Saqr صقر

Confidential

AI fix pack

Prepared for

shop.yourcompany.com

Security Assessment

Testing window	Report date	Report ID	Version
4 to 23 November 2026	23 November 2026	SQR-0001-01	1

Document control

About this report

Report	AI fix pack
Prepared for	shop.yourcompany.com
Engagement	Security Assessment
Tasks included	Quick task on shop.yourcompany.com, Full task on shop.yourcompany.com, Quick task on shop.yourcompany.com, Quick task on shop.yourcompany.com, Full task on shop.yourcompany.com (5 tasks)
Prepared by	Security Assessment Team
Report ID	SQR-0001-01
Version and date	Version 1, 23 November 2026
Classification	Confidential
Distribution	The people responsible for shop.yourcompany.com only. Do not forward outside the distribution without the owner's approval.

Contents

1 How to use this pack 3
2 Fix now 4
3 Fix next 7
4 Plan and harden 9

Section 1

How to use this pack

Each block below is a complete instruction for an AI coding assistant to fix one finding properly. Copy a block, paste it into the assistant inside the repository that serves the location it names, and review the change like any other pull request.

1. **One block at a time.** Start with the first block under Fix now. Give the assistant access to the code that serves the named location, then paste the block as it is.
2. **Review the change.** Read the diff, check the fix covers every place with the same pattern, and make sure no existing behavior broke.
3. **Run the tests.** The block asks for tests that fail before the fix and pass after it. Run the full suite before merging.
4. **Check the fix.** Fixed it? Ask Saqr's assistant to retest this finding, or run a new task. The next version of these reports shows it as fixed.

Each block contains	What it gives the assistant
What is wrong	The defect in plain terms, for this application.
Where	The affected endpoints or components, as Saqr recorded them.
Standards	The CWE class and the OWASP ASVS requirement the fix must meet.
Required secure behavior	What the code must do once fixed.
How to fix it	The fix approach that follows best practice for this weakness class.
Constraints	Keep existing behavior, avoid new dependencies, change only what the fix needs.
Tests and verification	The tests to add and how to confirm the fix, then how to have Saqr check it.

Safe to share with engineers. The blocks describe the weakness and the secure behavior expected, and pasting one changes nothing on your site. They hold no test inputs or steps to reproduce a finding; those stay in the technical report, which has a narrower distribution.

Priority	Blocks	Findings
Fix now	3	F-02, F-03, F-04
Fix next	2	F-05, F-06
Plan and harden	2	F-07, F-08

Section 2

Fix now

Critical and high severity findings. Hand these to your assistant first.

F-02 Invoice PDFs can be downloaded without signing in

▲ High shop.yourcompany.com/invoices/INV-2026-1043.pdf

```
Fix a security weakness: Invoice PDFs can be downloaded without signing in

What is wrong:
A function or endpoint does not check that the caller is signed in or has the required role, so it can
be used by anyone who reaches it. Saqr found this at shop.yourcompany.com/invoices/INV-2026-1043.pdf.
Its severity is high.

Finding F-02, severity High, from the Saqr report SQR-0001-01 for shop.yourcompany.com.

Where:
- shop.yourcompany.com/invoices/INV-2026-1043.pdf
Standards: CWE-862 Missing Authorization; OWASP ASVS V4 Access Control; OWASP Top 10 A01:2021 Broken
Access Control

Required secure behavior:
Every endpoint enforces authentication and the required role on the server, and anything not explicitly
allowed is denied.

How to fix it:
1. Add the authentication and role check to this endpoint using the application's standard authorization
layer.
2. Make deny the default for the router or controller so new endpoints require an explicit policy.
3. Hide the function in the user interface for users without the role, but rely only on the server
check.

Constraints:
- Keep existing behavior for legitimate users and keep public APIs and response formats unchanged unless
the fix requires it.
- Do not add new dependencies unless the platform has no maintained way to do this; if you add one,
explain why.
- Make the smallest change that fixes the weakness everywhere it occurs, and follow the code style of
the repository.
- Do not disable security controls, tests or linters to make the change pass.

Tests to add:
- Tests that an anonymous request and a request from a user without the role both receive 401 or 403.
- A test that a user with the role still succeeds.

How to verify:
- Run the full test suite and confirm the new tests fail before the fix and pass after it.
- Review the change for other places with the same pattern and fix them too.
- Fixed it? Ask Saqr's assistant to retest finding F-02, or run a new task, and confirm it is reported
fixed.
```

F-03 Another customer's order can be read by changing the order id

▲ High shop.yourcompany.com/api/orders/1042 and 1 more

```
Fix a security weakness: Another customer's order can be read by changing the order id

What is wrong:
The endpoint returns or changes a record chosen by an identifier in the request without checking that
the record belongs to the signed-in user. Saqr found this at shop.yourcompany.com/api/orders/1042 and 1
more place. Its severity is high.

Finding F-03, severity High, from the Saqr report SQR-0001-01 for shop.yourcompany.com.

Where:
- shop.yourcompany.com/api/orders/1042
- shop.yourcompany.com/api/orders/1043
Standards: CWE-639 Authorization Bypass Through User-Controlled Key; OWASP ASVS V4 Access Control; OWASP
Top 10 A01:2021 Broken Access Control

Required secure behavior:
Every request that takes an object identifier checks, on the server, that the current user may access
that object, and denies by default (OWASP ASVS V4.2).

How to fix it:
1. Load the record scoped to the current user or tenant (for example a query that filters by owner)
instead of by identifier alone.
2. Centralize the ownership or permission check in a policy or middleware used by every handler for this
resource.
3. Return 404 or 403 for records the user may not access, without revealing whether they exist.

Constraints:
- Keep existing behavior for legitimate users and keep public APIs and response formats unchanged unless
the fix requires it.
- Do not add new dependencies unless the platform has no maintained way to do this; if you add one,
explain why.
- Make the smallest change that fixes the weakness everywhere it occurs, and follow the code style of
the repository.
- Do not disable security controls, tests or linters to make the change pass.

Tests to add:
- A test where user A requests user B's record by identifier and receives 403 or 404.
- A test that user A can still read and change their own record.

How to verify:
- Run the full test suite and confirm the new tests fail before the fix and pass after it.
- Review the change for other places with the same pattern and fix them too.
- Fixed it? Ask Saqr's assistant to retest finding F-03, or run a new task, and confirm it is reported
fixed.
```

F-04 Stored script injection in the profile display name

▲ High PUT /api/profile

```
Fix a security weakness: Stored script injection in the profile display name

What is wrong:
Data from users or other untrusted sources is written into a page without encoding for its context, so a
browser can run it as script. Saqr found this at PUT /api/profile. Its severity is high.

Finding F-04, severity High, from the Saqr report SQR-0001-01 for shop.yourcompany.com.

Where:
- PUT /api/profile
Standards: CWE-79 Improper Neutralization of Input During Web Page Generation; OWASP ASVS V5 Validation,
Sanitization and Encoding; OWASP Top 10 A03:2021 Injection

Required secure behavior:
All untrusted data is encoded for the exact output context (HTML body, attribute, URL, JavaScript) or
rendered as text by the framework, and a strict Content Security Policy blocks inline script.

How to fix it:
1. Render the affected value through the framework's automatic escaping; remove any raw HTML insertion
(such as dangerouslySetInnerHTML, v-html, innerHTML or a template's safe or raw filter) for this value.
2. If rich text is genuinely required, sanitize it on the server with a maintained HTML sanitizer and a
small allow list of tags and attributes.
3. Validate the input's expected format on the server and reject values outside it.
4. Send a Content Security Policy that disallows inline script and restricts script sources to the
application's own origin.

Constraints:
- Keep existing behavior for legitimate users and keep public APIs and response formats unchanged unless
the fix requires it.
- Do not add new dependencies unless the platform has no maintained way to do this; if you add one,
explain why.
- Make the smallest change that fixes the weakness everywhere it occurs, and follow the code style of
the repository.
- Do not disable security controls, tests or linters to make the change pass.

Tests to add:
- A rendering test that a value containing HTML special characters appears as escaped text, not as
markup.
- A test that the Content Security Policy header is present on the affected page.

How to verify:
- Run the full test suite and confirm the new tests fail before the fix and pass after it.
- Review the change for other places with the same pattern and fix them too.
- Fixed it? Ask Saqr's assistant to retest finding F-04, or run a new task, and confirm it is reported
fixed.
```

Section 3

Fix next

Medium severity findings. Schedule these after the urgent fixes.

F-05 Error pages reveal stack traces and file paths

■ Medium shop.yourcompany.com/api/products/abc and 1 more

```
Fix a security weakness: Error pages reveal stack traces and file paths

What is wrong:
A response reveals information that should not be public, such as internal details, configuration, keys
or other users' data. Saqr found this at shop.yourcompany.com/api/products/abc and 1 more place. Its
severity is medium.

Finding F-05, severity Medium, from the Saqr report SQR-0001-01 for shop.yourcompany.com.

Where:
- shop.yourcompany.com/api/products/abc
- /srv/app/src/routes/products.js
Standards: CWE-200 Exposure of Sensitive Information to an Unauthorized Actor; OWASP ASVS V8 Data
Protection; OWASP Top 10 A01:2021 Broken Access Control

Required secure behavior:
Responses contain only the data the caller is allowed to see; sensitive data requires authentication and
is never sent to the client unnecessarily.

How to fix it:
1. Remove the sensitive fields from the response, using an explicit response type instead of serializing
whole objects.
2. Require authentication and authorization for any endpoint that must return sensitive data.
3. If a secret or key was exposed, rotate it.
4. Turn off verbose errors, stack traces and debug output in production.

Constraints:
- Keep existing behavior for legitimate users and keep public APIs and response formats unchanged unless
the fix requires it.
- Do not add new dependencies unless the platform has no maintained way to do this; if you add one,
explain why.
- Make the smallest change that fixes the weakness everywhere it occurs, and follow the code style of
the repository.
- Do not disable security controls, tests or linters to make the change pass.

Tests to add:
- A test that the response no longer contains the sensitive fields.
- A test that an anonymous caller cannot reach the protected data.

How to verify:
- Run the full test suite and confirm the new tests fail before the fix and pass after it.
- Review the change for other places with the same pattern and fix them too.
- Fixed it? Ask Saqr's assistant to retest finding F-05, or run a new task, and confirm it is reported
fixed.
```

F-06 No rate limit on the sign-in endpoint

Medium POST /api/auth/login

```
Fix a security weakness: No rate limit on the sign-in endpoint

What is wrong:
The sign-in or other sensitive endpoint accepts unlimited attempts, so passwords or codes can be guessed
by repetition. Saqr found this at POST /api/auth/login. Its severity is medium.

Finding F-06, severity Medium, from the Saqr report SQR-0001-01 for shop.yourcompany.com.

Where:
- POST /api/auth/login
Standards: CWE-307 Improper Restriction of Excessive Authentication Attempts; OWASP ASVS V2
Authentication; OWASP Top 10 A07:2021 Identification and Authentication Failures

Required secure behavior:
Repeated failures are throttled per account and per client, with progressive delays or temporary
lockout, without letting anyone lock other users out at will.

How to fix it:
1. Add rate limiting to the endpoint keyed by account and by client address, with a short window and a
low limit.
2. Add a progressive delay or temporary lockout after repeated failures, and notify the account owner.
3. Return the same response for an unknown account and a wrong password.

Constraints:
- Keep existing behavior for legitimate users and keep public APIs and response formats unchanged unless
the fix requires it.
- Do not add new dependencies unless the platform has no maintained way to do this; if you add one,
explain why.
- Make the smallest change that fixes the weakness everywhere it occurs, and follow the code style of
the repository.
- Do not disable security controls, tests or linters to make the change pass.

Tests to add:
- A test that requests beyond the limit receive 429 and that the limit resets after the window.
- A test that responses for an unknown account and a wrong password are identical.

How to verify:
- Run the full test suite and confirm the new tests fail before the fix and pass after it.
- Review the change for other places with the same pattern and fix them too.
- Fixed it? Ask Saqr's assistant to retest finding F-06, or run a new task, and confirm it is reported
fixed.
```

Section 4

Plan and harden

Low severity and informational findings. Fix these in normal maintenance.

F-07 Security headers are missing on every page

Low shop.yourcompany.com and 2 more

```
Fix a security weakness: Security headers are missing on every page

What is wrong:
Responses lack standard security headers, which removes browser protections against framing, content
sniffing, downgrade and script injection. Saqr found this at shop.yourcompany.com and 2 more places. Its
severity is low.

Finding F-07, severity Low, from the Saqr report SQR-0001-01 for shop.yourcompany.com.

Where:
- shop.yourcompany.com
- /login
- /account
Standards: CWE-693 Protection Mechanism Failure; OWASP ASVS V14 Configuration; OWASP Top 10 A05:2021
Security Misconfiguration

Required secure behavior:
Every response carries Strict-Transport-Security, a Content-Security-Policy, X-Content-Type-Options:
nosniff, frame protection (frame-ancestors or X-Frame-Options), and a Referrer-Policy (OWASP ASVS
V14.4).

How to fix it:
1. Set the headers once, in the web server, CDN or framework middleware, so they apply to every response
including errors and static files.
2. Start the Content-Security-Policy in report-only mode if needed, then enforce it once reports are
clean.
3. Set Strict-Transport-Security with a long max-age only after every subdomain serves HTTPS.

Constraints:
- Keep existing behavior for legitimate users and keep public APIs and response formats unchanged unless
the fix requires it.
- Do not add new dependencies unless the platform has no maintained way to do this; if you add one,
explain why.
- Make the smallest change that fixes the weakness everywhere it occurs, and follow the code style of
the repository.
- Do not disable security controls, tests or linters to make the change pass.

Tests to add:
- An integration test that requests a page, a static asset and an error response and asserts each
security header is present with the expected value.

How to verify:
- Run the full test suite and confirm the new tests fail before the fix and pass after it.
- Review the change for other places with the same pattern and fix them too.
- Fixed it? Ask Saqr's assistant to retest finding F-07, or run a new task, and confirm it is reported
fixed.
```

F-08 Session cookie is set without the Secure and HttpOnly flags

Low GET /login

```
Fix a security weakness: Session cookie is set without the Secure and HttpOnly flags

What is wrong:
A response reveals information that should not be public, such as internal details, configuration, keys
or other users' data. Saqr found this at GET /login. Its severity is low.

Finding F-08, severity Low, from the Saqr report SQR-0001-01 for shop.yourcompany.com.

Where:
- GET /login
Standards: CWE-200 Exposure of Sensitive Information to an Unauthorized Actor; OWASP ASVS V8 Data
Protection; OWASP Top 10 A01:2021 Broken Access Control

Required secure behavior:
Responses contain only the data the caller is allowed to see; sensitive data requires authentication and
is never sent to the client unnecessarily.

How to fix it:
1. Remove the sensitive fields from the response, using an explicit response type instead of serializing
whole objects.
2. Require authentication and authorization for any endpoint that must return sensitive data.
3. If a secret or key was exposed, rotate it.
4. Turn off verbose errors, stack traces and debug output in production.

Constraints:
- Keep existing behavior for legitimate users and keep public APIs and response formats unchanged unless
the fix requires it.
- Do not add new dependencies unless the platform has no maintained way to do this; if you add one,
explain why.
- Make the smallest change that fixes the weakness everywhere it occurs, and follow the code style of
the repository.
- Do not disable security controls, tests or linters to make the change pass.

Tests to add:
- A test that the response no longer contains the sensitive fields.
- A test that an anonymous caller cannot reach the protected data.

How to verify:
- Run the full test suite and confirm the new tests fail before the fix and pass after it.
- Review the change for other places with the same pattern and fix them too.
- Fixed it? Ask Saqr's assistant to retest finding F-08, or run a new task, and confirm it is reported
fixed.
```