

AutoBounty User Manual

Penetration testing, bug bounty and reporting platform Version 1.0 — September 2026
Platform URL: <https://autobounty.com.br>

This manual contains no IP addresses and no credentials. Whenever a password is required, it is provided out of band (by the administrator or at installation time).

Contents

1. Introduction
2. Penetration testing fundamentals
3. Bug bounty fundamentals
4. Platform concepts
5. Getting started (signup, login, MFA)
6. Dashboards
7. Clients and projects
8. Scopes: defining the target
9. Available tools
10. Running tests on a scope
11. Reports
12. Built-in assistant
13. Administration
14. Security and best practices
15. FAQ and troubleshooting
16. Glossary
17. References

1. Introduction

1.1. What is AutoBounty

AutoBounty is a web platform to **plan, run, track and report** penetration tests and bug bounty activities in one place. With it you can:

- Register **clients** and testing **projects**;
- Define **scopes** (authorized targets: web, network, mobile, API, cloud);
- Run **scans** with well-known tools (Nmap, Nuclei, sqlmap, Nikto, and more) straight from the browser;
- Review and edit **findings** (severity, CVSS, CWE, evidence);
- Generate **professional reports** in HTML and PDF, in Portuguese or English;
- Use a **built-in assistant** that explains tools, glossary terms and workflows;
- Manage **operators**, signup approvals, plans and MFA.

1.2. Who it is for

- **Administrators:** responsible for the platform, access approvals and the global overview.
- **Operators:** professionals executing tests on the scopes of the projects assigned to them.

1.3. How to use this manual

- New to pentest/bug bounty? Read sections 2 and 3 first.
- Already know the theory and only want to **use the platform**? Start at section 5 (getting started) and go through section 10 (running tests).
- Section 15 (FAQ) solves the most common problems.

1.4. Accessing the platform

Open your browser at **<https://autobounty.com.br>**. The landing page shows the slogan **Find. Prove. Secure.** plus links to **About**, **Terms of Use and Ethics Policy** and **Request Access**. Logged-in users go straight to their dashboard.

2. Penetration testing fundamentals

This section gives the theoretical base used throughout the manual.

2.1. What a pentest is

A **penetration test** is an authorized, simulated attack against a system (website, network, mobile app, API) with the goal of **finding vulnerabilities before malicious attackers do**. The result is a report listing findings, each one's severity, and remediation guidance.

Key differences:

Concept	Meaning
Pentest	Authorized simulation, closed scope, report and deadline
Bug bounty	Continuous program where the organization rewards flaws found within its rules
Scan	Automated discovery phase; does not replace manual analysis
Vulnerability	Exploitable weakness (e.g., SQL injection)
Finding	An item recorded on the platform with severity and evidence

2.2. Box types

- **Black box:** the tester knows nothing about the infrastructure — simulates an external attacker. The most common bug bounty model.

- **Gray box:** partial knowledge (e.g., regular user credentials, general structure).
- **White box:** full access to code and infrastructure — allows deeper coverage in less time.

2.3. Phases of a pentest

1. **Reconnaissance (recon):** gathering information about the target — domains, subdomains, technologies, exposed ports, public records. Typical tools: Nmap, WhatWeb, Sublist3r, Amass, The Harvester.
2. **Scanning:** automated vulnerability discovery — outdated services, exposed directories, misconfigurations, known exploit templates. Typical tools: Nuclei, Nikto, Dirb/Gobuster, SSLscan, WPScan.
3. **Exploitation:** practically confirming the flaw is real and what the impact is — e.g., proving SQL Injection can read data. Typical tools: sqlmap, XSSStrike, Metasploit, Hydra.
4. **Post-exploitation and pivoting:** assessing how far the attack can go (access to other systems, privileges) **within the scope**.
5. **Reporting:** recording each finding with evidence, severity (CVSS), classification (CWE/CVE), impact and recommendation — this is the deliverable the platform generates for you (section 11).

On AutoBounty, phases 1–3 are supported by the scan tools (section 10); manual exploitation and reporting complete the work.

2.4. OWASP Top 10 (overview)

The **OWASP Top 10** lists the most critical risks in web applications. Knowing it helps interpreting findings:

1. **Broken Access Control** — users reaching data/functions they should not.
2. **Cryptographic Failures** — sensitive data poorly protected (plaintext passwords, weak TLS).
3. **Injection** — SQL injection, XSS, commands injected into unsanitized input.
4. **Insecure Design** — architecture flaws (no rate limiting, exploitable business logic).
5. **Security Misconfiguration** — insecure defaults, exposed directories, debug enabled in production.
6. **Vulnerable and Outdated Components** — libraries/services with known CVEs.
7. **Identification and Authentication Failures** — weak MFA, predictable sessions, unlimited brute force.
8. **Software and Data Integrity Failures** — unverified updates, compromised dependencies.
9. **Security Logging and Monitoring Failures** — nothing is logged, incidents go unnoticed.
10. **Server-Side Request Forgery (SSRF)** — server tricked into requesting internal addresses.

2.5. Severity: CVSS, CWE and CVE

- **CVE (Common Vulnerabilities and Exposures):** unique public ID of a known vulnerability (e.g., CVE-2021-44228).
- **CWE (Common Weakness Enumeration):** classification of the **weakness** (e.g., CWE-89 = SQL Injection, CWE-79 = XSS, CWE-326 = weak cryptography). AutoBounty uses CWE in findings and reports.
- **CVSS (Common Vulnerability Scoring System):** **0 to 10** score with a vector (e.g., CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H). Higher is worse.

Severity scale used on the platform:

Severity	CVSS range (reference)	Practical meaning
Critical	9.0 – 10.0	Total remote compromise, mass data exposure
High	7.0 – 8.9	Serious impact with simple conditions
Medium	4.0 – 6.9	Exploitation with some limitations
Low	0.1 – 3.9	Reduced impact
Informational	0.0	Technical context/interest, no direct impact

Always interpret **context**: a “medium” in a critical system can matter more than a “high” in a test environment.

3. Bug bounty fundamentals

3.1. What a bug bounty program is

A **bug bounty program** is a continuous initiative where an organization invites researchers to find vulnerabilities in its systems, **within published rules**, in exchange for recognition and financial reward. Unlike a pentest (a closed project with a deadline), the program runs year-round and any eligible researcher may take part.

3.2. Program scope

Every program publishes a **scope** stating:

- **In scope**: domains, URLs, apps and vulnerability classes accepted;
- **Out of scope**: everything that will not be accepted (e.g., third-party subdomains, internal networks, phishing, DoS);
- **Specific rules**: which vulnerability classes count, whether aggressive fuzzing is allowed, forbidden tools, and so on.

Golden rule: if it is not in scope, do not test it. Testing targets outside the scope — even if the flaw is real — breaks the rules and can have legal consequences.

3.3. Authorization and rules of engagement

- Authorization **comes from the program itself** (or from a signed pentest contract with the client). Without it, any test is a crime.
- **Safe harbor**: most programs commit not to prosecute researchers who acted in good faith and within the rules.
- **Always prohibited**: causing downtime (DoS), exfiltrating real user data, modifying data, phishing, pivoting to third-party systems, selling the flaw.
- **Keep impact minimal**: prove the flaw with the least damage possible (e.g., show you can read **one** record, not dump the whole database).

3.4. Classification and reward

Severity (section 2.5) usually drives the payout. Programs use their own scales (“Critical/High/Medium/Low”) and may pay **bonuses** (first to report, flaw in a critical component, report quality). A good report includes:

1. Objective title and impact summary;
2. Reproduction steps (prerequisites, requests, captures);
3. Evidence (without third-party real data);
4. Suggested severity with justification (CVSS vector);
5. Remediation recommendation.

3.5. Responsible disclosure

- **Coordinated disclosure:** you report to the program and wait for the fix before publishing — the default and the preferred model.
- **Public disclosure:** only when agreed (or after the program's disclosure window). Never publish before that.
- **Basic rule:** never disclose exposed user data; demonstrate the risk with synthetic data.

3.6. Ethics, responsibility and legality

- Test **only authorized targets** — AutoBounty requires accepting the Terms of Use and Ethics Policy at signup precisely for this reason.
- Handle evidence carefully; treat everything as confidential.
- If you find a flaw exposing real people's data, prioritize the safe reporting channel and remediation.
- Using the platform against unauthorized targets **violates the terms and results in a ban** (plus civil/criminal liability).

4. Platform concepts

4.1. Core objects

Object	What it is	Where
Client	Organization/person the test is performed for	Clients menu
Project	A pentest/bug bounty engagement for a client, with status and dates	Projects menu
Scope	An authorized target inside a project (URL, domain, IP, network) with type, difficulty and ports	Inside the project
Tool	Scanning software available on the server (Nmap, Nuclei...)	Tools menu

Object	What it is	Where
Scan task	One tool execution against a scope, with status, raw output and findings	Scan Results
Finding	A relevant result recorded in a scan (title, severity, CVSS, CWE, evidence)	Scan detail
Report	Project document in HTML/PDF with executive summary, findings and conclusions	Report button

4.2. Access roles

Role	Capabilities
Administrator	Everything: global dashboards, operators, signup approvals, plans, password/MFA resets, all clients/projects/reports
Operator	Their own clients, projects and scans; creates scopes, runs scans, edits findings and generates reports

Administrators see **all** operators' projects; operators see only the projects assigned to them.

4.3. Access plans

The public signup requests a plan, approved by the administrator:

Plan	Duration
Monthly	30 days from approval
Annual	365 days
Lifetime	Never expires
No plan	Access not granted yet

An **expired** plan blocks login until the administrator renews it.

4.4. Statuses and severities

- **Project status:** Planning → Active → Scanning → Completed (or Cancelled). While scans run it becomes **Scanning** and reverts to the previous status afterwards.
- **Scan task status:** Pending → Running → Completed / Failed (or Cancelled).
- **Finding severity:** Critical, High, Medium, Low, Informational (colors kept across the platform and reports).

5. Getting started

5.1. Requesting an account

1. At **<https://autobounty.com.br>**, click **Request Access** (or go to `/accounts/signup/`).
2. Fill in username, email, phone, **password** (follow the rules shown) and pick your **plan**.
3. Read the **Terms of Use and Ethics Policy** (link on the page) and tick the acceptance box — the submit button only unlocks afterwards. Your acceptance is recorded with date and terms version (audit trail).
4. Submit. The account is created **inactive** and an access request enters the administrator's queue. You will see **“Signup requested! Wait for admin approval...”**.

While the request is pending, login is not released.

5.2. Approval by the administrator

The administrator opens **Access Requests** in the top menu, reviews each request (username, email, plan, accepted terms) and clicks **Approve** (adjusting the plan if needed) or **Reject** (with a reason). On approval the account is activated and the plan starts counting from that day.

5.3. Login

1. Go to **<https://autobounty.com.br/accounts/login/>** (or the **Sign in** button on the landing page).
2. Enter **username** and **password**.
3. If MFA is already active, enter the **6-digit code** from your authenticator app. The login only completes after MFA.
4. If the administrator reset your password, the **Forced password change** screen appears first — choose a new password to continue.

Sign in with GitHub: if the administrator configured OAuth, the **Sign in with GitHub** button replaces the password (MFA still applies). An unknown email on the platform creates a new access request, just like the regular signup.

5.4. Mandatory MFA (first login)

The platform requires **TOTP-based MFA** for every user:

1. Right after the password, the **Enable MFA** screen shows a **QR code**.
2. Use an authenticator app (Google Authenticator, Authy, Microsoft Authenticator, 1Password...), scan the QR and enter the 6-digit code to confirm.
3. Next, the **Recovery Codes** screen shows **8 codes**. **Save them somewhere safe** (print them, password manager): each works once when you have no phone at hand.

Notes:

- Wrong code attempts are limited per session; when exceeded, the MFA session restarts and you must log in again.
- **Lost your phone?** Use a recovery code or ask the administrator to **reset MFA** (section 13.3).

5.5. Language

At the top right, the **globe/language** switcher toggles **(BR) Português** and **(US) English** instantly (reports also come in both languages). Portuguese is the default.

6. Dashboards

After login you land on the dashboard matching your role.

6.1. Administrative dashboard (</dashboard/admin/>)

- **KPIs**: operators (total/active), clients, projects, scopes and scans;
- **Overview chart** with resource distribution and scan states (completed, running, pending, failed);
- **Access requests** pending with quick approval;
- **Signup history** and **plans section** for every user (edit plan, expire now);
- **Lists**: recent projects, recent clients, last 10 scans;
- **Shortcuts**: new operator, new client, new project, tools;
- **News and CVE cards** (feeds refreshed on demand).

6.2. Operator dashboard (</dashboard/operator/>)

- **KPIs** for your resources: clients, projects, scopes, scans and **total findings**;
- **Chart** of your resources overview;
- Your **recent projects** (with scope counts) and **latest scans** with links to their detail pages;
- Shortcuts to create client/project and open the tools catalog.

6.3. News and CVE feeds

The cards show security news headlines and recent vulnerabilities (CVEs with CVSS scores), fetched from public sources (RSS/NVD). They refresh via the panel (</dashboard/feeds/>) without reloading the page. Informational only — they do not replace following official sources.

6.4. Attack monitoring (card and map)

The dashboards include the **Attack Monitoring** card, showing what is hitting the server itself in near real time:

- **Totals for the last 7 days** by category: SSH bans (brute force), web scans, firewall (UFW) blocks and invalid logins on the platform;
- **Recent events** with IP, country, type and count;
- **Animated map** with the geographic origin of the attacks and a marker for the server itself; arcs highlight the most recent events.

How it works: a collector runs every minute (`collect_threats`, via a systemd timer), reads the fail2ban log, the UFW kernel journal and the Apache access logs, stores the IPs (with cached geolocation) and the card refreshes itself every 30 seconds. Invalid logins on the platform are recorded in real time, straight from the login page.

Typical interpretation: SSH brute force from botnets (banned by fail2ban), port scans from automatic sweepers (blocked by UFW) and web scanners hunting `/.env`, `wp-admin`, backups etc. None of it should reach the application — the card exists to **confirm the barriers are working**.

7. Clients and projects

7.1. Registering a client

1. **Clients** menu → **New Client** (/projects/clients/create/).
2. Fill in **Name**, **Company**, **Email** (required), **Phone**, **Contact person** and the **Owner** (platform user who will handle the client).
3. Save. The owning operator sees the client in their list; administrators see everyone.

7.2. Creating a project

1. **Projects** menu → **New Project** (/projects/create/).
2. Fields: - **Project Name** and **Description**; - **Client** (required — if none exists, the form links to client creation); - **Assigned operator** (who will run the tests); - **Status**: Planning, Active, Scanning, Completed or Cancelled (start with **Planning** or **Active**); - **Start date** and **End date** (optional).
3. Save. You land on the project list and can open the detail page.

7.3. Detail, editing and deletion

From the project list use **View** (opens detail), **Edit** and the trash icon **Remove** (confirmation screen). The project detail shows:

- Header with status, client, operator and dates;
- Cards for **scopes** and **latest scans**;
- Buttons **New scope**, **Scan Results** and **Report**.

8. Scopes: defining the target

The **scope** is the heart of the test: it says **what may be tested**. Define it carefully — it is the line between a legitimate test and an unauthorized one.

8.1. Creating a scope

On the project detail page → **New scope** (/projects/<id>/scopes/create/):

Field	What it is	Example value
Scope Name	Friendly identification	Corporate website
Target (URL/IP/Domain)	The exact target to test	https://example.com
Scope Type	Web Application, Network, Mobile App, API or Cloud	Web Application
Difficulty	Easy, Medium, Hard, Expert	Medium
Ports	For network scans: 80,443,8080 or blank (default)	443,8443

Field	What it is	Example value
Additional Notes	Restrictions, test credentials, warnings	Staging environment

8.2. Rules for a good scope

- **Authorized targets only:** in bug bounty, copy the program's scope verbatim; in contracted pentests, use the contract's targets. Never add a domain out of curiosity that does not belong to the client.
- **One scope per logical asset:** split web app, API and admin panel into separate scopes — it organizes scans and the final report.
- **Subdomain boundaries:** decide whether `api.example.com` is included; record it in **Notes**.
- **Moderate port lists:** listing every port makes Nmap slow; enter the relevant ones.
- **Operational restrictions:** record maintenance windows, targets that must not receive aggressive fuzzing, etc.

8.3. Editing and removing

On the scope card use **Edit** or **Remove** (confirmation required). Already executed scans remain in the project history.

9. Available tools

The **Tools** menu (`/projects/tools/`) lists the server catalog, filterable by category:

Category	Tools	Purpose
Reconnaissance	Nmap, Sublist3r, Amass, The Harvester, WhatWeb, HTTPX	Ports/services, subdomains, public emails, technologies
Web Scanning	Nikto, Nuclei, WPScan, SSLscan	Misconfigurations, known CVEs, WordPress, TLS
Exploitation	sqlmap, XSSStrike, Metasploit, Hydra	Confirm SQLi, XSS, exploits, (authorized) brute force
Fuzzing	Dirb, Gobuster, Wfuzz, FFuf	Hidden directories, files and parameters
Post-Exploitation	Buster	Further enumeration after initial access

Important notes:

- The installer already ships the essential tools (Nmap, Nikto, sqlmap, Dirb, Gobuster, WhatWeb, SSLscan, Hydra and **Nuclei**). Tools not yet installed **still appear in the catalog**: when run, the raw output explains what is missing and how to install it.
- The catalog is managed by the administrator (`seed_tools` command during install/update).

- Heavy tools (Metasploit, Amass) need more resources — run them with patience and inside the scope.

10. Running tests on a scope

This is the core section of the manual: the full test flow on the platform.

10.1. Flow overview

```
Client → Project → Scope → Select tools → Run pending
      → Results → Scan detail → Edit findings → Report
```

10.2. Step 1 — Open the tool selector

On the **project detail page**, in the scope card, click **Run Scan** (/projects/<project>/scan/<scope>/). The screen lists the active tools with checkboxes, grouped by category.

- Check **the tools suited to the scope type** (see the table in section 9 and the recommendation in 10.8), or
- Check **Run all** to create tasks for every active tool.

Click **Run**. The tasks are created with **Pending** status and you are taken to **Scan Results**.

10.3. Step 2 — Execute the pending tasks

On **Scan Results** (/projects/<id>/scan/results/):

1. Check the status filter and the stats on top (total findings and per-severity counters);
2. Click **Run Pending** (only this button starts execution);
3. A notice confirms the scans started **in the background** — you may close the page; the server keeps working;
4. **For each tool that starts running, a mini-terminal opens in the table row itself**, showing the log output **in real time** (refreshed every ~2 seconds), with the command on top and a spinning status bar; when it finishes, status, badges and totals are updated at once.

Behavior details:

- The **Run Pending** button shows **Running...** and ignores repeated clicks; the first submission is always sent to the server;
- Terminal windows can be opened/closed at any time with the **View log** button (terminal icon) in the **Actions** column — also for already completed scans, which show the stored final output; windows you left open are restored after a page refresh;
- The project switches to **Scanning** during execution and **reverts to its previous status** when finished;
- Double clicks do not duplicate tasks (they are “claimed” at start);
- One failing tool **does not break the others**;
- Each tool has its own timeout (1–5 minutes); a scan exceeding it is marked as failed with the timeout output.

10.4. Step 3 — Monitor and inspect results

In the scan list, each row shows tool, scope, status, time and severity badges. The **View log** button (terminal icon) reopens the tool's mini-terminal with its full output. Click **View** for the **scan detail**:

- **Executed command** (e.g., `nmap -sV -sC <target> -p 443`) — useful for auditing what the platform ran;
- Complete **raw output** of the tool;
- **Findings table**: title, severity badge, **CVSS**, **CWE** and evidence;
- **Edit** (🔗) button on each finding.

What each tool usually delivers:

Tool	Typical findings
Nmap	Open ports/services (informational); insecure services (medium); vulnerability signatures (high)
Nikto	Insecure configuration items, reflected XSS, information disclosure (medium/high)
Nuclei	Rich findings: name, description, CVSS + vector , CWE , CVE , references and proof command — the most complete source
sqlmap	Injectable parameters (CWE-89 , critical)
Dirb/Gobuster	Hidden content — <code>admin</code> , <code>backup</code> , <code>.env</code> directories (medium); files with secrets (high)
SSLscan	Obsolete SSL/TLS protocols (CWE-326), weak ciphers (CWE-327), missing certificate
WhatWeb	Identified technologies (informational); notable old versions (medium)
Sublist3r	Discovered subdomains (informational)
Hydra	Placeholder — brute force requires manual configuration (see 10.7)

The parsers are heuristic. Every automatic finding needs **human validation** before the report: check the raw output, confirm the flaw non-destructively, and edit the finding when needed.

10.5. Step 4 — Editing a finding

On the scan screen, click 🔗 **Edit** (`.../findings/<n>/edit/`). You can adjust:

- **Title** — objective and specific;
- **Severity** — Critical/High/Medium/Low/Informational (use the scale from section 2.5);
- **Detail** — what was found, impact and context;
- **CVSS (score and vector)** — e.g., `7.5 + CVSS:3.1/AV:N/AC:L/...`;
- **CWE** — accepts 89 and normalizes it to `CWE-89`;

- **Confidence** — how sure the scanner is (helps the reader).

On save, the **severity summary** is recalculated and immediately appears in badges, charts and the report. Best practices:

- Remove/downgrade false positives (e.g., a port misread as a vulnerability);
- Raise severity when evidence shows real higher impact;
- Always keep the **evidence** objective (response snippet, reproducible request).

10.6. Step 5 — Review everything before the report

1. Filter by **Critical/High** and confirm each one;
2. Standardize titles (pattern: “ **in** ”);
3. Ensure every finding has CWE or CVSS filled in;
4. Run complementary tools on scopes that felt empty (e.g., SSLscan if only Nmap ran).

10.7. Manual exploitation (when the platform does not do it)

Some tests are **inherently manual**. The platform does not automatically run:

- **Brute force (Hydra)**: build the command yourself with a wordlist and authorized test credentials — e.g.: `hydra -l user -P /path/wordlist.txt ssh://target;`
- **Metasploit exploitation** and post-exploitation: run it on the server inside the scope and record the outcome as a manual finding in the report;
- **Logical tests** (IDOR, authorization bypass, payment flow): always manual — describe the steps in the report.

10.8. Recommended tool combos per scope type

Type	Suggested combo
Web Application	Nuclei + Nikto + Dirb/Gobuster + WhatWeb (+ sqlmap if parameters exist)
Network	Nmap (+ SSLscan if TLS exists)
API	Nuclei + WhatWeb (+ sqlmap on endpoints with parameters)
Cloud	Nmap + Nuclei (review findings manually)
Mobile	WhatWeb against the backend API + Nuclei; the APK itself is manual analysis

10.9. Finding life cycle and retest

Every finding is **born as its own record** (the **Finding** object), derived from the scan output but with state the tool does not have: status, assignee, history and dates. The full flow is:

Open → In Progress → Fixed → Retest → Closed

Transition	When to use it
Start Remediation	someone took ownership of the finding
Mark as Fixed	the client/team confirmed the fix
Request Retest	ask for validation (stamps the request date)
Retest Approved	the retest did not reproduce it → close with evidence
Retest Failed	it still reproduces → back to Open
Close / Reopen	dismissal (false positive) or finding recurrence

How to use it:

1. On the project, click **Findings** — the list has status and severity filters, ordered from the most severe to the least;
2. Open the finding to see description, evidence, request/response, reproduction steps, the **full history** (who changed what and when) and to set the **assignee**;
3. Use the **Life Cycle** buttons with an optional note (e.g. “not reproduced after the 10/12 patch”).

Important rules:

- Transitions are validated on the server: you cannot skip stages (e.g. straight from Open to Retest);
- If a **fixed finding reappears in a new scan**, it goes back to Open automatically, with the **Reappeared in New Scan** event;
- Duplicate findings (same title/endpoint on the same scope) are **deduplicated across runs** — the record is updated, not copied;
- Technical editing (section 10.5) updates the record without duplicating it, and the report keeps reading the same data source.

11. Reports

11.1. Viewing and exporting

- On the project detail page or the results screen, click **Report** (/reports/<project>/).
- On the report page, use **PDF in Portuguese** and **PDF in English** (/reports/<project>/pdf/?lang=pt and ?lang=en) to download the client-ready document.

11.2. Report sections

1. **Project Information** — client, operator, status, dates;
2. **Summary** — totals by severity and stat cards;
3. **Test Scope** — targets, types, difficulty and scan counts per scope;
4. **Executive Summary** — text for leadership: overall risk level, findings by severity, highlights and priority recommendations;
5. **Detailed Findings** — each finding with evidence, CVSS, CWE, references and recommendation;

6. **Conclusion and Recommendations per Asset** — a verdict per scope (approved/with reservations/critical) with what to fix first;
7. **Methodology** — phases applied and tools used;
8. **Disclaimer** — limits and use of the document.

Repeat the finding review (section 10.6) **before** generating the PDF: the report uses exactly the data recorded on the platform.

12. Built-in assistant

On every authenticated screen there is a **chat panel** (bottom right). It works **offline** from a local knowledge base:

- **“How do I use Nuclei?”** — explains the tool with usage examples;
- **“What is CWE-89?”** — answers from the security glossary;
- **“I want to run a web pentest”** — suggests the step-by-step workflow;
- **“Give me suggestions”** — contextual tips for the current project/scope;
- **“Remember: ...”** — stores notes in the assistant's memory (visible in the memory button; deletable per item);
- +/- on answers teaches the base; **Clear conversation** resets the history.

Optionally, the administrator can enable free-form LLM conversation by setting `AI_PROVIDER=openai`, `AI_API_KEY` (and `AI_API_BASE` for other compatible providers) in the server environment. Without it, the assistant answers only from the local base.

The assistant is educational: it **does not replace** your technical validation nor the scope rules.

13. Administration

Everything below requires the **administrator** role.

13.1. Operators

Operators menu (`/accounts/operators/`):

- **New operator** — username, email, temporary password, role;
- **Edit** — data and status (active/inactive);
- **Remove** — with confirmation;
- The **MFA** column shows who has enabled it.

13.2. Administrator password reset

On the operators list or the dashboard, the **Reset password** button (`/accounts/users/<pk>/password-reset/`):

1. The admin **types a new temporary password**;
2. The target user's **active sessions are revoked** immediately;
3. On the next login, the **password change is mandatory** before using the platform;
4. Admins can also change their own password at `/accounts/password-change/`.

13.3. MFA and recovery

- **Reset MFA** (/accounts/users/<pk>/mfa-reset/): wipes the secret — the user is forced to configure MFA again at login;
- **Hardware reset** (/accounts/users/<pk>/hardware-reset/): clears old recovery codes; new ones are generated in the user's flow;
- Users can regenerate their own codes at /accounts/mfa/codes/ after logging in.

13.4. Access requests

Access Requests (/accounts/signup-requests/): a queue with username, email, plan, terms status and date. **Approve** activates the account with the chosen plan; **Reject** discards it with a record of who decided and when (full audit trail).

13.5. Access plans

In the administrative dashboard, under the users section:

- **Edit plan** — Monthly/Annual/Lifetime/No plan, recomputing the expiration date from today;
- **Expire now** — blocks access immediately (useful for audits and cancellations);
- Expired plan → login is blocked with a notice until renewal.

13.6. Advanced administration (Django)

/admin/ opens Django's native panel (login goes through the **same password + MFA flow**). Use it for fine-grained maintenance: users, tools, scan tasks and migrations. Prefer the platform screens day to day.

13.7. Publishing new languages

Panel translations live in translations/en.json and are compiled with python scripts/i18n_tool.py build. The language switcher only shows what is published.

14. Security and best practices

- **MFA always on** — never share the secret or recovery codes; keep them in a vault/password manager.
 - **Strong, unique passwords** — the platform enforces complexity and the admin forces a change on reset.
 - **HTTPS enforced** — the address uses TLS with HSTS; if the browser warns, do not ignore it.
 - **Least privilege** — create operators only for those who need them and deactivate idle accounts.
 - **Authorized scope** — always review the target before running any tool; record restrictions in the scope Notes.
 - **Sensitive evidence** — treat scan outputs and reports as the client's confidential information.
 - **Backup** — the server administrator keeps periodic database copies (see the project README.md); set up a tested restore routine.
 - **Audit** — signup decisions, resets and password changes are recorded; review them periodically.
-

15. FAQ and troubleshooting

I cannot log in / “waiting for approval”.

Your request is still pending. Wait for the administrator to approve it under **Access Requests**.

My plan expired.

Login is blocked with a notice. Ask the administrator to renew the plan (section 13.5).

I forgot my password.

The platform sends no reset email: the **administrator resets** it via the reset button and you are forced to choose a new password at login (section 13.2).

I lost the phone with my MFA.

Use a **recovery code** (section 5.4) or ask the administrator for an **MFA reset** (section 13.3).

The scan does not start.

Only **Pending** tasks run — use the **Run Pending** button on the results screen. If they are **Running**, the page refreshes itself.

Scan failed / “tool is not installed”.

Open the scan detail and read the raw output: it states which package is missing and the install command. Ask the server administrator to install it and re-run the task.

Findings look wrong or duplicated.

They are heuristics — filter, validate against the raw output and **edit the findings** (section 10.5). Remove false positives before the report.

I cannot see other operators' projects.

Expected behavior: operators see their own; administrators see all.

The PDF came out without findings.

No findings are recorded or the scans are still running — finish execution and review before generating.

I want the platform in English.

Use the language switcher (globe) at the top; reports also have an EN version.

Can I test any target?

No. Only authorized targets — within the program/contract scope (sections 3.2 and 8.2). Anything else is a violation and a crime.

16. Glossary

Term	Meaning
Finding	Vulnerability/observation recorded with evidence
Bug bounty	Continuous reward program for flaws within rules

Term	Meaning
CVSS	0-10 severity score for a vulnerability
CWE	Weakness classification (e.g., CWE-89 = SQL Injection)
CVE	Public ID of a known vulnerability
Scope	Set of authorized test targets
Fuzzing	Mass sending of inputs to find flaws
MFA/2FA	Second login step (here: 6-digit TOTP)
OWASP	Organization publishing web security standards
Pentest	Authorized, contracted penetration test
Recon	Information gathering phase about the target
Scan	Automated execution of a tool against the target
TOTP	Temporary code algorithm (30 s) used by MFA
XSS / SQLi	Cross-Site Scripting / SQL Injection
Safe harbor	Program's commitment not to prosecute in good faith
Executive summary	Report section written for leadership

17. References

- **OWASP Top 10** — <https://owasp.org/www-project-top-ten/>
- **OWASP Testing Guide** — <https://owasp.org/www-project-web-security-testing-guide/>
- **NIST SP 800-115** — Technical Guide to Information Security Testing
- **CVSS (FIRST)** — <https://www.first.org/cvss/>
- **CWE (MITRE)** — <https://cwe.mitre.org/>
- **CVE (MITRE)** — <https://cve.mitre.org/>
- **AutoBounty Terms of Use** — <https://autobounty.com.br/accounts/termos/>
- **Project operations README** — README.md at the package root (installation, backup, updates).