Scrut Automation

Vulnerability Assessment & Penetration Testing

Web Application Final Report

Presented to Elevaite365 Group AB

October 31, 2025

Report By – Riversys Technologies Private Limited (Scrut Automation)

Engagement Overview

Elevaite365 Group AB (Herein referred as Elevaite365) has engaged with Scrut Automation to conduct a penetration test of their Web Application. This report contains all the results of the report as well as all the action items that were included in the penetration test. The purpose of this report is to present the current security level of the external perimeters including gaps, vulnerabilities, and misconfigurations. The findings presented in this report should be fixed to improve the security level of the network systems.

Service Description

Web application Vulnerability Assessment and Penetration Testing (VAPT) is the process of simulating real-world attacks by using the same techniques as malicious hackers. For a security assessment that goes beyond a simple vulnerability scanner, you need experts in the industry. Scrut Automation conducts its penetration test by approaching the scope with both a manual and automatic approach.

Web Application Penetration Test

Our application-level penetration testing consists of both unauthenticated and authenticated testing using both automated and manual methods with particular emphasis placed on identifying vulnerabilities associated with the OWASP Top 10 Most Critical Application Vulnerabilities. It is important to note that a penetration test is not just an automated vulnerability scan, and a large portion of web application penetration testing is a manual process with a skilled engineer attempting to identify, exploit, and evaluate the associated risk of security issues.

Project Objectives

Scrut Automation consultants conduct all testing manually combined with custom and commercial tools that perform unique attack approaches on the network to make sure we cover the whole system in the test. Our expert knowledge and experience are the value we provide in our services.

Document Revision

DETAILS	
Title	Elevaite365- Web application Final Report
Version	1.0
Date	October 31, 2025
Submitted to	Mr. Borhan
Submitted By	Mr. Adithya H

Test Performed Details

Tester Name	Reviewer Name	Test Date	Version
Mr. Adithya H	Mr. Bharat Loiya	October 28, 2025	1.0
Mr. Adithya H	Mr. Bharat Loiya	October 31, 2025	1.1

Table of Content

1. **Executive Summary** 5

1.1 Summary 5

1.2 Approach 5

1.3 Disclaimer 5

1.4 Limitations 6

1.5 OWASPTOP10 6

1.6 Scope 6

1.7 Standards 7

1.8 VulnerabilityScoring 8

1.9 Key Findings 9

2. **Findings** 10

3. **Conclusions** 12

4. **Tools Used** 12

1. Executive Summary

1.1 Summary

This report presents the results of penetration testing conducted by Scrut Automation. The team carried out the penetration testing using automated tools and manual checks on Elevaite365 web application. The assessment was conducted at UAT Environment. The assessment started on October 27, 2025.

The purpose of this assessment was to (i) Test the application to identify technical vulnerabilities and discover whether a malicious user may leverage these flaws to compromise the security of Elevaite365,(ii) Provide recommendations for risk mitigation that may arise on successful exploitation of these vulnerabilities.

The subsequent sections of this document provide statistics of the vulnerabilities identified, severity and proof of findings of Elevaite365. The detailed technical findings section constitutes identified vulnerabilities with recommendations to mitigate security risks associated.

1.2 Approach

- Exploring various application functionalities to enumerate threat & vulnerability in alignment with Open Web Application Security Project (OWASP) Top 10 vulnerabilities.
- Performing information gathering/ fingerprinting to identify software used/ its version, web server details, ports, and services open, etc.
- Performing vulnerability scanning to identify common vulnerabilities in the application layer and by using Burp and various testing tools in the Kali Linux distribution in conjunction with a range of manual analysis. It should be noted that customized payloads and attack vectors were configured in Burp Suite to further enhance the identification of weakness in the application.
- Analysing the automated scan results for any vulnerabilities and ease of exploitability and providing proof of concept where safe exploits are possible.
- Post-Exploitation process will be performed once we get access to the device using identified vulnerabilities/exploits.
- Reporting identified vulnerabilities and recommended solutions to mitigate them; for ease of mitigation activities for application support personnel/ developers' further details of CWEs were added.

1.3 Disclaimer

This report and any supplements are Confidential and may be protected by one or more legal privileges. It is intended solely for the use of the relevant IT personnel in the organization. This report is prepared based on the IT environment that prevailed in the period of assessment. This report is not a guarantee or certification that all vulnerabilities have been discovered and reported in the findings. Subsequent reviews may report on previously unidentified findings or on new vulnerabilities. The sample screen shots should not be treated as the final vulnerabilities.

Gaps which we have identified can also get replicated in any part of the infrastructure. Organization should ensure that the Vulnerability Management Program should be adapted continuously rather than fixing just the issues identified within the areas of Elevaite365.

1.4 Limitations

The Scrut team did not have any limitations for this engagement.

1.5 OWASP TOP 10

- Broken Access Control
- Cryptographic Failures
- Injections
- Insecure Design
- Security Misconfiguration
- Vulnerable and Outdated Components
- Identification and Authentication Failures
- Software and Data Integrity Failures
- Security Logging and Monitoring Failures
- Server-Side Request Forgery

1.6 Scope

The scope included the following IP Addresses / Systems for vulnerability scanning and penetration testing.

Web Application Details:

S No.	Application Name	Application URLs
1	Elevaite365	https://dev.elevaite365.com/

1.7 Standards

OWASP: The OWASP web application security testing methodology and explains how to test for evidence of vulnerabilities within the application due to deficiencies with identified security controls.

The test is divided into 2 phases:

Phase one Passive mode:

In the passive mode, the tester tries to understand the application's logic and walkthrough with the application.

Phase two Active mode:

Information Gathering

Configuration and Deployment Management Testing

Identity Management Testing

Authentication Testing

Authorization Testing

Session Management Testing

Input Validation Testing

Error Handling

Cryptography

Business Logic Testing

Client-Side Testing

1.8 Vulnerability Scoring

The Risk level is divided in four categories:

Severity	DESCRIPTION
Critical	Critical vulnerabilities provide attackers with remote root or administrator capabilities. Malicious users have the ability to compromise the entire host. Easy to detect and exploit and result in large asset damage.
High	Exploitation of the vulnerability discovered on the system can directly lead an attacker to information allowing them to gain privileged access (e.g., administrator or root) to the system. These issues are often difficult to detect and exploit but can result in large asset damage.
Medium	The vulnerability discovered on the system can directly lead to an attacker gaining non-privileged access (e.g., as a standard user) to the system or the vulnerability provides access that can be leveraged within one step to gain administrator-level access. These issues are easy to detect and exploit, but typically result in small asset damage.
Low	The vulnerability discovered on the system provides low-level, but sufficient data to the attacker that may be used to launch a more informed attack against the target environment. In addition, the vulnerability may indirectly lead to an attacker gaining some form of access to the system. These issues can be difficult to detect and exploit and typically result in small asset damage.

1.9 Key Findings

A Medium severity vulnerability, Poor Unhandled Exception, was detected during initial testing OWASP TOP 10 Security misconfiguration. Re-testing determined that this vulnerability is not applicable to production environments.

Definitions:
Open: ***The vulnerability has been identified and documented, but no action has been taken to address it yet. It requires attention and mitigation to reduce the associated risk.***

Fixed: ***The vulnerability has been successfully addressed and the necessary steps have been taken to eliminate or remediate it. Appropriate measures, such as applying patches or updates, have been implemented.***

Closed: ***After thorough assessment, it has been determined that the reported vulnerability is not applicable, is not valid, or does not pose a genuine risk. No further action is required, and the vulnerability is considered closed.***

Business Use Case: ***After thorough assessment, it has been determined that the reported vulnerability is not applicable, as this is required by the client application to work, is flow, or business logic required by the application***

2. Findings

Poor Unhandled Exception

Severity: *Medium*
Re-Test Status: *Closed - Issue cannot be reproduced in production.*
URL: *dev-api.elevaite365.com*

Description: *When an application fails to handle exceptions properly, it can leak sensitive error messages, stack traces, or internal system details. Attackers can exploit these unhandled exceptions to gain insights into application logic, database structure, or even execute attacks like SQL injection or remote code execution.*

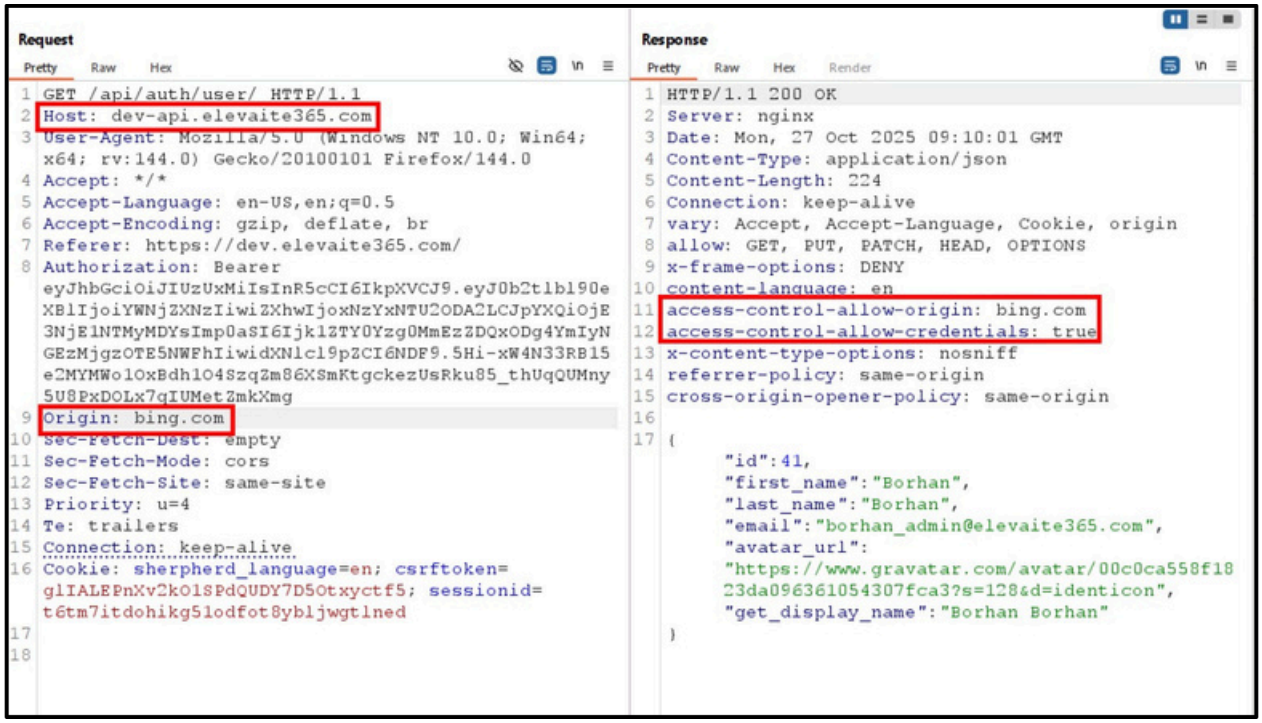
Impact: *Error messages may expose database queries, file paths, API endpoints, or system configurations.*

Recommendations: *Use Generic Error Messages*

References: *https://www.veracode.com/security/error-handling-flaws- information-and-how-fix-tutorial/*

Steps to reproduce: *Observe the error thrown by the application*

Proof of Concept:



Request

Pretty Raw Hex

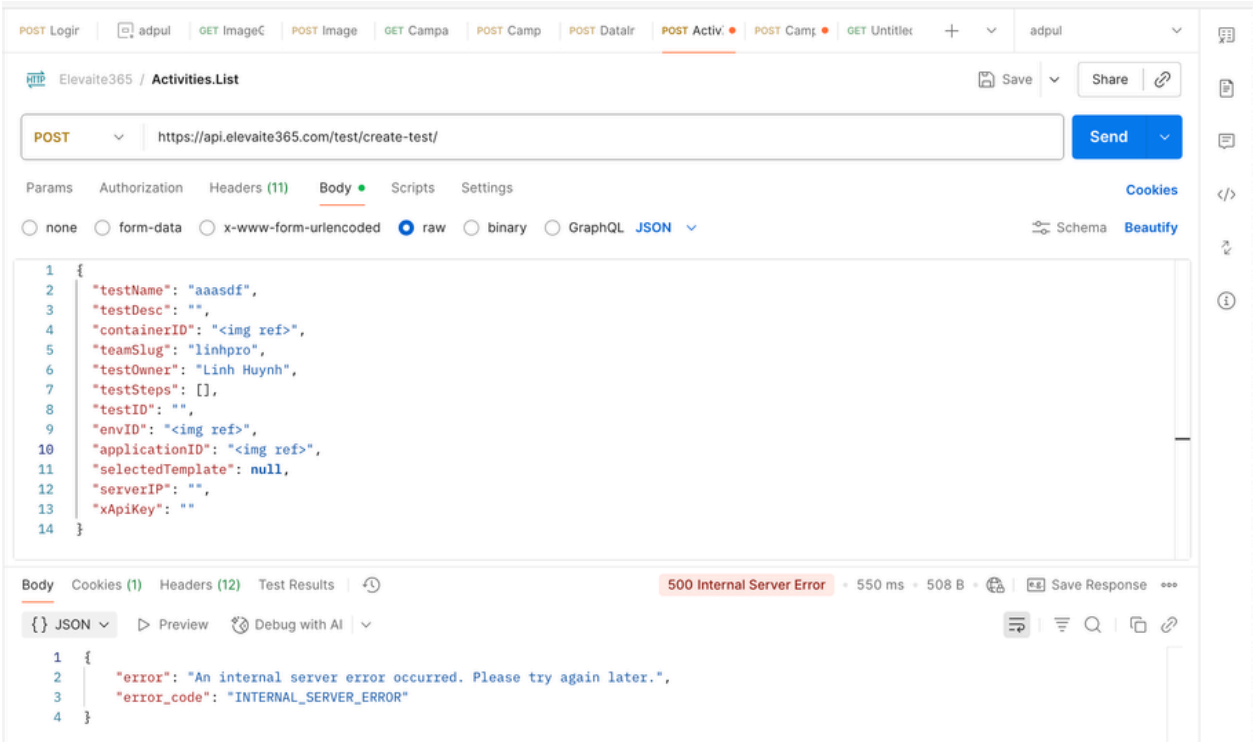
```
1 GET /api/auth/user/ HTTP/1.1
2 Host: dev-api.elevaite365.com
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64;
  x64; rv:144.0) Gecko/20100101 Firefox/144.0
4 Accept: */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Referer: https://dev.elevaite365.com/
8 Authorization: Bearer
  eyJhbGciOiJIUzUxMiIsInR5cCI6IkpXVCJ9.eyJ0b2tlbl90e
  XBlIjoiYWNjZXNzIiwiZXhwIjoxNzYxNTU2ODA2LCJpYXQiOjE
  3NjE1NTMyMDYsImp0aSI6Ijk1ZTY0Yzg0MmEzZDQxODg4YmIyN
  GEzMjgzOTE5NWFhIiwidXNlcl9pZCI6NDF9.5Hi-xW4N33RB15
  e2MYMWo1OxBdhlO4SzqZm86XSmKtgckezUsRku85_thUqQUMny
  5U8PxDOLx7qIUMetZmkXmg
9 Origin: bing.com
10 Sec-Fetch-Dest: empty
11 Sec-Fetch-Mode: cors
12 Sec-Fetch-Site: same-site
13 Priority: u=4
14 Te: trailers
15 Connection: keep-alive
16 Cookie: sherpherd_language=en; csrftoken=
  glIALEPnXv2kO1SPdQUDY7D50txyctf5; sessionid=
  t6tm7itdohikg51odfot8ybljwgtlned
17
18
```

Response

Pretty Raw Hex Render

```
1 HTTP/1.1 200 OK
2 Server: nginx
3 Date: Mon, 27 Oct 2025 09:10:01 GMT
4 Content-Type: application/json
5 Content-Length: 224
6 Connection: keep-alive
7 vary: Accept, Accept-Language, Cookie, origin
8 allow: GET, PUT, PATCH, HEAD, OPTIONS
9 x-frame-options: DENY
10 content-language: en
11 access-control-allow-origin: bing.com
12 access-control-allow-credentials: true
13 x-content-type-options: nosniff
14 referrer-policy: same-origin
15 cross-origin-opener-policy: same-origin
16
17 {
       "id":41,
       "first_name":"Borhan",
       "last_name":"Borhan",
       "email":"borhan_admin@elevaite365.com",
       "avatar_url":
       "https://www.gravatar.com/avatar/00c0ca558f18
       23da096361054307fca3?s=128&d=identicon",
       "get_display_name":"Borhan Borhan"
}
```

Re-Testing Proof of Concept:



POST Logir | adpul | GET ImageC | POST Image | GET Campa | POST Camp | POST Datalr | POST Activ. | POST Camp | GET Untitlec | adpul

Elevaite365 / **Activities.List** — Save | Share

POST https://api.elevaite365.com/test/create-test/ — Send

Params | Authorization | Headers (11) | Body | Scripts | Settings — Cookies

none | form-data | x-www-form-urlencoded | raw | binary | GraphQL JSON — Schema | Beautify

```
1  {
2    "testName": "aaasdf",
3    "testDesc": "",
4    "containerID": "<img ref>",
5    "teamSlug": "linhpro",
6    "testOwner": "Linh Huynh",
7    "testSteps": [],
8    "testID": "",
9    "envID": "<img ref>",
10   "applicationID": "<img ref>",
11   "selectedTemplate": null,
12   "serverIP": "",
13   "xApiKey": ""
14 }
```

Body | Cookies (1) | Headers (12) | Test Results — 500 Internal Server Error | 550 ms | 508 B | Save Response

{} JSON | Preview | Debug with AI

```
1 {
2    "error": "An internal server error occurred. Please try again later.",
3    "error_code": "INTERNAL_SERVER_ERROR"
4 }
```

3. Conclusions

The above Application had a Medium vulnerability. Hence, testing and re-testing has been completed on the Web application of Elevaite365 Group AB. Determination was that the vulnerability was caused by the debug configuration of the dev server, after re-testing it was verified that the vulnerability is not applicable and would not affect a production environment.

4. Tools Used

- *BurpSuiteLicensed*
- *Nessus Licensed*
- *OWASP ZAP Kali Linux – Open*
- *Source trusted*