Breached Labs

DEMO PENETRATION TEST

Sample Report // April 2025

completed by breachedlabs.com

BUSINESS CONFIDENTIAL | V1.0

I. Confidentiality & Distribution

contact@breachedlabs.com

1.1 Document Classification

THIS IS A CONFIDENTIAL DOCUMENT.
As such this document contains sensitive information regarding the security posture of Demonstration Organization's systems as evaluated during the penetration test conducted between **April 7, 2025** and **April 21, 2025**.

Unauthorized disclosure, copying, distribution, or use of this report or its contents is strictly prohibited. This report is intended solely for the use of the authorized individuals and entities listed in the distribution list below.

1.2 Distribution List

This report is intended for the exclusive use of the following authorized personnel within **Demonstration Organization** and **Breached Labs**:

- **Demonstration Organization:**
  - John Doe, Head of Engineering (john@demo.com)
  - John Appleseed, CISO (appleseed@demo.com)
- **Breached Labs:**
  - Lead Pentester (pentester@breachedlabs.com)
  - Quality Assurance Reviewer (quality@breachedlabs.com)

Recipients are responsible for maintaining the confidentiality of this document. Distribution beyond this list requires explicit written consent from both Demonstration Organization's executive management and Breached Labs.

1.3 Legal Disclaimer

This penetration test report describes the results of a security assessment performed on the systems specified within the agreed-upon scope and timeframe. The testing methodology employed simulates common attack techniques to identify potential vulnerabilities.

Breached Labs has made every effort to identify and report vulnerabilities within the scope of this engagement using industry-standard practices and tools. However, penetration testing provides a "point-in-time" assessment. It is not possible to guarantee that all existing vulnerabilities have been discovered, nor can this assessment guarantee that new vulnerabilities will not emerge in the future due to changes in the systems, configurations, or threat landscape.

The findings and recommendations presented herein are based on the information available and the state of the target systems during the testing period. Breached Labs is not responsible for any misuse of the information contained in this report or for any damages resulting from the exploitation of vulnerabilities, whether identified in this report or not.

The client (Demonstration Organization) retains full responsibility for the security of its systems, data, and infrastructure, including the implementation and verification of any remediation actions based on this report.

This report does not constitute a **guarantee of security or compliance** but serves as an assessment of the security posture based on the defined scope and methodology at the time of testing.

2. Executive Summary

contact@breachedlabs.com

2.1 Introduction & Engagement Overview

Breached Labs was engaged by Demonstration Organization to perform a penetration test targeting their primary web application and associated API infrastructure. This assessment aimed to identify **security vulnerabilities**, evaluate the **effectiveness of existing security controls**, and provide **actionable recommendations** to enhance the overall security posture of the application environment.

The engagement simulated attacks that could be leveraged by malicious actors with authenticated user access (**Grey Box** approach, reflecting potential insider threats or compromised user accounts).

2.1.1 Project Objectives

The engagements primary points of focus include:

- Identify vulnerabilities within the in-scope web application and API endpoints, focusing on common web vulnerabilities **(OWASP Top 10**), business logic flaws, and authorization bypasses.
- Assess the potential business impact of exploitable vulnerabilities, particularly concerning data confidentiality (customer data segregation), integrity, and application availability.
- Provide clear, prioritized, and actionable recommendations for remediation to assist Demonstration Organization's development and security teams.

2.1.2 Scope Summary

This penetration test focused on the security posture of the primary web application **(https://demo.com)** and its associated backend API infrastructure. The assessment included evaluation of the application's features and workflows accessible to a standard authenticated user. The API portion of the scope comprised approximately **100 endpoints**, whose definitions and expected interactions were detailed in the provided Postman collection and Swagger documentation.

To facilitate a realistic grey-box assessment simulating threats from authenticated users or compromised accounts, Demonstration Organization provided valid credentials for a standard user role. Our testing leveraged these credentials and the API documentation to examine authenticated functionality, business logic, authorization controls, and potential vulnerabilities accessible post-login.

2.1.3 Engagement Dates & Duration

The active testing phase of this engagement was conducted between April 7, 2025 and April 21, 2025 spanning a total of 2 weeks. Report generation and finalization occurred subsequently.

2.2 Overall Risk Posture Assessment

The overall risk posture for the assessed application environment (https://demo.com and API) has been determined to be **HIGH**. This conclusion is drawn from the identification of multiple significant vulnerabilities and a qualitative analysis of their potential business impact. The **HIGH** rating reflects a concerning pattern of weaknesses, primarily rooted in inadequate **authorization checks**, and insufficient **input validation**

These underlying issues manifest as critical vulnerabilities, including confirmed avenues for authenticated attackers to potentially **compromise sensitive data across tenant boundaries**, **circumvent core permission models**, and potentially **manipulate or disrupt backend database operations** via injection flaws, impacting service availability. Although foundational security measures like HTTPS are in place, the severity and nature of the identified vulnerabilities, particularly those allowing direct bypass of business logic and access controls, necessitate this **HIGH** risk rating and require immediate strategic attention.

2.2.1 Key Findings Overview

The most significant risks identified during the assessment include:

[x] Critical Authorization Bypass Enabling Cross-Tenant Data Access
Vulnerabilities were confirmed allowing authenticated users to potentially access or modify data belonging to other users by manipulating identifiers in API requests.

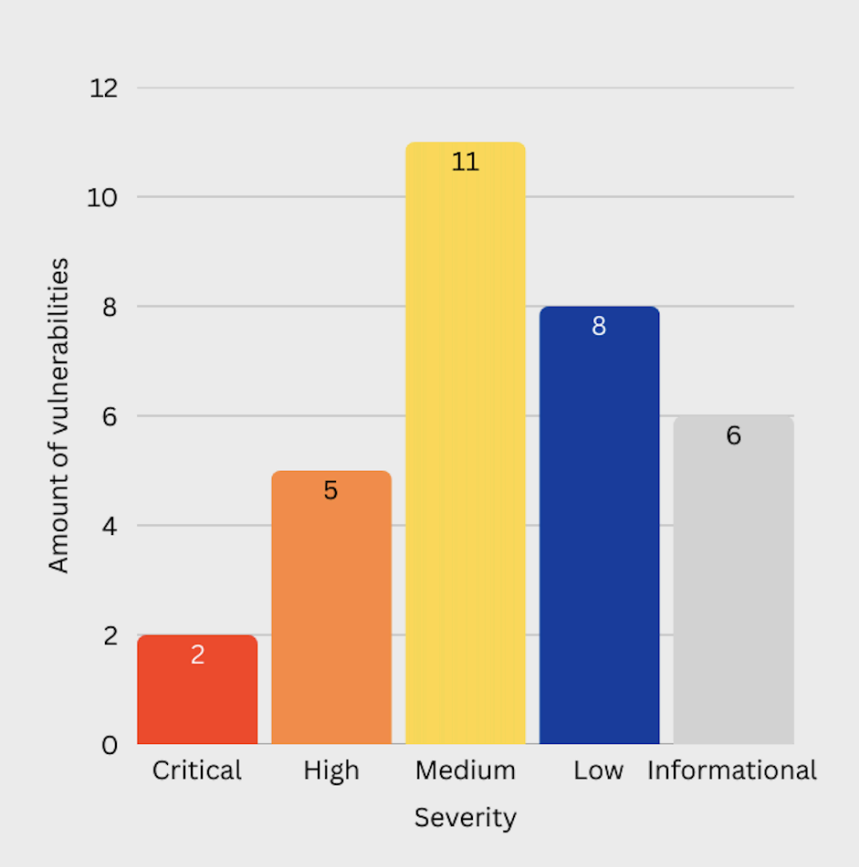
[x] High Severity SQL Injection in /api/registration endpoint
A high-impact SQL injection vulnerability was discovered, potentially allowing attackers to extract sensitive data directly from the backend PostgreSQL database.

[x] Insecure Direct Object References (IDOR) Leading to Privilege Escalation
Multiple instances were found where users could access or manipulate resources or perform actions intended for higher-privileged roles by modifying parameters.

[x] Significant Frontend Input Validation Weaknesses
Insufficient client-side and server-side validation on frontend components could lead to cross-site scripting attacks, impacting users.

2.2.2 Visual Risk Profile

The assessment revealed vulnerabilities across various severity levels, with a significant concentration in the higher-risk categories. This distribution underscores the need for prioritized remediation efforts



An analysis of the findings distribution reveals a critical concentration of risk demanding immediate attention. While this report details numerous vulnerabilities across the spectrum, including 14 informational and low-risk items, the most significant data point is that **nearly 25% of all identified findings (7 out of 32) are classified as either Critical or High severity.**

These are not peripheral issues; they represent exploitable weaknesses with the potential for severe business impact, such as significant data compromise or operational disruption. Consequently, these specific high-impact vulnerabilities constitute the most immediate threats discovered and **must be the primary focus of urgent remediation efforts.**

2.3 Business Impact Synopsis

Operational Disruption

-

Exploitation of SQL injection or severe authorization flaws could lead to database corruption, application downtime, or require significant effort for incident response and recovery, impacting service availability for legitimate users.

Reputational Damage & Loss of Trust

-

The confirmed ability for users to potentially access or modify data belonging to other customers represents a severe breach of trust. Public disclosure or discovery of such an incident could lead to significant reputational damage, customer churn, and negative publicity.

Data Breach & Compliance Costs

-

Unauthorized access to sensitive user or business data via SQL Injection or authorization bypasses could constitute a data breach. Even without specific mandates like PCI/GDPR explicitly stated as requirements, mishandling user data carries significant liability and potential financial penalties under general privacy principles. Remediation and potential incident response costs would also be substantial.

Compromised Business Logic

-

Flaws in privilege escalation and frontend validation could allow users to bypass intended workflows, potentially leading to fraud, data integrity issues, or abuse of application features.

2.4 Summary of Positive Security Findings

Despite the critical issues identified, the assessment also noted positive security practices:

- Consistent use of HTTPS across the application, ensuring data encryption in transit.
- Implementation of multi-factor authentication (MFA) options for user accounts adds a significant layer of protection against credential compromise.
- Basic input sanitization was observed in several areas, indicating some level of security awareness within development practices.
- Use of modern frameworks provides inherent protection against some common vulnerability classes.

3. Engagement Scope & Methodology

contact@breachedlabs.com

3.1 Detailed Scope Definition

This section formally outlines the agreed-upon scope, defining the exact targets, methodologies, and constraints under which this penetration test was conducted. It specifies the applications, network ranges, API endpoints, and user roles included within the assessment boundaries, alongside any explicit exclusions or operational restrictions. **Strict adherence** to this detailed scope throughout the engagement was paramount to ensure a focused, efficient, and relevant security assessment that directly aligns with the key project objectives and security concerns articulated by the Demonstration Organization.

3.1.1 In-Scope Assets

The following assets and conditions constituted the agreed-upon boundaries for all penetration testing activities conducted during this engagement:

Primary Web Application Interface:

The assessment comprehensively covered the user-facing web application accessible via its primary URL: `https://demo.com`. This included all pages, features, and workflows reachable from this entry point within the authenticated user context defined below.

Supporting API Infrastructure:

The backend Application Programming Interface (API) infrastructure, which provides core functionality and data services to the primary web application, was in scope. This specifically included the set of approximately 100 API endpoints defined and documented within the provided Postman collection and Swagger specification files. Interactions with these endpoints formed a significant part of the testing effort.

Authenticated User Roles & Perspectives:

Testing was conducted from the perspective of authenticated users possessing different privilege levels to assess access controls and potential privilege escalation vectors. Credentials were provided by Demonstration Organization and utilized for the following defined roles:

- **Standard User Role:** Representing typical user functionality (Provided Login: ‘PentestLogon’).
- **Elevated User Role:** Representing a user with additional administrative or management privileges within the application (Provided Login: ‘PentestElevated’).

Designated Testing Environment:

All testing activities were performed exclusively against the designated User Acceptance Testing (UAT) environment, intended to closely mirror the production configuration. Access to this environment was provided via the following address: `127.0.0.1:80`. No testing was conducted against production or other environments.

3.1.2 Out-of-Scope Assets & Restrictions

To ensure focus and prevent disruption to unrelated systems or services, the following were explicitly excluded from the scope:

- Underlying server infrastructure OS-level testing (unless an application vulnerability directly provided access).
- Corporate network infrastructure outside the defined UAT environment.
- Denial of Service (DoS) or Distributed Denial of Service (DDoS) attacks.
- Social engineering, phishing, or physical security assessments.
- Third-party applications, integrations, or external services linked from the primary application (unless explicitly stated otherwise).
- Destructive testing aimed at causing irreversible damage or prolonged outages.

3.2 Testing Methodology

To ensure thoroughness and accuracy within the defined scope, Breached Labs employed a **multi-faceted testing methodology** that integrates the strengths of both automated scanning and extensive manual analysis. Our process began with leveraging specialized automated tools to perform broad sweeps of the target application (https://demo.com) and API infrastructure, rapidly identifying known vulnerability patterns and potential misconfigurations. The results from this automated phase were then critically analyzed and used to inform **targeted manual investigation.**

This crucial manual phase involved our security experts performing deep dives into application functionality, simulating sophisticated attack techniques, validating automated findings (eliminating false positives), and probing for complex vulnerabilities requiring human intuition and contextual understanding. By systematically **combining the efficiency of automation with the critical thinking and adaptability of manual testing**, we achieve comprehensive coverage, providing the Demonstration Organization with a more accurate and actionable assessment of their security posture within the specific boundaries of this engagement.

3.2.1 Assessment Approach: Grey Box

This engagement utilized a **Grey Box** testing approach. This means our testers were provided with user-level credentials and access to documentation (API specifications) that a standard authenticated user might possess. Unlike a Black Box test (no prior knowledge), this allows for deeper testing of authenticated functionality, business logic, and authorization controls. Unlike a White Box test, full source code or administrator-level access was not provided, simulating an attacker who has **compromised a user account or has legitimate user access.**

3.2.2 Testing Frameworks Leveraged

Our methodology aligns with industry best practices and incorporates elements from recognized frameworks, including:

- **OWASP Top 10 2021:** Focused testing against the most critical web application security risks.
- **OWASP Web Security Testing Guide (WSTG):** Provided a comprehensive checklist for web application security testing techniques.
- **Penetration Testing Execution Standard (PTES):** Guided the overall phases and structure of the engagement.

3.2.3 Testing Phases

The penetration test followed these logical phases in sequential order:

1. **Planning & Reconnaissance:**
   Defining scope, objectives, rules of engagement, and initial information gathering using provided documentation and authenticated exploration.

2. **Scanning & Enumeration:**
   Utilizing automated tools and manual techniques to identify accessible application components, API endpoints, features, and potential attack surface areas from an authenticated perspective.

3. **Vulnerability Analysis:**
   Manually probing identified areas and using specialized tools to discover potential vulnerabilities like injection flaws, authorization issues, session management weaknesses, XSS, etc.

4. **Exploitation:**
   Attempting to safely exploit identified vulnerabilities to confirm their existence and assess their potential impact. Proof-of-concept exploits were developed where feasible.

5. **Post-Exploitation (Limited):**
   Where initial exploitation was successful, limited attempts were made to understand the extent of potential access or impact (e.g., assessing data exposure, potential for privilege escalation) within the grey box constraints.

6. **Reporting:**
   Consolidating all findings, providing detailed descriptions, impact assessments, reproduction steps, and actionable remediation recommendations.

3.3 Risk Rating Methodology

To provide actionable insights and facilitate prioritized remediation, all identified vulnerabilities were systematically classified based on their **potential risk specifically tailored** to the Demonstration Organization's context**.** This classification process involved a careful evaluation balancing two primary dimensions: **the likelihood of a threat actor successfully discovering and exploiting the vulnerability**, considering factors like complexity and required access, and **the potential business impact** should the vulnerability be exploited, encompassing consequences to data confidentiality, integrity, availability, operational continuity, and reputation. The resulting severity rating serves as a crucial guide for focusing resources on the most critical issues first.

3.3.1 Severity Levels Defined

Critical

Vulnerabilities that could lead to immediate, widespread compromise of sensitive data, complete system takeover, or severe disruption of critical business functions. Typically easily exploitable. (Example: Remote Code Execution, Critical SQL Injection, System-Wide Authentication Bypass).

High

Vulnerabilities that could lead to significant data exposure, unauthorized access to privileged functionality, or notable disruption of specific business processes. Often exploitable with moderate effort. (Example: Stored Cross-Site Scripting impacting admins, Privilege Escalation, Severe Authorization Flaws).

Medium

Vulnerabilities that could lead to limited data disclosure, compromise of individual user accounts, or minor service degradation. Exploitation might require more specific conditions or user interaction. (Example: Reflected Cross-Site Scripting, CSRF on important functions, Information Disclosure).

Low

Vulnerabilities with minimal direct impact, often violating best practices or contributing to reconnaissance efforts. Exploitation is typically difficult or yields little value. (Example: Software Version Disclosure, Weak TLS Ciphers, Missing Security Headers with low impact).

Informational

Observations that are not exploitable vulnerabilities but represent deviations from best practices, potential areas for future improvement, or positive security findings.

3.3.2 Factors Considered

The assigned severity level considers:

- **Likelihood:**
  The ease with which an attacker could discover and exploit the vulnerability. Factors include required privileges, attack complexity, and user interaction needed.

- **Impact:**
  The potential consequences if the vulnerability is successfully exploited. Factors include confidentiality (data accessed), integrity (data modified), availability (service disruption), and potential business/reputational harm.

- **CVSS v3.1:**
  The **Common Vulnerability Scoring System (CVSS) version 3.1** framework was used as a guideline to provide a standardized baseline score, considering metrics like Attack Vector (AV), Attack Complexity (AC), Privileges Required (PR), User Interaction (UI), Scope (S), Confidentiality (C), Integrity (I), and Availability (A). The final severity rating may be adjusted based on specific business context provided by the Demonstration Organization.

3.4 Tools Utilized

The assessment employed a combination of commercial, open-source, and custom tools, always augmented by extensive manual analysis and verification. Key tools included:

- **Intercepting Proxies:** Burp Suite Professional, OWASP ZAP
- **Vulnerability Scanners:** Burp Suite Professional (Scanner Module), Nuclei
- **API Testing Tools:** Postman, Burp Suite extensions (e.g., OpenAPI Parser, Param Miner)
- **Manual Exploitation Tools:** SQLMap, ffuf, dirbuster
- **Frameworks/Libraries:** Python scripting (for custom checks), various decoding/encoding tools.
- **Information Gathering:** Browser Developer Tools, Wappalyzer (limited context)

(Note: This list is representative. Actual tools vary per engagement. The emphasis is on the combination of automated support and manual expertise.)

4. Detailed Findings & Recommendations

contact@breachedlabs.com

4.1 Critical Risk Findings

4.1.1 Finding ID: BL-CRIT-001 - Critical Authorization Bypass Allows Cross-Tenant Data Modification

4.1.1.1 Description

The application's API endpoint responsible for updating store inventory details (`/api/v1/inventory/update`) fails to properly validate if the user making the request is authorized to manage the specific `storeId` provided within the API request body. An authenticated user belonging to one store (Tenant A) can successfully submit a request containing the `storeId` of a different, unrelated store (Tenant B), thereby modifying the inventory details (e.g., item quantity, price) of that other tenant. This constitutes a critical failure in the application's multi-tenant segregation controls and directly enables unauthorized modification of other customers' business data.

4.1.1.2 Affected Asset(s) / Location(s)

API Endpoint: `POST /api/v1/inventory/update`
Vulnerable Parameter: `storeId` (within the JSON request body)
Affected Roles: All authenticated user roles possessing permissions to utilize the inventory update feature (e.g., Standard User `PentestLogon`, Elevated User `PentestElevated`).

4.1.1.3 Business Impact

Successful exploitation of this vulnerability grants a malicious or compromised user account the ability to directly and arbitrarily alter critical business data (such as inventory levels, product pricing, or availability flags) belonging to **any other store (tenant)** within the platform. The potential business consequences are severe and include:

- **Operational Chaos:** Affected stores may face incorrect stock counts, leading to unfulfilled orders or inability to sell available items.
- **Financial Loss:** Incorrect pricing could lead to direct financial losses for the affected tenant or the platform.
- **Complete Loss of Customer Trust:** The discovery that data can be modified by other tenants would severely damage the platform's reputation and likely lead to significant customer churn.
- **Reputational Damage:** Public disclosure of such a fundamental flaw would result in substantial negative publicity.
- **Legal and Compliance Liability:** Failure to protect the integrity of customer data carries significant potential legal consequences, even in the absence of specific regulatory mandates mentioned in the scope.

4.1.1.4 Steps to Reproduce

The following steps detail how an authenticated user (**User A from Store 123**) can modify data belonging to another tenant (**Store 456**):

1. Log in to the application (`https://demo.com`) as User `PentestLogon` (associated with Store ID `123`).
2. Navigate to the application section responsible for managing inventory. Using an intercepting proxy (e.g., Burp Suite), capture the API request made when legitimately updating an inventory item for Store 123.
3. Observe the intercepted request to `POST /api/v1/inventory/update`. Note the original JSON body containing the legitimate `storeId`.

```
// Example Request Body Captured
{
    "itemId": "DEMO_ITEM_001",
    "quantity": 50,
    "storeId": 123 // User A's legitimate Store ID
  }
```

[**IMAGE REDACTED]**

4. Modify the value of the `storeId` parameter within the intercepted request's JSON body to target a different, known (or guessed) store ID, for example, `456`. Modify other parameters like `quantity` or `itemId` as desired for impact. Ensure User A's valid authentication token/session cookie remains in the request headers.

```
// Example Modified Request Body
{
    "itemId": "DEMO_ITEM_XYZ", // Item potentially belonging only to Store 456
    "quantity": 9999,       // Maliciously inflated quantity
    "storeId": 456         // Target Store ID belonging to another tenant
  }
```

[**IMAGE REDACTED]**

5. Forward the maliciously crafted request to the server via the intercepting proxy.
6. Observe the server's response. A success status code (e.g., `HTTP/1.1 200 OK` or `HTTP/1.1 204 No Content`) accompanied by a standard success message indicates the server improperly processed the unauthorized request.

[**IMAGE REDACTED**]

4.1.1.5 Remediation Recommendations

Implement robust, non-bypassable server-side authorization controls for all API endpoints handling tenant-specific data or actions. Specifically for this endpoint:

1. **Session-Based Ownership Check:** On the server-side, upon receiving a request to `/api/v1/inventory/update`, retrieve the authenticated user's identity and their definitively associated `storeId` directly from their validated session object or security context. **Do not trust the `storeId` provided within the request body for authorization decisions.**
2. **Enforce Match:** Compare the `storeId` retrieved from the user's secure session against the `storeId` present in the request body.
3. **Deny Mismatch:** If the `storeId` from the request body **does not exactly match** the `storeId` associated with the authenticated user's session, the request **must be rejected immediately**. Return an appropriate HTTP error status code, such as `HTTP 403 Forbidden` or `HTTP 404 Not Found` (to avoid disclosing existence of other stores), and log the authorization failure event server-side.
4. **Centralize Logic:** Encapsulate this tenant ownership verification logic within a reusable authorization middleware, filter, or decorator that can be applied consistently across all relevant API endpoints handling sensitive, tenant-scoped data. This avoids potential inconsistencies from implementing checks individually on each endpoint.

4.1.1.6 References

OWASP Top 10 2021: A01 Broken Access Control
https://owasp.org/Top10/A01_2021-Broken_Access_Control/

OWASP Cheatsheet Series: Authorization Testing Automation
https://cheatsheetseries.owasp.org/cheatsheets/Authorization_Testing_Automation_Cheat_Sheet.html

5. Prioritized Remediation Plan

contact@breachedlabs.com

5.1 Remediation Urgency Matrix

To facilitate strategic decision-making regarding remediation, the matrix below offers a high-level synthesis of the assessment's findings. It maps each vulnerability identified in Section 4 to its assigned **Severity** rating, alongside two key planning factors: an **Estimated Remediation Effort**: a relative gauge of complexity (Low, Medium, High), and the **Recommended Urgency** indicating the suggested timeframe for mitigation.

This overview is intended to assist the Demonstration Organization in allocating resources effectively and developing a phased remediation roadmap. It is crucial to remember that the Effort estimations are relative benchmarks, predicated on the assumption that developers tasked with the fixes have existing familiarity with the relevant sections of the application's codebase.

Finding ID	Vulnerability Title	Severity	Estimated Remediation Effort	Recommended Urgency
BL-CRIT-001	Critical Authorization Bypass Allows Cross-Tenant Data Mod	Critical	High	Immediate
BL-HIGH-001	SQL Injection in Product Search API	High	Medium	Immediate
BL-MED-001	Stored Cross-Site Scripting (XSS) in User Profile Desc	Medium	Low	Soon
BL-LOW-001	Missing Content-Security-Policy (CSP) Header	Low	Low	Plan
BL-INFO-001	Verbose Error Messages Reveal Stack Trace	Informational	Low	Plan

Effort Estimation Key:

- **Low:** Minor code change, configuration tweak, likely <1 day effort.
- **Medium:** Requires moderate code changes, potentially affecting multiple files or requiring careful testing, likely 1-3 days effort.
- **High:** May require architectural changes, significant refactoring, extensive testing, or affect core application logic, potentially >3 days effort.

Urgency Key:

- **Immediate:** Address within the next sprint or patch cycle (within days/1-2 weeks). Poses significant immediate risk.
- **Soon:** Address in the near term (within 1-2 months). Poses moderate risk or contributes significantly to exploitability of higher risks.
- **Plan:** Address as part of regular maintenance or future deployment cycles (within 3-6 months). Represents best practice or low-impact risk.

5.2 Recommended Remediation Roadmap

Given the range of findings, **we recommend structuring the remediation process into logical, manageable phases.** This methodical approach prevents the team from being overwhelmed and provides a clear roadmap for improvement. The core principle of this phasing is to **ensure that the highest impact risks, as defined by their severity rating and potential business consequences, receive immediate attention and are mitigated first.** By tackling vulnerabilities in this prioritized order, the Demonstration Organization can systematically address the most critical security gaps, track progress effectively, and build momentum towards resolving all identified issues in a controlled manner.

5.2.1 Phase 1: Immediate Threat Mitigation (Urgency: Immediate)

Focus: Critical & High severity vulnerabilities.

Actions:

Fix BL-CRIT-001 (AuthZ Bypass)

Implement strict server-side ownership validation for the inventory update API and any other APIs handling tenant-specific data. This likely requires careful analysis and potentially refactoring authorization logic.

Fix BL-HIGH-001 (SQL Injection)

Immediately implement parameterized queries for the product search API endpoint. Review other database interaction points for similar vulnerabilities.

Goal: Eliminate avenues for direct data compromise and cross-tenant data leakage.

5.2.2 Phase 2: Addressing Significant Gaps (Urgency: Soon)

Focus: Medium severity vulnerabilities.

Actions:

Fix BL-MED-001 (Stored XSS)
Implement robust contextual output encoding for the user profile description and review all other areas where user-supplied content is displayed.

Goal: Reduce the risk of session hijacking and user-level compromises.

5.2.3 Phase 3: Hardening and Best Practices (Urgency: Plan)

Focus: Low & Informational findings, reinforcing defenses.

Actions:

Implement BL-LOW-001 (CSP Header)
Define and deploy a Content-Security-Policy header to add a crucial layer of defense against XSS and related attacks.

Address BL-INFO-001 (Verbose Errors)
Configure the application and web server for production error handling, ensuring stack traces are logged server-side only and users receive generic error messages.

Goal: Implement defense-in-depth measures and adhere to security best practices.

5.2.4 Post-Remediation

The remediation process is incomplete without independent validation. Breached Labs **strongly recommends performing dedicated verification testing (re-testing)** subsequent to the deployment of remediation actions, **particularly** for the Critical (BL-CRIT-001) and High (BL-HIGH-001) severity findings identified in this report. The primary objective of this re-testing is twofold:

1. To **independently validate** that the implemented fixes effectively eliminate the original reported vulnerability under realistic testing conditions.
2. To **provide assurance** that the remediation efforts themselves have not inadvertently introduced new security weaknesses or caused functional regressions in related application areas. Simple developer checks may not be sufficient to catch bypasses or subtle errors introduced during the fix.

Given the potential impact of medium-severity issues, extending this verification process to key Medium findings (such as BL-MED-001 - Stored XSS) is also **strongly advised** to achieve greater confidence in the overall security improvement.

This structured, phased remediation approach, prioritized by demonstrable risk severity, allows the Demonstration Organization to **allocate its development and security resources most effectively.** It ensures that **immediate defensive efforts are concentrated on neutralizing the vulnerabilities posing the greatest and most imminent danger** to core business functions, sensitive data integrity, and ultimately, the trust of its customers. Addressing the highest risks first delivers the most significant reduction in overall risk exposure in the critical initial stages of the remediation lifecycle.

6. Conclusion

contact@breachedlabs.com

6.1 Summary of Security Posture

This report concludes the penetration test conducted against the Demonstration Organization's primary web application and associated API, performed under Grey Box testing conditions. Our assessment culminated in an overall security posture rating of **HIGH RISK** at the time of testing. This significant risk level is not assigned lightly but is primarily driven by the identification of critical vulnerabilities in fundamental security mechanisms. Specifically, the **broken authorization controls (BL-CRIT-001)**, which demonstrably enable potential **cross-tenant data modification**, and the presence of **high-risk SQL Injection (BL-HIGH-001)**, presenting a direct pathway to **widespread database compromise**, are the most critical contributors.

Furthermore, vulnerabilities such as **Stored Cross-Site Scripting (BL-MED-001)** compound the risk by exposing users to session hijacking and other client-side attacks. While the implementation of positive security controls, including consistent HTTPS usage and available Multi-Factor Authentication (MFA) options, was noted and is commendable, these foundational measures are unfortunately insufficient to counteract the severity of the identified vulnerabilities rooted in **core authorization logic and secure data handling practices.** These represent significant, exploitable weaknesses demanding **immediate attention and remediation** to prevent potential compromise and adequately protect sensitive customer data and application integrity.

6.2 Engagement Limitations & Assumptions

It is important to understand the context and limitations of this assessment:

Point-in-Time Assessment: The findings reflect the state of the application and environment only during the testing period (April 7, 2025 to April 21, 2025). Subsequent changes to code, configuration, or infrastructure may introduce new vulnerabilities or alter existing ones.

Scope Restrictions: Testing was confined to the assets explicitly defined in Section 3.1. Out-of-scope systems, third-party integrations, underlying infrastructure OS, and aspects like social engineering were not evaluated.

Environment Dependency: Testing was performed against the designated UAT environment. While intended to mirror production, subtle differences could exist, potentially affecting the applicability or exploitability of findings in the live environment.

Grey Box Constraints: Testing relied on the provided user credentials and documentation. Findings are representative of threats posed by authenticated users or attackers who have

obtained such credentials, but may not cover all vulnerabilities discoverable with full source code access (White Box) or administrator privileges.

Non-Exhaustive Testing: While comprehensive methodologies were employed, no penetration test can guarantee the discovery of *all* potential vulnerabilities. Time constraints and the inherent complexity of systems mean that undetected flaws might still exist.

No Destructive Testing: Testing was performed in a non-destructive manner to avoid impacting the stability of the test environment.

6.3 Recommended Next Steps

Based on the findings of this assessment, Breached Labs recommends Demonstration Organization take the following steps:

Prioritize Remediation: Address the identified vulnerabilities following the **Prioritized Remediation Plan (Section 5)**, focusing immediate efforts on the Critical (BL-CRIT-001) and High (BL-HIGH-001) risk findings.

Schedule Verification Testing (Re-testing): Engage Breached Labs or another qualified third party to perform verification testing once remediation for Critical and High findings is complete. This is essential to confirm the effectiveness of the fixes.

Review Strategic Recommendations: Evaluate and plan the implementation of the strategic recommendations outlined in **Section 6** (e.g., architectural review, enhancing SDLC security, improving logging) to address root causes and improve long-term security posture.

Internal Review & Training: Share relevant findings internally with development and operations teams. Use the report as a basis for targeted secure coding training, particularly focusing on authorization, input validation, and secure database interaction (parameterized queries).

Regular Assessment Cadence: Establish a program of regular penetration testing (e.g., annually or post-major releases) to proactively identify new vulnerabilities as the application evolves. Consider targeted assessments on new features before deployment.

Breached Labs is available to discuss these findings and recommendations in further detail and assist with planning and executing verification testing.

Breached Labs is a specialized offensive security boutique, focusing exclusively on high-fidelity penetration testing. We partner with businesses, IT providers, and MSPs, particularly those based in Ireland, providing deep, actionable insights derived from realistic attack simulations. As your dedicated testing partner, we identify vulnerabilities *before* attackers do, helping you validate controls and prevent breaches. Utilizing a rigorous, real-world methodology, we combine expert manual analysis with an attacker's mindset to ensure your security posture is effectively challenged against the evolving threat landscape. Our specialized focus enables us to become a trusted validation partner for organizations demanding thorough security assessments.

Certifications & Accreditations

Breached Labs staff validate their deep technical expertise through industry-recognized certifications, including elite qualifications such as the OSCE3. More importantly, we prioritize continuous learning, ensuring the team constantly pursues cutting-edge training on the newest threats and attack techniques. This commitment guarantees our testing methodologies are sharp, relevant, and capable of uncovering the vulnerabilities that matter most to our clients' security.

- OSCP - Offensive Security Certified Professional
- OSWE - Offensive Security Web Expert
- OSEP - Offensive Security Experienced Penetration Tester
- OSED - Offensive Security Exploit Developer
- OSCE3 - Offensive Security Certified Expert 3
- CompTIA Pentest+

contact@breachedlabs.com