATCH /autopilots/:id`

Updates an autopilot's configuration. The body is wrapped in an `autopilot` object and accepts the same fields as create. Passing `group_ids` **replaces** the full set of targeted groups. **Auth:** Bearer; **role:** admin/editor — and the autopilot must be editable (not `stopped`). Shallow route (no `account_id`).

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Autopilot id (`auto_…` or integer).
name	body	string	no	Display name. Max 80 chars; unique per account (case-insensitive).
all_groups	body	boolean	no	Target all groups.
group_ids	body	array	no	Prefixed group ids (`grp_…`). When present, **replaces** the current group set; unknown/foreign id → `422`.
intensity_count	body	integer	no	Campaigns per period (≥ 1; daily rate must stay ≤ 2/day).
intensity_period	body	string	no	`day` | `week` | `month` | `year`.
duration_kind	body	string	no	`continuous` | `until_date`.
ends_on	body	string (date)	conditional	Required future date when `duration_kind` is `until_date`.
ai_optimizer_enabled	body	boolean	no	Toggle AI optimization.
auto_include_new_members	body	boolean	no	Toggle auto-enrollment.
language	body	string	no	Target language code.
industry_code_id	body	integer	no	Industry code id.
end_action_type	body	string	no	`nothing` | `redirect_to_course` | `message_page` | `redirect_to_url`.

Name	In	Type	Required	Description
end_action_url	body	string	conditional	Required `http` / `https` URL when `end_action_type` is `redirect_to_url`.
end_action_html	body	string	conditional	Required when `end_action_type` is `message_page`.
course_id	body	string	conditional	Prefixed course id (`course_…`); must belong to the account or be global. Required when `end_action_type` is `redirect_to_course`.

Request

```
curl -X PATCH -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{ "autopilot": { "intensity_count": 1, "intensity_period": "week" } }' \
  https://platform.phishspot.com/api/v1/autopilots/9
```

Response `200 OK` — the updated autopilot object (same shape as `GET /autopilots/:id`).


```
{
  "id": 9,
  "account_id": 11,
  "name": "Finance team — quarterly drip",
  "state": "running",
  "all_groups": false,
  "intensity_count": 1,
  "intensity_period": "week",
  "duration_kind": "continuous",
  "ai_optimizer_enabled": true,
  "auto_include_new_members": true,
  "language": "en",
  "end_action_type": "redirect_to_course",
  "end_action_url": null,
  "created_at": "2026-06-02T10:15:00Z",
  "updated_at": "2026-06-02T11:00:00Z",
  "ends_on": null,
  "started_at": "2026-06-02T10:16:00Z",
  "daily_rate": 0.14,
  "progress_percentage": 100,
  "course_id": 14,
  "editable": true,
  "groups": [
    { "id": 3, "name": "Finance" }
  ],
  "recent_campaigns": []
}
```

Status codes

Code	When
200	Autopilot updated.
400	The `autopilot` body wrapper is missing entirely.
403	Token's user is a `member` (read-only), **or** the autopilot is `stopped` (stopped autopilots are read-only — delete only).
404	No autopilot with that id, or the token's user has no active membership in its account.
422	Validation failed — same triggers as create (name, daily-rate cap, `ends_on`, end-action requirements, unknown `group_ids` / `course_id`).

`DELETE /autopilots/:id`

Permanently deletes an autopilot and its group links; generated campaigns are detached (not deleted). A `running` autopilot cannot be deleted — stop it first. **Auth:** Bearer; **role:** admin/editor. Shallow route (no `account_id`).

No parameters beyond the bearer token.

Name	In	Type	Required	Description
id	path	string	yes	Autopilot id (`auto_…` or integer).

Request

```
curl -X DELETE -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/autopilots/9
```

Response `204 No Content` — empty body on success.

Status codes

Code	When
204	Autopilot deleted.
403	Token's user is a `member` (read-only) on the autopilot's account.
404	No autopilot with that id, or the token's user has no active membership in its account.
422	The autopilot is `running` — stop it before deleting. (A `paused` or `draft` autopilot can be deleted directly.)

`POST /autopilots/:id/start`

Activates the autopilot: sets `state` to `running` (stamping `started_at` on first start) so the platform begins generating and delivering campaigns at the configured cadence. Use to go live, or to resume a `paused` autopilot. **Auth:** Bearer; **role:** admin/editor — and the autopilot must be editable (not `stopped`). Shallow route.

> This is a live action: a successful call begins sending real simulated phishing email to targeted members.

No parameters beyond the bearer token.

Name	In	Type	Required	Description
id	path	string	yes	Autopilot id (`auto_…` or integer).

Request

```
curl -X POST -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/autopilots/9/start
```

Response `200 OK` — the autopilot object with `state: "running"` (same shape as `GET /autopilots/:id`).

```
{
  "id": 9,
  "name": "Finance team — quarterly drip",
  "state": "running",
  "started_at": "2026-06-02T10:16:00Z",
  "editable": true,
  "progress_percentage": 100,
  "daily_rate": 0.07
}
```

Status codes

Code	When
200	Autopilot started/resumed; now `running`.
403	Token's user is a `member` (read-only), **or** the autopilot is `stopped` (a stopped autopilot cannot be restarted).
404	No autopilot with that id, or the token's user has no active membership in its account.

`POST /autopilots/:id/pause`

Pauses a `running` autopilot: sets `state` to `paused` so no new campaigns are generated. Resume later with `start`. **Auth:** Bearer; **role:** admin/editor — and the autopilot must be editable (not `stopped`). Shallow route.

No parameters beyond the bearer token.

Name	In	Type	Required	Description
id	path	string	yes	Autopilot id (`auto_…` or integer).

Request

```
curl -X POST -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/autopilots/9/pause
```

Response `200 OK` — the autopilot object with `state: "paused"` (same shape as `GET /autopilots/:id`).

```
{
  "id": 9,
  "name": "Finance team — quarterly drip",
  "state": "paused",
  "started_at": "2026-06-02T10:16:00Z",
  "editable": true,
  "progress_percentage": 92
}
```

Status codes

Code	When
200	Autopilot paused.
403	Token's user is a `member` (read-only), **or** the autopilot is `stopped`.
404	No autopilot with that id, or the token's user has no active membership in its account.

`POST /autopilots/:id/stop`

Stops the autopilot permanently: sets `state` to `stopped`. A stopped autopilot becomes read-only — it can no longer be updated, started, paused, or stopped again, only viewed or deleted. **Auth:** Bearer; **role:** admin/editor — and the autopilot must still be editable (not already `stopped`). Shallow route.

No parameters beyond the bearer token.

Name	In	Type	Required	Description
id	path	string	yes	Autopilot id (`auto_…` or integer).

Request

```
curl -X POST -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/autopilots/9/stop
```

Response `200 OK` — the autopilot object with `state: "stopped"` and `editable: false` (same shape as `GET /autopilots/:id`).

```
{
  "id": 9,
  "name": "Finance team — quarterly drip",
  "state": "stopped",
  "started_at": "2026-06-02T10:16:00Z",
  "editable": false,
  "progress_percentage": null
}
```

Status codes

Code	When
200	Autopilot stopped.
403	Token's user is a `member` (read-only), **or** the autopilot is already `stopped`.
404	No autopilot with that id, or the token's user has no active membership in its account.

27.11 Sending domains

Two related resources control which domains a campaign can send from:

- **Platform domains** (`pdm_…`) are the attacker/landing domains PhishSpot operates. Most are platform-owned ("public" or assigned "private" domains); customers can also bring their own (BYOD) via `provision_byod`. Writing platform-domain records directly (`POST` / `PATCH` / `DELETE /platform_domains`) is reserved for **admin** users; provisioning a BYOD domain is open to any account member.
- **Secured domains** (`sdm_…`) are DNS-ownership proofs. A customer adds a domain they control, drops a TXT record, and verifies it — this is how PhishSpot confirms a sender is allowed to send "from" that domain.

`GET /api/v1/platform_domains`

Lists every platform domain visible to the calling token's account — all operational public domains plus any operational private domains assigned to the account. Results are ordered by name. **Auth:** Bearer; **role:** read (any role).

No parameters beyond the bearer token.

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/platform_domains
```

Response `200 OK` — a JSON array of platform-domain objects (each object uses the same fields as the show endpoint below).

Field	Type	Description
id	integer	Numeric id. (Use the `pdm_…` prefixed id in URLs.)
name	string	Domain name (e.g. `officelogin.in`).
public	boolean	Legacy public flag from the column.
mail	boolean	Whether this is the platform's mail domain.
state	string	Lifecycle state. One of `pending`, `checking`, `confirmed`, `purchasing`, `purchased`, `configuring_dns`, `dns_pending`, `configuring_postal`, `active`, `failed`.
expires_on	string|null	ISO8601 registration expiry, or null.
metadata	object	Free-form JSON (Cloudflare/Postal provisioning details, diagnostics, etc.).
created_at	string	ISO8601 timestamp.
updated_at	string	ISO8601 timestamp.
active	boolean	True when `state == "active"`.
byod	boolean	True for customer "bring your own domain" records.
sending_blocked	boolean	True when the domain is active but blocked from starting new sends.
nameservers	array	Cloudflare nameservers assigned to this domain's zone (empty unless captured).
cloudflare_error	string|null	Set if the Cloudflare zone could not be created/read.

```
[
  {
    "id": 42,
    "name": "officelogin.in",
    "public": true,
    "mail": false,
    "state": "active",
    "expires_on": "2027-03-01T00:00:00.000Z",
    "metadata": {},
    "created_at": "2026-01-10T09:00:00.000Z",
    "updated_at": "2026-05-30T14:22:00.000Z",
    "active": true,
    "byod": false,
    "sending_blocked": false,
    "nameservers": [],
    "cloudflare_error": null
  }
]
```


Status codes

Code	When
200	Domains returned (possibly an empty array).

`GET /api/v1/platform_domains/:id`

Fetches a single platform domain by id. Useful for inspecting state, BYOD nameservers, and `sending_blocked` before selecting it for a campaign. **Auth:** Bearer; **role:** read (any role).

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Platform-domain id (`pdm_…` or integer).

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/platform_domains/pdm_42
```

Response `200 OK` — a single platform-domain object (same fields as the list endpoint above).

```
{
  "id": 42,
  "name": "officelogin.in",
  "public": true,
  "mail": false,
  "state": "active",
  "expires_on": "2027-03-01T00:00:00.000Z",
  "metadata": {},
  "created_at": "2026-01-10T09:00:00.000Z",
  "updated_at": "2026-05-30T14:22:00.000Z",
  "active": true,
  "byod": false,
  "sending_blocked": false,
  "nameservers": [],
  "cloudflare_error": null
}
```


Status codes

Code	When
200	Domain found.
404	No platform domain with that id.

`POST /api/v1/platform_domains`

Creates a platform-owned domain record directly. This is an administrative operation for managing the platform's own domain pool — customers should use `provision_byod` instead. **Auth:** Bearer; **role:** admin.

Parameters — wrapped in a `platform_domain` object.

Name	In	Type	Required	Description
platform_domain.name	body	string	yes	Domain name. Lowercased/trimmed; must be a valid domain (3–253 chars, at least one dot, only `a–z 0–9 . –`, no leading/trailing dot or hyphen, no consecutive `..` / `––`, labels ≤63 chars). Must be globally unique (case-insensitive).
platform_domain.public	body	boolean	no	Legacy public flag.
platform_domain.metadata	body	object	no	Free-form JSON metadata.
platform_domain.expires_on	body	string	no	ISO8601 registration expiry.

`state`, `genre`, and `source` are not settable through the API; a record created here defaults to `state: "active"`, `genre: "public"`, `source: "manual"`.

Request

```
curl -X POST -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{ "platform_domain": { "name": "secure-portal.co", "public": true } }' \
  https://platform.phishspot.com/api/v1/platform_domains
```

Response `201 Created` — the created platform-domain object (same fields as the show endpoint).


```
{
  "id": 77,
  "name": "secure-portal.co",
  "public": true,
  "mail": false,
  "state": "active",
  "expires_on": null,
  "metadata": {},
  "created_at": "2026-06-02T10:00:00.000Z",
  "updated_at": "2026-06-02T10:00:00.000Z",
  "active": true,
  "byod": false,
  "sending_blocked": false,
  "nameservers": [],
  "cloudflare_error": null
}
```


Status codes

Code	When
201	Domain created.
403	Caller is not an admin.
422	Validation failed (e.g. blank/duplicate/invalid `name`). Body: `{ "errors": { "name": ["..."] } }`.

`PATCH /api/v1/platform_domains/:id`

Updates a platform-owned domain record. **Auth:** Bearer; **role:** admin.

Parameters — same wrapped `platform_domain` body as create; all fields optional.

Name	In	Type	Required	Description
id	path	string	yes	Platform-domain id (`pdm_…` or integer).
platform_domain.name	body	string	no	New domain name (same validation rules as create).

Name	In	Type	Required	Description
platform_domain.public	body	boolean	no	Legacy public flag.
platform_domain.metadata	body	object	no	Free-form JSON metadata.
platform_domain.expires_on	body	string	no	ISO8601 registration expiry.

```
Renaming or otherwise updating a domain that has an in-progress (locked) campaign is
rejected at the model level (`ActiveRecord::RecordNotDestroyed`), which surfaces as a
500-level error rather than 422. Avoid editing domains attached to running campaigns.
```

Request

```
curl -X PATCH -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{ "platform_domain": { "expires_on": "2028-01-01T00:00:00Z" } }' \
  https://platform.phishspot.com/api/v1/platform_domains/pdm_42
```

Response `200 OK` — the updated platform-domain object (same fields as the show endpoint).

Status codes

Code	When
200	Domain updated.
403	Caller is not an admin.
404	No platform domain with that id.
422	Validation failed. Body: `{ "errors": { ... } }`.

`DELETE /api/v1/platform_domains/:id`

Deletes a platform-owned domain. A domain can only be deleted if it has no blocking associations: platform-owned domains must have no campaigns and no assigned accounts; BYOD domains must have no active (in-progress/paused) campaigns. **Auth:** Bearer; **role:** admin.

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Platform-domain id (`pdm_…` or integer).

Request

```
curl -X DELETE -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/platform_domains/pdm_42
```

Response `204 No Content` — empty body on success.

Status codes

Code	When
204	Domain deleted.
403	Caller is not an admin.
404	No platform domain with that id.
422	Domain still has active campaigns (or other blocking associations). Body: `{ "error": "Cannot delete platform domain with active campaigns" }`.

`POST /api/v1/platform_domains/:id/check`

Re-checks the BYOD provisioning/health status of a domain the calling account owns, then returns the (reloaded) domain. Use this to poll a BYOD domain after delegating nameservers: an active domain runs a health check, a still-provisioning domain refreshes its setup status. Transient refresh failures are swallowed so polling never errors — you just get the last persisted state. **Auth:** Bearer; **role:** read (any role), but the token's user must belong to the domain's owner account.

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Platform-domain id (`pdm_…` or integer). Must be a BYOD domain owned by an account the caller belongs to.

Request

```
curl -X POST -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/platform_domains/pdm_88/check
```

Response `200 OK` — the refreshed platform-domain object (same fields as the show endpoint). Watch `state`, `active`, `nameservers`, `sending_blocked`, and `cloudflare_error` to track progress.

```
{
  "id": 88,
  "name": "mail.acme-customer.com",
  "public": false,
  "mail": false,
  "state": "dns_pending",
  "expires_on": null,
  "metadata": { "cloudflare_nameservers": ["kara.ns.cloudflare.com",
        "rob.ns.cloudflare.com"] },
  "created_at": "2026-06-01T08:00:00.000Z",
  "updated_at": "2026-06-02T09:30:00.000Z",
  "active": false,
  "byod": true,
  "sending_blocked": false,
  "nameservers": ["kara.ns.cloudflare.com", "rob.ns.cloudflare.com"],
  "cloudflare_error": null
}
```


Status codes

Code	When
200	Status refreshed and domain returned.
404	No such domain, or the domain has no owner account / the caller does not belong to its owner account.

`POST /api/v1/accounts/:account_id/platform_domains/provision_byod`

Provisions a customer “bring your own domain” (BYOD) sending domain for the account. Creates a private, BYOD platform domain in a provisioning state and returns the Cloudflare nameservers the domain owner must set at their registrar; delegation and verification then complete asynchronously (poll with `POST /platform_domains/:id/check`). Idempotent — re-provisioning a domain the account already owns just re-surfaces its nameservers. **Auth:** Bearer; **role:** any role, but the caller must belong to `account_id`.

Parameters

Name	In	Type	Required	Description
account_id	path	string	yes	Account id (`acct_…` or integer) the caller belongs to.
domain_name	body	string	yes	Domain to provision (e.g. `mail.acme-customer.com`). Lowercased/trimmed server-side. Not wrapped in any object — sent as a top-level key.

Request

```
curl -X POST -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{ "domain_name": "mail.acme-customer.com" }' \
  https://platform.phishspot.com/api/v1/accounts/11/platform_domains/provision_byod
```

Response `201 Created` — the provisioned platform-domain object plus provisioning guidance. All platform-domain fields from the show endpoint, plus:

Field	Type	Description
nameservers	array	Cloudflare nameservers the domain owner must set at their registrar. (Also present in the base object; repeated at the top level for convenience.)
next_step	string	Human-readable instruction describing the registrar change and the `check` endpoint to poll.


```
{
  "id": 88,
  "name": "mail.acme-customer.com",
  "public": null,
  "mail": false,
  "state": "dns_pending",
  "expires_on": null,
  "metadata": { "cloudflare_nameservers": ["kara.ns.cloudflare.com",
        "rob.ns.cloudflare.com"] },
  "created_at": "2026-06-02T09:00:00.000Z",
  "updated_at": "2026-06-02T09:00:00.000Z",
  "active": false,
  "byod": true,
  "sending_blocked": false,
  "nameservers": ["kara.ns.cloudflare.com", "rob.ns.cloudflare.com"],
  "cloudflare_error": null,
  "next_step": "At the registrar for mail.acme-customer.com, replace the nameservers
        with the ones above, then poll POST /api/v1/platform_domains/pdm_88/check."
}
```


Status codes

Code	When
201	Domain provisioned (or re-surfaced if already owned by this account).
403	Caller does not belong to `account_id` (Pundit `show?` denied).
404	`account_id` not found among the caller's accounts. Body: `{ "error": "Account not found" }`.

Code	When
422	Provisioning rejected. Body: `{ "error": "<message>" }`, one of: blank name (`"Domain name is required."`), already registered by another account (`"That domain is already registered in PhishSpot by another account."`), or invalid domain (`"That domain name is invalid."`).

`GET /api/v1/accounts/:account_id/secured_domains`

Lists the secured (ownership-verified) domains for an account, newest first. Optionally filter by verification state. **Auth:** Bearer; **role:** any role, but the caller must belong to `account_id`.

Parameters

Name	In	Type	Required	Description
account_id	path	string	yes	Account id (`acct_…` or integer) the caller belongs to.
state	query	string	no	Filter by verification state. One of `pending`, `verified`, `failed`. Omit for all.

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  "https://platform.phishspot.com/api/v1/accounts/11/secured_domains?state=verified"
```

Response `200 OK` — a JSON array of secured-domain objects.

Field	Type	Description
id	integer	Numeric id. (Use the `sdm_…` prefixed id in URLs.)
account_id	integer	Owning account id.
domain	string	The domain being verified (lowercased).
state	string	Verification state: `pending`, `verified`, or `failed`.
verification_attempts	integer	Number of DNS verification attempts made.
verified_at	string|null	ISO8601 timestamp of successful verification, or null.
created_at	string	ISO8601 timestamp.
updated_at	string	ISO8601 timestamp.
dns_record	object	The TXT record the owner must publish to prove ownership.
dns_record.type	string	Always `"TXT"`.
dns_record.name	string	Record host, e.g. `_phishspot-verify.example.com`.

Field	Type	Description
dns_record.value	string	Record value, e.g. `phishspot-verify=<64-hex-token>`.


```
[
  {
    "id": 5,
    "account_id": 11,
    "domain": "acme-customer.com",
    "state": "verified",
    "verification_attempts": 2,
    "verified_at": "2026-05-20T11:00:00.000Z",
    "created_at": "2026-05-19T16:30:00.000Z",
    "updated_at": "2026-05-20T11:00:00.000Z",
    "dns_record": {
      "type": "TXT",
      "name": "_phishspot-verify.acme-customer.com",
      "value": "phishspot-verify=3f9a...c2"
    }
  }
]
```


Status codes

Code	When
200	Domains returned (possibly an empty array).
403	Caller does not belong to `account_id`.
404	`account_id` not found among the caller's accounts. Body: `{ "error": "Account not found" }`.

`POST /api/v1/accounts/:account_id/secured_domains`

Adds a domain the customer controls and returns the TXT record to publish for ownership verification. Public email-provider domains (e.g. gmail.com) are rejected. **Auth:** Bearer; **role:** any role, but the caller must belong to `account_id`.

Parameters — wrapped in a `secured_domain` object.

Name	In	Type	Required	Description
account_id	path	string	yes	Account id (`acct_…` or integer) the caller belongs to.

Name	In	Type	Required	Description
secured_domain.domain	body	string	yes	Domain to verify (e.g. `acme-customer.com`). Lowercased/trimmed; must match the standard domain format; must be unique per account (case-insensitive); cannot be a blocked public-email-provider domain.

Request

```
curl -X POST -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{ "secured_domain": { "domain": "acme-customer.com" } }' \
  https://platform.phishspot.com/api/v1/accounts/11/secured_domains
```

Response `201 Created` — the created secured-domain object (same fields as the list endpoint), with `state: "pending"` and a freshly generated `dns_record` to publish.


```
{
  "id": 6,
  "account_id": 11,
  "domain": "acme-customer.com",
  "state": "pending",
  "verification_attempts": 0,
  "verified_at": null,
  "created_at": "2026-06-02T10:00:00.000Z",
  "updated_at": "2026-06-02T10:00:00.000Z",
  "dns_record": {
    "type": "TXT",
    "name": "_phishspot-verify.acme-customer.com",
    "value": "phishspot-verify=3f9a...c2"
  }
}
```


Status codes

Code	When
201	Domain added; publish the returned TXT record, then call `verify_dns`.
403	Caller does not belong to `account_id`.
404	`account_id` not found among the caller's accounts. Body: `{ "error": "Account not found" }`.
422	Validation failed — blank/invalid format, duplicate for this account, or a blocked public-email-provider domain. Body: `{ "errors": { "domain": ["..."] } }`.

`GET /api/v1/secured_domains/:id`

Fetches a single secured domain by id, including the TXT record needed for verification. Shallow route (not nested under account) — the caller must belong to the owning account. **Auth:** Bearer; **role:** any role (account member).

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Secured-domain id (`sdm_…` or integer).

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/secured_domains/sdm_6
```

Response `200 OK` — a single secured-domain object (same fields as the list endpoint above).

Status codes

Code	When
200	Domain found.
403	Pundit denied (should not normally occur — the policy permits any member).
404	No such domain, or the caller does not belong to its owning account.

`DELETE /api/v1/secured_domains/:id`

Removes a secured domain. Blocked while any active (in-progress/paused) campaign sends from an email address on that domain. Shallow route — the caller must belong to the owning account. **Auth:** Bearer; **role:** any role (account member).

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Secured-domain id (`sdm_…` or integer).

Request

```
curl -X DELETE -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/secured_domains/sdm_6
```

Response `204 No Content` — empty body on success.

Status codes

Code	When
204	Domain deleted.
403	Pundit denied (should not normally occur).
404	No such domain, or the caller does not belong to its owning account.
422	Domain is verified and in use by active campaigns. Body: `{ "error": "Cannot delete secured domain <domain> while it is used by active campaigns: <campaign names>" }`.

`POST /api/v1/secured_domains/:id/verify_dns`

Runs DNS verification: looks up the expected TXT record for the domain and, if found, marks the domain `verified`. Call this after publishing the TXT record returned at creation. Returns the (reloaded) domain so you can read the resulting `state`. Shallow route — the caller must belong to the owning account. **Auth:** Bearer; **role:** any role (account member).

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Secured-domain id (`sdm_…` or integer).

Request

```
curl -X POST -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/secured_domains/sdm_6/verify_dns
```

Response `200 OK` — the refreshed secured-domain object (same fields as the list endpoint). On success `state` becomes `"verified"` and `verified_at` is set; otherwise it stays `pending` / `failed` and `verification_attempts` increments.

```
{
  "id": 6,
  "account_id": 11,
  "domain": "acme-customer.com",
  "state": "verified",
  "verification_attempts": 1,
  "verified_at": "2026-06-02T10:05:00.000Z",
  "created_at": "2026-06-02T10:00:00.000Z",
  "updated_at": "2026-06-02T10:05:00.000Z",
  "dns_record": {
    "type": "TXT",
    "name": "_phishspot-verify.acme-customer.com",
    "value": "phishspot-verify=3f9a...c2"
  }
}
```

Status codes

Code	When
200	Verification ran; check `state` in the response.
403	Pundit denied (should not normally occur).
404	No such domain, or the caller does not belong to its owning account.

27.12 Reported messages

Reported messages are suspicious emails that employees flagged — either forwarded into the account's report inbox (`source: inbound_webhook`) or submitted through the Outlook add-in (`source: outlook_addin`). The read endpoints below return **metadata only** (sender, subject, received-at, source, reporter) — never the message body, headers, or attachments. They are scoped to accounts the calling token's user belongs to.

`GET /accounts/:account_id/reported_messages`

Lists reported messages for one account, newest first (ordered by `received_at` descending). Use it to feed a triage queue or to sync reports into your SOC tooling. **Auth:** Bearer; **role:** read (any role).

Parameters

Name	In	Type	Required	Description
account_id	path	string	yes	Account id (`acct_…` or integer). Must be an account the token's user belongs to.
source	query	string	no	Filter by intake source. One of `inbound_webhook`, `outlook_addin`. An unknown value returns `422`. Omit to return all sources.
limit	query	integer	no	Page size. Defaults to `50`; clamped to the range `1` – `500` (values `<1` or `0` fall back to `50`, values `>500` are capped at `500`).
page	query	integer	no	1-based page number. Defaults to `1`; values below `1` are treated as `1`. Offset is computed as `(page - 1) * limit`.

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  "https://platform.phishspot.com/api/v1/accounts/11/reported_messages?
        source=outlook_addin&limit=25&page=1"
```

Response `200 OK` — a JSON array of reported-message metadata objects (no envelope). Each element has these fields:

Field	Type	Description
id	integer	Reported-message id.
account_id	integer	Owning account id.
from_email	string	Sender address of the reported email.
from_name	string | null	Sender display name, if captured.
subject	string | null	Subject line of the reported email.
message_id	string | null	Original RFC `Message-ID`, if captured.
source	string	Intake source: `inbound_webhook` or `outlook_addin`.
received_at	string (ISO 8601)	When the original email was received.
created_at	string (ISO 8601)	When the report was ingested into PhishSpot.
from_domain	string	Lower-cased domain portion of `from_email` (text after `@`).
reporter_contact_email	string | null	Email of the account Contact who reported it (matched on `from_email`); `null` when no matching Contact exists.

```
[
  {
    "id": 4821,
    "account_id": 11,
    "from_email": "billing@suspicious-invoice.example",
    "from_name": "Accounts Payable",
    "subject": "Overdue invoice — action required",
    "message_id": "<CADnf9x1@mail.suspicious-invoice.example>",
    "source": "outlook_addin",
    "received_at": "2026-05-28T09:14:00.000Z",
    "created_at": "2026-05-28T09:15:32.000Z",
    "from_domain": "suspicious-invoice.example",
    "reporter_contact_email": "jane.doe@acme.test"
  }
]
```

Status codes

Code	When
200	Reports returned (array may be empty).

Code	When
403	The token's user is authorized but Pundit denies `AccountPolicy#show?` for this account.
404	`account_id` is not an account the token's user belongs to (returns `{ "error": "Account not found" }`).
422	`source` is present but not one of the valid sources (returns `{ "error": "Unknown source …; valid sources: inbound_webhook, outlook_addin." }`).

`GET /reported_messages/:id`

Fetches the metadata for a single reported message. This is a **shallow** route — it takes the report id directly, not an `account_id` path segment. Account isolation is enforced server-side: the lookup is scoped to the token user's accounts, so requesting a report in another account returns `404`. **Auth:** Bearer; **role:** read (any role).

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Reported-message id (`rep_…` or integer).

No parameters beyond the bearer token and the path id.

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/reported_messages/rep_4821
```

Response `200 OK` — a single reported-message metadata object with the same fields as one element of the index array:

Field	Type	Description
id	integer	Reported-message id.
account_id	integer	Owning account id.
from_email	string	Sender address of the reported email.
from_name	string | null	Sender display name, if captured.
subject	string | null	Subject line of the reported email.
message_id	string | null	Original RFC `Message-ID`, if captured.
source	string	Intake source: `inbound_webhook` or `outlook_addin`.
received_at	string (ISO 8601)	When the original email was received.

Field	Type	Description
created_at	string (ISO 8601)	When the report was ingested into PhishSpot.
from_domain	string	Lower-cased domain portion of `from_email`.
reporter_contact_email	string | null	Email of the matching account Contact, or `null`.


```
{
  "id": 4821,
  "account_id": 11,
  "from_email": "billing@suspicious-invoice.example",
  "from_name": "Accounts Payable",
  "subject": "Overdue invoice — action required",
  "message_id": "<CADnf9x1@mail.suspicious-invoice.example>",
  "source": "outlook_addin",
  "received_at": "2026-05-28T09:14:00.000Z",
  "created_at": "2026-05-28T09:15:32.000Z",
  "from_domain": "suspicious-invoice.example",
  "reporter_contact_email": "jane.doe@acme.test"
}
```


Status codes

Code	When
200	The report was found and returned.
404	No report with that id exists within the token user's accounts (returns `{ "error": "Resource not found" }`).

`POST /accounts/:account_id/reported_messages`

Add-in only. Ingests a newly reported message from the Outlook add-in. This endpoint does **not** use a normal API bearer token or Pundit — it requires an add-in capability token (`reported_messages:create`) whose pinned `account_id` matches the `:account_id` in the URL. Standard integrations do not call this; reads use the two endpoints above. The body is wrapped in a `reported_message` object (permitted keys: `from_email`, `from_name`, `subject`, `plain_body`, `html_body`, `received_at`, `message_id`, `headers`, plus an `attachments` array) and a successful call returns `201 Created` with `{ "id": "rep_…", "url": "…" }`. A capability/account mismatch returns `403`; validation failures return `422`.

> The add-in submission flow (token capabilities, pairing, and the full request shape) is documented separately under Reported messages (add-in intake). Use the read endpoints above for any reporting or triage integration.

Status codes

Code	When
201	The report was created.

Code	When
403	The token lacks the `reported_messages:create` capability, or its pinned `account_id` does not match the URL.
404	`account_id` does not resolve to an existing account (returns `{ "error": "Account not found" }`).
422	The ingest service rejected the payload (returns `{ "error": "…" }`).

27.13 Media library

The media library stores hosted image and CSS files for an account. Upload a file once, then reference its returned `url` inside campaign email HTML, landing pages, and templates. **Always embed the hosted `url`** — email clients (Gmail, Outlook) strip inline `data:` URIs, so base64-embedded images will not render.

Allowed content types: `image/png`, `image/jpg`, `image/jpeg`, `image/gif`, `image/svg+xml`, and `text/css`. Maximum file size is **5 MB**. Every media item must have a unique (case-insensitive) `name` within its account.

All media endpoints are usable by any member of the account (no admin/editor gate). The collection endpoints (`index`, `create`) are nested under an account; the single-item endpoints (`show`, `update`, `destroy`) are shallow (`/media/:id`) and resolve only media belonging to one of the token user's accounts — anything else returns `404`.

`GET /accounts/:account_id/media`

Lists all media items for the account, newest first. Use it to find a file's hosted `url` before embedding it in a campaign. **Auth:** Bearer; **role:** read (any role).

Parameters

Name	In	Type	Required	Description
account_id	path	string	yes	Account id (`acct_…` or integer).

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/accounts/11/media
```

Response `200 OK` — a JSON array of media objects (each is the shape below).

Field	Type	Description
id	integer	Media item id.

Field	Type	Description
account_id	integer	Owning account id.
name	string	Display name (unique per account, case-insensitive).
created_at	string	ISO 8601 timestamp.
updated_at	string	ISO 8601 timestamp.
url	string|null	Hosted file URL (relative path); embed this in HTML. `null` if no file is attached.
filename	string|null	Original uploaded filename.
content_type	string|null	MIME type, e.g. `image/png`.


```
[
  {
    "id": 42,
    "account_id": 11,
    "name": "phishing-logo",
    "created_at": "2026-06-02T10:15:00.000Z",
    "updated_at": "2026-06-02T10:15:00.000Z",
    "url": "/rails/active_storage/blobs/redirect/eyJfcmFpbHMi.../phishing-logo.png",
    "filename": "phishing-logo.png",
    "content_type": "image/png"
  }
]
```


Status codes

Code	When
200	Media list returned.
404	The `account_id` is not one of the token user's accounts.

`POST /accounts/:account_id/media`

Uploads a new file to the account's media library. **This request is `multipart/form-data`, not JSON** — the file is sent as a form field, not a base64 string in a JSON body. **Auth:** Bearer; **role:** any account member.

Parameters (form fields, wrapped in a `medium[...]` object)

Name	In	Type	Required	Description
account_id	path	string	yes	Account id (`acct_…` or integer).
medium[name]	body	string	yes	Display name; must be unique (case-insensitive) within the account.

Name	In	Type	Required	Description
medium[attachment]	body	file	yes	The file to upload. Must be one of `image/png`, `image/jpg`, `image/jpeg`, `image/gif`, `image/svg+xml`, `text/css`, and ≤ 5 MB.

Request

Note the `-F` flags (multipart). Do **not** set `Content-Type: application/json`; let curl set the multipart boundary. The file is read from disk with `@/path`.

```
curl -X POST -H "Authorization: Bearer $TOKEN" \
  -F "medium[name]=phishing-logo" \
  -F "medium[attachment]=@./phishing-logo.png;type=image/png" \
  https://platform.phishspot.com/api/v1/accounts/11/media
```

Response `201 Created` — the created media object (same shape as the list item above).

Field	Type	Description
id	integer	New media item id.
account_id	integer	Owning account id.
name	string	Display name.
created_at	string	ISO 8601 timestamp.
updated_at	string	ISO 8601 timestamp.
url	string	Hosted file URL — embed this in campaign HTML.
filename	string	Stored filename.
content_type	string	MIME type of the uploaded file.


```
{
  "id": 42,
  "account_id": 11,
  "name": "phishing-logo",
  "created_at": "2026-06-02T10:15:00.000Z",
  "updated_at": "2026-06-02T10:15:00.000Z",
  "url": "/rails/active_storage/blobs/redirect/eyJfcmFpbHMi.../phishing-logo.png",
  "filename": "phishing-logo.png",
  "content_type": "image/png"
}
```


> Prefix the returned relative `url` with your platform host (`https://platform.phishspot.com`) to get an absolute, externally reachable URL for use in email HTML.

Status codes

Code	When
201	Media created.
404	The `account_id` is not one of the token user's accounts.
422	Validation failed — missing/blank `name`, duplicate `name` in the account, missing `attachment`, disallowed content type, or file over 5 MB. Body: `{ "errors": { … } }`.

Example `422` body (missing attachment):

```
{ "errors": { "attachment": ["can't be blank"] } }
```

`GET /media/:id`

Returns a single media item by id. Resolves only media belonging to one of the token user's accounts. **Auth:** Bearer; **role:** read (any role).

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Media item id (`med_…` or integer).

No parameters beyond the bearer token.

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/media/42
```

Response `200 OK` — the media object (same fields as the `POST` response above).

```
{
  "id": 42,
  "account_id": 11,
  "name": "phishing-logo",
  "created_at": "2026-06-02T10:15:00.000Z",
  "updated_at": "2026-06-02T10:15:00.000Z",
  "url": "/rails/active_storage/blobs/redirect/eyJfcmFpbHMi.../phishing-logo.png",
  "filename": "phishing-logo.png",
  "content_type": "image/png"
}
```

Status codes

Code	When
200	Media item returned.
404	No media item with that id belongs to one of the token user's accounts.

`PATCH /media/:id`

Updates a media item. In practice only `name` is meaningful; you can also re-upload a file by sending a new `attachment` (multipart). **Auth:** Bearer; **role:** any account member.

Parameters (wrapped in a `medium[...]` object)

Name	In	Type	Required	Description
id	path	string	yes	Media item id (`med_…` or integer).
medium[name]	body	string	no	New display name; must stay unique (case-insensitive) within the account.
medium[attachment]	body	file	no	Replacement file (send as multipart). Same content-type and 5 MB limits as create.

Request

A name-only change can be sent as JSON:

```
curl -X PATCH -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{ "medium": { "name": "phishing-logo-v2" } }' \
  https://platform.phishspot.com/api/v1/media/42
```

To replace the file, send multipart instead (`-F "medium[attachment]=@./new.png;type=image/png"`).

Response `200 OK` — the updated media object (same fields as `show`).

```
{
  "id": 42,
  "account_id": 11,
  "name": "phishing-logo-v2",
  "created_at": "2026-06-02T10:15:00.000Z",
  "updated_at": "2026-06-02T11:02:00.000Z",
  "url": "/rails/active_storage/blobs/redirect/eyJfcmFpbHMi.../phishing-logo.png",
  "filename": "phishing-logo.png",
  "content_type": "image/png"
}
```

Status codes

Code	When
200	Media updated.
404	No media item with that id belongs to one of the token user's accounts.
422	Validation failed — blank `name`, duplicate `name`, or (on file replacement) disallowed content type / over 5 MB. Body: `{ "errors": { … } }`.

`DELETE /media/:id`

Permanently deletes a media item and its attached file. Any campaign HTML still referencing the file's `url` will break, so remove references first. **Auth:** Bearer; **role:** any account member.

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Media item id (`med_…` or integer).

No parameters beyond the bearer token.

Request

```
curl -X DELETE -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/media/42
```

Response `204 No Content` — empty body.

Status codes

Code	When
204	Media deleted.
404	No media item with that id belongs to one of the token user's accounts.

27.14 Webhooks

Manage outbound webhook subscriptions and inspect the events PhishSpot has generated for an account. An **endpoint** is a URL you register to receive signed HTTP POSTs; an **event** is an immutable record of something that happened (a campaign was created, a contact was deleted, etc.) that is fanned out to every enabled endpoint subscribed to its type. For the delivery mechanics — retry schedule, the `X-Webhook-Signature` HMAC header, and how to verify it with the `signing_secret` — see Webhook delivery & signatures.

:::note[Endpoints are account-global, not tenant-scoped] `Webhook::Endpoint` and `Webhook::Event` are not multi-tenant records, so the shallow routes (`/webhooks/endpoints/:id` , `/webhooks/events/:id`)

take a record id directly and have **no** `account_id` in the path. Authorization is still enforced per record: you must be a member of the account that owns the endpoint/event, otherwise you get `403`. The `:id` segment accepts either the prefixed id (`whep_…`) or the raw integer. :::

Available `event_type_ids` values. An endpoint subscribes by listing one or more of these strings. The same strings appear as the `event_type` on emitted events:

Event type	Fires when
`campaign.created`	A campaign is created.
`campaign.updated`	A campaign is updated.
`campaign.deleted`	A campaign is deleted.
`contact.created`	A contact is added.
`contact.updated`	A contact is updated.
`contact.deleted`	A contact is removed.
`deliverable.created`	A campaign deliverable (one recipient's send) is created.
`deliverable.updated`	A deliverable changes state (sent, opened, clicked, etc.).
`spam_whitelist.updated`	The account's spam/allow-list snapshot changes.

`GET /accounts/:account_id/webhooks/endpoints`

Lists every webhook endpoint registered on the account, newest first. Use this to render a management UI or reconcile your local config. **Auth:** Bearer; **role:** read (any role).

Parameters

Name	In	Type	Required	Description
account_id	path	string	yes	Account id (`acct_…` or integer).

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/accounts/11/webhooks/endpoints
```

Response `200 OK` — a JSON array of endpoint objects. The `signing_secret` is **omitted** from this list view (it is only ever returned by the single-endpoint views).

Field	Type	Description
id	integer	Endpoint id. The path/show segment also accepts the `whep_…` prefixed form.
account_id	integer	Owning account id.
name	string	Human label for the endpoint.
url	string	Destination URL that receives POSTs.

Field	Type	Description
event_type_ids	array of string	Subscribed event types (see table above).
enabled	boolean	Whether deliveries are currently being sent.
api_version	integer	Payload schema version (currently `1`).
created_at	string	ISO 8601 timestamp.
updated_at	string	ISO 8601 timestamp.
total_deliveries	integer	Count of all delivery records for this endpoint.
successful_deliveries	integer	Count of deliveries in `delivered` status.
failed_deliveries	integer	Count of deliveries in `failed` status.


```
[
  {
    "id": 42,
    "account_id": 11,
    "name": "Production listener",
    "url": "https://hooks.example.com/phishspot",
    "event_type_ids": ["campaign.created", "deliverable.updated"],
    "enabled": true,
    "api_version": 1,
    "created_at": "2026-05-30T09:14:22Z",
    "updated_at": "2026-06-01T12:03:10Z",
    "total_deliveries": 128,
    "successful_deliveries": 121,
    "failed_deliveries": 7
  }
]
```


Status codes

Code	When
200	Endpoints returned (empty array if none).
403	Caller is not a member of `account_id`.
404	`account_id` does not exist or caller is not a member.

`POST /accounts/:account_id/webhooks/endpoints`

Registers a new webhook endpoint. The response includes the `signing_secret` **exactly once** — store it immediately, as it is needed to verify delivery signatures and is never shown in full again except on `show` / `update` / `toggle` of that same endpoint. **Auth:** Bearer; **role:** read (any role — membership in the account is sufficient).

:::caution[Endpoints start disabled unless you opt in] A newly created endpoint defaults to `enabled: true` in the model, but pass `enabled: false` to register it dormant and switch it on later via the toggle endpoint once you have verified your receiver. The `url` is validated for safety: only `http` / `https`

schemes are accepted, and URLs pointing at `localhost`, `127.0.0.1`, `::1`, RFC 1918 / link-local / loopback / unique-local IP ranges, or any `*.phishspot.com` host are rejected with `422`. :::

Parameters

Body params are wrapped in a `webhook_endpoint` object.

Name	In	Type	Required	Description
account_id	path	string	yes	Account id (`acct_…` or integer).
webhook_endpoint.name	body	string	yes	Human label. Presence-validated.
webhook_endpoint.url	body	string	yes	Destination URL. Must be a valid `http` / `https` URL and pass the safety checks above.
webhook_endpoint.event_type_ids	body	array of string	yes	One or more event types to subscribe to (see table). Presence-validated — at least one required.
webhook_endpoint.enabled	body	boolean	no	Whether to start enabled. Defaults to `true`.

Request

```
curl -X POST -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{
        "webhook_endpoint": {
          "name": "Production listener",
          "url": "https://hooks.example.com/phishspot",
          "event_type_ids": ["campaign.created", "deliverable.updated"],
          "enabled": false
        }
      }' \
  https://platform.phishspot.com/api/v1/accounts/11/webhooks/endpoints
```

Response `201 Created` — the created endpoint, including the one-time `signing_secret`. Fields are the same as the list view plus `signing_secret`:

Field	Type	Description
id	integer	Endpoint id.
account_id	integer	Owning account id.
name	string	Human label.
url	string	Destination URL (trimmed/normalized).
event_type_ids	array of string	Subscribed event types.

Field	Type	Description
enabled	boolean	Whether deliveries are active.
api_version	integer	Payload schema version (`1`).
created_at	string	ISO 8601 timestamp.
updated_at	string	ISO 8601 timestamp.
signing_secret	string	64-char hex HMAC secret. **Returned here — save it now.**
total_deliveries	integer	`0` for a new endpoint.
successful_deliveries	integer	`0` for a new endpoint.
failed_deliveries	integer	`0` for a new endpoint.


```
{
  "id": 42,
  "account_id": 11,
  "name": "Production listener",
  "url": "https://hooks.example.com/phishspot",
  "event_type_ids": ["campaign.created", "deliverable.updated"],
  "enabled": false,
  "api_version": 1,
  "created_at": "2026-06-02T10:00:00Z",
  "updated_at": "2026-06-02T10:00:00Z",
  "signing_secret": "9f2c1e7b4a6d8f0c3e5a7b9d1f3c5e7a9b1d3f5c7e9a1b3d5f7c9e1a3b5d7f9c",
  "total_deliveries": 0,
  "successful_deliveries": 0,
  "failed_deliveries": 0
}
```


Status codes

Code	When
201	Endpoint created.
403	Caller is not a member of `account_id` .
404	`account_id` does not exist or caller is not a member.
422	Validation failed — missing `name` / `url` / `event_type_ids` , a malformed URL, or a URL that targets localhost / a private IP / a `*.phishspot.com` host.

`GET /webhooks/endpoints/:id`

Fetches a single endpoint, including its full `signing_secret` and lifetime delivery statistics. Use this to re-read the secret if you lost it, or to monitor delivery health. **Auth:** Bearer; **role:** read (any role — must be a member of the owning account).

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Endpoint id (`whep_…` or integer).

No parameters beyond the bearer token and path id.

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/webhooks/endpoints/whep_8xk2p9
```

Response `200 OK` — one endpoint object with the same fields as the create response (includes `signing_secret`).

```
{
  "id": 42,
  "account_id": 11,
  "name": "Production listener",
  "url": "https://hooks.example.com/phishspot",
  "event_type_ids": ["campaign.created", "deliverable.updated"],
  "enabled": true,
  "api_version": 1,
  "created_at": "2026-05-30T09:14:22Z",
  "updated_at": "2026-06-01T12:03:10Z",
  "signing_secret": "9f2c1e7b4a6d8f0c3e5a7b9d1f3c5e7a9b1d3f5c7e9a1b3d5f7c9e1a3b5d7f9c",
  "total_deliveries": 128,
  "successful_deliveries": 121,
  "failed_deliveries": 7
}
```

Status codes

Code	When
200	Endpoint returned.
403	Caller is not a member of the account that owns the endpoint.
404	No endpoint with that id.

`PATCH /webhooks/endpoints/:id`

Updates an endpoint's name, URL, subscribed event types, or enabled flag. The `signing_secret` is not regenerated and cannot be changed through this call. **Auth:** Bearer; **role:** read (any role — must be a member of the owning account).

Parameters

Body params are wrapped in a `webhook_endpoint` object; send only the fields you want to change.

Name	In	Type	Required	Description
id	path	string	yes	Endpoint id (`whep_…` or integer).
webhook_endpoint.name	body	string	no	New label. Cannot be blanked (presence-validated).
webhook_endpoint.url	body	string	no	New destination URL. Re-validated for safety (same rules as create).
webhook_endpoint.event_type_ids	body	array of string	no	Replacement set of subscribed event types. Cannot be emptied.
webhook_endpoint.enabled	body	boolean	no	Enable/disable.

Request

```
curl -X PATCH -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{
        "webhook_endpoint": {
          "event_type_ids": ["campaign.created", "campaign.updated", "campaign.deleted"]
        }
      }' \
  https://platform.phishspot.com/api/v1/webhooks/endpoints/whep_8xk2p9
```

Response `200 OK` — the updated endpoint object (same fields as `show`, including `signing_secret`).

```
{
  "id": 42,
  "account_id": 11,
  "name": "Production listener",
  "url": "https://hooks.example.com/phishspot",
  "event_type_ids": ["campaign.created", "campaign.updated", "campaign.deleted"],
  "enabled": true,
  "api_version": 1,
  "created_at": "2026-05-30T09:14:22Z",
  "updated_at": "2026-06-02T11:20:45Z",
  "signing_secret": "9f2c1e7b4a6d8f0c3e5a7b9d1f3c5e7a9b1d3f5c7e9a1b3d5f7c9e1a3b5d7f9c",
  "total_deliveries": 128,
  "successful_deliveries": 121,
  "failed_deliveries": 7
}
```

Status codes

Code	When
200	Endpoint updated.
403	Caller is not a member of the owning account.

Code	When
404	No endpoint with that id.
422	Validation failed — blank `name`, empty `event_type_ids`, or an invalid/unsafe `url`.

`DELETE /webhooks/endpoints/:id`

Permanently deletes an endpoint and all of its delivery records. Events themselves are not deleted. **Auth:** Bearer; **role:** read (any role — must be a member of the owning account).

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Endpoint id (`whep_…` or integer).

No parameters beyond the bearer token and path id.

Request

```
curl -X DELETE -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/webhooks/endpoints/whep_8xk2p9
```

Response `204 No Content` — empty body on success.

Status codes

Code	When
204	Endpoint deleted.
403	Caller is not a member of the owning account.
404	No endpoint with that id.

`POST /webhooks/endpoints/:id/toggle`

Flips the endpoint's `enabled` flag — enables a disabled endpoint or disables an enabled one. Use this to pause/resume deliveries without deleting the endpoint or losing its secret. **Auth:** Bearer; **role:** read (any role — must be a member of the owning account).

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Endpoint id (`whep_…` or integer).

No parameters beyond the bearer token and path id — the new state is derived from the current one.

Request

```
curl -X POST -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/webhooks/endpoints/whep_8xk2p9/toggle
```

Response `200 OK` — the endpoint with its flipped `enabled` value (same fields as `show`, including `signing_secret`).


```
{
  "id": 42,
  "account_id": 11,
  "name": "Production listener",
  "url": "https://hooks.example.com/phishspot",
  "event_type_ids": ["campaign.created", "deliverable.updated"],
  "enabled": false,
  "api_version": 1,
  "created_at": "2026-05-30T09:14:22Z",
  "updated_at": "2026-06-02T11:30:00Z",
  "signing_secret": "9f2c1e7b4a6d8f0c3e5a7b9d1f3c5e7a9b1d3f5c7e9a1b3d5f7c9e1a3b5d7f9c",
  "total_deliveries": 128,
  "successful_deliveries": 121,
  "failed_deliveries": 7
}
```


Status codes

Code	When
200	State flipped; updated endpoint returned.
403	Caller is not a member of the owning account.
404	No endpoint with that id.

`GET /accounts/:account_id/webhooks/events`

Lists the events generated for the account, newest first, paginated. Each event records what happened and bundles per-event delivery counts. Use this to audit what PhishSpot tried to send, independent of any one endpoint. **Auth:** Bearer; **role:** read (any role).

Parameters

Name	In	Type	Required	Description
account_id	path	string	yes	Account id (`acct_…` or integer).
event_type	query	string	no	Filter to a single event type (e.g. `deliverable.updated`). Omit for all types.
page	query	integer	no	Page number. Defaults to `1`.
per_page	query	integer	no	Items per page. Defaults to `50`.

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  "https://platform.phishspot.com/api/v1/accounts/11/webhooks/events?
        event_type=deliverable.updated&page=1&per_page=50"
```

Response `200 OK` — a JSON array of event objects.

Field	Type	Description
id	integer	Event id. The show segment also accepts the prefixed/raw form.
account_id	integer	Owning account id.
subject_id	integer	Id of the record the event is about (campaign, contact, deliverable, …).
subject_type	string	Class name of the subject (e.g. `Campaign`, `Contact`, `Deliverable`).
event_type	string	One of the event types in the table above.
api_version	integer	Payload schema version (`1`).
uuid	string	Stable unique id for this event (also used as `payload.id`).
created_at	string	ISO 8601 timestamp.
updated_at	string	ISO 8601 timestamp.
data	object	Event-specific data describing the change (shape varies by `event_type`).
payload	object	The exact JSON body delivered to endpoints (see below).
deliveries	object	Per-event delivery counts.
deliveries.total	integer	All delivery records for this event.
deliveries.delivered	integer	Deliveries in `delivered` status.
deliveries.failed	integer	Deliveries in `failed` status.
deliveries.pending	integer	Deliveries in `pending` status.

The `payload` object is what receivers actually get, with this shape: `id` (the event `uuid`), `type` (the `event_type`), `created_at` (ISO 8601), `data` (same as the top-level `data`), and `api_version`.

```
[
  {
    "id": 9001,
    "account_id": 11,
    "subject_id": 305,
    "subject_type": "Deliverable",
    "event_type": "deliverable.updated",
    "api_version": 1,
    "uuid": "3f1c8a02-7d4e-4b9a-9c1e-2a6b5d8f0e11",
    "created_at": "2026-06-02T09:58:12Z",
    "updated_at": "2026-06-02T09:58:12Z",
    "data": { "deliverable_id": "dlv_4k2m", "state": "clicked" },
    "payload": {
      "id": "3f1c8a02-7d4e-4b9a-9c1e-2a6b5d8f0e11",
      "type": "deliverable.updated",
      "created_at": "2026-06-02T09:58:12Z",
      "data": { "deliverable_id": "dlv_4k2m", "state": "clicked" },
      "api_version": 1
    },
    "deliveries": { "total": 2, "delivered": 1, "failed": 0, "pending": 1 }
  }
]
```


Status codes

Code	When
200	Events returned (empty array if none match).
403	Caller is not a member of `account_id`.
404	`account_id` does not exist or caller is not a member.

`GET /webhooks/events/:id`

Fetches a single event by id, including its full `data`, the delivered `payload`, and delivery counts. Use this to inspect exactly what was sent for one occurrence. **Auth:** Bearer; **role:** read (any role — must be a member of the owning account).

Parameters

Name	In	Type	Required	Description
id	path	string	yes	Event id (prefixed or raw integer).

No parameters beyond the bearer token and path id.

Request

```
curl -H "Authorization: Bearer $TOKEN" \
  https://platform.phishspot.com/api/v1/webhooks/events/9001
```

Response `200 OK` — one event object, identical in shape to a single element of the index array above.


```
{
  "id": 9001,
  "account_id": 11,
  "subject_id": 305,
  "subject_type": "Deliverable",
  "event_type": "deliverable.updated",
  "api_version": 1,
  "uuid": "3f1c8a02-7d4e-4b9a-9c1e-2a6b5d8f0e11",
  "created_at": "2026-06-02T09:58:12Z",
  "updated_at": "2026-06-02T09:58:12Z",
  "data": { "deliverable_id": "dlv_4k2m", "state": "clicked" },
  "payload": {
    "id": "3f1c8a02-7d4e-4b9a-9c1e-2a6b5d8f0e11",
    "type": "deliverable.updated",
    "created_at": "2026-06-02T09:58:12Z",
    "data": { "deliverable_id": "dlv_4k2m", "state": "clicked" },
    "api_version": 1
  },
  "deliveries": { "total": 2, "delivered": 1, "failed": 0, "pending": 1 }
}
```


Status codes

Code	When
200	Event returned.
403	Caller is not a member of the account that owns the event.
404	No event with that id.

27.15 Outlook Add-in version (public)

`GET /outlook/version`

Returns the current Outlook add-in release metadata. **No authentication required.** Cached ~5 minutes.

Parameters: none.

```
curl https://platform.phishspot.com/api/v1/outlook/version
```

Response `200 OK`

Field	Type	Description
`latest`	string	Latest add-in version.
`min_supported`	string	Oldest version still allowed to run.
`bundle_filename`	string	Sideload bundle filename.

Field	Type	Description
`bundle_sha256`	string	SHA-256 of the bundle.

```
{ "latest": "1.1.0", "min_supported": "1.0.0", "bundle_filename": "phishspot-outlook-
        sideload-v1.1.0.zip", "bundle_sha256": "…" }
```

Useful for inventory tooling that verifies which add-in version your fleet should run. See Chapter 20.

27.16 Spam-whitelist download (separate token system)

The mail-admin self-serve URL from Chapter 22 uses a different scheme — a 64-character token embedded in the path, **no** `Authorization` header:

`GET /integrations/spam/:token/:format`

Name	In	Type	Required	Description
`token`	path	string (64 hex)	yes	The whitelist token from the Integrations panel.
`format`	path	enum	no	One of `txt` (default), `json`, `csv`, `md`, `microsoft365`, `google-workspace`, `mimecast`, `proofpoint`, `postfix`, `spamassassin`.

Possession of the URL is the only credential — treat it as a password and rotate it from **Account settings → Integrations → Spam Filter Whitelist** if it leaks.

27.17 Rate limits

Rack::Attack throttles apply per source IP for unauthenticated endpoints and per token for authenticated ones:

Surface	Limit
Outlook pairing-code generation	10 / minute / IP
Outlook pairing-code polling	60 / minute / IP
Phishing report intake (add-in)	30 / minute / IP
Spam whitelist download	60 / minute / token

Exceeding a bucket returns `429 Too Many Requests` with a `Retry-After` header. The general authenticated API isn't otherwise rate-limited beyond infrastructure-level abuse protection; keep usage under a few hundred requests/minute/token or contact us to raise it.

27.18 Cross-references

- Creating and managing API tokens in the admin UI: Chapter 14 API Tokens.
- Drive these same capabilities from an AI client with natural language: Chapter 29 MCP Server.
- The push-based counterpart to polling these endpoints: Chapter 26 Webhooks.
- The spam-whitelist endpoint in context: Chapter 22 Spam Filter Whitelist.
- The Outlook add-in that consumes `outlook/version`: Chapter 20 Outlook Add-in.

Entra ID: tradeoffs to consider before connecting

PhishSpot can connect to Microsoft Entra ID (formerly Azure AD) to pull users and groups directly from your directory — see Chapter 25 Directory Sync for how. We support the integration, customers use it in production, and it works as documented.

This chapter exists because, for most organisations, **we don't think you should turn it on.** The decision to grant directory-read scopes to a third-party SaaS is not a small one, and the convenience benefit is smaller than it looks once you account for the operational, security and compliance work it actually creates. The simpler alternative — quarterly CSV import — meets every realistic need of a phishing-simulation program at a fraction of the risk.

Read this before you click **Connect to Microsoft**. If you've already connected, read it before your next OAuth-grant review.

28.1 TL;DR — what we recommend

- **Default recommendation: do not connect Entra ID.** Use CSV import from your HR system instead — see Chapter 5 Contacts and the bulk operations described in §5.6.
- **Three reasons in one breath:** (1) the OAuth grant expands your attack surface by giving PhishSpot read access to your whole directory; (2) the integration couples your phishing program to Microsoft's availability and your tenant's policy decisions; (3) a phishing-simulation platform that trains employees to be suspicious of Microsoft-themed prompts shouldn't itself be a Microsoft-themed prompt.
- **If you connect anyway:** scope-review with your security team, register the processing with your DPO, time-box the OAuth grant and re-audit every six months. See §28.9.

28.2 What connecting Entra actually grants

The Entra connection in PhishSpot requests three Microsoft Graph scopes during admin consent:

- `User.Read.All` — read the **full** user object for every account in your tenant.
- `Group.Read.All` — read every security and Microsoft 365 group definition.
- `Directory.Read.All` — read group memberships and organisation-wide directory data.

PhishSpot uses a small slice of this — email, first/last name, job title, department, location, telephone, the `accountEnabled` flag and group membership. It ignores the rest. But the scope is **read everything**: mobile phone, manager chain, licence assignments, mailbox settings, sign-in metadata, and the dozens of other attributes Entra surfaces for every user. The OAuth token PhishSpot stores after admin consent can fetch any of those fields, at any time, until you revoke it.

CSV import inverts this. You choose which columns to export from your HR system (`first_name, last_name, email, department, title`). Nothing else reaches PhishSpot — not because we'd refuse it, but because you never sent it.

This is the **data minimization principle** in GDPR Article 5(1)(c): personal data shall be "adequate, relevant and limited to what is necessary in relation to the purposes for which they are processed." Granting `Directory.Read.All` for the purpose "run phishing simulations" is, on the face of it, hard to defend.

28.3 Security exposure

Three sub-arguments stack here.

The OAuth token expands your attack surface. PhishSpot stores a tenant-scoped Microsoft Graph token in its database. If PhishSpot is compromised — by a vulnerability in our code, a misconfigured cloud resource, a privileged-user mistake on our side — the attacker walks away with read-only directory access to **your** organisation. Before you connected, your directory was guarded by your own security perimeter. After, it's guarded by ours too. You've doubled the number of organisations that need to stay secure for your directory to stay private.

The phishing-simulation irony. The whole point of PhishSpot is to teach employees to be suspicious of unexpected Microsoft prompts: the "Your password expires today" email, the "Click here to view a shared document on OneDrive" link, the surprise sign-in screen. By prompting users for SSO via Microsoft to access **our** platform, we train them in the opposite direction — that an MS-branded sign-in prompt, in the context of phishspot.com, is normal and expected. We have built the trojan horse our own training warns against. (This irony applies more strongly to Chapter 24 SSO with Microsoft 365; the same principle is relevant here because Entra integration and SSO share the same OAuth surface.)

The audit trail outlives the integration. When you grant admin consent to PhishSpot, the OAuth grant is recorded in your Entra audit log forever. Disconnecting later doesn't delete that record. Five years from now, a new compliance auditor asks "why did this organisation grant `Directory.Read.All` to a third-party phishing vendor?" — and you, or your successor, has to explain.

28.4 Operational coupling

When you depend on a CSV file, you depend on a CSV file. When you depend on Entra sync, you depend on:

- **Microsoft Graph availability.** A Graph outage — even a partial one — degrades or halts your sync. PhishSpot retries, but if the outage spans multiple days, your contact list drifts away from reality.
- **Conditional Access policies.** Your tenant's CA policies can silently block PhishSpot's daemon flow. The next scheduled sync fails with an opaque error, but you don't see it until the next time you open the integration page — which, for most admins, is "when something looks broken."
- **Microsoft Graph schema changes.** Microsoft updates the Graph API on its own cadence. A field rename, a deprecation, a permission re-mapping — and our sync logic has to adapt. We do that work, but the gap between "Microsoft ships the change" and "PhishSpot ships the fix" can be days.

- **Token lifetime and refresh.** App tokens expire. We refresh them. Most of the time. When a refresh fails — admin password rotated, conditional access tightened, app marked as risky in MDM — your sync goes dark.

CSV has none of these failure modes. You upload, the data is loaded, and the only thing that ever changes is when you upload a new file.

28.5 GDPR / RODO — data minimization

We already mentioned GDPR Article 5(1)(c) in §28.2. There's more.

- **Lawful basis.** Article 6 requires you to have a lawful basis for processing personal data. "Legitimate interest" is the usual basis for phishing simulation — but legitimate interest requires a balancing test, and "we pulled the whole directory because it was convenient" doesn't survive balancing when "we pulled the five fields we needed via CSV" was right there.
- **Record of processing activities (Article 30).** Connecting Entra to PhishSpot is a new processing activity, with a new data flow, new sub-processor, new retention question. It belongs in your RoPA. CSV imports usually fold into an existing "phishing simulation" RoPA entry; Entra adds a fresh one.
- **DPO review.** In most EU and UK organisations, granting `Directory.Read.All` to a SaaS vendor triggers a Data Protection Impact Assessment under Article 35. That's weeks of process, often months. CSV import, with the same data subjects, often doesn't trigger a DPIA at all — the data scope is narrower and the sub-processor relationship is simpler.

28.6 The hidden admin cost

The pitch for Entra sync is "set it once, forget it." The reality is more work.

- Global Administrator must consent. This is one or two people in most orgs — and their calendars are full.
- DPO and legal review of the integration, RoPA entry, DPIA (sometimes).
- Conditional Access policy carve-outs, if your defaults block app-only flows.
- Quarterly audit of OAuth grants in the tenant (good security hygiene).
- Re-consent when Microsoft renames scopes or requires updated app registrations.
- Incident response when sync silently fails for a week and the next campaign targets ex-employees.

Versus the CSV alternative: any admin exports a five-column CSV from the HR system once a quarter, uploads it, done. No global-admin involvement, no DPO call, no audit follow-up. The "saved time" of Entra automation is mostly an illusion once you count the work it generates.

28.7 The simulation-program irony, in detail

This argument deserves its own section because it's the one PhishSpot-specific reason that doesn't apply to other SaaS integrations.

A successful awareness program teaches employees a few reflexes:

- An unexpected Microsoft-branded sign-in prompt deserves a second look.
- A "consent to grant this app access to your account" screen is something to read before clicking.
- A prompt that arrives via email and asks for credentials is presumed hostile until verified.

PhishSpot's Entra integration and Microsoft 365 SSO contradict each of these reflexes in our own UX:

- We **send** users a Microsoft sign-in prompt at `platform.phishspot.com/users/sign_in`. They click it, complete MFA, and the "trust the Microsoft prompt in this context" reflex strengthens.
- We **ask** the Global Administrator to click through a Microsoft consent screen granting our app `Directory.Read.All`. The reflex we want — "read this screen, question this scope" — is in tension with the integration we're asking them to complete.

The result is an employee population that's better trained for the simulations we send, but slightly worse trained for the real attacks they're supposed to be defended against. The training transfers less well to "an unfamiliar MS-themed prompt outside PhishSpot" because we made one **inside** PhishSpot familiar.

This isn't a fatal flaw — most awareness programs accept some imperfection — but it's a reason to default to the simpler integration path that doesn't produce the contradiction.

28.8 What we recommend

For nearly every organisation we work with, the right path is:

1. **CSV import via Chapter 5 Contacts.** Pick the columns you actually need (`first_name, last_name, email, department, title, groups`). Export from your HR system or Active Directory via PowerShell. Upload through the PhishSpot UI.
2. **Quarterly refresh.** Re-export and re-import once a quarter, or after major org changes (large hiring waves, reorganisations, M&A). The bulk operations in §5.6 handle removals.
3. **Group membership in the CSV.** PhishSpot accepts a comma-separated `groups` column — same outcome as Entra sync's group mirroring, without the sync.
4. **Risk scoring still works.** Risk score is computed by PhishSpot from campaign results, not from Entra. Switching off Entra sync changes nothing about how risk score behaves.

This path covers 95% of phishing-simulation needs. It's the path our long-running customers use even when their tenants are perfectly capable of supporting Entra sync.

28.9 If you connect anyway

We support Entra sync. Customers connect it. Sometimes there are genuine reasons: an organisation of 5,000+ employees where quarterly CSV doesn't track org changes fast enough, a regulated environment where IT mandates directory-driven onboarding for all SaaS, a phased rollout where the simulation program is one of many tools sharing the same Entra pipeline.

If you decide to connect Entra ID, do it deliberately:

- **Scope review with security.** Have your security team confirm `User.Read.All` / `Group.Read.All` / `Directory.Read.All` are acceptable under your vendor risk framework.
- **Time-box the OAuth grant.** Set a calendar reminder to review the grant every six months. If you've stopped using sync, revoke.
- **Register processing.** Add an entry to your RoPA. If you operate in the EU/UK and your DPO requires one, complete a DPIA.
- **Monitor sync health.** The activity log in §25.6 is the only signal of degradation. Configure your operations team to check it after every scheduled run, or set up a webhook on a `contact.updated` quota.
- **Keep CSV ready.** Have a current CSV export available for the day you want to disconnect — onboarding new contacts from CSV after Entra is gone is easier if you've kept the file.

Then proceed to Chapter 25 Directory Sync for the technical instructions.

28.10 Cross-references

- Chapter 5 Contacts — the CSV import path we recommend instead.
- Chapter 25 Directory Sync — technical instructions for the Entra integration this chapter argues against.
- Chapter 24 SSO with Microsoft 365 — separate decision from directory sync; the simulation-irony argument in §28.7 applies to SSO too.
- GDPR Article 5 — data minimization principle in EU law.

The same arguments apply to any future Google Workspace directory sync we might offer. They are not arguments against Microsoft as a vendor — they are arguments against OAuth-based directory sync between two unrelated SaaS products in general, and against the specific mismatch between phishing-simulation training and tight Microsoft integration in particular.

MCP Server (AI Integration)

PhishSpot ships an MCP (Model Context Protocol) server, so an AI client such as Claude can drive PhishSpot on your behalf — using natural language, against the same data and rules as the web app. The tool surface now mirrors most of what a human can do in the admin app: browse the template library, build and schedule campaigns, upload hosted images, manage contacts/groups/courses/domains/webhooks/autopilots, and read every result and trend.

29.1 Endpoint

The MCP server is available at:

```
https://platform.phishspot.com/mcp
```

It speaks JSON-RPC over HTTP. Point any MCP-capable client at that URL using the **HTTP transport**.

29.2 Authentication

The MCP server reuses PhishSpot **API tokens**. A token must be explicitly granted MCP access.

1. Go to **Settings** → **API Tokens** → **New Token**.
2. Give it a name and tick **Allow MCP access**.
3. Copy the token value (shown once).

> A token can act on **every account you belong to**, and MCP **write** actions require an **admin or editor** role in the target account. Treat the token like a password.

29.3 Connecting Claude

For **Claude Code**, run (replace `YOUR_TOKEN`):

```
claude mcp add --transport http phishspot https://platform.phishspot.com/mcp \
  --header "Authorization: Bearer YOUR_TOKEN"
```

For **Claude Desktop** or another client, add a server entry:

```
{
  "mcpServers": {
    "phishspot": {
      "type": "http",
      "url": "https://platform.phishspot.com/mcp",
      "headers": { "Authorization": "Bearer YOUR_TOKEN" }
    }
  }
}
```

29.4 Safety: what sends and what doesn't

Almost everything the AI can do is **read-only** or **build-only**. The `build_*`, `create_*` and `set_*` campaign tools prepare a campaign up to the **review** step and **never send email to recipients** — a person launches each campaign from the PhishSpot UI.

A small, clearly-labelled set of **action tools** can trigger real email, so an AI can run a campaign end-to-end when you ask it to. Their descriptions open with a warning, and they require an admin/editor role:

Action tool	Effect
`schedule_campaign`	**Sends real email** — schedules a ready campaign to actually launch at a given time.
`reschedule_campaign`	**Sends real email** — moves a scheduled campaign to a new send time.
`start_autopilot`	**Starts a live program** — activates an autopilot that generates and sends recurring campaigns on a schedule.

`cancel_schedule`, `pause_autopilot` and `stop_autopilot` are the safe counterparts — they **stop** sends. Adding a sending domain provisions it and returns nameservers to set at your registrar; it does **not** register or buy the domain.

> If you want the AI to never send on its own, simply don't ask it to schedule or start anything — every other tool leaves a human in control of the launch button.

29.5 Available tools

Almost every account-scoped tool takes an `account_id` (`acct_…`). Call **`whoami`** first to discover the accounts and roles your token can use. Tools are grouped below by capability.

Identity & sending domains

Tool	What it does
`whoami`	List the authenticated user and the accounts/roles the token can act on.

Tool	What it does
`list_sending_domains`	List active and provisioning sending domains for an account.
`provision_sending_domain`	Add a BYOD sending domain and return the nameservers to set at the registrar.
`check_sending_domain`	Poll a sending domain's delegation, mail records, and whether it is sendable.
`list_platform_domains`	List every domain visible to the account (shared + BYOD) with state and sendability.
`get_platform_domain`	Full detail for one domain: verification status, expected DNS records, diagnostics, block reason.

Contacts & groups

Tool	What it does
`list_contacts`	List contacts in an account (paginated).
`import_contacts`	Import contacts from CSV or JSON; the `groups` column models waves/ segments.
`update_contact`	Update a contact's fields and/or replace its group membership.
`delete_contacts`	Delete contacts (skips any locked by an active campaign).
`list_groups`	List contact groups in an account.
`create_group`	Create a new contact group.
`delete_group`	Delete a group (unless it is locked by an active campaign).
`add_contacts_to_group`	Add contacts to a group (skips duplicates).
`remove_contacts_from_group`	Remove contacts from a group.

Phishing template library

Tool	What it does
`list_phishing_templates`	List curated or custom templates, filterable by category and search.
`get_phishing_template`	Get one template's full email + landing HTML/CSS and post-click action.
`list_phishing_categories`	List the template category tree (only leaf categories hold templates).
`build_campaign_from_template`	Build a draft campaign from a template; optionally add all contacts and stop at review. Never sends.

E-learning courses

Tool	What it does
`list_courses`	List courses usable by the account (own + global) with block counts and completion stats.
`get_course`	Get one course's details and an ordered summary of its blocks.

Media library (image hosting)

Tool	What it does
`upload_media`	Upload an image or CSS file (from a URL or base64) and get a **hosted URL** for emails/landings.
`list_media`	List the account's hosted media.
`delete_media`	Delete a media file.

> Email clients (Gmail, Outlook) strip `data:` image URIs, so embed `upload_media`'s hosted URL in campaign HTML rather than inlining base64.

Campaign building

Tool	What it does
`list_campaigns`	List campaigns with state and wizard progress.
`get_campaign`	Full status of one campaign, including what still blocks launch.
`create_campaign`	Create a draft campaign (settings).
`set_campaign_email`	Set the email subject and HTML body.
`set_campaign_landing`	Set the landing page and sending/landing domain.
`set_campaign_post_click`	Set the post-click action (training, awareness page, or redirect).
`add_campaign_recipients`	Add recipients (all, a group, or specific contacts). Leaves the campaign at review.
`build_campaign_from_spec`	Build a whole draft campaign in one call (settings → recipients).
`duplicate_campaign`	Duplicate a campaign into a fresh draft (with recipients). Never sends.

Campaign scheduling

Tool	What it does
`schedule_campaign`	⚠ **Sends real email** — schedule a ready campaign to launch at a given time.
`reschedule_campaign`	⚠ **Sends real email** — move a scheduled campaign to a new send time.
`cancel_schedule`	Cancel a pending scheduled send, returning the campaign to draft.

Results & reporting (read-only)

Tool	What it does
`get_campaign_results`	Engagement funnel plus per-group and per-department breakdowns.
`get_campaign_recipients`	Per-recipient delivery stage, training status and reply flag (filterable).
`get_recipient_timeline`	Chronological event timeline for one contact in a campaign.

Tool	What it does
`get_campaign_replies`	Replies recipients sent back to the phishing email.
`list_account_trends`	Phishing-susceptibility trends across campaigns over a date range.
`list_events`	Raw engagement events, filterable by campaign / contact / type.
`list_reported_messages`	Suspicious emails employees reported (sender/subject metadata only).

Webhooks

Tool	What it does
`list_webhook_endpoints`	List outbound webhook endpoints.
`get_webhook_endpoint`	One endpoint plus its recent deliveries (signing secret masked).
`create_webhook_endpoint`	Create a (disabled) endpoint; returns the signing secret once.
`update_webhook_endpoint`	Update an endpoint's name, URL or event subscriptions.
`delete_webhook_endpoint`	Delete an endpoint and its delivery history.
`toggle_webhook_endpoint`	Enable or disable an endpoint.
`list_webhook_event_types`	List the event types you can subscribe to (no account needed).

Autopilots (automated programs)

Tool	What it does
`list_autopilots`	List autopilot programs and their state/progress.
`get_autopilot`	One autopilot's config, target groups and recent campaigns.
`create_autopilot`	Create an autopilot in **draft**. Does not start it.
`update_autopilot`	Update an editable (non-stopped) autopilot.
`delete_autopilot`	Delete an autopilot (not while it is running).
`start_autopilot`	⚠ **Starts a live program** that sends recurring phishing campaigns on a schedule.
`pause_autopilot`	Pause a running autopilot.
`stop_autopilot`	Stop an autopilot permanently (irreversible).

29.6 Adding a sending domain (BYOD)

To send a campaign from your own domain (for example `your-org.com`):

1. Ask the AI to call `provision_sending_domain` with the domain.
2. Set the returned **nameservers** at your domain registrar.
3. Poll with `check_sending_domain` until it reports the domain is **active** and **sendable**.

Once active, the domain appears in `list_sending_domains` / `list_platform_domains` and can be used as a campaign's sending/landing domain. See also Domains.

29.7 Example: build a campaign from a template

A typical AI-driven flow, all build-only until you choose to schedule:

1. `whoami` → pick the `account_id`.
2. `list_phishing_categories` and `list_phishing_templates` → choose a template.
3. `build_campaign_from_template` (optionally `quick_launch`) → a draft campaign with recipients, sitting at review.
4. `get_campaign` → confirm there are no readiness errors.
5. Either launch it yourself in the UI, or ask the AI to `schedule_campaign` for a specific time (**this sends for real**).
6. After it runs, `get_campaign_results`, `get_campaign_recipients` and `list_account_trends` summarize who fell for it.

Every tool ships live on the platform with each deploy — no extra setup, no migration, and no per-tool configuration is required.

Designing Effective Campaigns

Chapter 4 walks you through *where to click* in the campaign wizard. This guide covers *what to put there* — how to design a simulation that behaves like a real attack, personalizes correctly, tracks every interaction, and ends in a teachable moment. It assumes you've read Campaigns and focuses on the design decisions inside each step.

A campaign in PhishSpot has five moving parts, and good design means making them consistent with one another:

1. **Sender identity** — the name and address the email appears to come from.
2. **The email** — subject and HTML body, personalized per recipient.
3. **The landing page** — what the recipient sees after they click (optional).
4. **The end action** — the "teachable moment" after a click or form submit.
5. **The recipients** — who receives it, and how the message is tailored to them.

The rest of this chapter takes each in turn, with emphasis on the platform mechanics — the merge tags and tracking keys — that the manual hasn't documented in depth before.

30.1 Personalization with merge tags (the "keys")

Merge tags are the `{{placeholders}}` you drop into your content that PhishSpot replaces with each recipient's real data at send time. They are the single most effective lever you have: a message addressed to "Jan" from "your IT team at [your company]" is dramatically more convincing than a generic blast.

Syntax. Wrap the tag name in double curly braces: `{{first_name}}`. Tags are **case-insensitive** and tolerate surrounding spaces — `{{First_Name}}` and `{{ first_name }}` both work. A tag PhishSpot doesn't recognize is left in the message **literally**, so a typo like `{{frist_name}}` will ship as visible text. Always send a test (see §30.6) to catch these.

The available tags differ between the **email** and the **landing page**, because each is rendered in a different context. Use only the tags valid for where you're writing — the editor validates this and won't let you save an email that references a landing-only tag.

Email subject & body — available tags:

Tag	Replaced with	Example
`{{first_name}}`	Recipient's first name	Jan
`{{last_name}}`	Recipient's last name	Kowalski
`{{full_name}}`	First + last name	Jan Kowalski
`{{email}}`	Recipient's email address	jan.kowalski@firma.pl
`{{position}}`	Recipient's job position	Senior Analyst

Tag	Replaced with	Example
`{{department}}`	Recipient's department	Finance
`{{company}}`	Your account name	Acme Sp. z o.o.
`{{campaign_name}}`	The campaign's name	Q2 Invoice Test
`{{landing_url}}`	The recipient's tracked link	https://officelogin.in/l/ab12cd34?d=…

Landing page & awareness message — available tags:

Tag	Replaced with
`{{first_name}}`, `{{last_name}}`, `{{full_name}}`, `{{email}}`	As above
`{{company}}`	Your account name
`{{landing_url}}`	The recipient's tracked link
`{{elearning_url}}`	The recipient's training link (used on the awareness page)

> `{{landing_url}}` is the tracked phishing link — there is no `{{phishing_url}}` tag. `{{company}}` resolves to *your* account name (the organization running the simulation), which is what makes "a message from your own company" pretexts work. For the full reference, see Template Variables.

Design tips:

- Personalize the **subject**, not just the body — `Jan, akcja wymagana na Twoim koncie` lifts open rates more than a generic subject.
- Put `{{landing_url}}` behind a real-looking button or link, never as raw text. Recipients rarely click a visible URL.
- Use `{{department}}` / `{{position}}` to make a pretext role-appropriate (an invoice for Finance, a benefits notice for HR). See Social Engineering & Persuasion for targeting strategy.
- Leave a tag out rather than risk it being empty — an awkward "Dear ," (no first name) is a giveaway. Empty values render as blank, so confirm your contact data is complete for the fields you rely on.

30.2 How tracking works, and why it shapes your design

PhishSpot records each recipient's journey through the campaign funnel — **Sent** → **Delivered** → **Opened** → **Clicked** → **Submitted** → **Trained**. Understanding *how* each stage is detected helps you design content that measures what you actually care about.

- **Opened** is detected by an invisible 1×1 tracking pixel embedded automatically in every email. The recipient's mail client loads it when the message is displayed. Because many clients (notably Apple Mail Privacy Protection and some corporate gateways) block or pre-fetch remote images,

treat the open rate as a *soft* signal — don't design a campaign whose success depends on open tracking alone.

- **Clicked** is detected when the recipient visits their personalized link. That link carries an opaque per-recipient **key** — the deliverable ID — as the `d` query parameter:

  ```
  https://<your-landing-domain>/l/<random-path>?d=<deliverable-id>
  ```

  This is the value behind `{{landing_url}}`. The key identifies exactly one (campaign, recipient) pair, which is how PhishSpot attributes a click to a specific person without putting their email address in the URL. **Never hand-edit or hard-code a link in your email** — always insert `{{landing_url}}` so the per-recipient key is preserved. A static URL would attribute every click to nobody.

- **Submitted** is detected when the recipient submits a form on your landing page (§30.3). Every field they type is captured, except routing and security fields — so you measure not just *that* someone submitted, but *what* they were willing to give up.

Tracking-only campaigns. If you only want to measure who clicks — without hosting a fake login — you can disable the landing page (Step 3). The link still records the click via its key, then immediately runs the end action (e.g. straight to an awareness page). This is the lightest-touch design and avoids capturing any credentials at all.

30.3 Designing the landing page

When enabled, the landing page is the destination of `{{landing_url}}`. It's plain HTML you control, hosted on your selected platform domain. Common patterns:

- **Login clone** — a copy of a Microsoft 365, Google, or internal portal sign-in screen. The classic credential-harvest test.
- **File-share / document** — "a document was shared with you, sign in to view."
- **Notification / action page** — "confirm your details," "review this policy."

Forms. You don't need to wire up a form action — PhishSpot rewrites any `<form>` on the page to post back to the tracking URL automatically. Whatever fields you include (username, password, etc.) are captured into the recipient's event record on submit, minus internal routing/security fields. Design the form to mirror whatever you're imitating.

You can use the landing-context merge tags (§30.1) here too — e.g. pre-filling `{{email}}` in a username field is a strong realism touch.

> Decide deliberately whether to capture passwords. Capturing the *fact* of submission is usually enough to drive training, and storing real passwords — even briefly, even your own employees' — raises the stakes of the exercise. Many programs capture submission without retaining the password value. Align this with your security and HR policies.

For the technical side of making the page (and especially the email) render correctly, see Email Client Compatibility.

30.4 The teachable moment (end action)

What happens after a recipient clicks or submits is where the simulation turns into training. PhishSpot offers four end actions (Step 4):

End action	Behavior	Use when
Nothing	Blank page	You only need the click/submit recorded; minimal disruption
Redirect to course	Sends the recipient to an assigned e-learning course	You want immediate, in-context training — the strongest learning moment
Awareness message page	Shows a custom "this was a simulation" page (HTML you write, supports merge tags)	You want a branded, reassuring explainer without a full course
Redirect to URL	Sends them to any external URL	You want them to land on, e.g., the real portal or an internal policy page

The most effective design is **redirect to course**: the recipient is most receptive in the seconds after they realize they fell for it. Pair the campaign with a short, relevant course — see Courses.

On the awareness page you can use `{{first_name}}` to address the recipient and `{{elearning_url}}` to offer an optional follow-up lesson. Keep the tone non-punitive (more on this in Social Engineering & Persuasion §32.7).

30.5 Sender identity & deliverability

The most carefully crafted email is worthless if it lands in spam. Two fields define the sender (Step 1):

- **Display name** (`from_name`) — what appears as the sender, e.g. `IT Security`.
- **From email** (`from_email`) — the address, which must be on the **platform domain** you select for the campaign.

A few realities to design around:

- The sending domain must be **active and not blocked** to launch. Pick a domain whose name supports your pretext — `officelogin.in` reads very differently from `random-string.xyz`. See Domains for provisioning and BYOD setup.
- Deliverability depends on the domain's SPF/DKIM/DMARC and reputation, configured during domain setup. A brand-new domain with no warm-up may go to spam regardless of content.
- If your test lands in spam, consult Spam Filter Whitelist — you may need recipients' mail admin to allow the simulation source.

Match the display name and domain to the pretext: a “Microsoft” email from `acme-internal.com` is incoherent and trains recipients on the wrong signal.

30.6 Test and verify before launch

Never launch a campaign you haven’t seen rendered. Before Step 6:

1. **Send a test email** to yourself (Campaign Actions → Send Test Email). Confirm: merge tags resolved (no stray `{{…}}`), the link works and lands on the right page, images load, and formatting holds.
2. **Check both desktop and mobile.** After launch, the per-recipient preview in Reports & Analytics §11.5 shows the exact email each person received with a desktop/mobile toggle — use it to verify rendering and personalization.
3. **Walk the full path** — click your own test link, submit the form, and confirm the end action fires (course, awareness page, or redirect) as intended.

A quick pre-launch checklist:

- [ ] Subject and body personalize correctly (test email received and read)
- [ ] `{{landing_url}}` is behind a button/link, click is tracked
- [ ] Landing page renders; form posts; submission recorded
- [ ] End action fires and points to the right course/page/URL
- [ ] Sender domain is active; test didn’t land in spam
- [ ] Email renders cleanly on desktop **and** mobile

See also: Campaigns · Template Variables · Email Client Compatibility · Social Engineering & Persuasion · Domains · Reports & Analytics

Email Client Compatibility

A simulation email only works if it renders the way you designed it. Unlike a web page — which runs in one of a handful of modern browsers — an email is opened in dozens of clients, each with its own rendering engine, and many of them are years behind the web. This guide covers the practical rules for writing email HTML that survives the trip to the inbox. It pairs with Designing Effective Campaigns, which covers the *content* side.

You write the email body in the HTML code editor in **Step 2** of the campaign wizard (see Campaigns §4.2). Everything below is about what HTML/CSS to put there.

31.1 Why email HTML is not web HTML

There is no single "email standard." Each client decides how much HTML and CSS it supports:

Client / engine	What to expect
Outlook (Windows, 2007–2021)	Renders with Microsoft Word's engine. No support for `float`, `position`, background images (without workarounds), modern CSS, or reliable `margin` / `padding` on `<div>`. The single most restrictive target.
Outlook 365 / outlook.com / new Outlook	Webview-based, much better — but still strips `<style>` in some contexts and rewrites CSS.
Gmail (web & app)	Good CSS support, but historically strips `<head>` / `<style>` in some views, clips long messages, and proxies images.
Apple Mail (macOS & iOS)	Best-in-class rendering, near-browser. Honors `@media`. Privacy Protection pre-loads images (affecting open tracking).
Mobile webview clients (Gmail/Outlook apps, Samsung Mail)	Variable; assume narrow viewport and inconsistent `@media` support.

The golden rule: design for the worst client your audience uses, then enhance. In a corporate phishing simulation that almost always means **desktop Outlook on Windows**. If your email works in Outlook, it works almost everywhere.

31.2 Use tables for layout, not `div`s

Modern CSS layout (`flexbox`, `grid`, `float`) does not work in Outlook. The reliable, decades-proven approach is **nested HTML tables** with a fixed-width content column (typically 600px) centered in a full-width background table.

```
<!-- Full-width wrapper holds the background; inner table holds the content. -->
<table role="presentation" width="100%" cellpadding="0" cellspacing="0" border="0"
       style="background-color:#f4f4f4;">
  <tr>
    <td align="center" style="padding:24px 12px;">
      <!-- Fixed 600px content column, centered -->
      <table role="presentation" width="600" cellpadding="0" cellspacing="0" border="0"
             style="width:600px; max-width:600px; background-color:#ffffff;">
        <tr>
          <td style="padding:32px; font-family:Arial, sans-serif; font-size:16px;
                     line-height:24px; color:#333333;">
            Hello {{first_name}}, your account requires attention…
          </td>
        </tr>
      </table>
    </td>
  </tr>
</table>
```

Key points:

- Set `cellpadding="0" cellspacing="0" border="0"` on every table — Outlook adds default spacing otherwise.
- `role="presentation"` tells screen readers the table is for layout, not data.
- Put spacing in `<td>` `padding`, not `margin` on inner elements — `margin` is unreliable in Outlook.
- Use the `width` **attribute** *and* a `width` / `max-width` style; Outlook reads the attribute, modern clients read the style.

31.3 Inline your CSS

Many clients strip `<style>` blocks from `<head>` (Gmail historically does in several views). The safe default is **inline styles** on each element:

```
<td style="font-family:Arial,sans-serif; font-size:16px; color:#333; padding:16px;">…</
        td>
```

Reserve a `<style>` block in `<head>` **only** for things that must use it — chiefly `@media` rules for responsiveness (§31.5) and `:hover` states — and treat them as progressive enhancement that some clients will ignore. Never rely on a class defined in `<head>` for layout that must work everywhere.

Other inline rules:

- Use web-safe fonts (Arial, Helvetica, Georgia, Tahoma, Verdana) with a generic fallback. Custom web fonts work only in a few clients.
- Always set explicit `font-family`, `font-size`, `line-height`, and `color` on text cells — don't inherit.

31.4 Outlook-specific survival kit

The Windows Outlook (Word) engine causes most “it looked fine in my test but broke for half the recipients” problems. Practical defenses:

- **Bulletproof buttons.** A CSS-styled `<a>` button collapses in Outlook. Use a table-based button, optionally with VML for Outlook:

```
<table role="presentation" cellpadding="0" cellspacing="0" border="0">
  <tr>
    <td align="center" bgcolor="#0067b8"
        style="border-radius:4px; mso-padding-alt:14px 28px;">
      <a href="{{landing_url}}"
         style="display:inline-block; padding:14px 28px; font-family:Arial,sans-
        serif;
                font-size:16px; color:#ffffff; text-decoration:none;">
        Verify your account
      </a>
    </td>
  </tr>
</table>
```

  Note `mso-padding-alt` — an Outlook-only property that restores padding the Word engine drops.

- **Conditional comments.** Target Outlook specifically with `<!--[if mso]> … <![endif]-->` to add fixes (e.g. fixed widths, VML) that other clients ignore.

- **Avoid `background-image`** on elements — Outlook ignores most of them. Use a solid `bgcolor` or a real `<img>` instead.

- **Image gaps.** Outlook adds whitespace under images. Set `display:block;` and `border:0;` on every `<img>`, and `font-size:0; line-height:0;` on image-only cells.

- **Don’t rely on `border-radius`, `box-shadow`, gradients, or `max-width` in Outlook** — they’re ignored. Treat them as nice-to-have for capable clients.

31.5 Responsiveness: desktop and mobile

Your simulation must be usable on a phone — a large share of recipients read email on mobile first, and a broken mobile layout reads as “fake.”

- **Start with a single fixed-width column (≤600px)** that already looks acceptable on desktop and degrades gracefully — many mobile clients don’t honor `@media` reliably, so the base layout must stand on its own.

- **Enhance with `@media` queries** in a `<head>` `<style>` block for clients that support them (Apple Mail, most native mobile apps):

```
<style>
  @media only screen and (max-width:600px) {
    .container { width:100% !important; }
    .stack     { display:block !important; width:100% !important; }
    .mobile-pad { padding:16px !important; }
  }
</style>
```

- **Stack columns on mobile.** A two-column desktop row should become two full-width stacked blocks on a phone (the `.stack` pattern above).
- **Tap targets:** make buttons at least ~44px tall and full-width-friendly on mobile.
- **Font sizes:** body text ≥14px (16px is safer); anything smaller is unreadable on a phone and looks suspicious.

31.6 Images

- **Host images, never embed them.** Base64 `data:` URIs are stripped by Gmail and Outlook. Upload images to the Media Library §10 and reference the hosted URL. (This is also why the platform serves campaign images from a real URL.)
- **Always include `alt` text.** Many clients block images by default, so the email's first impression is often *images-off*. Meaningful `alt` text on the logo and key images keeps the message coherent — and a good pretext should still read with images disabled.
- **Set explicit `width` and `height`** attributes so the layout doesn't jump while images load and so blocked-image placeholders are correctly sized.
- **Retina:** export images at 2× the displayed size and constrain with `width` / `height` for crispness on high-DPI screens.
- **Don't build the whole email as one big image** — it triggers spam filters, breaks with images off, and isn't selectable/personalizable.

31.7 Dark mode

Many clients (Apple Mail, Outlook, Gmail app) recolor emails in dark mode, sometimes inverting your backgrounds and text unpredictably.

- Don't assume a white background — a logo that's dark-on-transparent disappears on a dark background. Use a logo with a safe background or a version that works on both.
- Test in dark mode on at least Apple Mail and the Gmail mobile app.
- Avoid pure `#000000` text on pure `#ffffff`; slightly off values invert more gracefully.

31.8 Links and preheader

- **Keep URLs whole.** Don't break a link across lines or insert it as raw text mid-sentence. In PhishSpot you insert the tracked link via `{{landing_url}}` (see §30.2) — the platform handles the per-recipient key, so you only place the merge tag behind your button or link.
- **Preheader text** — the preview snippet shown in the inbox list — is the first line of body text by default. Add a deliberate, hidden-or-visible preheader near the top so the inbox preview supports your pretext rather than showing "View in browser" or a stray URL:

```
<div style="display:none; max-height:0; overflow:hidden; mso-hide:all;">
  Action required on your account before Friday.
</div>
```

31.9 Test, every time

There is no substitute for seeing the email in real clients:

1. **Send a test email** from the campaign (Campaign Actions → Send Test Email) to real inboxes you control — ideally one each of **Outlook desktop, Gmail (web + app), and Apple Mail (Mac + iOS)**.
2. **Check the per-recipient preview** in Reports & Analytics §11.5, which renders the exact email each recipient received with a **desktop/mobile toggle**.
3. **View with images off** and **in dark mode** at least once.

A pre-send rendering checklist:

- [ ] Layout built with nested tables, fixed ≤600px content column
- [ ] All CSS that matters is inline; `<style>` only for `@media` / `:hover`
- [ ] Buttons are table/VML-based (survive Outlook)
- [ ] Images hosted (no base64), with `alt`, explicit dimensions, `display:block`
- [ ] Renders on desktop **and** mobile; columns stack; text ≥14px
- [ ] Looks acceptable with images off and in dark mode
- [ ] Preheader set; tracked link inserted via `{{landing_url}}`

See also: Designing Effective Campaigns · Campaigns · Media Library · Reports & Analytics · Spam Filter Whitelist

Social Engineering & Persuasion

A phishing simulation is only useful if it's convincing. A message nobody falls for measures nothing; a message everyone falls for teaches nothing if it's a cheap trick. This guide explains *why* people click — the persuasion principles real attackers exploit — and how to apply them in PhishSpot to build simulations that produce honest results and, crucially, a teachable moment.

This is written for security and awareness teams running **authorized** simulations against their own organization. The goal is never to embarrass employees — it's to find and close the gaps before a real attacker does. Keep Designing Effective Campaigns open alongside this: that guide covers the *mechanics*, this one covers the *psychology*.

32.1 Why people click

Falling for phishing is rarely about intelligence. Attackers succeed by triggering fast, automatic decisions — exploiting how busy people process a flood of email on autopilot. The same six principles of influence (popularized by Robert Cialdini) underpin almost every effective phish:

Principle	The lever	Example pretext
Authority	We comply with figures of power	"Message from the CEO," "IT Security policy update"
Urgency / scarcity	A deadline short-circuits scrutiny	"Your account will be locked in 24 hours"
Social proof	We follow what others do	"12 colleagues have already completed this"
Reciprocity	We return favors	"Here's your bonus statement — confirm to receive it"
Fear	Threat narrows attention	"Suspicious login detected on your account"
Curiosity	An open loop demands closing	"A document was shared with you"

Each maps cleanly onto pretexts you can build in PhishSpot. The most effective simulations combine **one** dominant principle with strong realism — stacking three or four makes a message read as a scam.

32.2 Designing a believable pretext

A pretext is the *story* the email tells. Believability comes from consistency, not cleverness:

- **Plausible context.** The message should fit something the recipient actually expects — an invoice for someone in Finance, a shared file for a collaborator, a password-expiry notice that mirrors your real IT process.
- **Coherent sender.** The display name, address, and domain must match the story (see §30.5). A "Microsoft" email from an unrelated domain trains the wrong lesson.

- **Right tone and branding.** Match the voice and visual style of whatever you're impersonating. A corporate IT notice is terse and formal; a consumer brand is friendly. Use the email HTML techniques to reproduce a brand's look convincingly.
- **A single, clear call to action.** Real phishing asks for one thing. Multiple asks dilute urgency and raise suspicion.
- **Just enough personalization.** Use `{{first_name}}` and role data so it feels addressed to the person — but don't over-personalize in a way a real external attacker couldn't (that tests a different threat model; see §32.3).

32.3 Targeting and difficulty

Not every recipient should get the same email. Tailoring the pretext to the audience both improves realism and teaches you where the real risk sits.

- **Role / department awareness.** Use contact data (`{{department}}`, `{{position}}`) to choose pretexts that fit: invoice and payment lures for Finance, credential-reset lures for IT-heavy teams, benefits and HR notices for everyone, "executive request" lures for assistants and finance approvers (the classic Business Email Compromise vector). Segment recipients into Groups and run targeted campaigns.
- **Generic vs. spear.** A broad, lightly personalized lure ("your mailbox is full") models commodity phishing. A spear-phishing simulation references a real project, vendor, or person — much harder to spot, and the right test for high-value targets. Decide deliberately which threat model you're measuring.
- **Difficulty laddering.** Over a program, escalate. Start with easy, obvious lures to set a baseline and build the reporting habit; progress to subtler, well-branded, context-aware messages. A repeated easy test inflates your numbers without improving resilience.

32.4 The red flags you're planting

Every simulated phish should contain the *teachable signals* you want employees to learn to spot. Design them in on purpose, then map results back to them:

- **Mismatched / look-alike sender domain** (`micros0ft-support.com`).
- **Urgency and threats** ("act now or lose access").
- **Generic greeting** — which is exactly why you control it: send some recipients a `{{first_name}}` version and some a generic one, and compare. If personalization sharply raises your click rate, that's a finding worth teaching.
- **Unexpected attachment or link**, hover-mismatch between link text and destination.
- **Requests for credentials or payment** that bypass normal process.

When you debrief (via the course or awareness page), point employees at the specific flags *that message* contained. Concrete beats abstract: "this email asked you to log in via a link with a misspelled domain" lands better than "be careful online."

32.5 Localization and culture

A translated phish is a weak phish. Idiom, formality, and local business norms all signal authenticity:

- Polish recipients respond to copy written in natural Polish, with the right register and local references — not machine-translated English. PhishSpot's curated Polish templates are **written by a Polish-speaking team for this reason**, not auto-translated (see Phishing Templates).
- Localize the pretext, not just the language: the brands, banks, courier services, and government bodies impersonated should be the ones your recipients actually use.
- For multinational audiences, segment by language/region and run parallel localized campaigns rather than one lowest-common-denominator email.

32.6 Learning from the results

Persuasion is a hypothesis; the Reports & Analytics funnel is the test. Read results as signal, not a scoreboard:

- Compare **click** vs. **submit** rates — many people click out of curiosity but stop at handing over credentials. The gap tells you where awareness is holding.
- Break results down by **department/group** to find concentrated risk, not just an org-wide average.
- Watch the **trend over time** across a program — resilience improving campaign over campaign is the real success metric, not a single low rate.
- Track **reporting**, not just clicking — an employee who reports the simulation is the win condition. See Reported Messages.

32.7 Ethics and program hygiene

The line between a useful simulation and a harmful one is the care you take. A program that humiliates people destroys trust and drives under-reporting — the opposite of what you want.

- **Stay in scope and authorized.** Simulate only against your own organization, with leadership and (where required) works-council/HR sign-off. This is security awareness training, not an attack.
- **Avoid cruel pretexts.** Don't dangle bonuses, raises, layoffs, COVID/medical results, or anything that exploits genuine personal anxiety. Realistic ≠ heartless; these themes cause real distress and backlash and have made headlines for the wrong reasons.
- **Train, don't shame.** Make the end action a constructive moment — a short course or a reassuring "this was a simulation, here's what to watch for" — never a public list of who failed.
- **Reinforce reporting.** Reward the behavior you want: praise people who report, make reporting easy (see the Outlook Add-in), and treat a click as a coaching opportunity, not a mark against someone.
- **Protect the data.** Reconsider capturing real passwords (see §30.3); often recording *that* someone submitted is enough to drive training without storing sensitive values.

Run this way, a simulation does what it's meant to: it turns a moment of "I almost fell for that" into a durable habit of caution — and gives you the data to prove the program is working.

See also: Designing Effective Campaigns · Email Client Compatibility · Phishing Templates · Courses · Groups · Reports & Analytics · Reported Messages

Keyboard Shortcuts & Tips

- Use the account switcher (top of sidebar) to quickly move between teams.
- The Monaco code editor supports standard keyboard shortcuts: Ctrl+Z to undo, Ctrl+S to save, Ctrl+F to search.
- Click column headers in list views to sort data.
- Use browser back/forward buttons — PhishSpot uses standard URL navigation.

Glossary

Term	Definition
Campaign	A phishing simulation exercise sent to a group of recipients
Contact	A person in your organization who may receive phishing simulations
Group	A named collection of contacts used for targeting campaigns
Template	A reusable phishing email design (curated or custom)
Landing Page	The fake web page displayed when a recipient clicks the phishing link
Secured Domain	A verified email sending domain owned by your organization
Platform Domain	A domain used to host phishing landing pages
Course	A security awareness training module shown after a phishing interaction
Block	An individual content section within a course (lesson or quiz)
Funnel	A visualization of the conversion stages: Sent → Opened → Clicked → Submitted
Risk Score	A calculated metric showing how vulnerable a contact is based on their campaign responses
Webhook	An automated HTTP notification sent to an external system when events occur
API Token	A credential used to authenticate programmatic access to the PhishSpot API

End of Document