



NxSAM Application Security — Quick Guide

Quick reference guide for the Application Security module

Product release	**NxSAM 2026.09**
Verified against code on	**2026-09-10**
Document revision	**b441b08-dirty**
Audience	**End users**
Document owner	**eSec Forte Technologies — Product Documentation**

Screenshots are taken from a demonstration data set. Figures are illustrative.

eSec Forte Technologies

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

Table of Contents

Application Security

- Overview
- Assess an application, end to end
- How an audit is created, and what happens next
- Reading the Application Security screens
- Application Security: every screen, column and action
- Application Security: find the right screen

Reference

- Glossary

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

Application Security

Application Security

Application Security covers weaknesses in software your organisation builds or runs — web applications, mobile apps and thick clients. Findings arrive from code scanners, dynamic scanners and manual testing, and are tracked together against the application they affect.

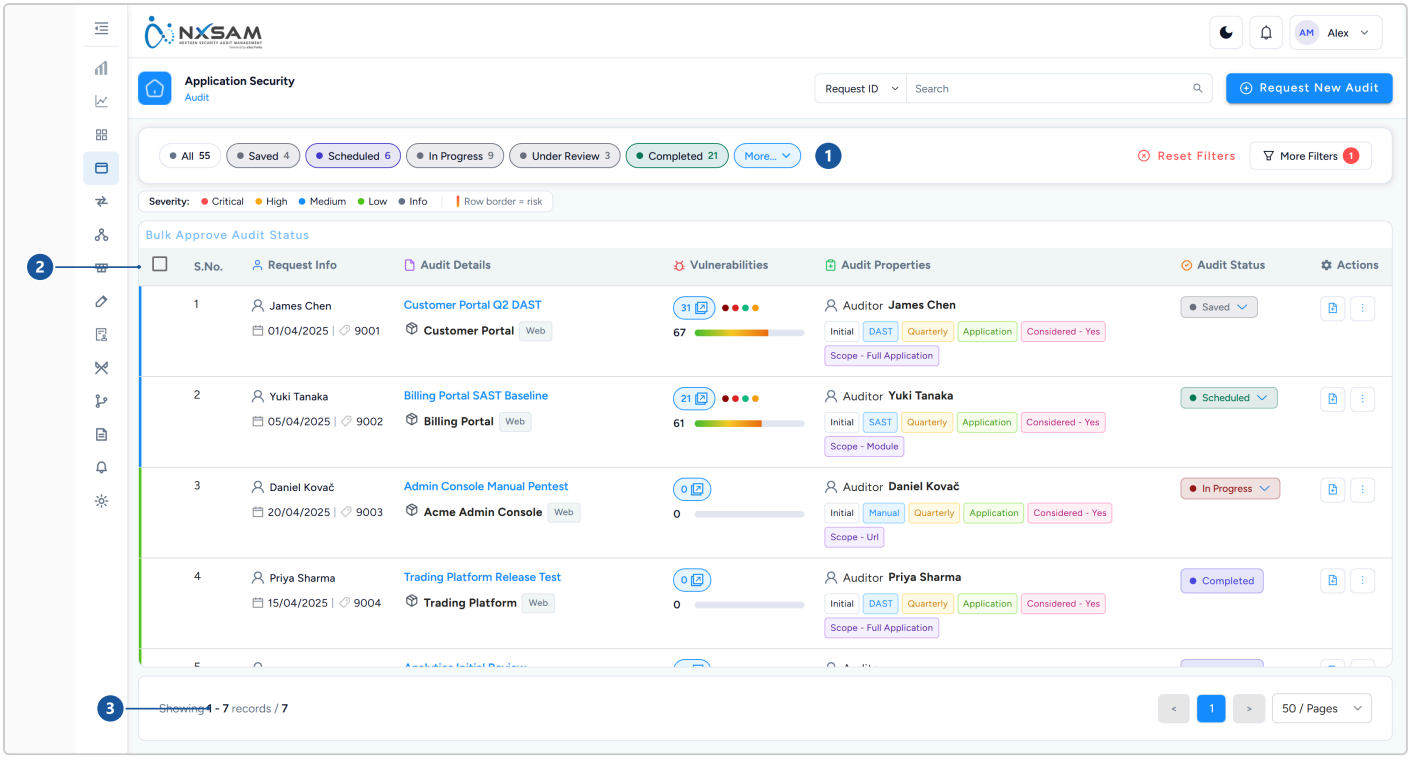
Where findings come from

Kind	Tools	What they see
Static (SAST)	Checkmarx, Fortify, Snyk Code	The source code. Finds flaws before the application runs.
Dynamic (DAST)	Burp Suite, Netsparker, Acunetix, AppScan	The running application, from the outside.
Composition (SCA)	Checkmarx SCA, Dependency Track, JFrog Xray	Third-party libraries and their known vulnerabilities.
Secrets	Gitleaks	Credentials committed into source control.
Manual	Your own testers	Anything the tools miss — logic flaws in particular.

The same application usually has findings from several of these at once, which is the point: they see different things, and no single tool is sufficient.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

Applications and audits



The AppSec audit list.

1 status filter chips · **2** what each audit records · **3** page size

An **application** is the thing you are protecting; an **audit** is one assessment of it. Applications persist; audits accumulate against them over time.

This is why the same application shows several audits, and why a finding always belongs to a specific audit — it records not only what is wrong but when and how it was discovered.

An application with many URLs or modules can be assessed as a whole or broken into parts, depending on how it was set up.

The working pattern

1. An audit is created for the application.
2. Findings arrive — imported from a tool, or entered by a tester.
3. Each finding is **reviewed and approved** into the audit. Until then it is not counted in dashboards.
4. Approved findings get an SLA by severity and become visible to the application owner.
5. The owner fixes; a retest confirms; the finding is validated and closed.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

Approval is a real step, not a formality

Findings awaiting approval are visible inside the audit but absent from dashboards and posture counts. If a colleague says "the dashboard is missing findings I can see", approval status is the first thing to check.

Severity and what to do first

Severity comes from the tool that reported it, mapped onto NxSAM's five levels. Tools disagree — the same weakness may be High in one scanner and Medium in another — so severity is a starting point, not a verdict.

Two things temper it in practice:

- **Asset criticality.** A Medium finding on a payment application usually matters more than a High on an internal wiki.
- **Exploitability.** Whether the weakness is reachable in your deployment.

False positives

Static scanners in particular report findings that are not genuine. NxSAM handles this two ways:

- **Automatically at import**, where the integration is configured with rules that classify certain findings as false positives on the way in.
- **Manually**, by marking a finding as a false positive after review.

Either way the finding is retained rather than deleted, so the decision is auditable and the same finding is not re-litigated after every scan.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

Assess an application, end to end

From onboarding an application to closing its findings, with what changes at each step.

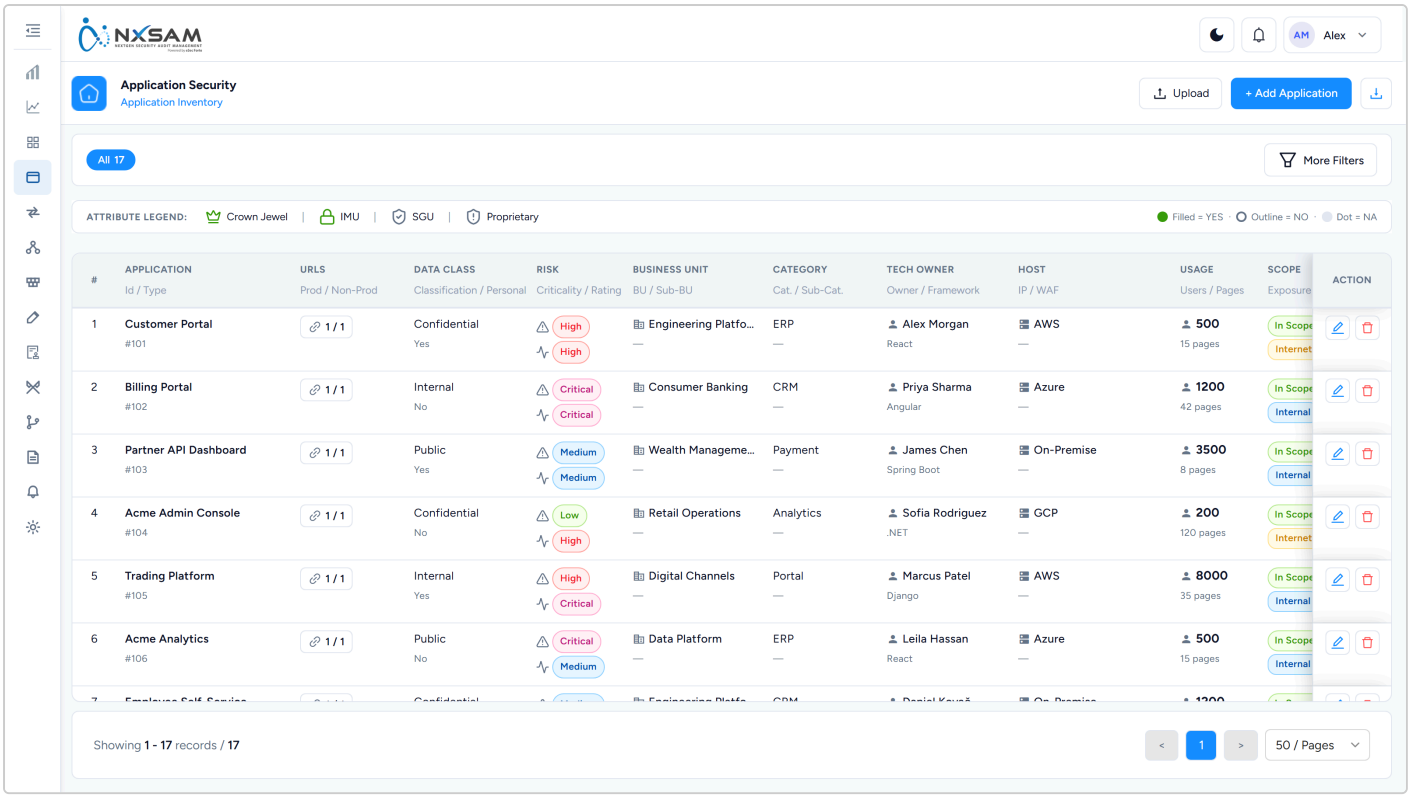
Before you start

You need	Why
The application's business context	It is what turns a finding into a decision
An audit type and category	They decide which properties are mandatory
An auditor account	An audit without one cannot be worked

1. Onboard the application

Application Security → Application Inventory → Add.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.



Application Security → Application Inventory

The inventory is filled by whoever onboards the application, not by any scanner. Two fields are routinely skipped and routinely wanted during an incident:

- **WAF** — whether one sits in front of the application.
- **Acceptable Downtime** — how long it can be offline.

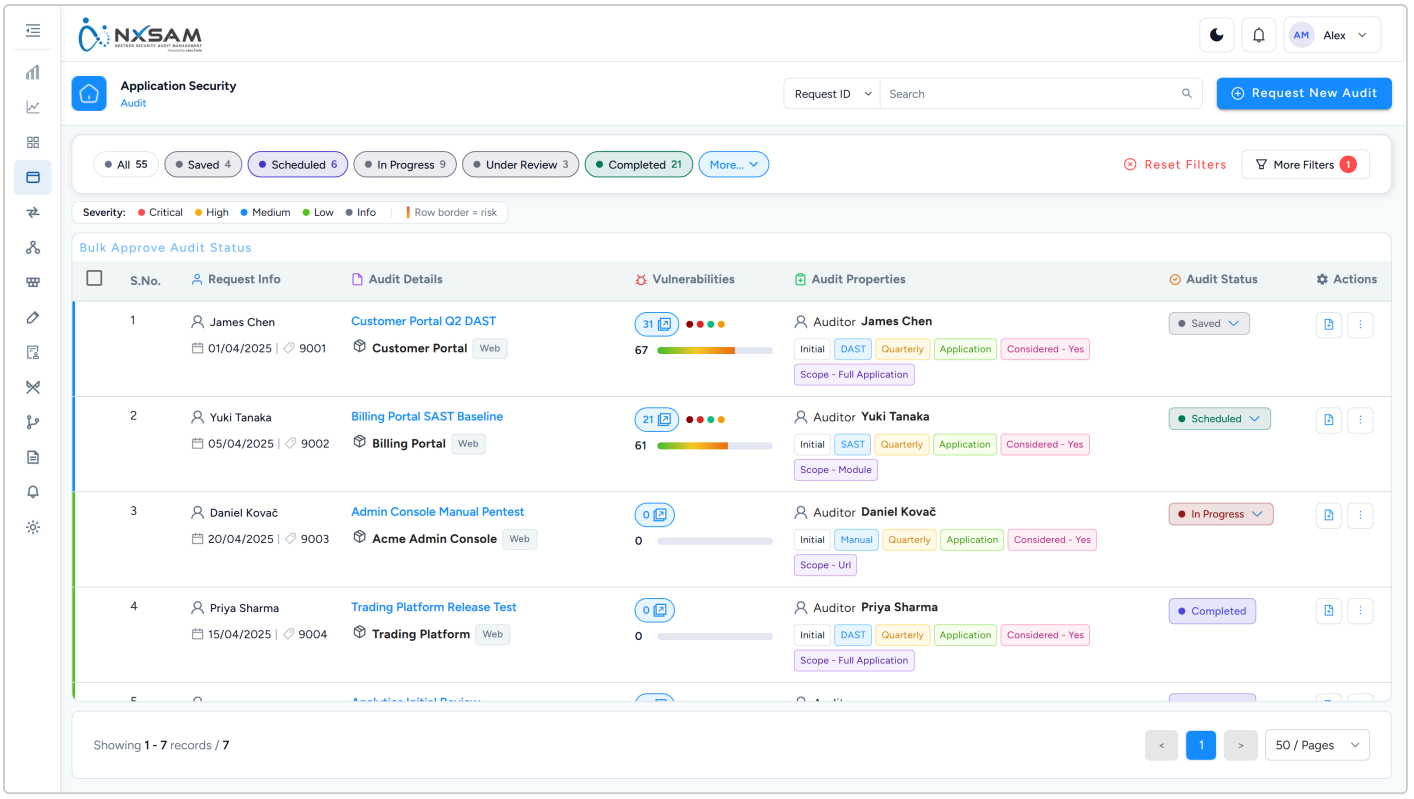
Together they decide what remediation is even possible. A Critical behind a WAF with zero acceptable downtime needs a virtual patch; the same Critical on an internal tool needs a deploy.

---

2. Request the audit

Application Security → Audits → Request New Audit.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.



Application Security → Audits

Choose the application, audit type, category and cycle.

After you save: the audit exists and appears in the list, with its properties shown as chips. Audits that arrive by import or from an integration often have fewer chips — the column was not in the file, or the tool did not know.

Assign auditors before the assessment is due to start.

---

3. Findings arrive

Findings are added by hand during a manual assessment, or imported from a scanner.

A scanner's count and NxSAM's count can differ legitimately

Integrations such as Checkmarx can classify findings on the way in, using your own triage rules. A finding auto-marked **false positive never enters the open population** — it is not behind a filter, and nothing on the screen explains its absence.

"The scan had 800 findings and NxSAM shows 600" is usually this, not data loss.

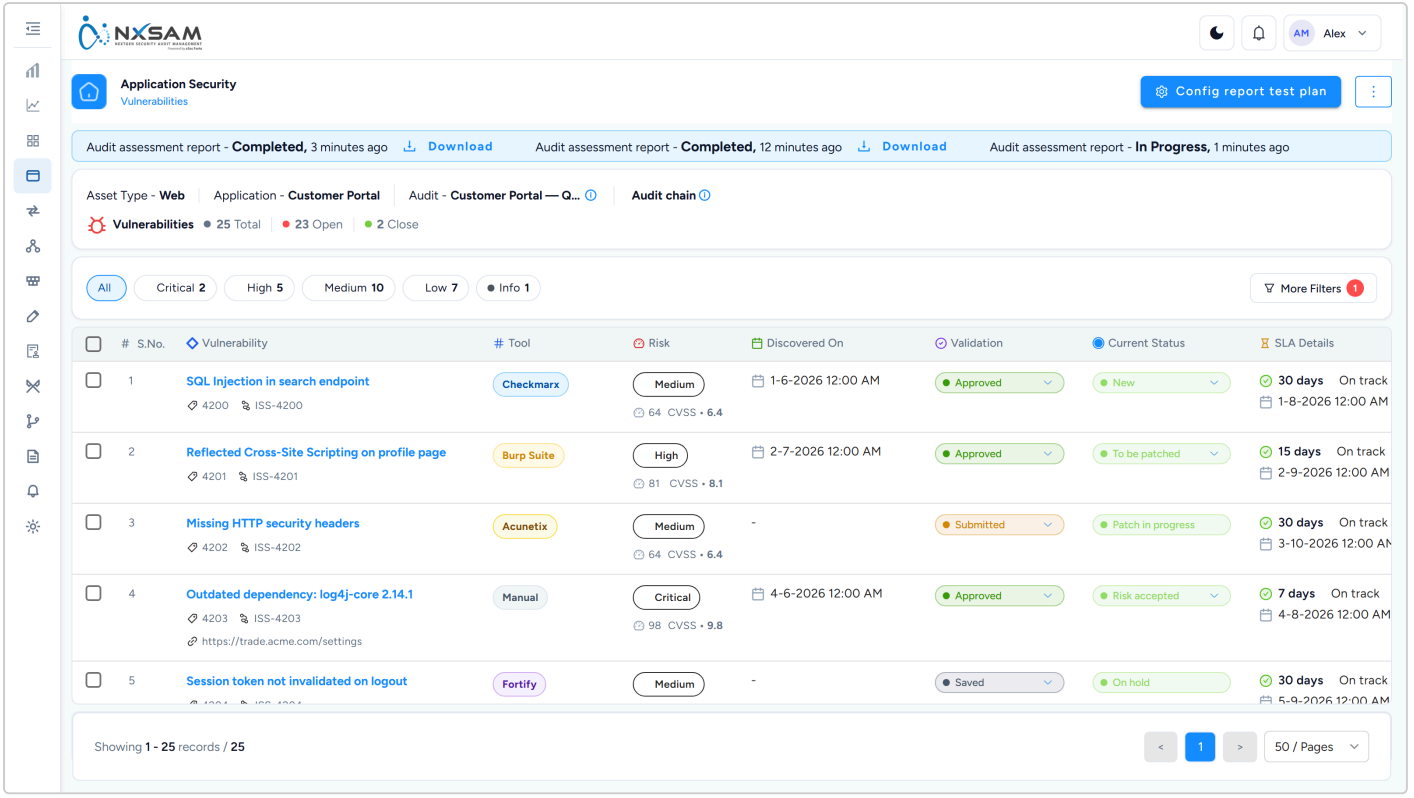
Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

4. Record the evidence

Each finding carries **Artifacts** — step, description and images, in order.

A finding whose artifacts are a single unlabelled screenshot cannot be re-tested by anyone but its author. This is the difference between an assessment that survives a handover and one that does not.

5. Approve



Application Security → Vulnerabilities

Move Validation to Approved.

After approval: the finding counts, **Discovered On is populated**, and its SLA starts.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

> ⚠ **Edit Vulnerability is not Edit Vulnerability Master Details**
>
> They sit next to each other in the same menu and look alike.
>
> **Edit Vulnerability** changes the row you opened. **Edit Vulnerability Master Details** changes the catalogue entry inherited by *every* occurrence of that vulnerability, on every asset, in every audit.
>
> Fixing a typo in the master is right. Recording something specific to one application through it is not, and there is no per-occurrence copy to fall back on.

---

6. Remediate and close

Move **Current Status** as work proceeds. Only **Validated** and **Exception Handled** close a finding — *Risk accepted* still counts as open, because accepting a risk is a decision about treatment rather than a fix.

Where the same weakness affects several applications built from one template, **Copy Vulnerability** reproduces it onto another asset while keeping each copy a genuinely separate finding with its own SLA and owner.

---

7. Re-test

Create a **confirmatory** audit against the parent. The form opens pre-filled with the name and today's date; both can be changed.

Then use **Compare** on the vulnerability screen to put the two audits side by side. It answers the only question a re-test asks — which of the original findings are actually gone — without reading two lists in parallel.

---

8. Complete

Move the audit to a final state. The date it reaches that state is the date every aggregate view files it under.

Only **considered** audits reach dashboards and reports. An audit marked *Considered — No* keeps its findings and shows them on the module screen while contributing to nothing aggregated — which is correct for a superseded audit or a trial run.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

When something looks wrong

Symptom	First thing to check
Finding not counted	Validation is not yet Approved
Discovered On is blank	Same — it populates on approval
Audit in the wrong month	The date it reached a final state
Text changed on findings you never touched	Someone edited the vulnerability **master**
Fewer findings than the scan reported	The import classification rules
An audit cannot be worked	No auditor assigned

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

How an audit is created, and what happens next

There are four ways an audit comes into existence, and they behave differently once created. A caller saying "the audit is wrong" usually needs a different answer depending on which of the four produced it.

The four routes in

1. Requested in the UI

Request New Audit on the module screen. Someone picks the application, the audit type, the category and the cycle, and the audit is created in `Saved`.

This is the only route where all the properties are chosen deliberately, which is why audits created this way have complete Audit Properties chips and imported ones often do not.

Every field on the request form, and what choosing it decides

The form has three parts: the first narrows *what* is being assessed, the second defines *how*, the third says *who*.

Context — each choice filters the next, so working top to bottom is faster than it looks:

Field	What choosing it decides
Org	The organisation the audit belongs to. Everything below is filtered to it
Application Type	Web, mobile, thick client and so on. It filters the application list, and it decides which audit properties become mandatory further down
Business Unit, **Projects**	Narrow the application list further. They change what you can pick, not the audit itself
Application Name	Required. The asset being assessed. Decommissioned applications are deliberately not offered

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

Audit definition — the choices that change how the audit behaves afterwards:

Field	What choosing it decides
Methodology	Required. Black box, grey box or white box — how much the tester was given. It bounds what the assessment could have found, which is the first thing an auditor asks about a result
Audit Type	Required. Your organisation's own list, administered under Configurations
Audit Cycle	Required. A scheduled round or a one-off — **and it decides which Audit Scope values are offered**. An ad-hoc audit may be scoped freely; a calendar-cycle audit is held to the full scope. A scope you expected can be missing simply because of the cycle chosen above it
Audit Scope	Required. Full or partial coverage. Partial scope is why "we tested that" can be true and still not cover the thing being asked about
Audit Category	Required. The classification the audit is reported under
Scope Length	How much was assessed — screens, endpoints or modules in scope. It is what makes two audits comparable when one covered four screens and the other forty
URL	The address assessed. On a multi-URL application this is the one this audit covers, not the whole application
Audit Name	Free text. On a re-test it arrives pre-filled with *Confirmatory* and the parent's identifier
Start Date, **Closure Date**	Planned dates. The closure date is a plan — not the date the audit counts from, which is when it reaches a final state

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

Field	What choosing it decides
Audit properties	Your own fields, defined under Configurations. Which are mandatory depends on the asset type and audit type **together**, which is why two audits can demand different fields
Remarks	Free text carried with the audit
Upload Documents	Scope documents, authorisation, rules of engagement — attached here they stay with the audit

Assignment:

Field	What choosing it decides
Auditor	Who works the audit. It can be left empty, and that is the commonest reason an approved audit sits with nothing happening
Comment	Sent to the approver with the request

2. Imported in bulk

A file of audits, staged and queued like any other import. Used when onboarding a programme of work that already exists elsewhere — a year of assessments being migrated in.

Two consequences:

- Properties absent from the file are absent from the audit. A blank cycle or category chip usually means the column was not in the import, not that someone forgot to set it.
- Import is asynchronous. Audits appear when the worker processes the batch, not when the upload returns.

3. Created by a tool integration

Some integrations create an audit for each scan they find. A Burp Suite scan, for instance, becomes an audit against the application it targeted.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

This is the route that surprises people: **audits appear that nobody created**. That is the integration doing its job. The audit is named after the scan, and its properties come from what the tool knew, which is usually less than a person would have entered.

If a module fills with audits after an integration is connected, this is why — and the fix is scoping what the integration collects, not deleting audits.

4. As a re-test of an earlier audit

A **confirmatory** audit is created against a parent audit to verify that fixes hold. The form opens pre-filled — the name as *Confirmatory* plus the parent's identifier, and the start date as today — both of which you can change before saving. Naming that parent is what makes the audit's chip read **Confirmatory** rather than **Initial**, and it inherits the parent's scope.

The chain matters for reporting: a confirmatory audit is evidence about the original, not an independent assessment, and the two should not be counted as two rounds of testing.

What "considered" decides

Only one audit per asset should normally count towards the organisation's figures — the latest one that reached a final state. That is what `Considered` marks.

> ⚠ **A considered audit is chosen, not automatic**
>
> An audit only becomes considered when it reaches a final state, and the date it did so is the date aggregate views use for it — not the date it was created, and not the date the scan ran.
>
> So a scan run in March, imported in April and completed in May counts as May in the dashboards. Support tickets about "findings appearing in the wrong month" are almost always this.

An audit marked **Considered — No** keeps all its findings and appears in the module, and is excluded from dashboards, posture counts and reports. Superseded audits, trial runs and assessments of decommissioned environments are the normal reasons.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

Status, and who can move it

An audit moves through its statuses in a defined order, and the dropdown offers only the legal next steps from where it is. An option that is missing is a transition that is not allowed from the current state, or one your role cannot perform — not a defect.

Some transitions require a comment — whether one is needed is a property of the status being moved *to*, so moving an approved audit to **Hold** asks for one, and the same is true of any status an administrator has marked as needing a reason. Bulk Approve applies one status change to every selected row, which is the reason the checkbox column exists and the usual tool after a large import.

A reverted audit comes back with its start date cleared

When an audit is sent back with **Reverted**, reopening it for edit unlocks the fields and blanks the **start date** deliberately. The audit is being re-run, so the original start date is no longer true, and leaving it in place would silently misstate how long the work took.

Set it again before saving — the form will not accept the audit without it.

Deleting an audit

Deletion is soft: the record is hidden rather than destroyed. That is why a deleted audit's name can still appear on historical findings and older reports, and why deleting an audit does not retrospectively change a report already produced.

If an audit was deleted to correct a mistake and its findings are still visible somewhere, that is the design rather than a failure to delete.

What to ask when an audit looks wrong

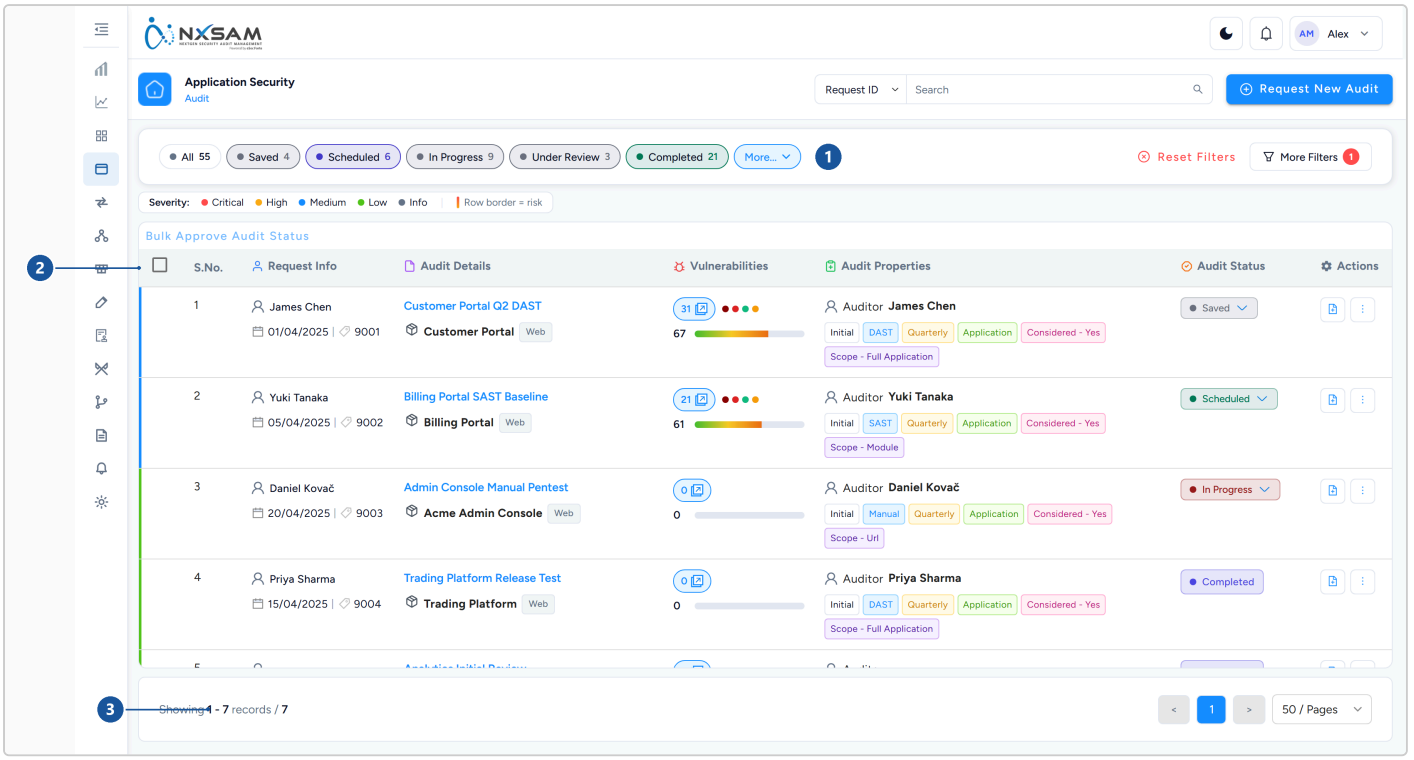
1. **Which route created it?** The chips tell you — a confirmatory chip, or missing properties suggesting an import or an integration.
2. **Is it considered, and when did it reach a final state?** This answers most "wrong period" and "not in the dashboard" questions.
3. **Is the status where the caller thinks it is?** Findings are not counted anywhere until they are approved, regardless of the audit's own status.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

Reading the Application Security screens

Both screens pack a lot into each row, and most of it is unlabelled — the chips, dots and bars all carry meaning that is not written on screen. This page names every element and says what it is telling you.

The audit list, row by row



One row of the audit list carries six separate pieces of state.

1 status filter chips · **2** what each audit records · **3** page size

The status chips above the table

`All`, `Saved`, `Scheduled`, `In Progress`, `Under Review`, `Completed`, and more behind `More…`. Each carries a count, and the counts are of **all** audits matching your other filters — not of the page you are looking at.

Clicking one filters the list. `All` clears the status filter but leaves the other filters in place, which is why the count sometimes does not match what you expect: `Reset Filters` is what clears everything.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

The severity legend and the row border

`Severity: Critical · High · Medium · Low · Info` followed by **Row border = risk**.

The coloured bar down the left edge of each row is the audit's highest open severity. It is the fastest way to scan the list — a red edge means at least one Critical is open in that audit, regardless of what the counts column shows.

Request Info

Who requested the audit, the date, and the request ID. The ID is what to quote in a support ticket: it identifies the audit unambiguously where names repeat.

Audit Details

The audit name, then the asset it runs against with its type badge (`Web`, `Mobile`, `Thick Client`, `API`). One application can have many audits; the audit name is what distinguishes them, so a naming convention pays off here.

Vulnerabilities

Three separate things in one column:

Element	Meaning
Count badge	Open findings in this audit. The arrow opens the finding list
Coloured dots	Which severities are present — one dot per severity that has at least one open finding
Number and bar	The audit's risk score, 0–100, coloured along a green-to-red ramp

The dots answer "how bad", the number answers "how bad overall". An audit with 31 findings and a score of 67 has more, milder findings than one with 8 findings and a score of 90.

Audit Properties

The chips that generate the most questions. Each is a separate property:

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

Chip	What it means
Initial / **Confirmatory**	Whether this is the first assessment or a re-test verifying earlier fixes
DAST / **SAST** / **Manual**	The kind of testing — dynamic, static, or hand-performed
Quarterly / **Annual** / …	The audit cycle it belongs to
Application / **Module** / **URL**	The bifurcation: whether the app was assessed whole or split into parts
Considered — Yes/No	Whether this audit counts towards aggregate figures
Scope — …	What was in scope for this particular run

> ⚠ **Considered is the one that surprises people**
>
> An audit marked **Considered — No** is fully visible here, with all its findings, and is **excluded from dashboards, posture counts and reports**.
>
> That is deliberate: a superseded audit, a trial run, or an assessment of a decommissioned environment should not move the organisation's numbers. But it means "the dashboard does not match the audit list" is expected behaviour for those audits, not a defect.
>
> Only an audit in a final state can be considered, and the date it reached that state is what aggregate views use as its date.

Audit Status

A dropdown, not a label — the status can be changed from here where your role permits it. Which statuses you can move to depends on the current one: the list offers only the legal next steps, so an option missing from the dropdown means that transition is not allowed from where the audit currently is, not that it is broken.

Some transitions require a comment; the product will ask.

Actions

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

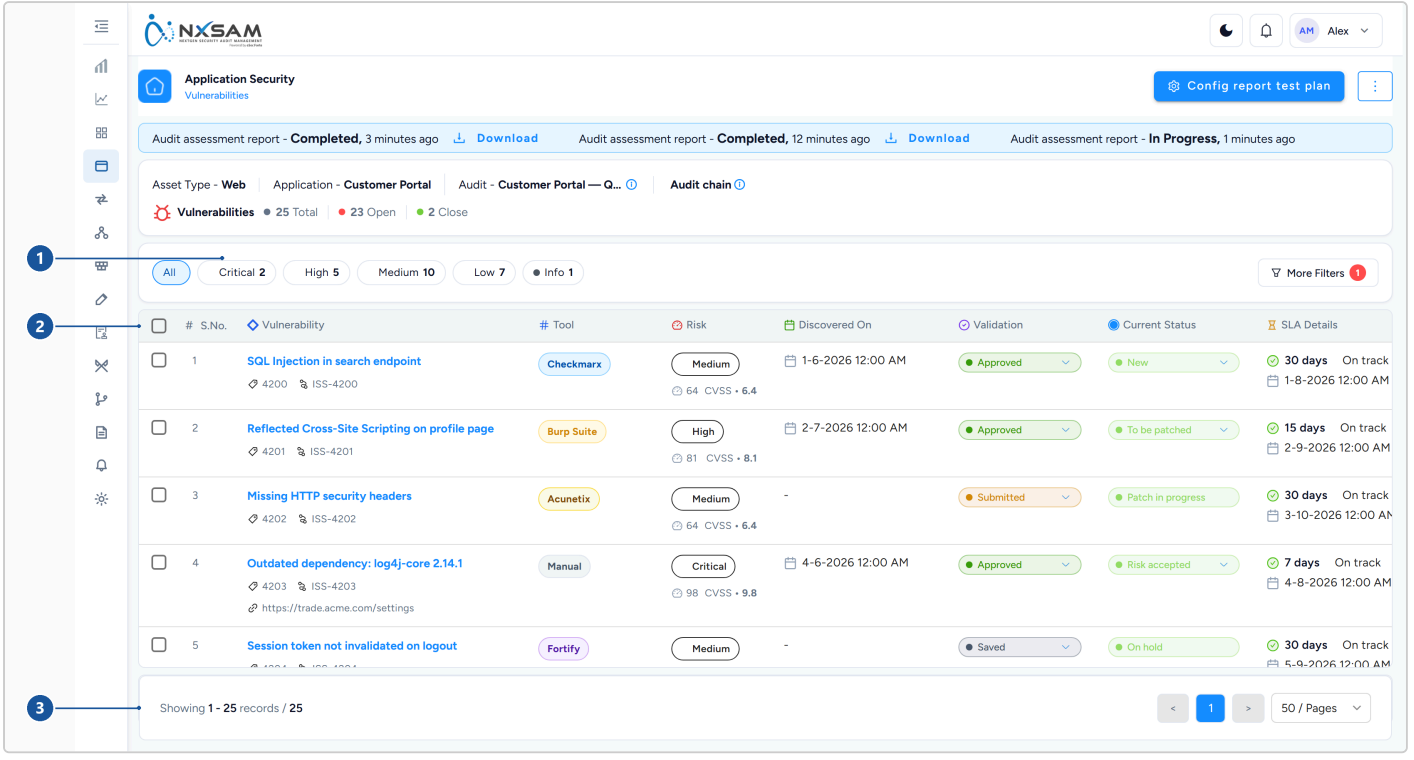
The per-row menu. What appears depends on the audit's status and your permissions, which is why two people looking at the same row see different options. Typical entries are managing the audit's scope, assigning an auditor, editing the audit, and downloading its report.

An action that is absent is almost always a permission or a status question rather than a fault — check the audit's state first.

Bulk Approve Audit Status

Acts on the rows selected by the checkboxes. Useful after a large import when many audits need the same status change, and the reason the checkbox column exists.

The vulnerability list



Findings within an audit. This is where an analyst spends most of the day.

1 filter by severity · **2** finding, asset and status · **3** paging

This is reached by drilling into an audit — the count badge in the Vulnerabilities column above. It always shows findings **within a scope**, so a finding you expect to see and cannot is often in a different audit rather than missing.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

Two status columns, not one

The column people read as "status" is actually two:

- **Validation status** — is this finding genuine and agreed? There are exactly five: `Saved`, `Submitted`, `Approved`, `Closed`, `False Positive`.
- **Client status** — what is being done about it? Fourteen values: `New`, `To be patched`, `Patch in progress`, `Patched`, `Not Patched`, `On hold`, `Reopen`, `Validated`, `Not Applicable`, `Risk accepted`, `Risk acceptance`, `Duplicate`, `Exception Handled`, `Expired`.

Only `Validated` and `Exception Handled` count as **closed**. Everything else — including `Patched` and `Risk accepted` — is still an open finding as far as every count in the product is concerned, which is why a list that looks resolved can report every row as open.

Both are dropdowns offering only the legal next steps. They move independently, and the distinction matters more than anything else on the screen — see the two status fields.

The remaining columns

Column	Notes
Issue / track ID	Quote this in tickets; it survives re-scans
Title	The finding, usually with its category and OWASP mapping
Severity	Derived from the score, so it changes if the score mapping is edited
CVSS / risk score	The raw score, and NxSAM's contextual score
Asset	Which application, and the affected URL where the tool reported one
Discovered on	**Last** seen, not first — a re-scan updates it
SLA expiry	Read the SLA from here, not by counting days from discovery
Tool	Which scanner reported it. The same weakness from two tools should be one finding; if it is two, check the vulnerability master mapping

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

Discovered today and overdue at the same time

Not a contradiction. A re-opened finding gets a new discovery date but keeps its original SLA expiry. The expiry column is authoritative.

Discovered On is blank until the finding is approved

The date is gated on validation status. A finding still `Saved` or `Submitted` shows an empty Discovered On cell — visible in the screenshot above on rows 3 and 5, which are the two rows not yet approved.

This is deliberate rather than missing data: before approval the finding is not counted anywhere, so it has no agreed discovery date to show. It appears the moment the finding is approved, and the SLA clock starts from it.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

Application Security: every screen, column and action

Complements Reading the Application Security screens, which covers the audit and vulnerability lists in narrative form. This page is the exhaustive list — every column and every menu item, including the ones that are easy to confuse with each other.

Application Inventory

The widest table in the product: seventeen columns, most of them stacked pairs. It is the business context that turns a finding into a decision, and it is filled by whoever onboards the application rather than by any scanner.

Column	Holds	Why it matters
Application — Id / Type	Application name, its NxSAM identifier and asset type	Identity and asset type
URLs — Prod / Non-Prod	The production and non-production addresses	Which environment a finding was raised against
Data Class — Classification / Personal Data	Data classification, and whether personal data is handled	Whether a breach here is a regulatory event
Risk — Criticality / Rating	Business criticality, and the rating derived from it	Business criticality, and the resulting rating
Business Unit — BU / Sub-BU	Owning business unit and sub-unit	Who owns the remediation budget
Category — Cat. / Sub-Cat.	Application category and sub-category	Application classification

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

Column	Holds	Why it matters
Tech Owner — Owner / Framework	The technology owner and the framework it is built on	Who can actually fix it, and in what
Host — IP / WAF	Address, deployment location, and whether a WAF is in front	Where it runs and whether a WAF sits in front
Usage — Users / Pages	Number of users and pages	Blast radius, and rough test effort
Scope — Exposure	Assessment scope, and internet-facing or internal	What was assessed, and how reachable it is
Attributes — CJ / PROP	Customer-journey and personal-information flags	Whether the application sits on a customer journey or handles personal data
Attributes — IMU / SGU	Internal management unit and service group unit	The internal ownership hierarchy
Status — Remarks	Lifecycle state and any note recorded against it	Lifecycle state and any note
Acceptable Downtime	The outage the business will tolerate	Decides whether a fix can be deployed immediately
End User Base — Size	Scale of the user population	How many people a change affects
Repository — Source URL	Where the source code lives	Needed for static scanning

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

WAF and Acceptable Downtime change what remediation is possible

These two are the operational reality behind a finding. A Critical on an application behind a WAF with zero acceptable downtime cannot be fixed the same way as one on an internal tool that can be restarted at will — the first needs a virtual patch, the second a deploy.

They are the fields most often left blank at onboarding and most often wanted during an incident.

The audit list

Columns and the twelve row actions are covered in Reading the Application Security screens. Two are worth naming here because they are not obvious:

- **Manage Test Summary** — a free-text record of what testing was actually performed. For a manual assessment this is the evidence of scope; without it, an audit cannot be reviewed after the fact.
- **View Audit Chain** — the original audit and every re-audit against the same scope, in order. The answer to "why are there four audits for this application".
- **Assign Auditors** — who performs the assessment. Until someone is assigned the audit can be requested and approved but not worked, which is a common reason an audit sits in `Approved` with nothing happening.
- **Vulnerability Overview** — the severity and asset breakdown for the audit without opening its finding list, useful when triaging a queue of audits rather than one.

As in API Security, the module ships two audit-table components with slightly different menus. Both perform the same operations.

The vulnerability list

Column	Notes
Vulnerability	The finding, with category and OWASP mapping
Tool	Which scanner reported it — a numeric master-type code rendered as a name

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

Column	Notes
Risk	Severity, derived from the score
Discovered On	Blank until the finding is approved
Validation	`Saved`, `Submitted`, `Approved`, `Closed`, `False Positive`
Current Status	The client status — fourteen values
SLA Details	Duration and expiry
Artifacts	Evidence attached to the finding

Row actions — nine

Action	What it does
Edit Vulnerability	This occurrence
Edit Vulnerability Master Details	The catalogue entry every occurrence inherits
Copy Vulnerability	Duplicates the finding onto another asset or audit
Edit CVSS Metrics / **View CVSS Metrics**	The CVSS vector, component by component
Edit Sequence	The order findings appear in the generated report
Vulnerability History	Every change to the finding, across audits

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

Action	What it does
View Status History	Status changes with who made them
Delete Vulnerability	Removes it from the audit

> ⚠ **Edit Vulnerability Master Details is not Edit Vulnerability**
>
> They sit adjacent in the same menu. The first changes the master catalogue — the title, description and remediation inherited by **every** occurrence of that vulnerability across every audit and asset. The second changes only the row you opened it from.
>
> Fixing a typo in the master is correct. Recording something specific to one application through it is not, and the dialog does not make the difference obvious.

Copy Vulnerability exists because the same weakness often affects several applications built from one template. Copying preserves the description and remediation while creating a genuinely separate finding, each with its own SLA and owner.

Page-level actions on the vulnerability screen

Action	What it does
Export Vulnerabilities	Downloads the current list, respecting the filters applied
Compare	Two audits side by side — what is new, fixed and unchanged
Trend analysis	How the finding profile has moved over successive audits

Compare is the re-test tool. After a confirmatory audit it answers the only question that matters — which of the original findings are actually gone — without reading two lists in parallel.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

Artifacts

Step, **Description**, **Images**: the reproduction path, in order. A finding whose artifacts are a single unlabelled screenshot cannot be re-tested by anyone but its author.

Vulnerability history

Audit ID, **Audit Details**, **Risk**, **Vulnerability Status**, **Created On**, **Status Changed By** — and it spans audits, so it shows a finding closed in one assessment and re-opened by the next.

Add Old Vulnerabilities

`vId`, **Vulnerability**, **Audit Name**, **Risk**, **Client Status**. Carries outstanding findings from earlier audits into a new one, preserving their history and original discovery date. Re-entering them by hand creates fresh findings that age separately.

Edit Tags

Tag ID, **Tag Name**. Tags are shared, so renaming one changes it everywhere.

The dashboard

Every widget exports individually via **Download PDF** and **Download PNG**.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

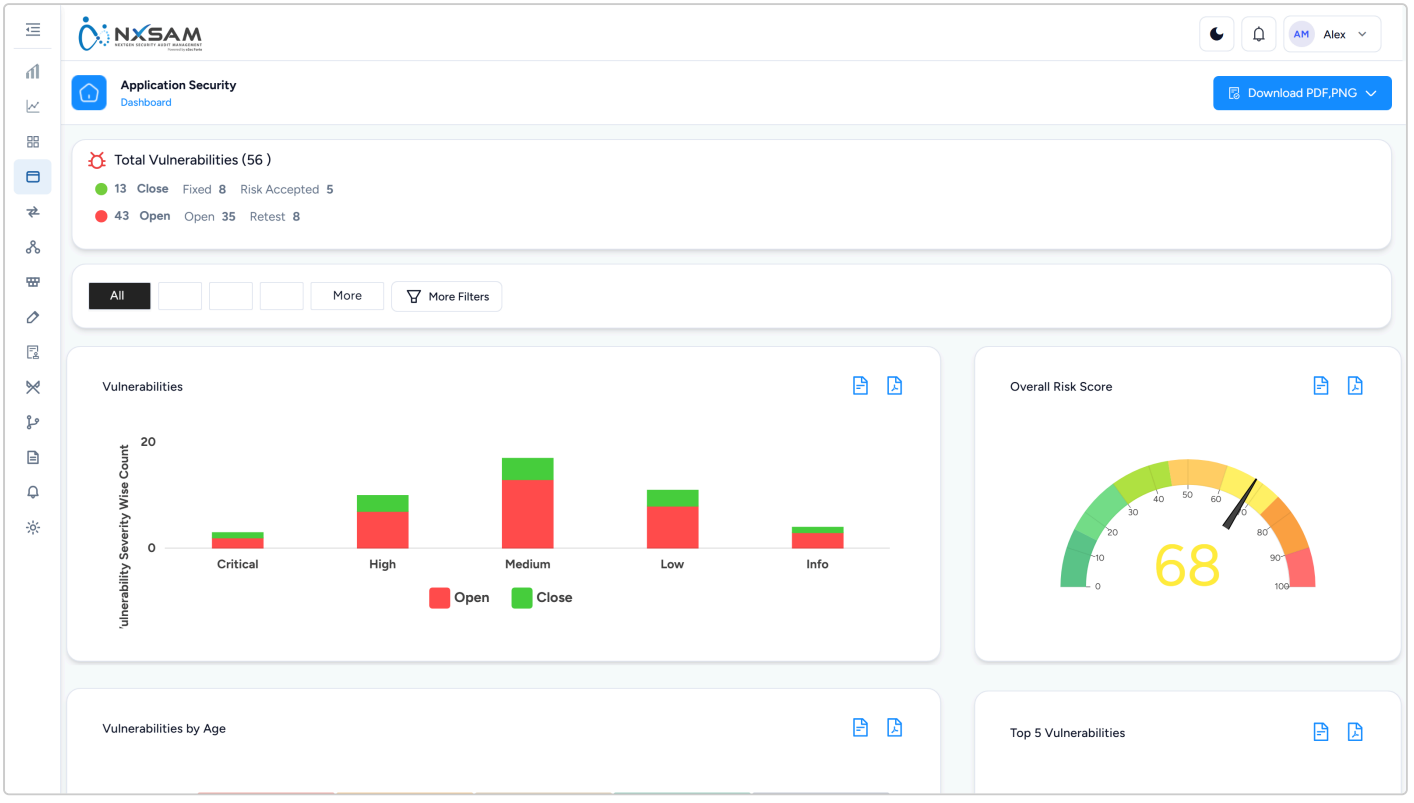
Application Security: find the right screen

Every screen in this module, what each one is for, and where it lives. Use it to find the right screen quickly, and to confirm you are on the one a colleague means — several screens in this product have similar names and different jobs.

Screenshots come from a demonstration data set, so the figures are illustrative. Click any screenshot to open it at full resolution.

AppSec Dashboard

Application security posture — severity mix, age, top applications and the audit timeline.



Application Security → Dashboards

Reference

Glossary

Terms as NxSAM uses them. Several are ordinary security words with a narrower meaning here, and those are the ones worth reading even if the term is familiar.

Assets and scope

Asset — anything that can hold a vulnerability: a web or mobile application, an API, a network host, a thick client, or a container image. Each type is a separate inventory with its own columns and its own licensed limit.

Asset property — an attribute recorded against an asset, such as criticality or whether it is internet-facing. Properties are more than description: they narrow SLA durations and feed the contextual risk score.

Bifurcation — splitting one application into separately assessed parts, so a large system can be tested module by module rather than as one unit.

Collection — a grouping of container images that comes from the scanning platform rather than from NxSAM, and is not editable here.

Scope — what an assessment covers. For applications this is the application; for compliance it is a host list, which is why compliance scope disputes are settled with an exported list of addresses.

Assessments

Audit — one assessment of one asset. It has a type, a category, a cycle, a scope, and a status of its own, separate from any finding's status.

Audit cycle — which round of a recurring programme an audit belongs to.

Confirmatory audit — a re-test created against a parent audit to verify that fixes hold. It is evidence about the original rather than an independent assessment, so the two should not be counted as two rounds of testing.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

Considered — the marker deciding whether an audit contributes to aggregate figures. Normally the latest audit per asset that reached a final state. An audit that is *not* considered still shows its findings on the module screen while contributing to no dashboard or report.

Initial audit — a first assessment, as opposed to a confirmatory one.

Findings

Vulnerability (also *finding*, *issue*) — one weakness on one asset, found by one assessment.

Vulnerability master — the catalogue entry (title, description, remediation) inherited by every occurrence of that vulnerability everywhere. Editing the master changes all of them.

Posture — Policy Compliance's unit: one control checked against one host. Every check produces a posture whether it passed or failed, which is why this module's volumes dwarf the others.

Artifact — the evidence attached to a finding: the steps, description and images that let someone else reproduce it.

Instance — which occurrence, where the same control or finding applies more than once to the same asset.

Status

Validation status — whether a finding has been confirmed. Nothing is counted anywhere until this reaches *Approved*, which is also when **Discovered On** is populated.

Client status (also *current status*) — what is being done about the finding. Of its values, only two close it.

Closed type — the flag deciding whether a status stops something counting as open. It is configuration rather than a fixed rule, so a deployment can change what "closed" means.

Posture status — what a compliance check returned: pass, fail or **error**. Error means the check could not run — a scanner or credentials problem rather than a compliance result.

Sub status — the qualifier beneath a main status: which kind of acceptance, or which stage of an approval.

Time and SLA

SLA duration — how long a finding of a given severity on a given asset type may remain open. Configurable, and narrowable by asset property.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

SLA expiry — the resulting deadline. Changing an SLA duration re-derives expiry dates, which can make a batch of findings overdue with no data change at all.

Discovered On — when a finding was first observed. Blank until the finding is approved.

Last observed — the most recent sighting, which is how a finding is known still to exist.

Overdue — past SLA expiry and not closed.

Decisions

Deviation (also *risk acceptance*, *RA*) — a recorded decision not to fix something, with an approval chain and an expiry. It does **not** close the finding: the exposure is still live and still counts as open.

Exception — a status marking an agreed exception rather than a resolution. Reporting that conflates the two overstates remediation.

Auto reject end date — when a temporary acceptance lapses on its own.

Integrations and processing

Integration — a connection to a scanning tool, either pulling from its API or parsing an uploaded report.

Vulnerability key — the columns deciding whether an incoming finding is one already seen. Changing it redefines "the same finding" and splits history.

Queue — asynchronous work: imports, syncs, report generation. A screen that seems not to have updated usually has a queue entry that explains it.

Data process — how far a tool-fed audit has got through the import pipeline; the difference between "the upload finished" and "the data is ready".

Beat — a scheduled job maintaining derived data, such as recalculating SLA states as dates pass. A stopped beat produces figures wrong by a fixed interval and raises no error.

Governance

RLCM — Red Line Compliance Management: assessment of controls against frameworks, rather than of systems for weaknesses.

TPRM — Third-Party Risk Management: the same control assessment applied to suppliers. It shares screens with RLCM and differs by the audit's type.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

UCF — Unified Control Framework: the mapping that lets one control satisfy several frameworks at once, so answering it once covers each of them.

Control — a requirement being assessed, as opposed to a weakness being found.

Standard — the framework a control belongs to, such as ISO/IEC 27001 or PCI DSS.

Reporting

Considered audit filter — the switch on Report Search that reproduces either the dashboard's numbers or the module screens'. Comparing the two settles most disagreements about counts.

Design template — the layout a report renders through, separate from the template that decides its content.

Share view against share copy — sharing a template as a view keeps one definition that your edits reach; sharing a copy creates an independent template that will drift.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.