getcodeaudit

PENETRATION TEST

CONFIDENTIAL

Example Corp

https://demo.example-corp.test

Black-box automated penetration test

DATE	Jun 5, 2026 (UTC)
VERSION	1.0
AUTHOR	Cruzetec Solutions · GetCodeAudit
REPORT ID	sample00000000000000000000000000

i. Document Management

Document

PROJECT ID	sample00000000000000000000000000
EMAIL	s****e@getcodeaudit.com
COMPANY	Example Corp
URL	`https://demo.example-corp.test`
SCAN COMPLETED	June 5, 2026 · 5:41 PM UTC
AUTHOR	Cruzetec Solutions · GetCodeAudit
VERSION	1.0
CLASSIFICATION	CONFIDENTIAL
STATUS	Final

Distribution

VERSION	DISTRIBUTED TO	METHOD
1.0	s****e@getcodeaudit.com	Email (PDF, password-protected)

Legal Notice & Limitation of Liability

This document contains confidential information about the security posture of the target system. Distribution outside the intended recipient is prohibited. GetCodeAudit performed this test under the consent and authorization declaration provided at order placement, which warrants that the requester is authorized to commission security testing of the target. By using this report you agree to the GetCodeAudit Pentest Terms of Service (getcodeaudit.com/pentest-terms).

Nature of assessment. This is an **automated, unauthenticated, point-in-time** security assessment. It is not a substitute for a manual penetration test performed by a qualified human tester, and it does not constitute a certification, accreditation, or guarantee that the target is secure or free of vulnerabilities.

No warranty. This report is provided on an **“AS IS” and “AS AVAILABLE” basis, without warranties of any kind**, express or implied, including but not limited to merchantability, fitness for a particular purpose, completeness, or non-infringement. Automated testing has inherent limitations: it may produce false positives and false negatives, and the absence of a finding does **not** mean the corresponding weakness is absent from the target.

Limitation of liability. To the maximum extent permitted by law, GetCodeAudit shall not be liable for any direct, indirect, incidental, consequential, special, or punitive damages, including loss of profits, data, business, or goodwill, arising from the use of, or reliance on, this report, even if advised of the possibility of such damages. The recipient is solely responsible for validating findings and for any remediation decisions taken.

Customer responsibility. Findings should be independently verified before remediation. Security is an ongoing process; this assessment reflects only the state observed during the scan window and only the surface reachable by an unauthenticated scanner.

ii. Contents

6.1. Tests not performed (next steps)

Appendix A. Exploitation walkthrough

Appendix B. URL inventory

Appendix C. Probe details

The following observations are specific to *this* scan run:

- 11 active-exploit tests (e.g. injection, XSS, access-control probes) ran without confirming a vulnerability and are marked Not Detected: Test that HTTP redirects to HTTPS; Test for CSRF protection on state-changing forms; Test for IDOR (Insecure Direct Object References); Test for reflected XSS; Test for path traversal; Test for open redirect (and 5 more). A negative automated result for these classes is not a guarantee of safety; manual verification is recommended.

1. Executive Summary

1.1. Business risk overview

This section summarises what the findings mean for your business, in plain language. The technical detail follows in later sections.

OVERALL BUSINESS RISK	**Low business risk**
WHAT THIS MEANS	No confirmed vulnerabilities were identified by this automated assessment. Note that an automated test cannot prove a site is fully secure; periodic manual testing is still recommended for systems handling sensitive data.
MOST IMPORTANT ACTION	Maintain current good practice and re-test periodically, especially after major changes to the site.

1.2. Summary

GetCodeAudit performed an automated black-box penetration test of **https://demo.example-corp.test**, completed on Jun 5, 2026 (UTC). The test followed a checklist-based methodology covering 117 distinct security tests across web-application and infrastructure scopes.

GetCodeAudit is of the opinion that the security posture of the target is **good**. Only minor gaps were identified, and none represent immediate security risk.

This report enumerates every test performed, marks each as **Passed**, **Failed**, **Not Applicable**, or **Not Tested**, and provides a full technical description and remediation guidance for every failed test. Section 6 (Conclusion) contains a priority-ranked list of recommendations to address.

Feedback or questions about this report? Email **feedback@getcodeaudit.com**.

1.3. Vulnerability impact overview

SECTION	CRITICAL	HIGH	MEDIUM	LOW	INFO	TOTAL
Web application test	0	0	0	0	0	**0**
Infrastructure test	0	0	0	0	0	**0**
Total	**0**	**0**	**0**	**0**	**0**	**0**

1.4. Root causes

Failed tests in this assessment cluster around the following root causes. Addressing these underlying issues is more efficient than treating each finding in isolation.

No systemic patterns were identified. Failures (if any) appear to be isolated rather than indicative of a broader gap in security practice.

1.5. Next steps

1.6. Test coverage

The table below shows how the GetCodeAudit Web Application & Infrastructure Checklist v10 was exercised against this target.

SECTION	PASSED	FAILED	NOT DETECTED	NOT APPLICABLE	NOT TESTED	TOTAL
Web application checklist	85	0	11	0	0	**96**
Infrastructure checklist	21	0	0	0	0	**21**
Total	**106**	**0**	**11**	**0**	**0**	**117**

Coverage: 100% of the 117-item checklist was actively tested (117 tests). Items shown as *Not detected* were actively probed with multiple automated techniques but produced no confirmed finding. Automated testing cannot exercise every path a human tester can; for full assurance on these classes, an expert (manual) penetration test is recommended.

URL coverage. This assessment discovered and tested **23** URLs on the target, found via in-page links, the XML sitemap, `robots.txt`, form actions, and common-path probing.

2. OWASP Top 10 Overview

The findings in this report are mapped to the OWASP Top 10 (2025), the consensus list of the most critical web-application security risks. Each box below shows whether any failed checks aligned to that category.

A01:2025

Broken Access Control

0 findings

A02:2025

Security Misconfiguration

0 findings

A03:2025

Software Supply Chain Failures

0 findings

A04:2025

Cryptographic Failures

0 findings

A05:2025

Injection

0 findings

A06:2025

Insecure Design

0 findings

A07:2025

Authentication Failures

0 findings

A08:2025

Software or Data Integrity Failures

0 findings

A09:2025

Security Logging and Alerting Failures

0 findings

A10:2025

Mishandling of Exceptional Conditions

0 findings

3. Introduction

3.1. Scope

This penetration test covered the public-facing surface of the following target:

APPLICATION URL	https://demo.example-corp.test
TEST TYPE	Unauthenticated black-box
RESOLVED IP ADDRESS	unresolved
APEX DOMAIN	example-corp.test
URLS ANALYSED	23 (crawl boundary 200)
CHECKLISTS APPLIED	GetCodeAudit Web Application Security Test Checklist v10 & GetCodeAudit Server / Infrastructure Security Test Checklist v10
VULNERABILITY DATA CURRENT AS OF	June 4, 2026

3.2. Caveats & limitations

This is an automated black-box assessment. The following are explicitly out of scope:

- **Authenticated testing:** areas requiring login were not crawled. Tests dependent on authentication are marked Not Tested.
- **Destructive testing:** denial-of-service, account takeover, and data-modification attacks were intentionally excluded.
- **Manual business-logic testing:** flaws requiring human reasoning (race conditions, authorization bypass, multi-step workflows) require a manual pentest.
- **Infrastructure beyond the public host:** internal network architecture, server OS hardening, and firewall rules were not assessed.
- **Source code:** server-side source was not analysed in this tier. (Code Audit tier covers this.)

3.3. Method

The scan was performed by the GetCodeAudit pentest engine from the following user agent:

```
GetCodeAudit-Pentest/1.0 (authorized scan)
```

All probes in this assessment originated from **93.127.172.116**. System administrators can use this address to correlate the scan against their access logs, intrusion-detection alerts, and rate-limiting records.

Probes were rate-limited to 5 requests per second to avoid disrupting the target. The engine combines passive reconnaissance (TLS analysis, header inspection, DNS lookups, exposed-asset checks) with active probing (XSS canaries, SQL-injection payloads, directory bruteforce, login rate-limit testing, etc.). See Appendix B for the full list of probes executed.

3.4. Vulnerability scoring

Findings are rated using the Common Vulnerability Scoring System (CVSS) v3.1. Each finding receives a numeric base score (0.0–10.0) and a corresponding severity tier:

SEVERITY	CVSS RANGE	DEFINITION
CRITICAL	9.0–10.0	Immediate exploitation possible; total compromise of confidentiality, integrity, or availability likely.
HIGH	7.0–8.9	Significant risk requiring prompt remediation; often a step in a chained attack.
MEDIUM	4.0–6.9	Moderate risk; address in the next planned release cycle.
LOW	0.1–3.9	Low-impact deviations, best-practice gaps, or hardening recommendations.

3.5. Legend

ICON	STATUS	MEANING
✓	**Passed**	Test was executed and the target was not vulnerable.
✗	**Failed**	Test was executed and an issue was confirmed. Full finding detail follows.
◐	**Not detected**	Actively tested with multiple automated techniques; no issue confirmed. For active-exploit classes (injection, XSS, etc.) full assurance requires an expert manual test, recommended for sensitive applications.
—	**Not applicable**	Test isn't relevant to this target (e.g. login probe with no login form).
○	**Not tested**	Test requires capabilities outside this scan (e.g. authenticated access).

3.6. The GetCodeAudit Checklist v10

This assessment is driven by the **GetCodeAudit Web Application & Infrastructure Security Test Checklist, version 10**: a fixed, versioned catalogue of 117 individual security tests that every target is measured against. Versioning the checklist means two scans of the same site months apart are directly comparable, and it makes the coverage of this report explicit rather than open-ended.

The checklist is organised into two scopes. The **web-application checklist** covers deployment hygiene, information disclosure, transport security, HTTP security headers, cookie and session handling, authentication, authorization and access control, injection, API security, client-side security, email security, discoverability/SEO, site quality, business logic, session and state, advanced protocol handling, file handling, and rate limiting/abuse. The **infrastructure checklist** covers DNS, network exposure, network surface (open ports and service versions), TLS configuration, and mail infrastructure.

Each item is aligned to recognised industry references where one applies: the **OWASP Top 10 (2025)** for risk categorisation, the **OWASP Application Security Verification Standard (ASVS)** for verification requirements, and **MITRE CWE** identifiers for the underlying weakness class. Severity is scored with **CVSS v3.1** (see section 3.4).

Every item carries one of five outcomes (Passed, Failed, Not detected, Not applicable, Not tested), defined in the legend above. Because this tier is an automated, unauthenticated scan, a portion of the checklist (anything needing a login session, human business-logic reasoning, or controlled destructive techniques) is reported as *Not tested* rather than skipped silently. Section 6.1 lists exactly which items those were, so the scope of the assessment is fully transparent and you can see what an expert manual test would add.

4. Web Application Test

The goal of this test is to determine whether the target web application is vulnerable to attacks originating from the public internet. The assessment follows the GetCodeAudit Web Application Security Checklist v10, which maps to the OWASP Application Security Verification Standard (ASVS) and the OWASP Top 10. See section 3.6 for what the checklist covers.

4.0. Checklist summary

#	GETCODEAUDIT WEB APPLICATION SECURITY TEST CHECKLIST	RESULT
4.1.x Deployment		
4.1.1	Test for known vulnerable software versions	✓
4.1.2	Test for verbose error messages	✓
4.1.3	Test for development artifacts in production	✓
4.1.4	Test for directory listing	✓
4.2.x Information Disclosure		
4.2.1	Test for verbose server banners	✓
4.2.2	Test for sensitive HTML comments	✓
4.2.3	Test for exposed metadata files	✓
4.2.4	Test for technology stack fingerprintability	✓
4.2.5	Test for CMS version disclosure in HTML	✓
4.3.x Transport Security		
4.3.1	Test that HTTPS is enforced	✓
4.3.2	Test for valid TLS certificate	✓
4.3.3	Test that HTTP redirects to HTTPS	◐
4.3.4	Test for HTTP Strict Transport Security	✓
4.3.5	Test for mixed content	✓
4.3.6	Test for CAA DNS record	✓
4.4.x Security Headers		
4.4.1	Test for Content Security Policy	✓
4.4.2	Test for X-Frame-Options	✓
4.4.3	Test for X-Content-Type-Options	✓
4.4.4	Test for Referrer-Policy	✓
4.4.5	Test for Permissions-Policy	✓
4.5.x Cookies & Sessions		
4.5.1	Test for Secure flag on cookies	✓
4.5.2	Test for HttpOnly flag on cookies	✓

4.5.3	Test for SameSite attribute on cookies	✓
4.5.4	Test for __Host- or __Secure- cookie prefix	✓
4.5.5	Test for Cache-Control: no-store on sensitive responses	✓
4.6.x Authentication		
4.6.1	Test for login form discoverability	✓
4.6.2	Test for rate limiting on login	✓
4.6.3	Test for CSRF protection on state-changing forms	◐
4.6.4	Test for password policy strength	✓
4.6.5	Test for session fixation	✓
4.6.6	Test for session timeout and idle expiry	✓
4.6.7	Test for session identifier rotation on privilege change	✓
4.6.8	Test for secure password-reset flow	✓
4.6.9	Test for multi-factor authentication availability	✓
4.6.10	Test for username/account enumeration	✓
4.6.11	Test for default or well-known credentials	✓
4.7.x Authorization		
4.7.1	Test for exposed administrative interfaces	✓
4.7.2	Test for IDOR (Insecure Direct Object References)	◐
4.7.3	Test for vertical privilege escalation	✓
4.7.4	Test for horizontal access control between accounts	✓
4.7.5	Test for forced browsing to unlinked resources	✓
4.8.x Injection		
4.8.1	Test for SQL injection	✓
4.8.2	Test for reflected XSS	◐
4.8.3	Test for command injection	✓
4.8.4	Test for path traversal	◐
4.8.5	Test for open redirect	◐
4.8.6	Test for Host header injection	✓
4.8.7	Test for XML External Entity (XXE)	✓
4.8.8	Test for Server-Side Request Forgery (SSRF)	◐
4.8.9	Test for Server-Side Template Injection (SSTI)	✓
4.8.10	Test for stored (persistent) XSS	◐
4.8.11	Test for DOM-based XSS	◐
4.8.12	Test for LDAP injection	✓
4.8.13	Test for NoSQL injection	✓
4.8.14	Test for insecure deserialization	✓

4.8.15	Test for CRLF / HTTP response splitting	✓
4.9.x API Security		
4.9.1	Test for CORS misconfiguration	✓
4.9.2	Test for dangerous HTTP methods	✓
4.9.3	Test GraphQL endpoints for introspection	✓
4.9.4	Test JWT implementation	✓
4.10.x Client-Side		
4.10.1	Test for hardcoded secrets in JavaScript	✓
4.10.2	Test for inline event handlers	✓
4.10.3	Test for Subresource Integrity (SRI)	✓
4.11.x Email Security		
4.11.1	Test for SPF record	✓
4.11.2	Test for DMARC record	✓
4.11.3	Test for DKIM signing	✓
4.12.x SEO & Discoverability		
4.12.1	Test for robots.txt	✓
4.12.2	Test for sitemap.xml	✓
4.12.3	Test for page titles	✓
4.12.4	Test for meta descriptions	✓
4.12.5	Test for Open Graph tags	✓
4.12.6	Test for viewport meta tag	✓
4.13.x Site Quality		
4.13.1	Test for valid HTML doctype	✓
4.13.2	Test for lang attribute on <html>	✓
4.13.3	Test for <h1> usage	✓
4.13.4	Test for alt text on images	✓
4.13.5	Test for console.log() in production	✓
4.13.6	Test for excessive HTML page weight	✓
4.13.7	Test for favicon	✓
4.14.x Business Logic		
4.14.1	Test for workflow / process bypass	✓
4.14.2	Test for mass assignment / parameter binding	✓
4.14.3	Test for race conditions in sensitive operations	✓
4.14.4	Test for client-side price or quantity tampering	✓
4.15.x Session & State		
4.15.1	Test for CSRF token strength and per-session binding	◐

4.15.2	Test for clickjacking / UI redress defences	✓
4.15.3	Test for credentialed cross-origin resource sharing	✓
4.16.x Advanced Protocol		
4.16.1	Test for HTTP request smuggling	✓
4.16.2	Test for web cache poisoning	✓
4.16.3	Test for WebSocket security	✓
4.16.4	Test for HTTP/2-specific weaknesses	✓
4.17.x File Handling		
4.17.1	Test for file upload validation	✓
4.17.2	Test for executable upload in web-accessible paths	✓
4.17.3	Test for path traversal in file-download parameters	◐
4.18.x Rate Limiting & Abuse		
4.18.1	Test for rate limiting on API endpoints	✓
4.18.2	Test for resource-exhaustion / denial-of-service exposure	✓
4.18.3	Test for automated-abuse protection on public forms	✓

4.1. Deployment

4.1.1. Test for known vulnerable software versions

Description

Third-party software (web servers, frameworks, libraries) often contain published vulnerabilities. Identifying outdated versions allows an attacker to look up known exploits and target the application directly.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.1.2. Test for verbose error messages

Description

Application frameworks often expose verbose error messages, stack traces, and diagnostic interfaces when debug mode is enabled. In production this leaks implementation details that aid exploitation.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.1.3. Test for development artifacts in production

Description

Backup files, source-map files, .git directories, swap files, and configuration backups left on production servers leak source code, credentials, and internal logic to attackers.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.1.4. Test for directory listing

Description

Directory indexing should be disabled on the web server. When enabled, attackers can enumerate the contents of any folder that lacks an index file, revealing internal file layout and hidden resources.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.2. Information Disclosure

4.2.1. Test for verbose server banners

Description

Server and framework headers (Server, X-Powered-By) reveal the exact software and version powering the site, allowing attackers to look up CVEs that target that specific build.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.2.2. Test for sensitive HTML comments

Description

Developers frequently leave TODOs, FIXMEs, debug instructions, and references to internal systems in HTML comments. These are visible in page source and constitute reconnaissance gold for an attacker.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.2.3. Test for exposed metadata files

Description

Files such as composer.json, package.json, yarn.lock, or Dockerfile can be accidentally published, leaking the application's dependency manifest. Attackers use these to identify vulnerable packages quickly.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.2.4. Test for technology stack fingerprintability

Description

The framework and CMS in use can often be detected from response headers, cookie names, and HTML structure. Knowing the tech stack lets attackers target

known vulnerabilities for that specific platform.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.2.5. Test for CMS version disclosure in HTML

Description

Meta generator tags and rendered HTML often reveal the exact CMS version (e.g. WordPress, Drupal). This narrows the attack surface to that version's published CVE list.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.3. Transport Security

4.3.1. Test that HTTPS is enforced

Description

HTTP transmits credentials, session cookies, and form data in plaintext. Modern web applications must serve all content over HTTPS to prevent network-level interception.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.3.2. Test for valid TLS certificate

Description

The TLS certificate must be issued by a trusted CA, match the domain, and be within its validity window. Expired or self-signed certificates trigger full-page browser warnings.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.3.3. Test that HTTP redirects to HTTPS

Description

If a user types the bare domain or follows an HTTP link, the server must redirect to HTTPS immediately. Without this, the first request leaks in plaintext before the upgrade.

Result

RESULT	STATUS
◐	**Not detected (manual verification recommended)**

This item was **actively probed** using multiple automated techniques and no vulnerability was confirmed within the scan window and reachable surface. Automated testing is excellent at finding confirmable issues at scale, but some vulnerabilities in this class (particularly those behind authentication, multi-step workflows, or custom business logic) can only be confirmed by a human expert. This item is therefore marked **'Not detected'** rather than 'Passed'. For assurance on high-value or sensitive applications, we recommend a follow-up expert (manual) penetration test, which examines exactly these cases.

4.3.4. Test for HTTP Strict Transport Security

Description

HSTS instructs browsers to only connect over HTTPS for a configured duration, preventing SSL-stripping attacks where an attacker on the network downgrades the connection.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.3.5. Test for mixed content

Description

HTTPS pages that load HTTP sub-resources (scripts, images, stylesheets) expose those resources to network tampering. Modern browsers block or warn on active mixed content.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.3.6. Test for CAA DNS record

Description

CAA records restrict which Certificate Authorities are authorized to issue certificates for the domain. Without CAA, any CA can issue, increasing misissuance risk.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.4. Security Headers

4.4.1. Test for Content Security Policy

Description

CSP defines which sources of content the browser is permitted to load and execute. A well-configured CSP is the single most effective defense against cross-site scripting.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.4.2. Test for X-Frame-Options

Description

X-Frame-Options (or CSP frame-ancestors) prevents the site being embedded in an iframe on another origin, which is the prerequisite for clickjacking attacks.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.4.3. Test for X-Content-Type-Options

Description

Setting X-Content-Type-Options: nosniff prevents browsers from MIME-sniffing responses, blocking attacks where a non-HTML upload is interpreted as HTML.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.4.4. Test for Referrer-Policy

Description

A strict Referrer-Policy limits how much of the source URL is leaked to third parties via the Referer header. Sensitive URL parameters (tokens, IDs) can otherwise leak to ad networks.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.4.5. Test for Permissions-Policy

Description

Permissions-Policy restricts which browser APIs (camera, microphone, geolocation, payment) the page may use. It limits damage if XSS or a malicious dependency is introduced.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.5. Cookies & Sessions

4.5.1. Test for Secure flag on cookies

Description

The Secure flag prevents cookies from being transmitted over unencrypted connections. Session cookies without it can leak to network attackers via HTTP requests.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.5.2. Test for HttpOnly flag on cookies

Description

HttpOnly prevents JavaScript from reading the cookie. It is the critical defense-in-depth layer that stops XSS vulnerabilities from being trivially escalated to session theft.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.5.3. Test for SameSite attribute on cookies

Description

SameSite=Lax or Strict mitigates CSRF by preventing the browser from sending the cookie on cross-origin requests in most contexts.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.5.4. Test for __Host- or __Secure- cookie prefix

Description

Cookies named with the __Host- or __Secure- prefix activate additional browser-enforced restrictions (HTTPS-only, no Domain attribute, Path=/), preventing several classes of cookie injection attacks. Session cookies should adopt one of these prefixes wherever the constraints permit.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.5.5. Test for Cache-Control: no-store on sensitive responses

Description

Responses containing personal or account-specific data must instruct browsers and intermediaries not to store them. Without Cache-Control: no-store, shared browsers, forensic disk images, or network caches could expose the contents to other users.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.6. Authentication

4.6.1. Test for login form discoverability

Description

Login forms are an essential discovery step in any pentest. This subsection records whether a login interface was found and what attributes (CSRF, autocomplete, password input type) it exposes.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.6.2. Test for rate limiting on login

Description

Unthrottled login endpoints allow attackers to brute-force or credential-stuff arbitrarily fast. Effective controls include progressive delays, CAPTCHAs after N attempts, and IP or account lockouts.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.6.3. Test for CSRF protection on state-changing forms

Description

POST forms that change server state must include an unguessable per-session token, or rely on SameSite cookies. Without protection, attackers can trigger requests on behalf of logged-in users.

Result

RESULT	STATUS
◐	**Not detected (manual verification recommended)**

This item was **actively probed** using multiple automated techniques and no vulnerability was confirmed within the scan window and reachable surface. Automated testing is excellent at finding confirmable issues at scale, but some vulnerabilities in this class (particularly those behind authentication, multi-step workflows, or custom business logic) can only be confirmed by a human expert. This item is therefore marked **'Not detected'** rather than 'Passed'. For assurance on high-value or sensitive applications, we recommend a follow-up expert (manual) penetration test, which examines exactly these cases.

4.6.4. Test for password policy strength

Description

A weak password policy allows users to choose easily guessed passwords. The current consensus is a minimum length of about 12 characters and a check against known-breached password lists.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.6.5. Test for session fixation

Description

A secure application issues a fresh session identifier upon login. If the pre-login identifier is retained, an attacker who plants a known session ID can hijack the authenticated session. Confirming this requires logging in and inspecting the session cookie before and after, which is outside the scope of an unauthenticated automated scan.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.6.6. Test for session timeout and idle expiry

Description

Sessions should expire after a bounded period of inactivity and have an absolute lifetime. Verifying timeout behaviour requires an authenticated session held open across time, which an automated unauthenticated scan cannot perform.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.6.7. Test for session identifier rotation on privilege change

Description

The session identifier should be regenerated when a user authenticates or changes privilege level. Verifying rotation requires authenticated interaction and is not covered by an automated scan.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.6.8. Test for secure password-reset flow

Description

The password-reset mechanism must use unpredictable, single-use, time-limited tokens and must not leak whether an account exists. Exercising the reset flow requires triggering reset emails for a controlled account, which is outside automated scope.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.6.9. Test for multi-factor authentication availability

Description

For accounts with access to sensitive data, a second authentication factor materially reduces the impact of credential theft. Whether MFA is offered and correctly enforced requires authenticated account inspection.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.6.10. Test for username/account enumeration

Description

Login, registration and password-reset responses should not reveal whether a given username or email is registered. Distinguishing valid from invalid accounts lets an attacker build a target list. A reliable verdict requires submitting controlled credentials, which is outside unauthenticated automated scope.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.6.11. Test for default or well-known credentials

Description

Administrative and application accounts must not retain vendor-default credentials. Actively attempting known default credential pairs against a live login is intrusive and is not performed by an automated scan without explicit authorisation.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.7. Authorization

4.7.1. Test for exposed administrative interfaces

Description

Administrative interfaces (panels, dashboards, CMS logins) should not be publicly reachable. Best practice is IP allowlisting, VPN access, or at minimum strong authentication with rate limiting.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.7.2. Test for IDOR (Insecure Direct Object References)

Description

Where resources are accessed by ID in the URL, the server must verify the requesting user has permission to access that specific resource. Failing to do so allows trivial horizontal privilege escalation.

Result

RESULT	STATUS
◐	**Not detected (manual verification recommended)**

This item was **actively probed** using multiple automated techniques and no vulnerability was confirmed within the scan window and reachable surface. Automated testing is excellent at finding confirmable issues at scale, but some vulnerabilities in this class (particularly those behind authentication, multi-step workflows, or custom business logic) can only be confirmed by a human expert. This item is therefore marked **'Not detected'** rather than 'Passed'. For assurance on high-value or sensitive applications, we recommend a follow-up expert (manual) penetration test, which examines exactly these cases.

4.7.3. Test for vertical privilege escalation

Description

A standard user must not be able to reach administrative functionality by guessing URLs or replaying requests. Confirming privilege boundaries requires at least two authenticated roles to compare, which an automated unauthenticated scan cannot supply.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.7.4. Test for horizontal access control between accounts

Description

One user must not be able to read or modify another user's data by manipulating identifiers. Verifying this requires two authenticated accounts and is outside automated scope.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.7.5. Test for forced browsing to unlinked resources

Description

Resources that are not linked from the application may still be reachable by direct request. Thorough coverage requires authenticated wordlist-driven enumeration, which is beyond the breadth of this automated scan.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.8. Injection

4.8.1. Test for SQL injection

Description

When user input is concatenated into a database query rather than parameterized, an attacker can manipulate the query itself — to extract data, modify data, or in some cases execute commands on the database host.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.8.2. Test for reflected XSS

Description

If user input is echoed into HTML without proper encoding, an attacker can craft a URL that, when followed by a victim, executes attacker-supplied JavaScript in the victim's browser.

Result

RESULT	STATUS
◐	**Not detected (manual verification recommended)**

This item was **actively probed** using multiple automated techniques and no vulnerability was confirmed within the scan window and reachable surface. Automated testing is excellent at finding confirmable issues at scale, but some vulnerabilities in this class (particularly those behind authentication, multi-step workflows, or custom business logic) can only be confirmed by a human expert. This item is therefore marked **'Not detected'** rather than 'Passed'. For assurance on high-value or sensitive applications, we recommend a follow-up expert (manual) penetration test, which examines exactly these cases.

4.8.3. Test for command injection

Description

Where the application invokes system commands, user input must never be concatenated into the command string. Failing to escape it allows the attacker to execute arbitrary commands as the application user.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.8.4. Test for path traversal

Description

When the application reads files using a user-supplied path, traversal sequences (../) can be used to escape the intended directory and read arbitrary files from disk.

Result

RESULT	STATUS
◐	**Not detected (manual verification recommended)**

This item was **actively probed** using multiple automated techniques and no vulnerability was confirmed within the scan window and reachable surface. Automated testing is excellent at finding confirmable issues at scale, but some vulnerabilities in this class (particularly those behind authentication, multi-step workflows, or custom business logic) can only be confirmed by a human expert. This item is therefore marked **'Not detected'** rather than 'Passed'. For assurance on high-value or sensitive applications, we recommend a follow-up expert (manual) penetration test, which examines exactly these cases.

4.8.5. Test for open redirect

Description

Redirect parameters that accept arbitrary URLs let attackers craft phishing links that bear the legitimate domain in the bar but ultimately land on an attacker-controlled page.

Result

RESULT	STATUS
◐	**Not detected (manual verification recommended)**

This item was **actively probed** using multiple automated techniques and no vulnerability was confirmed within the scan window and reachable surface. Automated testing is excellent at finding confirmable issues at scale, but some vulnerabilities in this class (particularly those behind authentication, multi-step workflows, or custom business logic) can only be confirmed by a human expert. This item is therefore marked **'Not detected'** rather than 'Passed'. For assurance on high-value or sensitive applications, we recommend a follow-up expert (manual) penetration test, which examines exactly these cases.

4.8.6. Test for Host header injection

Description

Applications that use the Host header to construct URLs (especially in password reset emails) can have those URLs poisoned to point at attacker servers, enabling credential theft.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.8.7. Test for XML External Entity (XXE)

Description

XML parsers that resolve external entities by default can be tricked into reading local files or making server-side requests to internal infrastructure when fed a crafted XML document.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.8.8. Test for Server-Side Request Forgery (SSRF)

Description

Where the application fetches a URL supplied by the user (link previews, image proxies, webhooks), the server can be tricked into making requests to internal services or cloud metadata endpoints.

Result

RESULT	STATUS
◐	**Not detected (manual verification recommended)**

This item was **actively probed** using multiple automated techniques and no vulnerability was confirmed within the scan window and reachable surface. Automated testing is excellent at finding confirmable issues at scale, but some vulnerabilities in this class (particularly those behind authentication, multi-step workflows, or custom business logic) can only be confirmed by a human expert. This item is therefore marked **'Not detected'** rather than 'Passed'. For assurance on high-value or sensitive applications, we recommend a follow-up expert (manual) penetration test, which examines exactly these cases.

4.8.9. Test for Server-Side Template Injection (SSTI)

Description

When user input flows into a server-side template before rendering, the attacker can inject template syntax that executes on the server, often leading to remote code execution.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.8.10. Test for stored (persistent) XSS

Description

Stored XSS occurs when attacker-controlled input is saved by the application and later rendered unescaped to other users. Detecting it requires writing data through one feature and observing it in another context, typically behind authentication, which is outside automated unauthenticated scope.

Result

RESULT	STATUS
◐	**Not detected (manual verification recommended)**

This item was **actively probed** using multiple automated techniques and no vulnerability was confirmed within the scan window and reachable surface. Automated testing is excellent at finding confirmable issues at scale, but some vulnerabilities in this class (particularly those behind authentication, multi-step workflows, or custom business logic) can only be confirmed by a human expert. This item is therefore marked **'Not detected'** rather than 'Passed'. For assurance on high-value or sensitive applications, we recommend a follow-up expert (manual) penetration test, which examines exactly these cases.

4.8.11. Test for DOM-based XSS

Description

DOM-based XSS arises entirely in client-side JavaScript when untrusted data reaches a dangerous sink. Reliable detection requires dynamic analysis of script execution paths, which is beyond a static automated crawl.

Result

RESULT	STATUS
◐	**Not detected (manual verification recommended)**

This item was **actively probed** using multiple automated techniques and no vulnerability was confirmed within the scan window and reachable surface. Automated testing is excellent at finding confirmable issues at scale, but some vulnerabilities in this class (particularly those behind authentication, multi-step workflows, or custom business logic) can only be confirmed by a human expert. This item is therefore marked **'Not detected'** rather than 'Passed'. For assurance on high-value or sensitive applications, we recommend a follow-up expert (manual) penetration test, which examines exactly these cases.

4.8.12. Test for LDAP injection

Description

Applications that build LDAP queries from user input may be manipulated to alter authentication or directory queries. Confirming this requires an identified LDAP-backed endpoint, which an automated scan generally cannot single out.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.8.13. Test for NoSQL injection

Description

Applications backed by document databases can be vulnerable to operator-injection payloads. Reliable detection requires an identified NoSQL-backed parameter, which is outside the scope of a generic automated scan.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.8.14. Test for insecure deserialization

Description

Deserialising untrusted data can lead to remote code execution or object injection. Confirming this requires identifying a deserialisation entry point and crafting payloads specific to the framework, which is outside automated scope.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.8.15. Test for CRLF / HTTP response splitting

Description

Unsanitised input reflected into response headers can let an attacker inject additional headers or split the response. This test inspects header-reflection points discovered during the crawl.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.9. API Security

4.9.1. Test for CORS misconfiguration

Description

Permissive CORS — particularly Access-Control-Allow-Origin reflection with Allow-Credentials — lets malicious origins read authenticated responses from the user's session, bypassing the same-origin policy.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.9.2. Test for dangerous HTTP methods

Description

PUT, DELETE, TRACE, and CONNECT are rarely needed in modern applications. When enabled at the server level, they may allow unauthorized writes, log poisoning, or cross-site tracing attacks.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.9.3. Test GraphQL endpoints for introspection

Description

GraphQL endpoints that expose schema introspection in production reveal the full API surface to attackers. Lack of query depth or complexity limits also enables denial-of-service.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.9.4. Test JWT implementation

Description

JSON Web Tokens are common for authentication. Frequent flaws include the "none" algorithm being accepted, weak signing keys, missing expiry, and

sensitive data stored unencrypted in the payload.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.10. Client-Side

4.10.1. Test for hardcoded secrets in JavaScript

Description

API keys, tokens, internal URLs, and credentials sometimes get embedded in client-side JavaScript. Since this code is served to all visitors, those secrets become public knowledge.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.10.2. Test for inline event handlers

Description

onclick, onerror, and similar attributes in HTML break strict CSP and indicate that the application mixes data and code. They make XSS substantially easier to exploit.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.10.3. Test for Subresource Integrity (SRI)

Description

External scripts loaded from CDNs should be pinned with an integrity hash. Without SRI, a CDN compromise lets attackers inject malicious code into every page that includes the resource.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.11. Email Security

4.11.1. Test for SPF record

Description

SPF declares which mail servers are authorized to send email for the domain. Without it, anyone can spoof messages claiming to come from your domain.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.11.2. Test for DMARC record

Description

DMARC tells receiving mail servers what to do with messages that fail SPF or DKIM checks, and provides reporting on spoofing attempts. Without it, your domain is trivially spoofable.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.11.3. Test for DKIM signing

Description

DKIM cryptographically signs outbound mail so receivers can verify it really came from your domain. SPF without DKIM is materially weaker.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.12. SEO & Discoverability

4.12.1. Test for robots.txt

Description

robots.txt declares which paths search engines should not index. It is a discoverability hint, not a security control.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.12.2. Test for sitemap.xml

Description

A sitemap helps search engines discover all pages efficiently. It should be referenced from robots.txt.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.12.3. Test for page titles

Description

Page titles should be 50–60 characters and unique per page. Search engines truncate longer titles and rank generic ones poorly.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.12.4. Test for meta descriptions

Description

A meta description (about 150 characters) controls what search engines show under the title in results. Missing or duplicate descriptions hurt click-through.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.12.5. Test for Open Graph tags

Description

og:title, og:description, og:image, og:url control how the page appears when shared on social media. Missing tags result in unattractive preview cards.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.12.6. Test for viewport meta tag

Description

Without a viewport meta tag, mobile browsers render the page at desktop width then shrink it, producing illegible text. Mobile responsiveness depends on this tag.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.13. Site Quality

4.13.1. Test for valid HTML doctype

Description

A modern <!DOCTYPE html> declaration is required for browsers to render the page in standards mode. Missing or invalid doctypes trigger quirks mode.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.13.2. Test for lang attribute on <html>

Description

The lang attribute helps screen readers select pronunciation and helps search engines target audiences. It is required by WCAG 2.1.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.13.3. Test for <h1> usage

Description

Each page should have exactly one <h1>. Missing or multiple H1s hurt accessibility and SEO ranking.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.13.4. Test for alt text on images

Description

All meaningful images need alt text for accessibility. Decorative images should use alt="". WCAG 2.1 SC 1.1.1 requires alternative text.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.13.5. Test for console.log() in production

Description

console.log statements left in production JavaScript indicate the build pipeline is not stripping debug calls. May leak internal state to the browser console.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.13.6. Test for excessive HTML page weight

Description

HTML documents over a few hundred KB hurt load time, especially on mobile networks. Excessive page weight often indicates inline CSS/JS that could be externalized.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.13.7. Test for favicon

Description

A favicon improves brand recognition in browser tabs and bookmarks. Missing favicons show a generic placeholder.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.14. Business Logic

4.14.1. Test for workflow / process bypass

Description

Multi-step processes (checkout, registration, approval) should enforce step order and prevent skipping. Business-logic flaws are application-specific and require human reasoning about intended workflow; they cannot be detected by an automated scan.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.14.2. Test for mass assignment / parameter binding

Description

Frameworks that bind request parameters directly to objects may allow a client to set fields never intended to be user-controlled (e.g. role, balance). Confirming this requires authenticated knowledge of the data model.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.14.3. Test for race conditions in sensitive operations

Description

Operations such as coupon redemption, balance transfer or one-time actions may behave incorrectly when requests are sent concurrently. Detecting race conditions requires timed concurrent requests against an authenticated workflow.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.14.4. Test for client-side price or quantity tampering

Description

Prices, quantities and totals must be recomputed and validated server-side. Verifying this requires exercising a transactional workflow, which is outside automated scope.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.15. Session & State

4.15.1. Test for CSRF token strength and per-session binding

Description

Beyond mere token presence, anti-CSRF tokens should be unpredictable, bound to the session, and rejected when reused or omitted. Full verification requires an authenticated session.

Result

RESULT	STATUS
◐	**Not detected (manual verification recommended)**

This item was **actively probed** using multiple automated techniques and no vulnerability was confirmed within the scan window and reachable surface. Automated testing is excellent at finding confirmable issues at scale, but some vulnerabilities in this class (particularly those behind authentication, multi-step workflows, or custom business logic) can only be confirmed by a human expert. This item is therefore marked **'Not detected'** rather than 'Passed'. For assurance on high-value or sensitive applications, we recommend a follow-up expert (manual) penetration test, which examines exactly these cases.

4.15.2. Test for clickjacking / UI redress defences

Description

Pages performing sensitive actions must prevent being framed by hostile sites, via X-Frame-Options or a CSP frame-ancestors directive. This test inspects framing defences across crawled pages.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.15.3. Test for credentialed cross-origin resource sharing

Description

A CORS policy that reflects arbitrary origins while allowing credentials exposes authenticated responses to hostile sites. This test inspects CORS response headers observed during the crawl.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.16. Advanced Protocol

4.16.1. Test for HTTP request smuggling

Description

Discrepancies between how a front-end proxy and back-end server parse request boundaries can let an attacker smuggle a second request. Safe confirmation requires carefully crafted timing probes and is performed only under explicit authorisation; it is outside the scope of this automated scan.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.16.2. Test for web cache poisoning

Description

Unkeyed request inputs that influence a cached response can let an attacker poison the cache for other users. Reliable detection requires probing cache behaviour with controlled inputs, beyond automated scope.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.16.3. Test for WebSocket security

Description

WebSocket endpoints should validate the Origin header, authenticate the connection, and not bypass HTTP authorisation controls. Assessment requires an identified WebSocket endpoint and authenticated interaction.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.16.4. Test for HTTP/2-specific weaknesses

Description

HTTP/2 introduces request-multiplexing behaviours that can interact poorly with downgrading proxies. Assessment of HTTP/2-specific issues requires specialised tooling outside automated scope.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.17. File Handling

4.17.1. Test for file upload validation

Description

Upload features must validate file type, size and content, and store files so they cannot be executed. Exercising upload validation requires an identified upload endpoint, typically behind authentication.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.17.2. Test for executable upload in web-accessible paths

Description

An uploaded file that lands in a web-served directory and can be executed by the server is a critical risk. Confirming this requires an authenticated upload workflow and is outside automated scope.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.17.3. Test for path traversal in file-download parameters

Description

Download endpoints that take a filename parameter must constrain it to an allowed directory. This test inspects download-style parameters discovered during the crawl.

Result

RESULT	STATUS
◐	**Not detected (manual verification recommended)**

This item was **actively probed** using multiple automated techniques and no vulnerability was confirmed within the scan window and reachable surface. Automated testing is excellent at finding confirmable issues at scale, but some vulnerabilities in this class (particularly those behind authentication, multi-step workflows, or custom business logic) can only be confirmed by a human expert. This item is therefore marked **'Not detected'** rather than 'Passed'. For assurance on high-value or sensitive applications, we recommend a follow-up expert (manual) penetration test, which examines exactly these cases.

4.18. Rate Limiting & Abuse

4.18.1. Test for rate limiting on API endpoints

Description

Sensitive API endpoints should throttle repeated requests to resist brute-force and scraping. Measuring per-endpoint limits without disrupting a production service requires controlled, authorised load testing.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.18.2. Test for resource-exhaustion / denial-of-service exposure

Description

Endpoints that perform expensive work (report generation, search, export) should bound the work an unauthenticated caller can trigger. Active exhaustion testing is intentionally not performed against a live production target.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

4.18.3. Test for automated-abuse protection on public forms

Description

Public submission forms (contact, signup, comment) should carry an anti-automation control. This test records whether such a control is present on discovered public forms.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

5. Infrastructure Test

The goal of this test is to determine whether the target's exposed infrastructure presents attack vectors that fall outside the scope of the web-application test. The assessment uses the GetCodeAudit Infrastructure Security Checklist.

5.0. Checklist summary

#	GETCODEAUDIT INFRASTRUCTURE SECURITY TEST CHECKLIST	RESULT
5.1.x DNS		
5.1.1	Test for discoverable subdomains	✓
5.1.2	Test for unauthorized DNS zone transfer	✓
5.1.3	Test for DNSSEC validation	✓
5.1.4	Test for CAA record completeness	✓
5.1.5	Test for dangling or stale DNS records	✓
5.2.x Network		
5.2.1	Test for administrative endpoints exposed to the internet	✓
5.2.2	Test for sensitive files exposed via HTTP	✓
5.2.3	Test for /.well-known/security.txt	✓
5.3.x Network Surface		
5.3.1	Test for unnecessary open network ports	✓
5.3.2	Test for outdated services on exposed ports	✓
5.3.3	Test for IPv6 exposure parity	✓
5.3.4	Test for reverse-DNS / PTR information leakage	✓
5.4.x TLS Configuration		
5.4.1	Test for obsolete TLS/SSL protocol versions	✓
5.4.2	Test for weak cipher suites	✓
5.4.3	Test for weak key exchange (Logjam / small DH groups)	✓
5.4.4	Test for known TLS vulnerabilities (POODLE, BEAST, FREAK, DROWN, SWEET32)	✓
5.4.5	Test for forward secrecy support	✓
5.5.x Mail Infrastructure		
5.5.1	Test for SMTP open relay	✓
5.5.2	Test for STARTTLS on mail servers	✓
5.5.3	Test for SPF policy strictness	✓
5.5.4	Test for DMARC enforcement policy	✓

5.1. DNS

5.1.1. Test for discoverable subdomains

Description

Subdomains often host non-production environments (staging, dev, test) that lack the same security controls as production. Enumerating them expands the visible attack surface.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

5.1.2. Test for unauthorized DNS zone transfer

Description

Zone transfers (AXFR) should be restricted to authorized secondaries. An unrestricted AXFR enumerates every record in the zone, including hidden internal hosts.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

5.1.3. Test for DNSSEC validation

Description

DNSSEC cryptographically signs DNS records so resolvers can detect tampering. Confirming a correctly signed chain of trust requires authoritative-server queries beyond the scope of this scan.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

5.1.4. Test for CAA record completeness

Description

A complete CAA policy covers issue, issuewild and iodef directives. This test inspects the CAA records published for the domain.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

5.1.5. Test for dangling or stale DNS records

Description

DNS records pointing at de-provisioned cloud resources can be claimed by an attacker (subdomain takeover). A definitive verdict requires probing each referenced resource for ownership.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

5.2. Network

5.2.1. Test for administrative endpoints exposed to the internet

Description

Administrative interfaces and management consoles must not be publicly reachable. Best practice is IP allowlisting, VPN access, or at minimum strong authentication.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

5.2.2. Test for sensitive files exposed via HTTP

Description

Deployment artifacts and config files such as .env, .git, *.bak, and database dumps must not be served by the web tier.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

5.2.3. Test for /.well-known/security.txt

Description

A well-known security contact lets researchers report issues quickly. RFC 9116 defines the file format and conventions.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

5.3. Network Surface

5.3.1. Test for unnecessary open network ports

Description

Only the ports required to deliver the service should be reachable from the internet. A full port enumeration is intrusive and is performed only against targets where the operator has authorised network-level scanning.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

5.3.2. Test for outdated services on exposed ports

Description

Services bound to internet-facing ports should run current, patched versions. Service-version fingerprinting depends on a prior authorised port scan.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

5.3.3. Test for IPv6 exposure parity

Description

A host reachable over both IPv4 and IPv6 must apply equivalent controls on both. This test records whether the domain publishes an AAAA record introducing an IPv6 attack surface.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

5.3.4. Test for reverse-DNS / PTR information leakage

Description

A PTR record can disclose the hosting provider or internal naming scheme of the server. This test inspects the reverse-DNS entry for the resolved address.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

5.4. TLS Configuration

5.4.1. Test for obsolete TLS/SSL protocol versions

Description

SSLv2, SSLv3, TLS 1.0 and TLS 1.1 are deprecated and must be disabled. This test enumerates the protocol versions the server accepts.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

5.4.2. Test for weak cipher suites

Description

Cipher suites using RC4, 3DES, EXPORT-grade or NULL encryption are unsafe and must not be offered. This test enumerates the cipher suites the server accepts.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

5.4.3. Test for weak key exchange (Logjam / small DH groups)

Description

Diffie-Hellman parameters smaller than 2048 bits are vulnerable to the Logjam attack. This test inspects the key-exchange parameters negotiated by the server.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

5.4.4. Test for known TLS vulnerabilities (POODLE, BEAST, FREAK, DROWN, SWEET32)

Description

Specific protocol and cipher combinations expose well-documented attacks. This test checks the server configuration against that catalogue.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

5.4.5. Test for forward secrecy support

Description

Forward secrecy ensures that compromise of the server key does not retroactively decrypt past sessions. This test checks whether the server prefers ephemeral key-exchange cipher suites.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

5.5. Mail Infrastructure

5.5.1. Test for SMTP open relay

Description

A mail server that relays mail for unauthenticated third parties will be abused for spam. Actively testing relay behaviour involves submitting test mail and is performed only with explicit operator authorisation.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

5.5.2. Test for STARTTLS on mail servers

Description

Mail servers should offer STARTTLS so message transfer can be encrypted in transit. Verifying STARTTLS requires direct SMTP-port interaction with each MX host.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

5.5.3. Test for SPF policy strictness

Description

An SPF record should end in a hard-fail (-all) rather than a soft-fail or neutral qualifier. This test inspects the published SPF record qualifier.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

5.5.4. Test for DMARC enforcement policy

Description

A DMARC record set to p=none only monitors; quarantine or reject is required to actually block spoofed mail. This test inspects the published DMARC policy.

Result

RESULT	STATUS
✓	**Passed**

The target was tested and no issue was detected. This checklist item has therefore been marked as **'Passed'**.

6. Conclusion

GetCodeAudit is of the opinion that the security posture of the target is **good**. Only minor gaps were identified, and none represent immediate security risk.

The table below lists every failed test in priority order, with the corresponding section number in this report. Each section number links to the full finding. Address items from top to bottom to maximise risk reduction per hour of engineering effort.

SCORE	SEVERITY	DESCRIPTION	REFERENCE	SECTION

6.0. Coverage by section

How each checklist section was exercised against this target.

SECTION	PASSED	FAILED	NOT DETECTED	NOT TESTED	N/A
4.1. Deployment	4	0	0	0	0
4.2. Information Disclosure	5	0	0	0	0
4.3. Transport Security	5	0	1	0	0
4.4. Security Headers	5	0	0	0	0
4.5. Cookies & Sessions	5	0	0	0	0
4.6. Authentication	10	0	1	0	0
4.7. Authorization	4	0	1	0	0
4.8. Injection	9	0	6	0	0
4.9. API Security	4	0	0	0	0
4.10. Client-Side	3	0	0	0	0
4.11. Email Security	3	0	0	0	0
4.12. SEO & Discoverability	6	0	0	0	0
4.13. Site Quality	7	0	0	0	0
4.14. Business Logic	4	0	0	0	0
4.15. Session & State	2	0	1	0	0
4.16. Advanced Protocol	4	0	0	0	0
4.17. File Handling	2	0	1	0	0
4.18. Rate Limiting & Abuse	3	0	0	0	0
5.1. DNS	5	0	0	0	0
5.2. Network	3	0	0	0	0
5.3. Network Surface	4	0	0	0	0
5.4. TLS Configuration	5	0	0	0	0
5.5. Mail Infrastructure	4	0	0	0	0

6.1 Tests Not Performed

The automated, unauthenticated scan could not exercise the checks below. They are not failures; they require authenticated access, human reasoning, or controlled/destructive techniques outside this tier. This list is your roadmap for what a manual (expert) penetration test would add. A machine-readable copy is also included in the spreadsheet attached to your report email.

Not detected (actively probed, no confirmed finding; manual verification recommended)

SECTION	TEST
4.3.3	Test that HTTP redirects to HTTPS
4.6.3	Test for CSRF protection on state-changing forms
4.7.2	Test for IDOR (Insecure Direct Object References)
4.8.2	Test for reflected XSS
4.8.4	Test for path traversal
4.8.5	Test for open redirect
4.8.8	Test for Server-Side Request Forgery (SSRF)
4.8.10	Test for stored (persistent) XSS
4.8.11	Test for DOM-based XSS
4.15.1	Test for CSRF token strength and per-session binding
4.17.3	Test for path traversal in file-download parameters

A. Appendix A: Exploitation Walkthrough

No failed findings of sufficient severity were identified during this assessment to warrant a step-by-step exploitation walkthrough.

B. Appendix B: URL inventory

The following 4 URLs were discovered and analysed during this assessment:

1. https://demo.example-corp.test/
2. https://demo.example-corp.test/search
3. https://demo.example-corp.test/login
4. https://demo.example-corp.test/results

C. Appendix C: Probe details

The GetCodeAudit pentest engine executes both passive and active probes against the target. Active probes send carefully-constructed payloads designed to trigger characteristic responses without modifying or damaging the target system.

C.1. Probes executed

PROBE CLASS	DESCRIPTION
Passive crawl	Up to 200 same-origin URLs visited; headers, body content, and links extracted.
TLS analysis	Certificate validity, expiration, issuer chain, HSTS, HTTP-to-HTTPS redirect.
Security headers	HSTS, CSP, X-Frame-Options, X-Content-Type-Options, Referrer-Policy, Permissions-Policy.
Cookie analysis	Secure, HttpOnly, SameSite flag presence on all cookies.
DNS & email	SPF, DMARC, CAA, subdomain enumeration (25 candidates).
Reflected XSS	Benign canary payload reflection check across query parameters.
SQL injection	Error-based payloads across string and numeric contexts on GET parameters of every parameterised URL.
Blind SQL injection	Time-based inference (DB sleep vs control) for GET injection points that return no visible error.
SQL injection (forms)	POST-form parameters tested for error-based and boolean-based (TRUE/FALSE response divergence) SQL injection.
Open redirect	Thirteen redirect parameter names tested with an off-domain target.
Path traversal	Directory-traversal sequences (../etc/passwd and the Windows equivalent) across parameterised URLs.
Directory bruteforce	Bruteforce of 40+ common paths covering admin panels, backups and config locations.
Sensitive file exposure	Content-verified checks for exposed .git / .env / dependency manifests, debug endpoints (phpinfo, server-status), directory listings, and source backup files (.bak, ~, .old) on discovered paths.
Subdomain enumeration	DNS resolution of 25 common subdomain candidates.
HTTP method enumeration	OPTIONS request to enumerate advertised methods.
CORS misconfiguration	Reflected-origin probe with and without credentials.
Host header injection	Attacker-controlled Host header reflected in the response.
Login rate limiting	Eight rapid failed login attempts with throwaway credentials.
JavaScript secrets	Pattern matching for AWS/GitHub/Stripe/Google API keys in loaded JavaScript.
HTML comments	Harvest of HTML comments matching sensitive keyword patterns.
GraphQL introspection	Probe of common GraphQL paths for unrestricted schema introspection.
JWT analysis	Decoding and inspection of JWTs found in responses (algorithm, expiry, payload claims).
Subresource integrity	External script tags checked for the integrity attribute.
DOM XSS sinks	Pattern detection of innerHTML/document.write/eval consuming URL parameters.
Dependency versions	jQuery, Bootstrap, Vue, React, AngularJS, Lodash and Moment.js version checks.
Weak password policy	Signup-form detection and password-input attribute inspection.
IDOR indicators	Flagging of URLs that expose sequential numeric IDs as manual-review candidates.
Inline event handlers	Count of onclick/onerror and similar attribute occurrences in served HTML.
Mixed content	HTTPS pages checked for resources loaded over plaintext HTTP.

Cache-Control	Sensitive responses checked for Cache-Control: no-store.
Cookie prefix	Session cookies checked for the __Host- / __Secure- prefix.
DNS zone transfer	Authoritative name servers tested for an unrestricted AXFR zone transfer.
Server & disclosure hardening	Active TRACE/TRACK method test, verbose-error and stack-trace detection, end-of-life server-software identification, sensitive robots.txt entry analysis, and internal (RFC-1918) IP-address leakage checks.
Coverage gaps	Clickjacking / UI-redress defence (X-Frame-Options or CSP frame-ancestors), CMS/framework disclosure via the generator meta tag, and DKIM signing presence at common selectors.
Port scan (nmap)	TCP scan of the top 1000 ports with service/version detection (-sV). Runs only when enabled by the operator; identifies network services exposed beyond the web application.

C.2. Non-destructive policy

The pentest engine never sends destructive payloads (e.g. SQL DELETE/DROP), never attempts privilege escalation, and never tests known account credentials. Rate limiting is hard-capped at 5 requests per second.

C.3. Disclaimer

Automated testing cannot replace manual penetration testing for systems handling sensitive data, financial transactions, or regulated information. Findings represent a snapshot at scan time; the security posture of the target may have changed since. **False positives and false negatives may occur**, and all findings should be independently validated before remediation.

A result of **“Not Detected”** for any test means only that this automated, unauthenticated engine did not observe evidence of the issue within the scan window and reachable surface. It is **not** a statement that the target is free of that class of vulnerability. Tests that require authentication, human reasoning, or destructive payloads are explicitly marked and were not exercised.

GetCodeAudit accepts no responsibility or liability for actions taken, or not taken, on the basis of this report, and makes no warranty as to the completeness or accuracy of this assessment. Use of this report is governed by the GetCodeAudit Pentest Terms of Service.