



WEB APPLICATION SECURITY ASSESSMENT

Prepared for:

EXAMPLE CORP

Prepared by:

DeepDefence Technologies

Date prepared:

29 September 2026

Version:

V1.0 — Initial

Classification:

Confidential — Internal Use Only

For any questions related to this report, please contact support@deepdefence.co

1. Executive Summary

1.1 Overview

The assessment identified 3 security finding(s) across the assessed application. The findings have been categorized by severity to help prioritize remediation.

1.2 Overall Risk Rating

The findings identified during the assessment are summarized below by severity category.

Critical	High	Medium	Low
1	2	0	0

1.3 Recommended Priorities

Prioritize remediation of Critical and High severity findings, followed by Medium and Low severity findings based on their business impact and applicability.

1.4 Positive Observations

The assessment also considered areas where the application demonstrated positive security practices. Specific positive observations should be documented here where applicable.

2. Scope and Methodology

2.1 Scope

#	Target URL
1	https://app.example.com
Assessment Property	Details
Testing Type	Black-box
Authentication	Authenticated — Standard User

The assessment was performed against the target URL(s) listed above as black-box testing. The assessment did not include access to the application's source code or internal implementation.

2.2 Methodology

Testing combined automated security assessment with manual analyst review and validation of identified findings.

Identified findings were reviewed to validate their applicability and reduce false positives before being included in the report.

Severity ratings were determined using the Common Vulnerability Scoring System (CVSS) v3.1 where applicable.

3. Findings Summary

#	Finding	Severity	CVSS Score	CWE
1	Reflected Cross-Site Scripting (XSS)	High	8.0	CWE-79
2	SQL Injection	Critical	9.8	CWE-89
3	SQL Injection — Database Error Disclosure	High	8.8	CWE-89

4. Detailed Findings

4.1 Reflected Cross-Site Scripting (XSS)

Severity	CVSS Score	CWE
High	8.0	CWE-79

Description

The application reflects user-controlled input into an HTML response without sufficient output encoding. An attacker may be able to inject client-side script into a crafted URL and cause that script to execute in another user's browser context.

Affected Locations

#	Method	Endpoint	Parameter
1	GET	https://app.example.com/process/view_source.php	id
2	GET	https://app.example.com/process/view_source.php	security

Evidence & Validation

Test input: </h2><script>alert(1);</script><h2>

Observed evidence: </h2><script>alert(1);</script><h2>

Technical Evidence: [Instance 1](#) | [Instance 2](#)

Technical Details

CVSS Vector: CVSS:3.1/AV:N/AC:L/PR:L/UI:R/S:U/C:H/I:H/A:H

Remediation

Apply context-appropriate output encoding to all untrusted data before inserting it into HTML, JavaScript, CSS, URLs, or other interpreted contexts. Validate input where appropriate and implement a restrictive Content Security Policy as a defense-in-depth control.

References

<https://owasp.org/www-community/attacks/xss/>

<https://cwe.mitre.org/data/definitions/79.html>

4.2 SQL Injection

Severity	CVSS Score	CWE
Critical	9.8	CWE-89

Description

The application appears to incorporate user-controlled input into a database query without sufficient parameterization or input handling. Testing demonstrated that manipulating the affected parameter changed the application's database-driven response, indicating that SQL syntax may be influencing query execution.

Affected Locations

#	Method	Endpoint	Parameter
1	GET	https://app.example.com/process/xd/	default

Evidence & Validation

Test input: French' AND '1'='1' --

Technical Evidence: [Instance 1](#)

Technical Details

CVSS Vector: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

Remediation

Use parameterized queries or prepared statements for all database operations. Avoid constructing SQL statements through string concatenation. Apply server-side input validation and ensure database accounts operate with the minimum privileges required by the application.

References

4.3 SQL Injection — Database Error Disclosure

Severity	CVSS Score	CWE
High	8.8	CWE-89

Description

The application appears vulnerable to SQL injection through multiple parameters. Manipulated input caused database error messages to be returned in the HTTP response, exposing SQL syntax information and internal application details.

Affected Locations

#	Method	Endpoint	Parameter
1	POST	https://app.example.com/process/auth/change_user_details.php	first_name
2	POST	https://app.example.com/process/auth/change_user_details.php	id
3	POST	https://app.example.com/process/auth/change_user_details.php	surname

Evidence & Validation

Test input: '

Observed evidence: You have an error in your SQL syntax

Technical Evidence: [Instance 1](#) | [Instance 2](#) | [Instance 3](#)

Technical Details

CVSS Vector: CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H

Remediation

Use parameterized queries or prepared statements for all database operations and avoid constructing SQL through string concatenation. Suppress database error details from production responses and return generic application errors instead. Database accounts should also follow least-privilege principles.

References

https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html

5. Remediation Roadmap

The following roadmap provides a practical prioritization framework for addressing the findings identified during the assessment.

Finding	Issue	Priority	Recommended Action
4.1	Reflected Cross-Site Scripting (XSS)	High	Remediate and retest
4.2	SQL Injection	Immediate	Remediate and retest
4.3	SQL Injection — Database Error Disclosure	High	Remediate and retest

6. Retest and Closure

After remediation, a focused retest should be performed to verify that the identified vulnerabilities have been effectively addressed and that remediation has not introduced new security issues.

A finding should be considered closed only after the affected functionality has been retested and the remediation has been confirmed.

7. Severity Definitions

Severity	Description
Critical	Severe security impact requiring immediate attention.
High	Significant security risk that should be addressed promptly.
Medium	Meaningful security weakness requiring remediation.
Low	Limited security impact or lower-risk weakness.
Informational	Observation that does not by itself represent a security vulnerability.

Appendix A — Technical Evidence

This appendix provides concise technical evidence supporting the findings. Request and response data has been reduced to the portions relevant to the observed security condition. Full underlying assessment data is retained separately and is not reproduced here in full.

Finding 4.1: Reflected Cross-Site Scripting (XSS)

Instance 1

Request

```
GET https://app.example.com/process/view_source.php
Parameter: id
Test input: </h2><script>alert(1);</scRipt><h2>
```

Response

Property	Value
Status	200 OK
Content-Type	text/html; charset=utf-8

Relevant response excerpt

```
... s="body_padded">
    <h1>Unknown Vulnerability Source</h1>

    <h2>vulnerabilities/</h2><script>alert(1);</scRipt><h2>/source/impossible.php</h2>
    <div id="code">
        <table width='100%' bgcolor='wh ...
```

Observed evidence: </h2><script>alert(1);</scRipt><h2>

Instance 2

Request

```
GET https://app.example.com/process/view_source.php
Parameter: security
Test input: </h2><script>alert(1);</scRipt><h2>
```

Response

Property	Value
Status	200 OK
Content-Type	text/html; charset=utf-8

Relevant response excerpt

```
... <h1>Open HTTP Redirect Source</h1>
```

```
<h2>vulnerabilities/open_redirect/source/</h2><script>alert(1);</script><h2>.php</h2>
<div id="code">
    <table width='100%' bgcolor='white' style="border ...
```

Observed evidence: </h2><script>alert(1);</script><h2>

Finding 4.2: SQL Injection

Instance 1

Request

```
GET https://app.example.com/process/xd/
Parameter: default
Test input: French' AND '1'='1' --
```

Response

Property	Value
Status	200 OK
Content-Type	text/html; charset=utf-8

Relevant response excerpt

```
... Security Level:</em> impossible<br /><em>Locale:</em> en<br /><em>SQLi DB:</em> mysql</div>
</div>

<div id="footer">

    <p>Test Vulnerable Web Application T ...
```

Validation observation: The page results were successfully manipulated using the boolean conditions [French' AND '1'='1' --] and [French' AND '1'='2' --]

The parameter value being modified was stripped from the HTML output for the purposes of the comparison.

Data was returned for the original parameter.

The vulnerability was detected by successfully restricting the data originally returned, by manipulating the parameter.

Finding 4.3: SQL Injection — Database Error Disclosure

Instance 1

Request

```
POST https://app.example.com/process/auth/change_user_details.php
Parameter: first_name
Test input: '
```

Request body:

```
{"id":1,"first_name":"","surname":"adminjBHMsOSmcvzQvvYuDNTaljbNaUECvZjE"}
```

Response

Property	Value
Status	200 OK
Content-Type	text/html; charset=UTF-8

Relevant response excerpt

```
<br />
<b>Fatal error</b>: Uncaught mysqli_sql_exception: You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right ...
```

Observed evidence: You have an error in your SQL syntax

Instance 2

Request

```
POST https://app.example.com/process/auth/change_user_details.php
Parameter: id
Test input: '

Request body:
{"id":"","first_name":"adminddZkTvstrtwYJSDIzlajyLX","surname":"adminjBHMsOSmoYBWDQfLUuYHEHOJ"}
```

Response

Property	Value
Status	200 OK
Content-Type	text/html; charset=UTF-8

Relevant response excerpt

```
<br />
<b>Fatal error</b>: Uncaught mysqli_sql_exception: You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right ...
```

Observed evidence: You have an error in your SQL syntax

Instance 3

Request

```
POST https://app.example.com/process/auth/change_user_details.php
Parameter: surname
Test input: '

Request body:
{"id":1,"first_name":"adminddZkTvstSabkqtnlkwmPWWtaTscKVTOL","surname":""}
```

Response

Property	Value
Status	200 OK
Content-Type	text/html; charset=UTF-8

Relevant response excerpt

Fatal error: Uncaught mysqli_sql_exception: You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right ...

Observed evidence: You have an error in your SQL syntax