



AKS IT SERVICES

Web Application Security Audit Report:

SBTET Telangana Portal

Report Release Date	22.01.2025
Type of Audit	Application Security
Type of Audit Report	Initial Audit Report
Period	20.01.2025 – 22.01.2025

Report Prepared By:

AKS Information Technology Services Pvt. Ltd.

www.aksitservices.co.in

E-Mail: info@aksitservices.co.in

NON-DISCLOSURE STATEMENT

This report is the sole property of Elite Soft All information obtained during the assessment is deemed privileged information and not for public dissemination. **AKS Information Technology Services Pvt. Ltd.** pledges its commitment that this information will remain strictly confidential. It will not be discussed or disclosed to any third party without the express written consent of Elite Soft except required by the government regulator (Cert-In) or by the order of the Court.

Document Control

Document Preparation	
Document Title	Web Application Security Audit Report Ver 1.0 SBTET Telangana Portal
Document ID	--
Document Version	1.0
Prepared by	Mr. Sampath Kumar
Reviewed by	Mr. Vinayak Kshirasagar
Approved by	Mr. Vishrant Ojha
Released by	Mr. Sampath Kumar
Release date	22.01.2025

Document Change History		
Version	Date	Remarks / Reason of change
1.0	22.01.2025	Initial release

Document Distribution List			
Name	Organization	Designation	Email Id
Akhil Bejjinki	Elite Soft	Senior Web Developer	bakhil@elitesoft.in
viswanth peddirsi	Elite Soft	Senior Web Developer	viswanthpeddirsi@gmail.com

Table of Contents

Table of Contents.....	4
Introduction	5
Engagement Scope.....	6
Details of the Auditing team.....	7
Audit Activities and Timelines	8
Tools/ Software Used	9
Executive Summary	10
Detailed Findings.....	16
Appendix 'A'	67
Appendix 'B'	72

Introduction

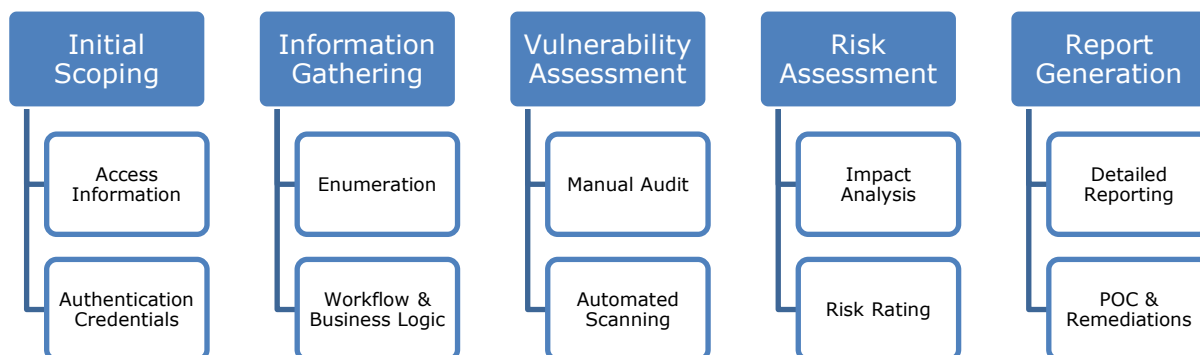
Objectives:

The key objective of this Web Application Security Audit was to identify whether any vulnerabilities exist in the Web Application and to exploit those that can be seen and compromised by malicious users. Additionally, the objective of this activity was to ensure the security of the network and web server from external threats through the Web Application.

Methodology & Standard:

Security Consultants at AKS IT Services Pvt. Ltd. Used the OWASP Web Application Security Testing Methodology for conducting the security audit of the in-scope Web Application.

The OWASP Web Application Methodology is based on the 'grey box' approach. The testing model consists of the following phases:



Standard:

The Open Worldwide Application Security Project (OWASP) standard was used for conducting the final level security audit of the SBTET Telangana Portal web application. The assessment was aimed at identifying the vulnerabilities that are defined in the OWASP Common Weakness Enumeration, and other common global best practices.

[Appendix A](#): Details of the OWASP Top 10:2021 Standard

Engagement Scope

S. No	Asset Description	Criticality of Asset	Internal IP Address	URL	Public IP Address	Location	Hash Value of final audit report	Version
1.	Web application	NA	NA	http://eliteqa.in/index.html#!/index/WebsiteLogin	NA	Remote	NA	NA

Details of the Auditing team

S. No	Name	Designation	Email Id	Professional Qualifications / Certifications	Whether the resource has been listed in the Snapshot information published on CERT-In's website (Yes/No)
1	Mr. Sampath Kumar	Infosec Consultant	Sampath.kumar@aksitservices.co.in	CEH	No
2	Mr. Vinayak kshirasagar	Team Lead – Application Security	vinayak.kshirasagar@aksitservices.co.in	CEH, CAP, eWPTX	Yes
3	Mr. Vishrant Ojha	Assistant Manager – Application Security	vishrant.ojha@aksitservices.co.in	CEH	Yes

Audit Activities and Timelines

Audit Activity	Timelines
Phase I	
Auditor Assigned	09.01.2025
Audit Initiated	20.01.2025
Audit Report Preparation	22.01.2025
Initial Audit Report Published	22.01.2025

Tools/ Software Used

S. No	Name of Tool/Software used	Version of the tool/Software used	Open Source/Licensed
1	Burp Suite Professional	2024.7.3	Commercial
2	Acunetix Vulnerability Scanner	24.1	Commercial
3	GoBuster	3.6.0	Open Source
4	SQLMAP	1.7	Open Source
5	Nmap	7.94	Open Source

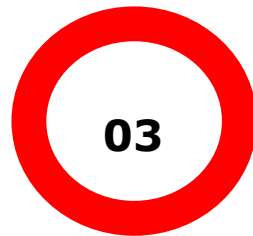
[Appendix C](#): Description of the tools

<Confidential>

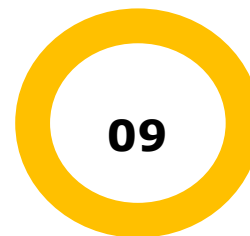
Executive Summary



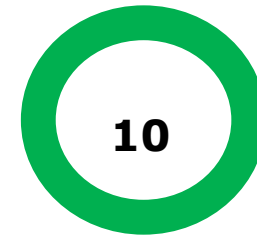
Total



High



Medium



Low

S. No	Affected Asset i.e. IP/URL/Application etc	Observation/Vulnerability title	CVE/CWE	Severity	Recommendation	Reference	New or Repeat or closed observation
1	http://eliteqa.in/index.html#!/index/WebSiteLogin	Response manipulation	CWE-294	High	As implementing encryption of every response is associated with an integrity check, enforcing encryption to secure the confidential data can also engender integrity checks. Companies do not widely use this method but companies must properly consider it as a way to counter data manipulation attacks. The approach of logging activity can also provide security against Response manipulation attacks. IT team must design the internal supervision to verify the information. They need to continually monitor the prioritized logs.	Case I	New
2	http://eliteqa.in/index.html#!/index/WebSiteLogin	Privilege Escalation	CWE-269	High	Create an access control matrix and map each role of the application as per the transactions allowed for each role. The authentication and authorization policies be role based, to minimize the effort required to maintain these policies. The policies should be highly configurable, in order to minimize any hard coded aspects of the policy. The enforcement mechanism(s) should	Case II	New

<Confidential>

					deny all access by default, requiring explicit grants to specific users and roles for access to every page. If the page is involved in a workflow, check to make sure the conditions are in the proper state to allow access.		
3	http://eliteqa.in/index.html#!/index/W ebsiteLogin	Failure to Restrict URL Access	CWE- 644	High	Unless a resource is intended to be publicly accessible, deny access by default. At the code level, make it mandatory for developers to declare the access that is allowed for each resource, and deny access by default..	Case III	New
4	http://eliteqa.in/</>	Improper Error Handling	CWE- 755	Medium	Ensure that a customized error message is shown for any error that has occurred, which gives out very limited information. Disable or limit detailed error handling. In particular, do not display debug information to end users, stack traces, or path information. Application should make secure to prevent revealing of any kind of error and Hardening process should be carried out periodically.	Case IV	New
5	http://eliteqa.in/index.html#!/index/W ebsiteLogin	Improper captcha implemented	CWE- 200	Medium	1. A CAPTCHA should not be used more than once. 2. CAPTCHA should be validated properly. 3. CAPTCHA must be alphanumeric and a minimum of 6 characters long. CAPTCHA Specifications <ul style="list-style-type: none">▪ CAPTCHA should be case-sensitive.▪ CAPTCHA should be image-based.▪ CAPTCHA should be randomly generated from the server and not from client side.▪ After each incorrect user credential, the server should return Login page with a new CAPTCHA	Case V	New

<Confidential>

6	http://eliteqa.in/index.html#!/index/W ebsiteLogin	Improper Session Timeout	CWE- 613	Medium	Application should automatically log out the user and destroy the session after 20 mins of inactivity.	Case VI	New
7	http://eliteqa.in/index.html#!/index/W ebsiteLogin	No Lockout Policy	CWE- 645	Medium	The user should be blocked after a failure of three login attempts. The blocked user can be unblocked by a user with administrative privileges or can be unblocked after certain period of time. Alternatively implement CAPTCHA at login page.	Case VII	New
8	http://eliteqa.in/index.html#!/index/W ebsiteLogin	Vulnerable and Outdated Components	CWE- 937	Medium	Upgrade the components used to the latest version. If upgradation is not possible, download the security patches and hide the version details.	Case VIII	New
9	http://eliteqa.in/index.html#!/index/W ebsiteLogin	Malicious file upload	CWE- 434	Medium	Application should check allowed File extension and File type (MIME Type) in the upload module using white-list filter at server side. File to be uploaded should be restricted to a particular size. Server-side check for not allowing long filename with double extension/double -dot(.)/nullbyte (%00)/meta characters. Assign only Read and Write permissions to the upload folders as required.	Case IX	New
10	http://eliteqa.in/index.html#!/index/W ebsiteLogin	Internal Path Disclosure	CWE- 200	Medium	File Upload path or any internal path of the application should not be visible in the application.	Case X	New
11	http://eliteqa.in/index.html#!/ForgetP assword	OTP Flooding	CWE- 319	Medium	Create logic that will allow the application to send maximum 3-5 OTP per mobile number post that block the mobile number for a particular time frame.	Case XI	New

<Confidential>

12	http://eliteqa.in/index.html#!/index/WebsiteLogin	Username Enumeration	CWE-204	Low	<p>Login Make sure to return a generic "No such username or password" message when a login failure occurs. Make sure the HTTP response, and the time taken to respond are no different when a username does not exist, and an incorrect password is entered.</p> <p>Password Reset Make sure your "forgotten password" page does not reveal usernames. If your password reset process involves sending an email, have the user enter their email address. Then send an email with a password reset link if the account exists.</p> <p>Registration Avoid having your site tell people that a supplied username is already taken. If your usernames are email addresses, send a password reset email if a user tries to sign-up with an existing address. If usernames are not email addresses, protect your sign-up page with a CAPTCHA. Profile Pages If your users have profile pages, make sure they are only visible to other users who are already logged in. If you hide a profile page, ensure a hidden profile is indistinguishable from a non-existent profile.</p>	Case XII	New
13	http://eliteqa.in/index.html#!/index/WebsiteLogin	Sensitive File Disclosure	CWE-200	Low	Access to sensitive files should disabled by default. Alternatively remove the files if not required.	Case XIII	New

<Confidential>

14	http://eliteqa.in/index.html#!/index/WebsiteLogin	Email Harvesting	CWE-200	Low	The application should properly customize the email addresses while posting on the website as: Email addresses should be posted as an image not as a hyperlink. Alternatively, instead of @symbol, [at] should be used. Similarly, the dot character (.) should be replaced by [dot]. So abc@nic.in should be written as abc[at]nic[dot]in.	Case XIV	New
15	http://eliteqa.in/index.html#!/index/WebsiteLogin	Debug Method Enabled	CWE-11	Low	To disable debugging, open the Web.config file for the application, and find the <compilation> element within the <system.web> section. Set the debug attribute to "false". Note that it is also possible to enable debugging for all applications within the Machine.config file. You should confirm that the debug attribute in the <compilation> element has not been set to "true" within the Machine.config file. It is strongly recommended that you refer to your platform's documentation relating to this issue, and do not rely solely on the above remediation.	Case XV	New
16	http://eliteqa.in/index.html#!/index/WebsiteLogin	Lack of security Headers	CWE-644	Low	Implement security headers such as X-XSS-Protection, Content-Security-Policy, Referrer Policy, X-Content-Type-Options, Permission Policy and Strict-transport-layer-protection	Case XVI	New
17	http://eliteqa.in/index.html#!/index/WebsiteLogin	Clickjacking	CWE-1021	Low	Preventing the browser from loading the page in frame using the X-Frame-Options or Content Security Policy (frame-ancestors) HTTP headers. Preventing session cookies from being included when the page is loaded in a frame using the SameSite cookie attribute. Implementing JavaScript code in the page to attempt to prevent it being loaded in a frame (known as a "frame-buster").	Case XVII	New

<Confidential>

18	http://eliteqa.in/index.html#!/index/WebsiteLogin	Version Disclouser	CWE-200	Low	Hide version details in the response	Case XVIII	New
19	http://eliteqa.in/index.html#!/index/WebsiteLogin	Aadhaar Number in plain text	CWE-319	Low	As per the Aadhaar Act and Regulations by UIDAI, Aadhaar number must not be stored in the database/files without the consent of the user. If the application requires to store Aadhaar number in permanent storage, Aadhaar seeding process must be followed. Permanent storages for further delivery of services, Aadhaar seeding process must be followed. Aadhaar number should be used as per the Aadhaar Act and Regulations.	Case XIX	New
20	http://eliteqa.in/index.html#!/index/WebsiteLogin	Application accepting same password	CWE-620	Low	Implement proper validation for disallowing the user to create same password.	Case XX	New
21	http://eliteqa.in/index.html#!/index/WebsiteLogin	Autocomplete Enabled on Password Field	CWE-200	Low	To prevent browsers from storing credentials entered into HTML forms, include the attribute autocomplete="off" within the FORM tag (to protect all form fields) or within the relevant INPUT tags (to protect specific individual fields).	Case XXI	New
22	http://eliteqa.in/index.html#!/index/WebsiteLogin	Security Logging and Monitoring Failures	CWE-778	Low	Information to be logged includes the following: IP address of the originating Source, Date, Time, Username (No Password), session details, Referrer, Process id, URL, User Agent, Countries if any in addition to other details to be logged in the website. Logging of Authentication Process which includes number of successful and failed login attempts. To create audit logs, use auto numbering so that every logged entry has an un-editable log number. Then if one audit entry is deleted a gap in the numbering sequence will appear. Report of the website logs to be generated weekly by the administrator to keep track of the website activities.	Case XXII	New

Detailed Findings

Case I

i. Affected Asset i.e. IP/URL/Application etc.

http://eliteqa.in/index.html#!/index/WebsiteLogin

ii. Observation/ Vulnerability title

Response Manipulation

iii. Detailed observation / Vulnerable point

Response manipulation attacks where an adversary does not take the data, but instead make subtle, stealthy tweaks to data for some type of gain, can be just as crippling for organizations compared to theft.

iv. CVE/CWE

CWE-294

v. Control Objective

NA

vi. Control Name

NA

vii. Audit Requirement

NA

viii. Severity

High

ix. Recommendation

As implementing encryption of every response is associated with an integrity check, enforcing encryption to secure the confidential data can also engender integrity checks. Companies do not widely use this method but companies must properly consider it as a way to counter data manipulation attacks.

The approach of logging activity can also provide security against Response manipulation attacks. IT team must design the internal

supervision to verify the information. They need to continually monitor the prioritized logs.

x. Reference

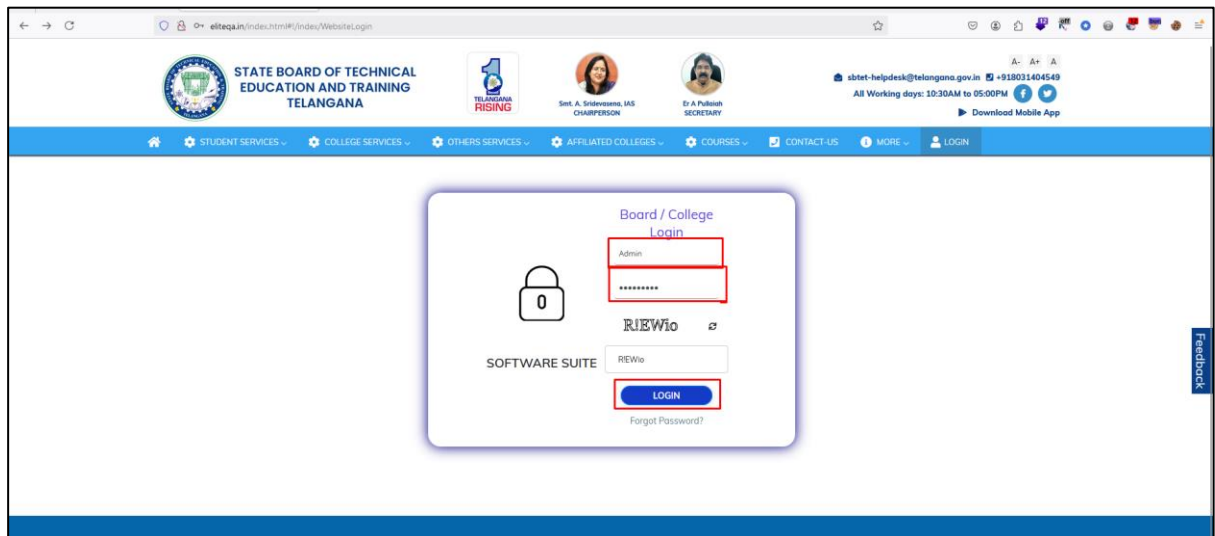
Case I

xi. New or Repeat observation

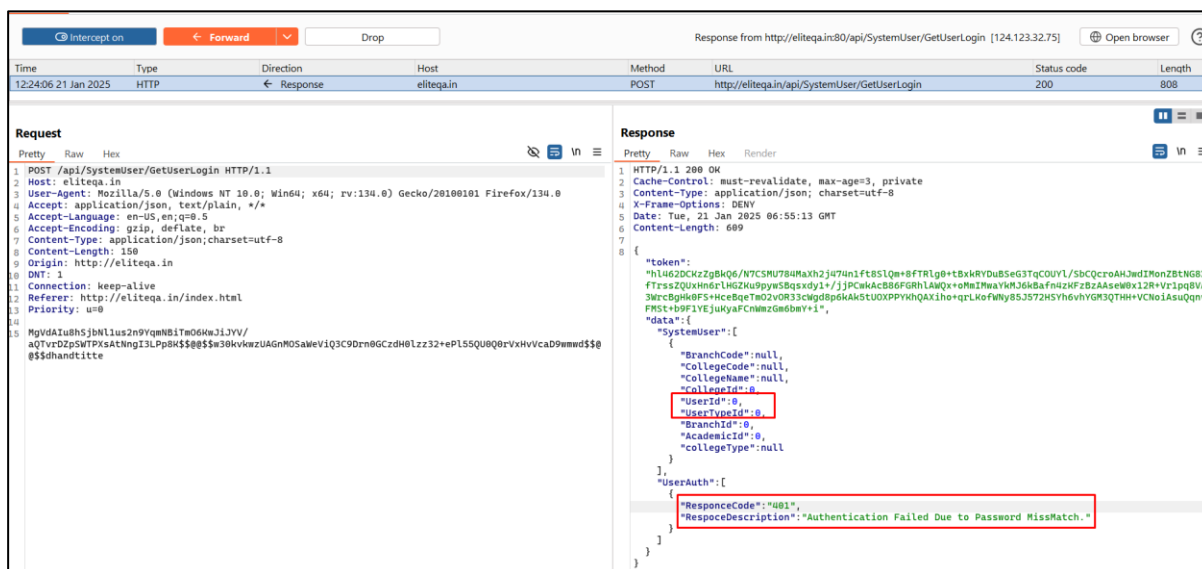
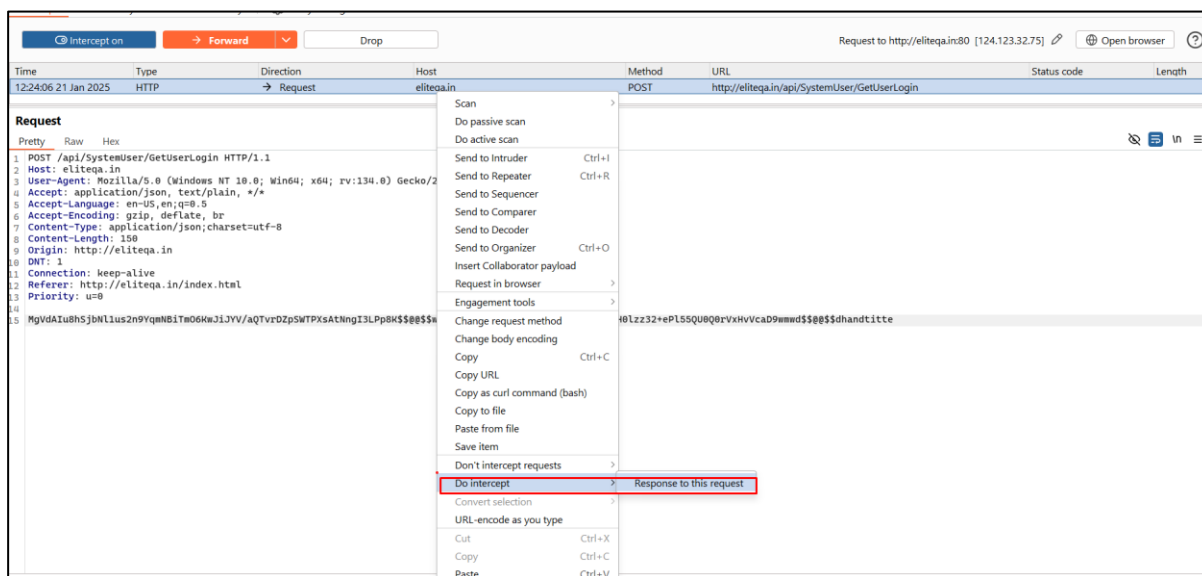
New

xii. References to evidence / Proof of Concept

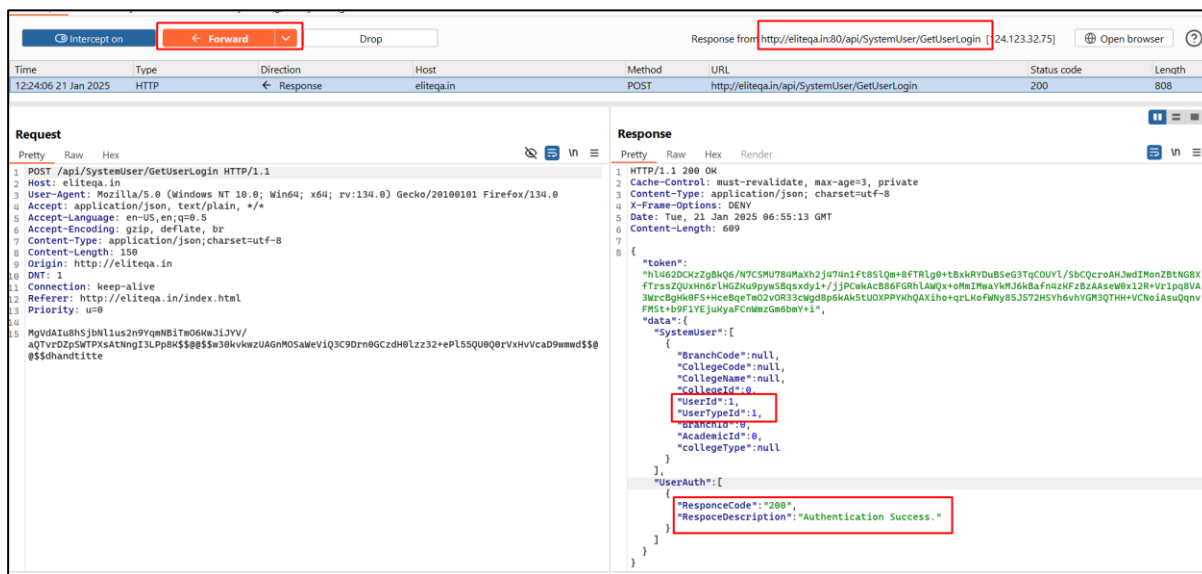
Step I: Go to login page and give the valid username and invalid Password and click on the login and Intercept the request as shown in the snapshot below:



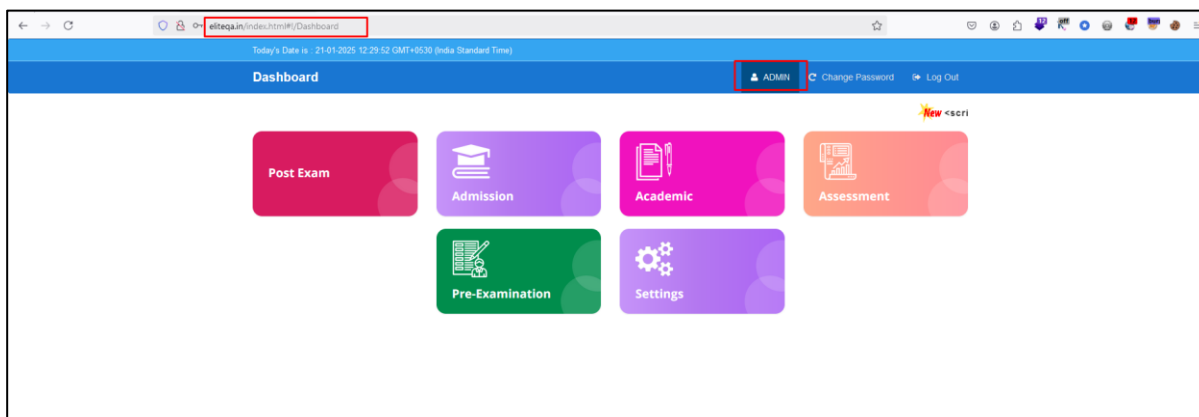
Step II: Here, Open the Options and Click on the Do intercept in response to this request and Forward the request As shown in the snapshot Below



Step III: Observe the response, Change the values from **UserId=1**, **UsertypeId=1** and Response Code:**200** and **Authentication Success**, in Response, forward the request as shown in the snapshot Below



Step IV: we observed that the password was bypassed through response manipulation, and the application is Logged in successfully, as shown in the snapshot below



Note: Kindly patch this vulnerability throughout the application.

Case II

i. Affected Asset i.e. IP/URL/Application etc.

https:// http://eliteqa.in/index.html#!/index/WebsiteLogin

ii. Observation/ Vulnerability title

Privilege Escalation

iii. Detailed observation / Vulnerable point

Frequently, an application only protects sensitive functionality by preventing the display of links or URLs to unauthorized users. Attackers can use this weakness to access and perform unauthorized operations by accessing those URLs directly.

iv. CVE/CWE

CWE-269

v. Control Objective

NA

vi. Control Name

NA

vii. Audit Requirement

NA

viii. Severity

High

ix. Recommendation

Create an access control matrix and map each role of the application as per the transactions allowed for each role.

The authentication and authorization policies be role based, to minimize the effort required to maintain these policies.

The policies should be highly configurable, in order to minimize any hard coded aspects of the policy.

The enforcement mechanism(s) should deny all access by default, requiring explicit grants to specific users and roles for access to every page.

If the page is involved in a workflow, check to make sure the conditions are in the proper state to allow access.

x. Reference

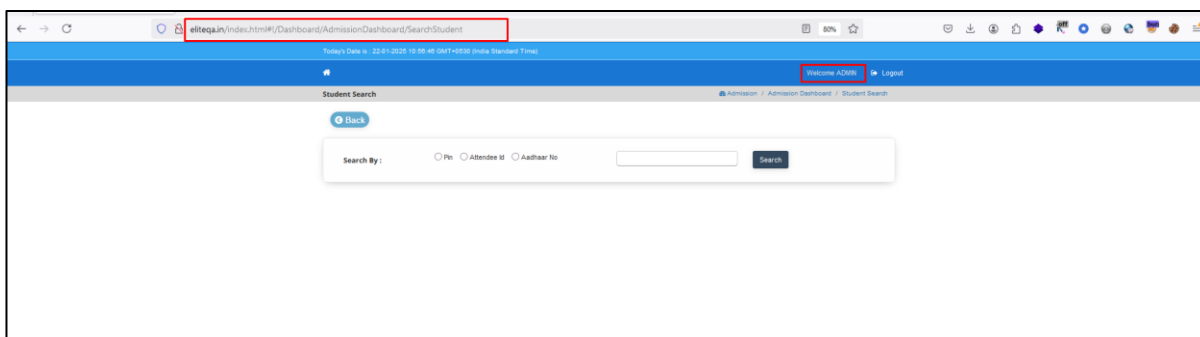
Case II

xi. New or Repeat observation

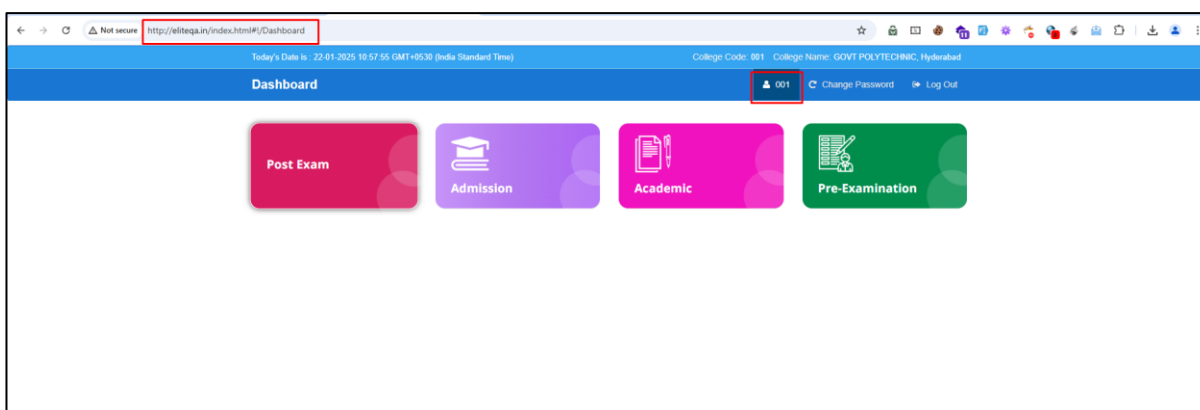
New

xii. References to evidence / Proof of Concept

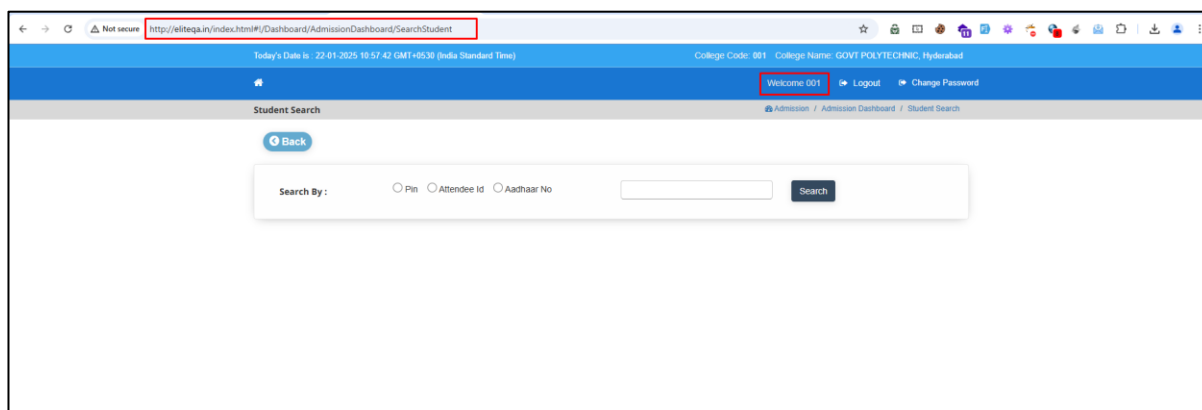
Step I: Login with **Admin** and Go to **Student Search** and Copy the **Url** as shown as snapshot below:



Step II: Then Login with **College User(001)** in **different browser** as shown as snapshot below:



Step III: After that, paste the previously copied Url in College user(001) browser and Here We observed that Normal user as able to access the high Privilege user functionalities as shown as snapshot below:



NOTE: Kindly implement the patch throughout the application

Case III

i. Affected Asset i.e. IP/URL/Application etc.

<http://eliteqa.in/index.html#!/index/WebsiteLogin>

ii. Observation/ Vulnerability title

Failure to Restrict URL Access

iii. Detailed observation / Vulnerable point

Frequently, an application only protects sensitive functionality by preventing the display of links or URLs to unauthorized users. Attackers can use this weakness to access and perform unauthorized operations by accessing those URLs directly.

iv. CVE/CWE

CWE-425

v. Control Objective #

NA

vi. Control Name #

NA

vii. Audit Requirement #

NA

viii. Severity

High

ix. Recommendation

Unless a resource is intended to be publicly accessible, deny access by default.

At the code level, make it mandatory for developers to declare the access that is allowed for each resource, and deny access by default.

x. Reference

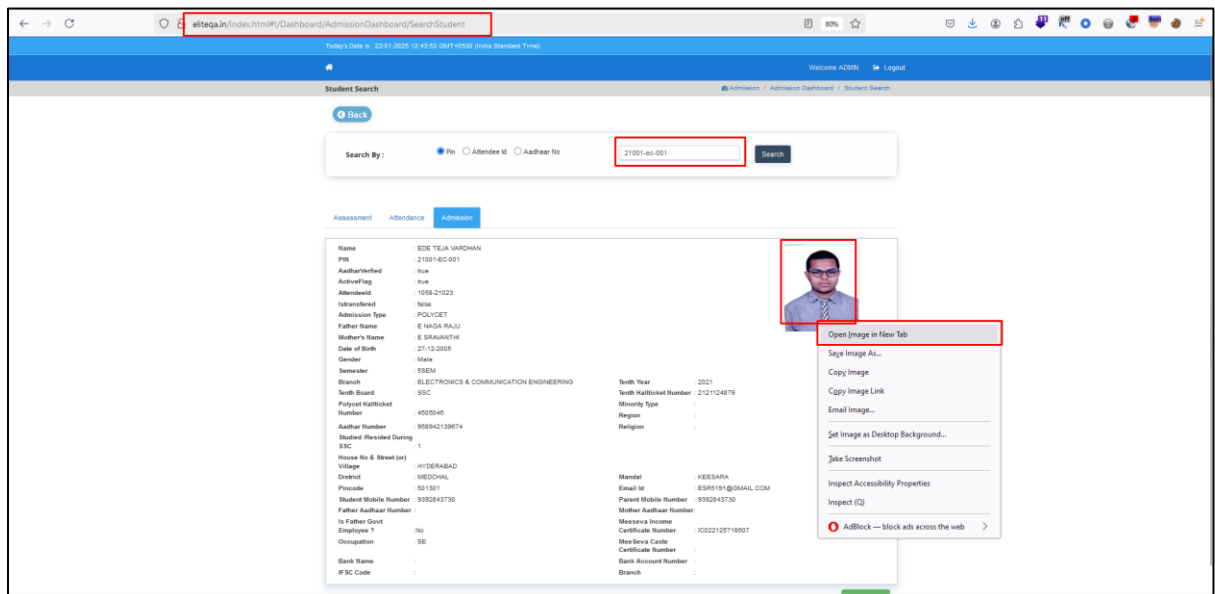
Case III

xi. New or Repeat observation

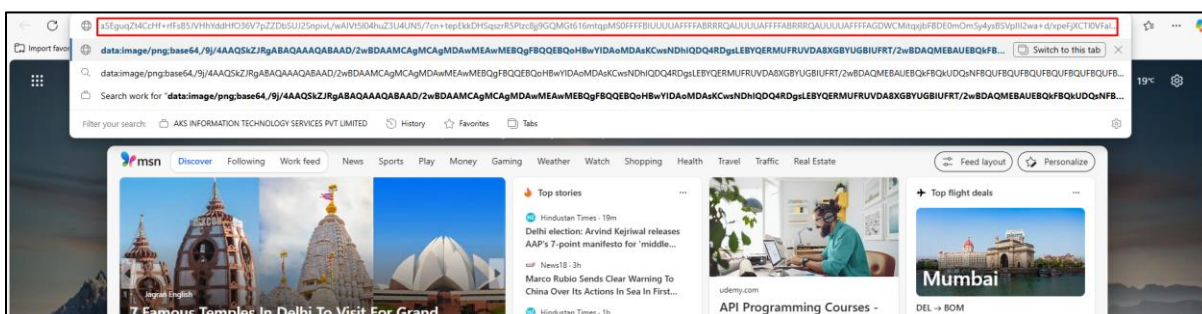
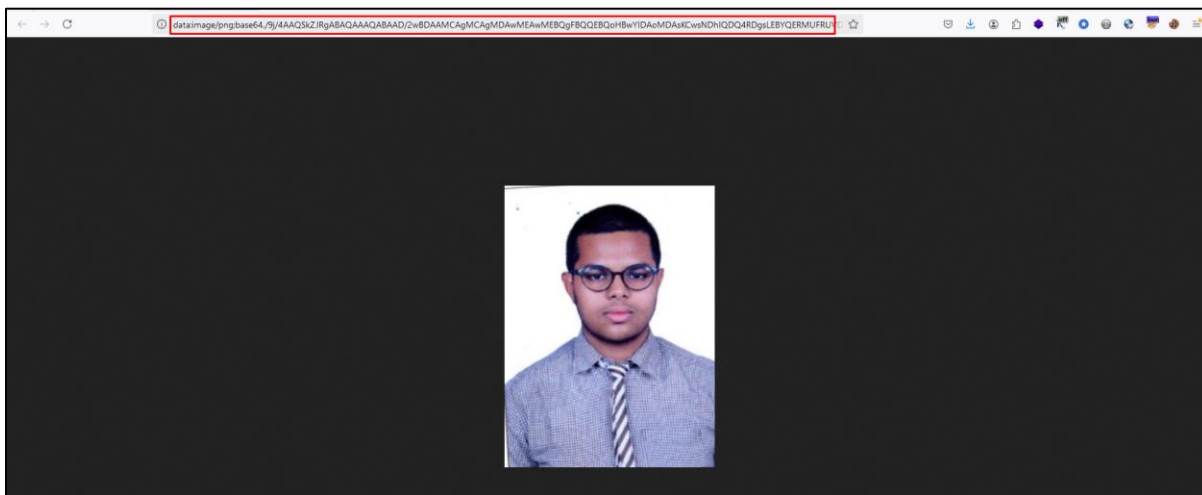
New

xii. References to evidence / Proof of Concept

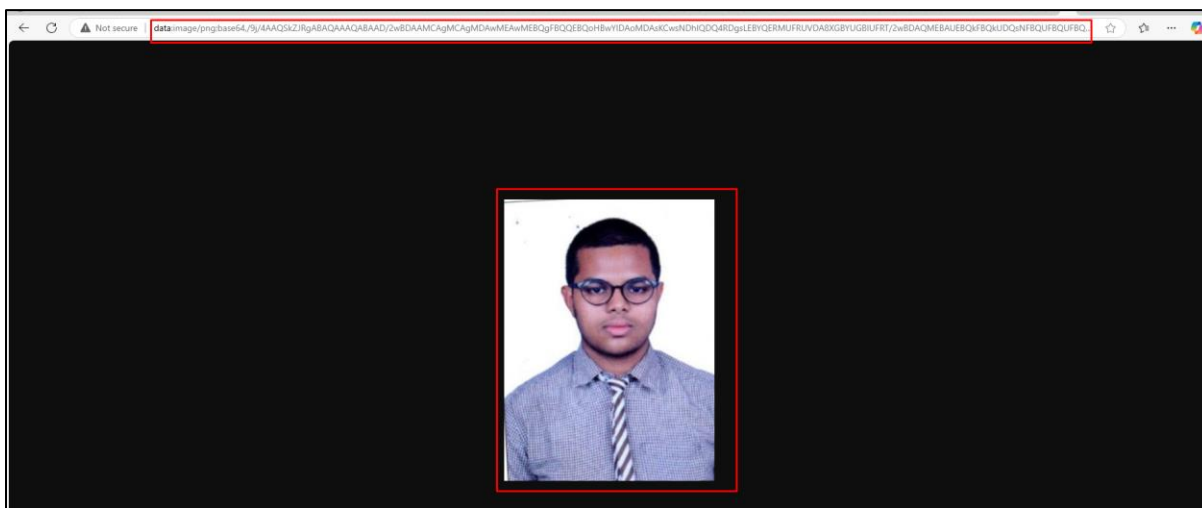
Step 1: Login with Admin and Go Student Search and Give the Details of student and open the student image in new tab As shown in the snapshot:



Step III: Now Copy the URL and paste it another browser where the user is not logged in as shown in the snapshot below:



Step IV: On observation we found that the URL is accessible without authentication as shown in the snapshot below:



NOTE: Kindly Patch this vulnerability Throughout the Application.

Case IV

i. Affected Asset i.e. IP/URL/Application etc.

http://eliteqa.in/</>

i. Observation/ Vulnerability title

Improper Error Handling

ii. Detailed observation / Vulnerable point

Applications can unintentionally leak information about their configuration, internal workings, or violate privacy through a variety of application problems. Attackers use this weakness to steal sensitive data or conduct more serious attacks.

iii. CVE/CWE

CWE-755

iv. Control Objective #

NA

v. Control Name #

NA

vi. Audit Requirement #

NA

vii. Severity

Medium

viii. Recommendation

Ensure that a customized error message is shown for any error that has occurred, which gives out very limited information.

Disable or limit detailed error handling. In particular, do not display debug information to end users, stack traces, or path information.

Application should make secure to prevent revealing of any kind of error and Hardening process should be carried out periodically.

ix. Reference

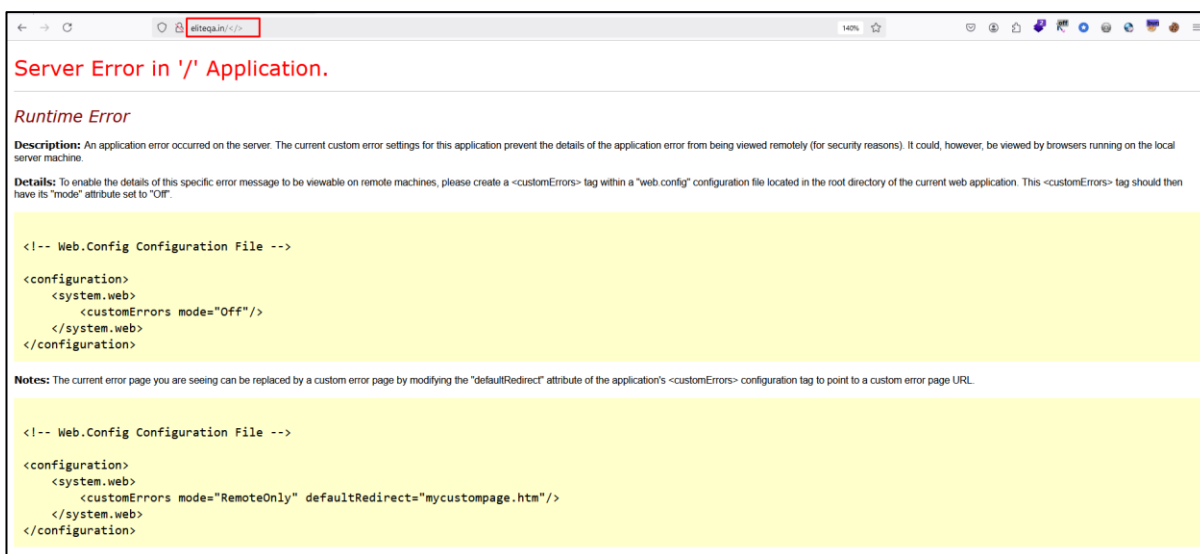
Case IV

x. New or Repeat observation

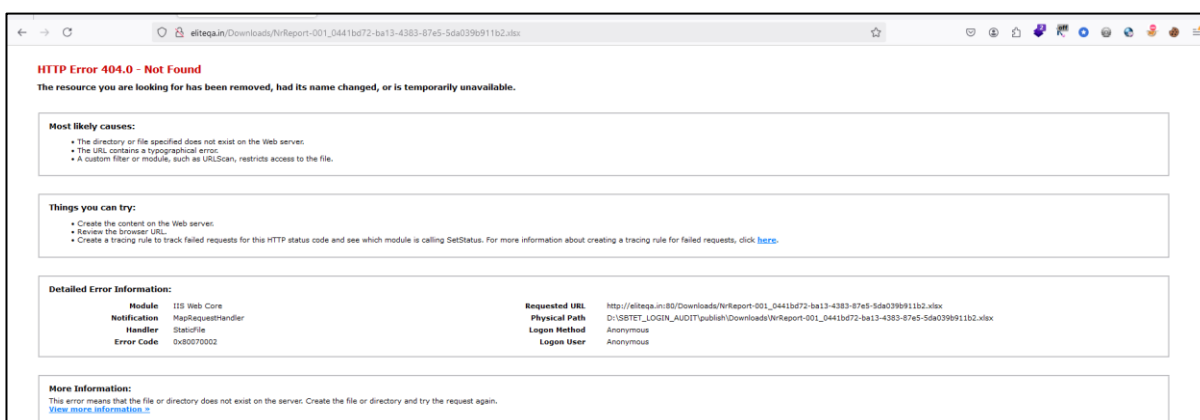
New

xi. References to evidence / Proof of Concept

Step I: Go to URL: "<http://eliteqa.in/>",here We observed that application throws the error as shown in the snapshot below



Step II: Go to URL: "http://eliteqa.in/Downloads/NrReport-001_0441bd72-ba13-4383-87e5-5da039b911b2.xlsx",here We observed that application throws the error as shown in the snapshot below



NOTE: Kindly Patch this vulnerability Throughout the Application.

Case V

i. Affected Asset i.e. IP/URL/Application etc.

<http://eliteqa.in/index.html#!/index/WebsiteLogin>

ii. Observation/ Vulnerability title

Improper captcha implemented

iii. Detailed observation / Vulnerable point

The CAPTCHA images are generated using a simple algorithm that produces a predictable pattern. The images are 6 characters long and consist of a combination of uppercase and lowercase letters, as well as numbers. The characters are displayed in a straightforward manner, with no distortion or rotation

iv. CVE/CWE

CWE-200

v. Control Objective

NA

vi. Control Name

NA

vii. Audit Requirement

NA

viii. Severity

Medium

ix. Recommendation

1. A CAPTCHA should not be used more than once.
2. CAPTCHA should be validated properly.
3. CAPTCHA must be alphanumeric and a minimum of 6 characters long.

CAPTCHA Specifications:

- CAPTCHA should be of 6 characters alphanumeric in length.
- CAPTCHA should be case-sensitive.
- CAPTCHA should be image-based.

- CAPTCHA should be randomly generated from the server and not from client side.
- After each incorrect user credential, the server should return Login page with a new CAPTCHA

x. Reference

Case V

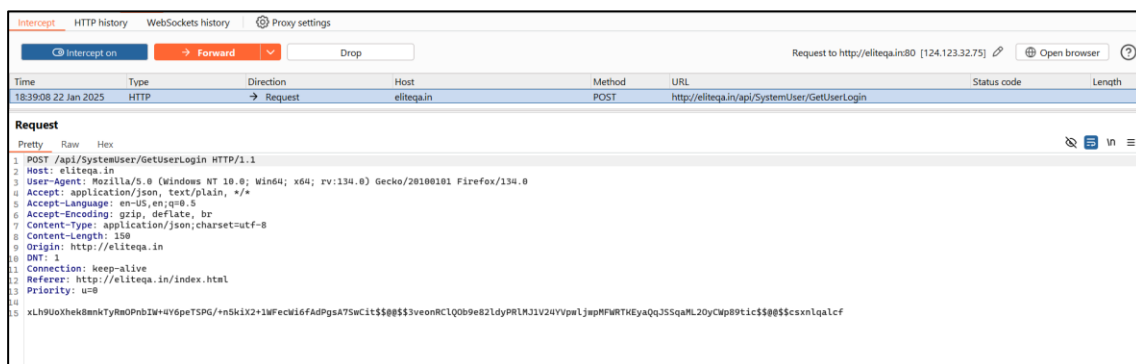
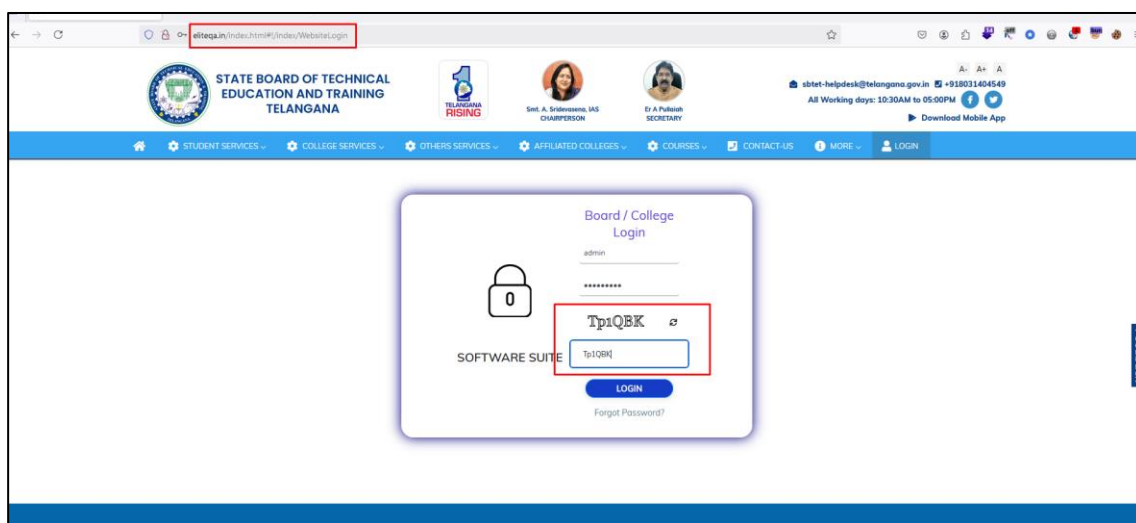
xi. New or Repeat observation

New

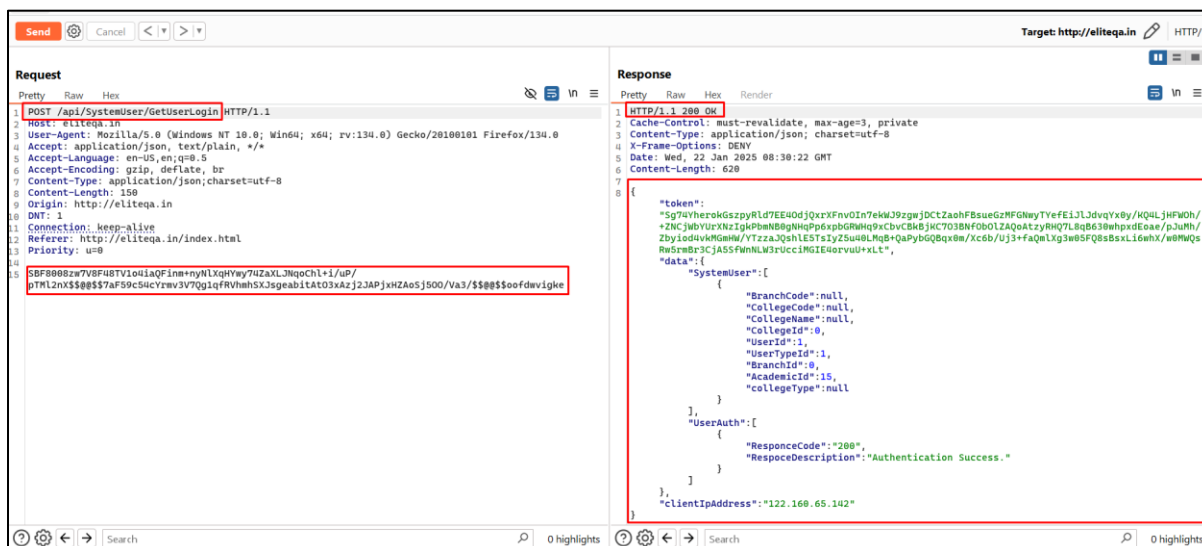
xii. References to evidence / Proof of Concept:

Step I: Open the URL

"**http://eliteqa.in/index.html#!/index/WebsiteLogin**" and login and intercept the request and Send to Repeater as shown in the below snapshot:



Step II: Send the request twice and Here, We observed that Captcha is Not Validating the server side as shown in the snapshot Below



Note: Kindly Validate the captcha in server side also

Case VI

i. Affected Asset i.e. IP/URL/Application etc.

http://eliteqa.in/index.html#!/index/WebsiteLogin

ii. Observation/ Vulnerability title

Improper Session Timeout

iii. Detailed observation / Vulnerable point

The application does not automatically logs out the user after 20 mins of inactivity

iv. CVE/CWE

CWE-613

v. Control Objective #

NA

vi. Control Name #

NA

vii. Audit Requirement #

NA

viii. Severity

Medium

ix. Recommendation

Application should automatically log out the user and destroy the session after 20 mins of inactivity.

x. Reference

Case VI

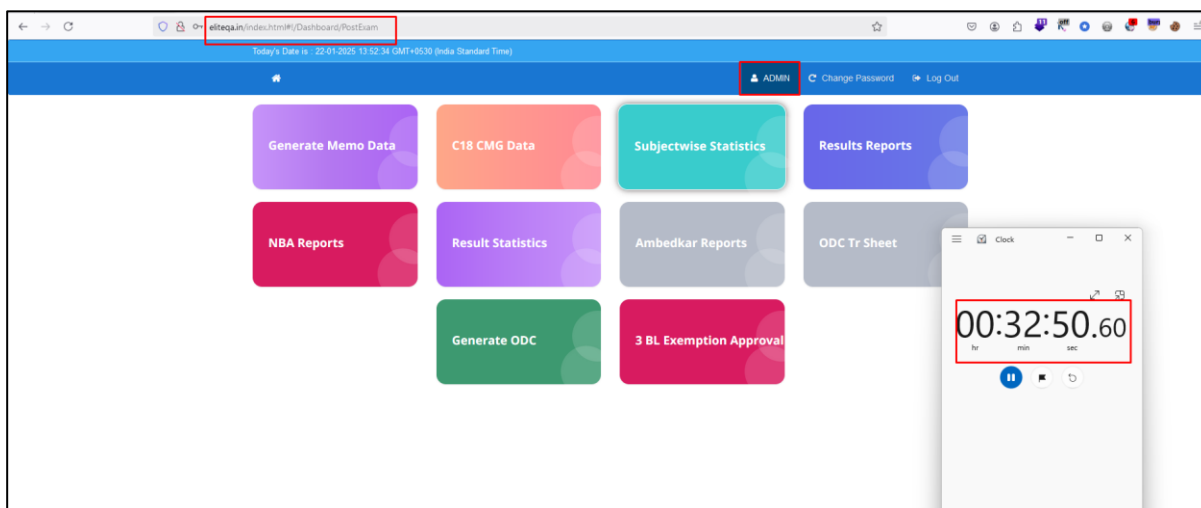
xi. New or Repeat observation

New

xii. References to evidence / Proof of Concept

Step I: Open the URL

"http://eliteqa.in/index.html#!/index/WebsiteLogin' and login Here we observed that session is not logout after 20 minutes as shown in the below snapshot.



Kindly Patch this vulnerability Throughout the Application

Case VII

i. Affected Asset i.e. IP/URL/Application etc.

http://eliteqa.in/index.html#!/index/WebsiteLogin

ii. Observation/ Vulnerability title

No Lockout Policy

iii. Detailed observation / Vulnerable point

Weak password allow in the application can make attacker to guess the password and brute force for the password if username is known, if no account lockout policy is implemented in the application to block user after particular invalid attempts.

iv. CVE/CWE

CWE- 645

v. Control Objective

NA

vi. Control Name

NA

vii. Audit Requirement

NA

viii. Severity

Medium

ix. Recommendation

The user should be blocked after a failure of three login attempts. The blocked user can be unblocked by a user with administrative privileges or can be unblocked after certain period of time.
Alternatively implement CAPTCHA at login page.

x. Reference

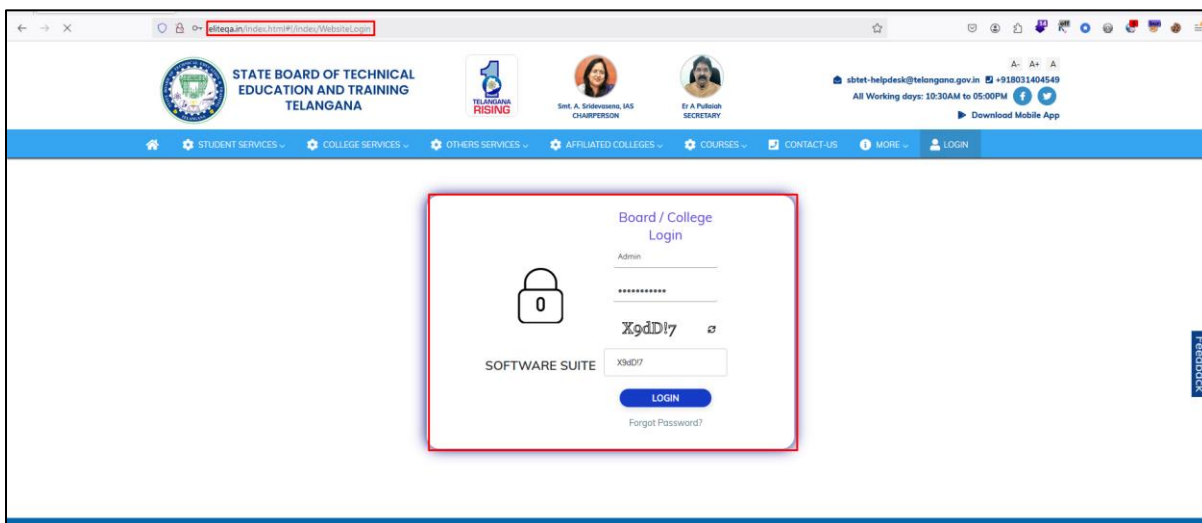
Case VII

xi. New or Repeat observation

New

xii. References to evidence / Proof of Concept

Step I: We observed that after a few invalid attempts, the accounts are not blocked as shown in the snapshot below.



Kindly Patch this vulnerability Throughout the Application

Case VIII

i. Affected Asset i.e. IP/URL/Application etc.

http://eliteqa.in/index.html#!/index/WebsiteLogin

ii. Observation/ Vulnerability title

Vulnerable and Outdated Components

iii. Detailed observation / Vulnerable point

Vulnerable components, such as libraries, frameworks, and other software modules almost always run with full privilege. So, if exploited, they can cause serious data loss or server takeover. Applications using these vulnerable components may undermine their defenses and enable a range of possible attacks and impacts

iv. CVE/CWE

CWE- 937

v. Control Objective

NA

vi. Control Name

NA

vii. Audit Requirement

NA

viii. Severity

Medium

ix. Recommendation

Upgrade the components used to the latest version.
If upgradation is not possible, download the security patches and hide the version details.

x. Reference

Case VIII

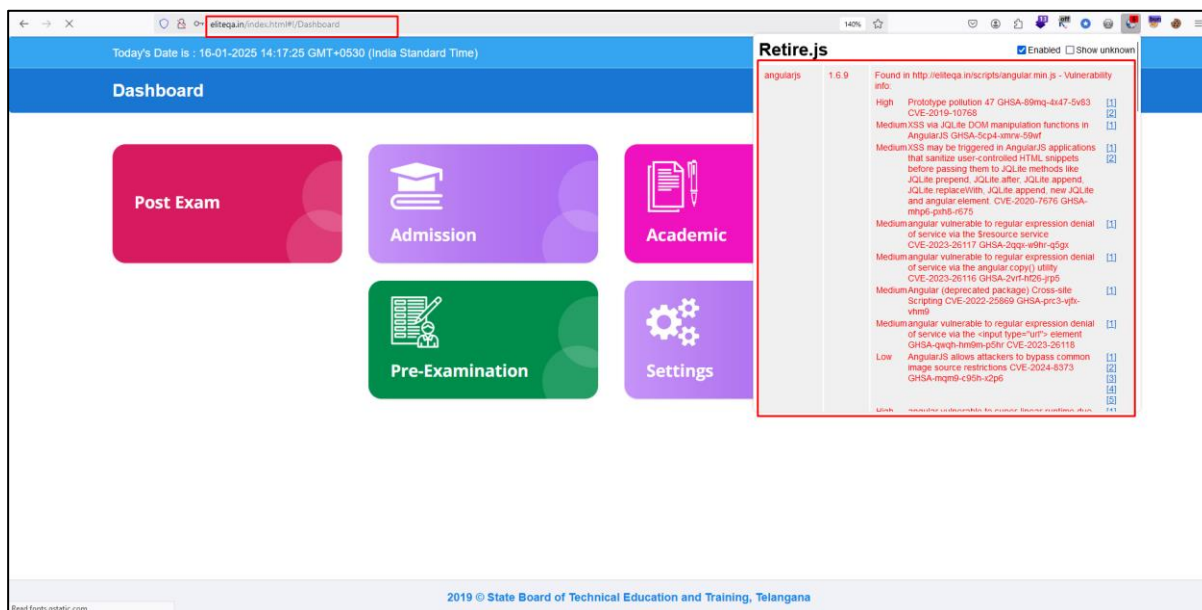
xi. New or Repeat observation

New

xii. References to evidence / Proof of Concept

Step I: Open the url

"<http://eliteqa.in/index.html#!/index/WebsiteLogin>" using retire.js extension we observed that application is using vulnerable and outdated components as shown in the snapshot below:



Kindly Patch this vulnerability Throughout the Application

Case IX

i. Affected Asset i.e. IP/URL/Application etc.

<http://eliteqa.in/index.html#!/index/WebsiteLogin>

ii. Observation/ Vulnerability title

Malicious file upload

iii. Detailed observation / Vulnerable point

Code vulnerable to remote file inclusion (RFI) allows attackers to include hostile code and data, resulting in devastating attacks, such as total server compromise. Malicious file execution attacks affect PHP, XML and any framework which accepts filenames or files from users.

iv. CVE/CWE

CWE-434

v. Control Objective

NA

vi. Control Name

NA

vii. Audit Requirement

NA

viii. Severity

Medium

ix. Recommendation

Application should check allowed File extension and File type (MIME Type) in the upload module using white-list filter at server side.

File to be uploaded should be restricted to a particular size.

Server-side check for not allowing long filename with double extension/double

-dot(./)/nullbyte (%00)/meta characters.

Assign only Read and Write permissions to the upload folders as required.

Reference

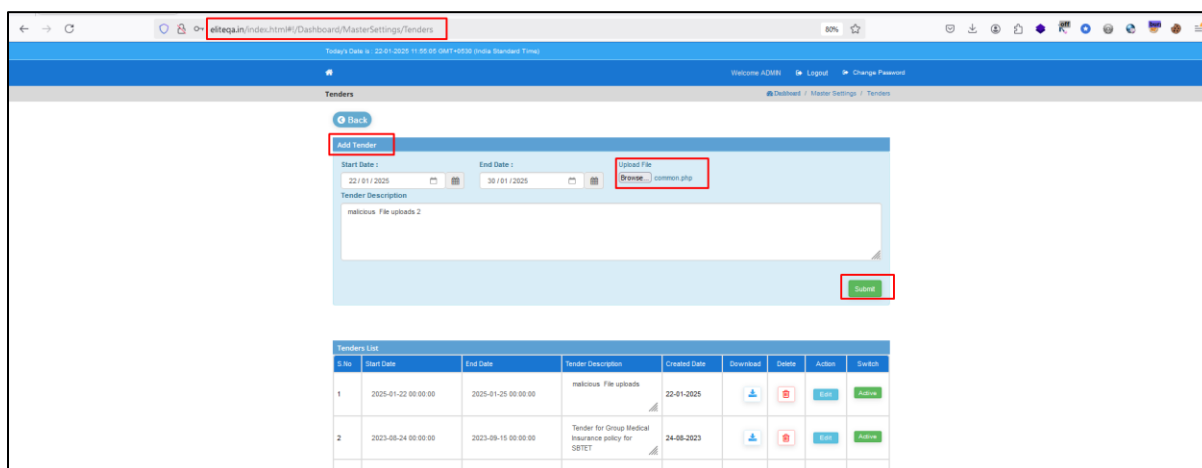
Case IX

x. New or Repeat observation

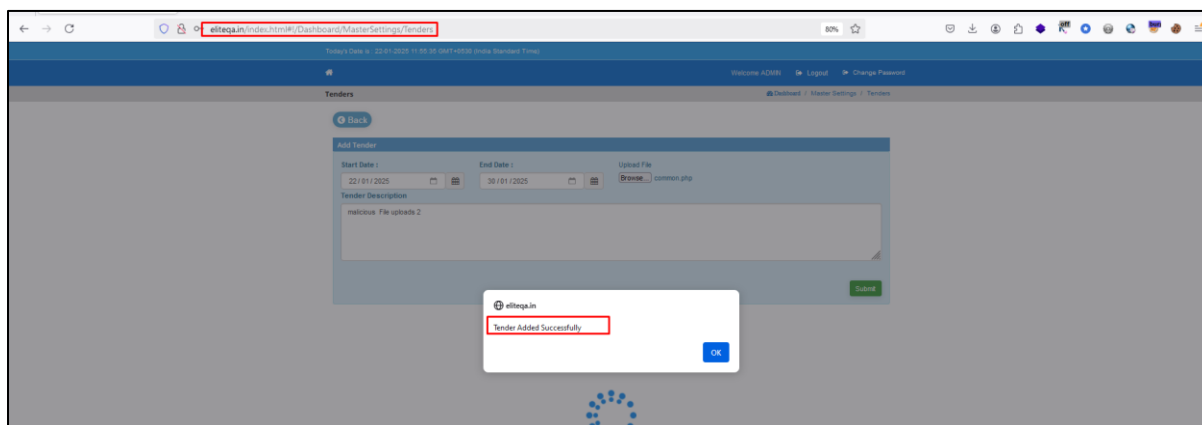
New

xi. References to evidence / Proof of Concept

Step I: Login with **admin** and go to **Setting > Tender** and Here fill the Details and upload a php File as shown in the snapshot below:



Step II: File got uploaded, Here We Observed that application accepting the malicious file as shown in the snapshot Below:



Kindly Patch this vulnerability Throughout the Application

Case X

i. Affected Asset i.e. IP/URL/Application etc.

http://eliteqa.in/index.html#!/index/WebsiteLogin

ii. Observation/ Vulnerability title

Internal Path Disclosure

iii. Detailed observation / Vulnerable point

Path Disclosure vulnerabilities enable the attacker to see the path to the webroot/file. e.g.: /home/omg/htdocs/file/. Certain vulnerabilities, such as using the load file() (within a SQL Injection) query to view the page source, require the attacker to have the full path to the file they wish to view.

iv. CVE/CWE

CWE- 613

v. Control Objective

NA

vi. Control Name

NA

vii. Audit Requirement

NA

viii. Severity

Medium

ix. Recommendation

File Upload path or any internal path of the application should not be visible in the application.

x. Reference

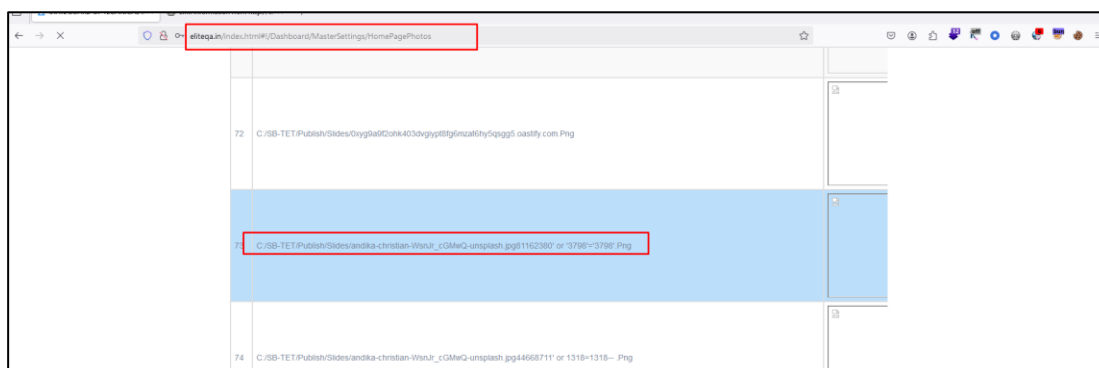
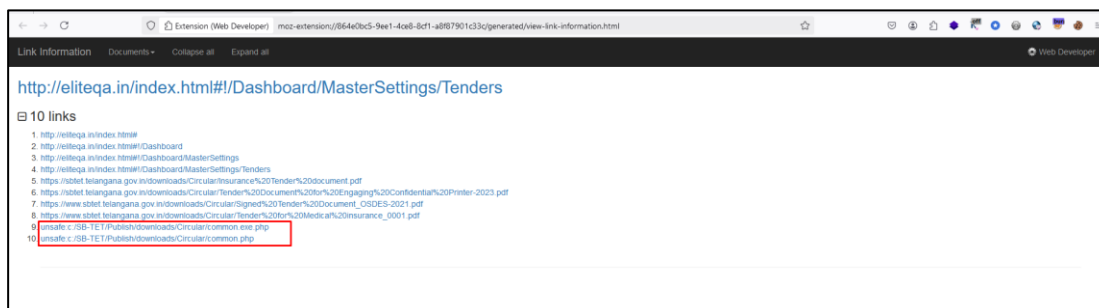
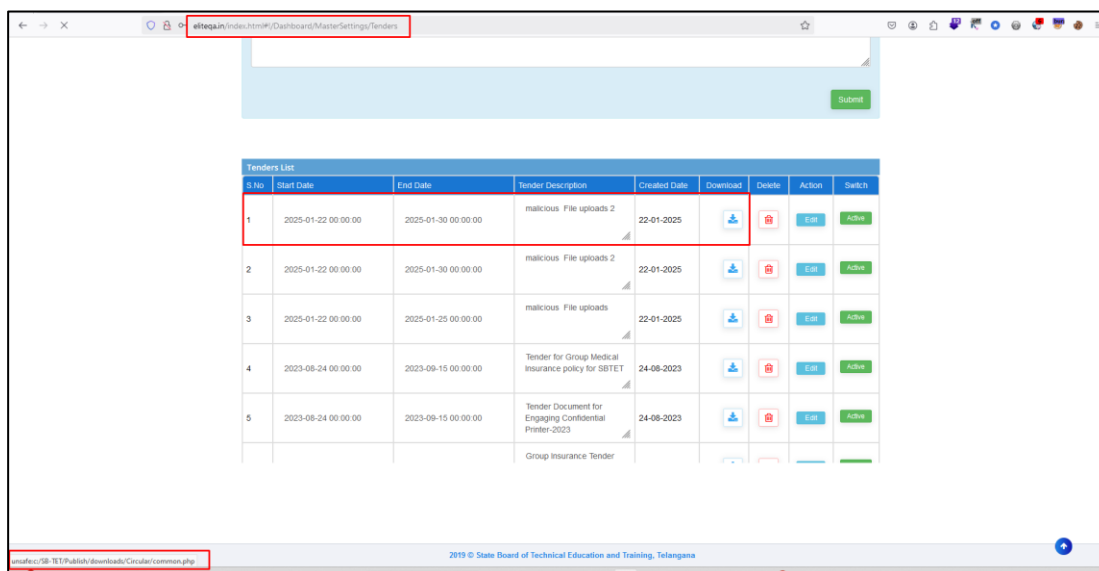
Case X

xi. New or Repeat observation

New

xii. References to evidence / Proof of Concept

Step I: Login with **admin** and go to **Setting > Tender** and Here fill the Details and upload a file and application discloses the internal path as shown in the snapshots below:



Kindly Patch this vulnerability Throughout the Application

Case XI

i. Affected Asset i.e. IP/URL/Application etc.

http://eliteqa.in/index.html#!/ForgetPassword/

ii. Observation/ Vulnerability title

OTP Flooding

iii. Detailed observation / Vulnerable point

During testing, we observed that the process of OTP generation is vulnerable to OTP flooding attacks, as team was able to replay same request, which sent more than 100 messages to the user mobile device on the same or different number, leading to financial loss or reputation loss to the organization.

iv. CVE/CWE

CWE- 400

v. Control Objective #

NA

vi. Control Name #

NA

vii. Audit Requirement #

NA

viii. Severity

Medium

ix. Recommendation

Create logic that will allow the application to send maximum 3-5 OTP per mobile number post that block the mobile number for a particular time frame.

x. Reference

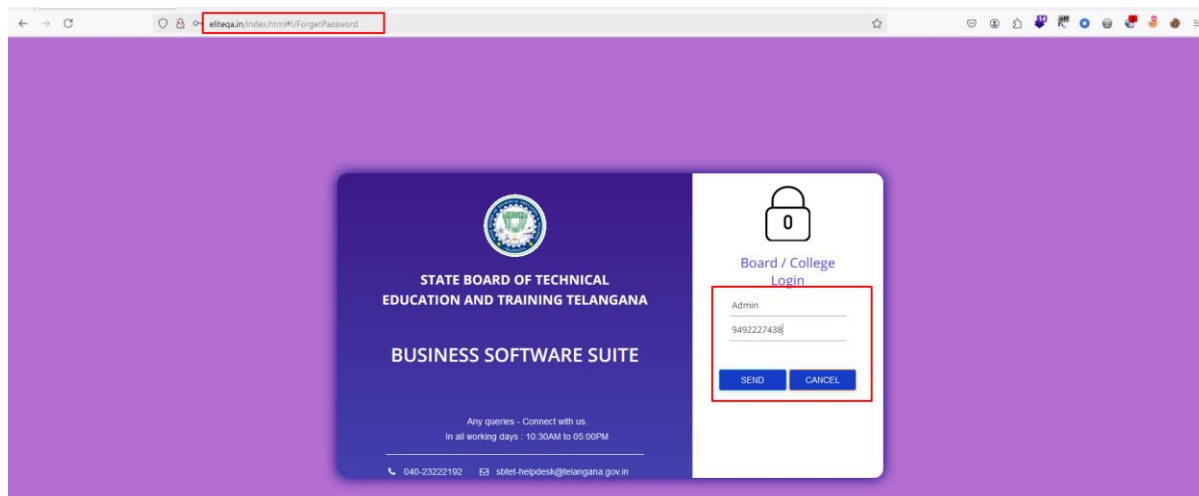
Case XI

xi. New or Repeat observation

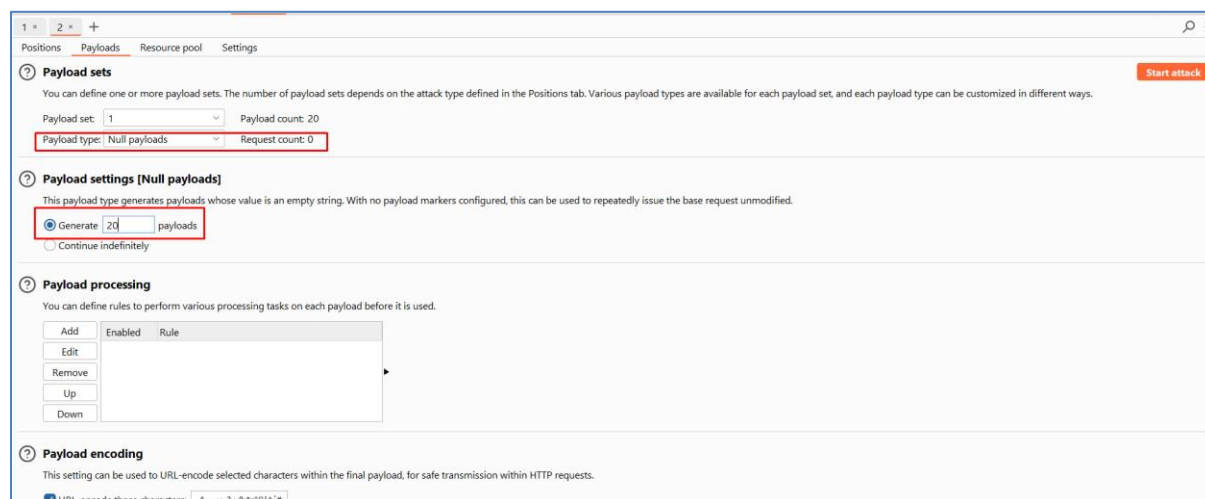
New

xii. References to evidence / Proof of Concept

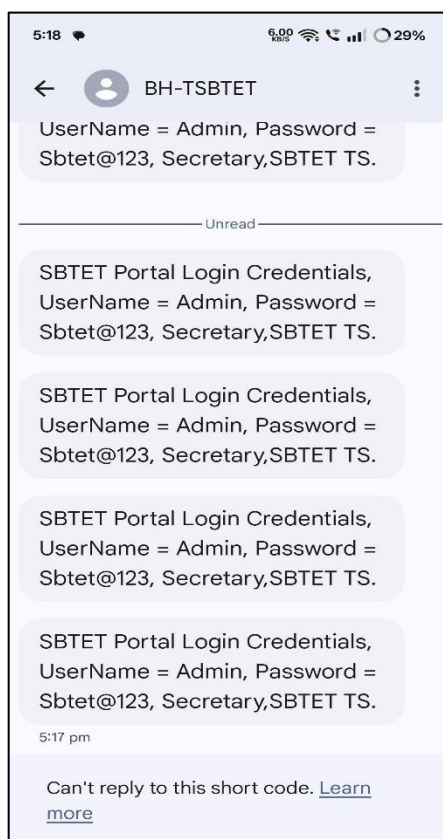
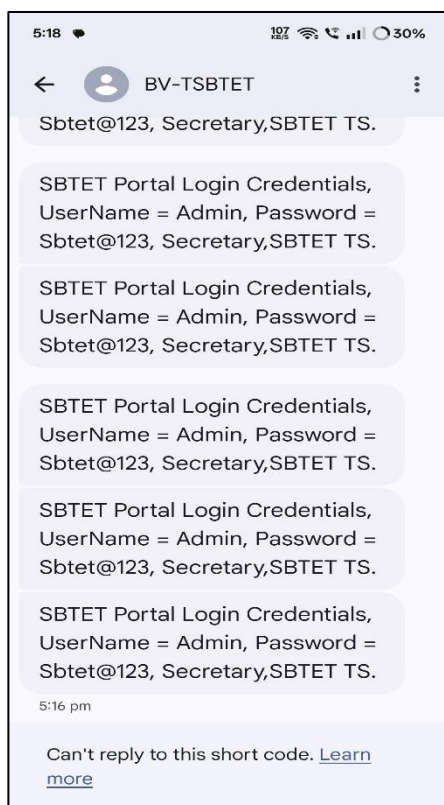
Step I: Go the URL "http://eliteqa.in/index.html#!/ForgetPassword/" and give the details and intercept the request and send the request to intruder as shown in the snapshot below



Step II: Here Set the payload as Null payload and generate the payloads as 20 as shown in the snapshot below



Step III: Here We received the more the 7 OTPs at a time as shown in the snapshot Below:



Kindly Patch this vulnerability Throughout the Application

Case XII

i. Affected Asset i.e. IP/URL/Application etc.

http://eliteqa.in/index.html#!/index/WebsiteLogin

ii. Observation/ Vulnerability title

Username Enumeration

iii. Detailed observation / Vulnerable point

One of the most common and underestimated web application vulnerabilities that is user enumeration. Simply put, we can figure out a list of valid user accounts that are allowed to login to an application. This isn't just assuming there's a common user called "Admin" and attempting to guess the password for that account. Instead, this is the ability to compile a list of valid users based on a flaw in the registration process, login sequence, or password reset functionality.

iv. CVE/CWE

CWE- 204

v. Control Objective

NA

vi. Control Name

NA

vii. Audit Requirement

NA

viii. Severity

Medium

ix. Recommendation

Login

Make sure to return a generic "No such username or password" message when a login failure occurs. Make sure the HTTP response, and the time taken to respond are no different when a username does not exist, and an incorrect password is entered.

Password Reset

Make sure your “forgotten password” page does not reveal usernames. If your password reset process involves sending an email, have the user enter their email address. Then send an email with a password reset link if the account exists.

Registration

Avoid having your site tell people that a supplied username is already taken. If your usernames are email addresses, send a password reset email if a user tries to sign-up with an existing address. If usernames are not email addresses, protect your sign-up page with a CAPTCHA. Profile Pages If your users have profile pages, make sure they are only visible to other users who are already logged in. If you hide a profile page, ensure a hidden profile is indistinguishable from a non-existent profile.

x. Reference

Case XII

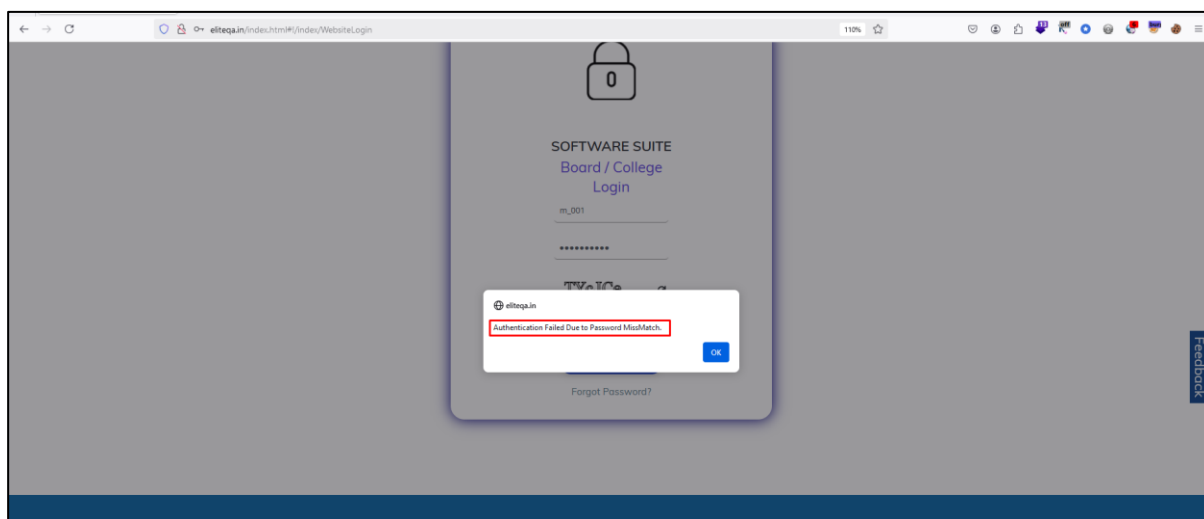
xi. New or Repeat observation

New

xii. References to evidence / Proof of Concept

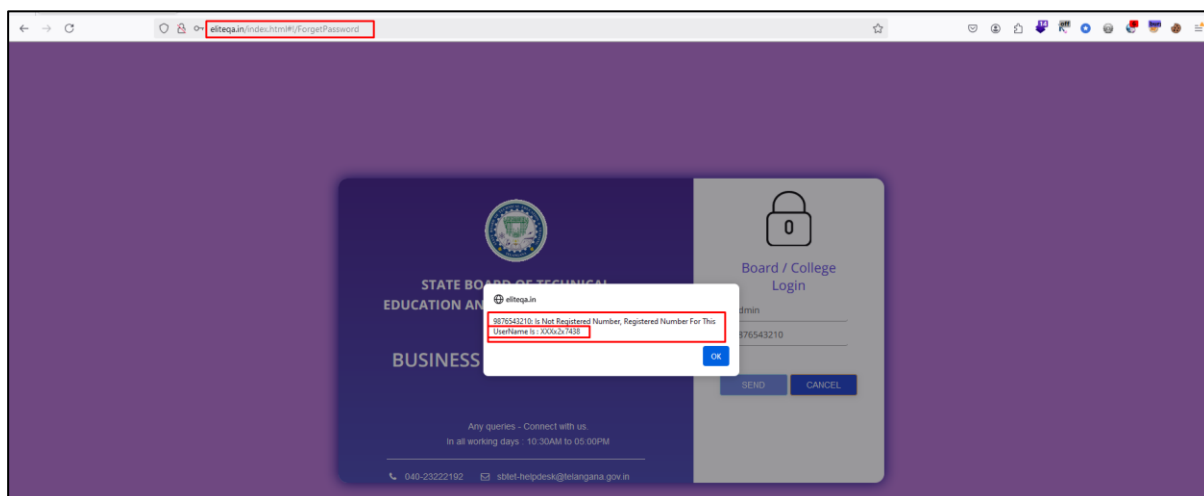
Step I: Open the URL

“<http://eliteqa.in/index.html#!/index/WebsiteLogin>”, Enter the valid username and invalid password and here We observed that application throws the error like password mismatch Which Means server Enumerates the username as shown in the snapshot



Step II: Open the Url

"http://eliteqa.in/index.html#!/ForgetPassword" , Enter the valid username and invalid mobile number ,here We observed that application throws the error like Registered number Which Means server Enumerates the username as shown in the snapshot



Kindly Generate a generic error message the Application

Case XIII

i. Affected Asset i.e. IP/URL/Application etc.

http://eliteqa.in/app/services/AppSettings.js

http://eliteqa.in/app/app.js

ii. Observation/ Vulnerability title

Sensitive File Disclosure

iii. Detailed observation / Vulnerable point

Sensitive File Disclosure occurs when an application does not adequately protect sensitive files. The files (webstat, php info, server status) etc. are exposed without authentication.

iv. CVE/CWE

CWE- 200

v. Control Objective #

NA

vi. Control Name #

NA

vii. Audit Requirement #

NA

viii. Severity

Low

ix. Recommendation

Access to sensitive files should disabled by default. Alternatively remove the files if not required.

x. Reference

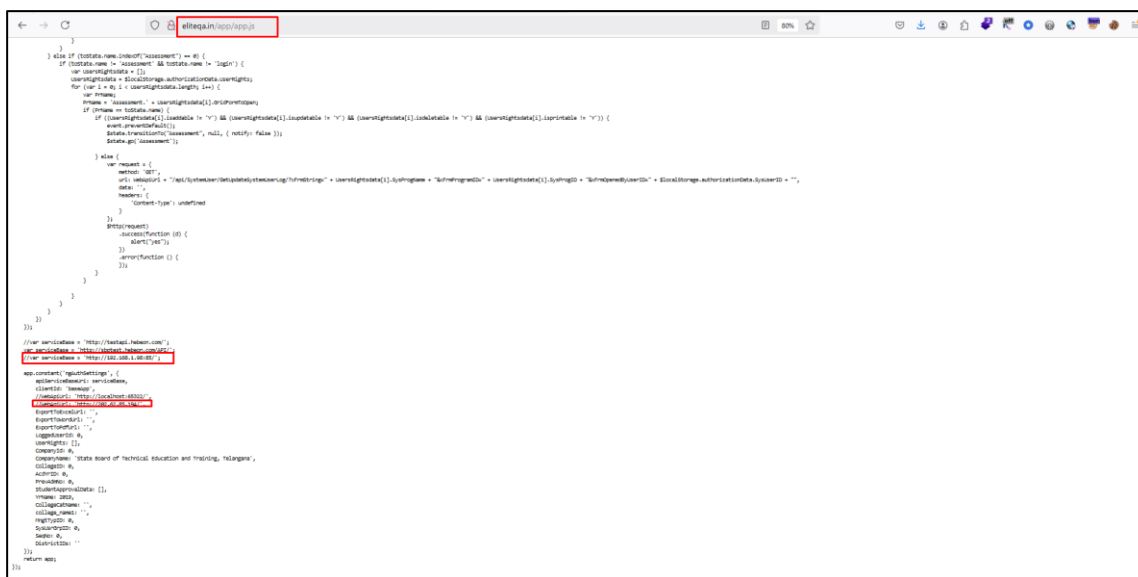
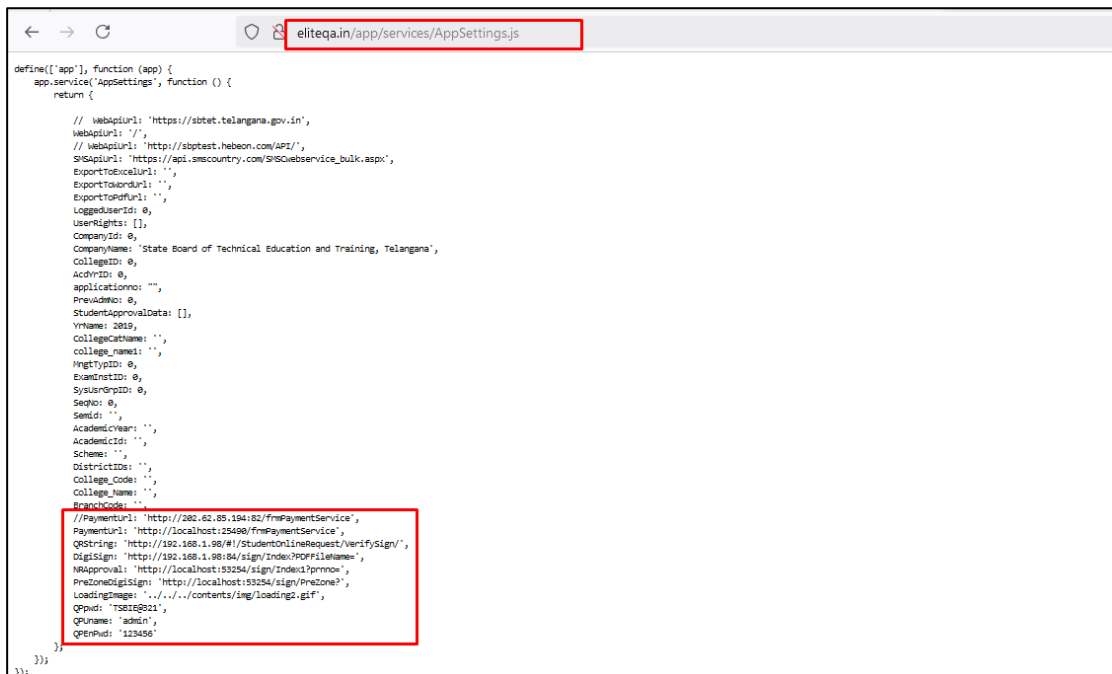
Case XIII

xi. New or Repeat observation

New

xii. References to evidence / Proof of Concept

Step I: Go to url "**http://eliteqa.in/app/services/AppSettings.js**" and **http://eliteqa.in/app/app.js** and here we observed that sensitive file as shown in the snapshot below



Kindly Patch this vulnerability Throughout the Application

Case XIV

i. Affected Asset i.e. IP/URL/Application etc.

<http://eliteqa.in/index.html#!/ForgetPassword>

ii. Observation/ Vulnerability title

Email Harvesting

iii. Detailed observation / Vulnerable point

An attacker can send SPAM mail to the application by using Email Harvester. The email harvesters are programs that scour the internet looking for email addresses on any website they come across.

iv. CVE/CWE

CWE- 200

v. Control Objective #

NA

vi. Control Name #

NA

vii. Audit Requirement #

NA

viii. Severity

Low

ix. Recommendation

The application should properly customize the email addresses while posting on the website as: Email addresses should be posted as an image not as a hyperlink. Alternatively, instead of @symbol, [at] should be used. Similarly, the dot character (.) should be replaced by [dot]. So abc@nic.in should be written as abc[at]nic[dot]in.

x. Reference

Case XIV

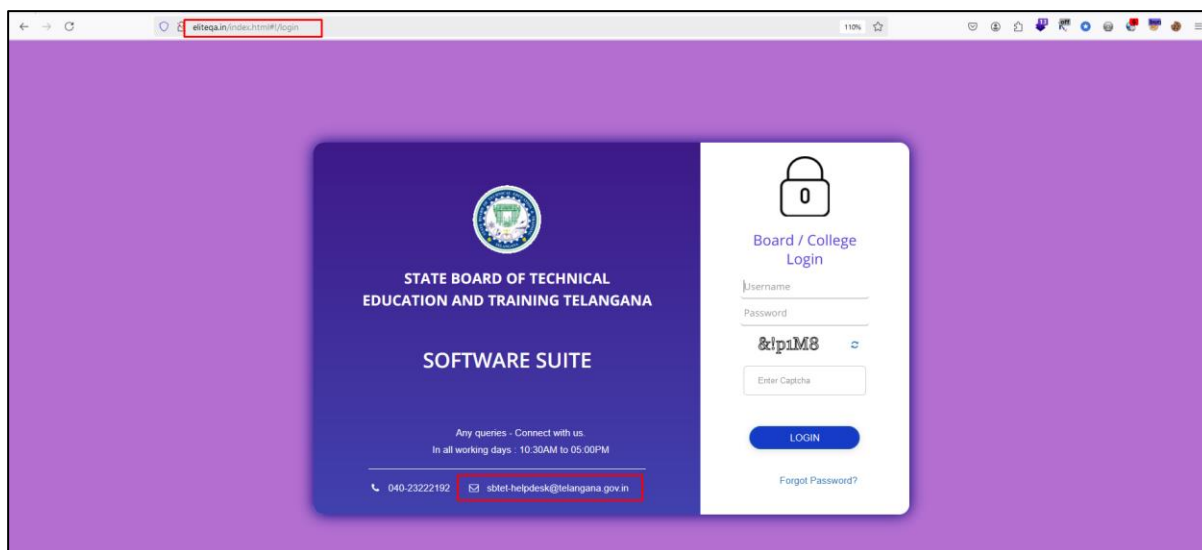
xi. New or Repeat observation

New

xii. References to evidence / Proof of Concept

Step I: open the URL

"**http://eliteqa.in/index.html#!/ForgetPassword**", Here we Observed that the Here We observed that Email address as not in customize format



Case XV

i. Affected Asset i.e. IP/URL/Application etc.

http://eliteqa.in/index.html#!/index/WebsiteLogin

ii. Observation/ Vulnerability title

Debug Method Enabled

iii. Detailed observation / Vulnerable point

ASP.NET allows remote debugging of web applications, if configured to do so. By default, debugging is subject to access control and requires platform-level authentication.

iv. CVE/CWE

CWE- 11

v. Control Objective

NA

vi. Control Name

NA

vii. Audit Requirement

NA

viii. Severity

Low

ix. Recommendation

To disable debugging, open the Web.config file for the application, and find the <compilation> element within the <system.web> section. Set the debug attribute to ""false"". Note that it is also possible to enable debugging for all applications within the Machine.config file. You should confirm that the debug attribute in the <compilation> element has not been set to ""true"" within the Machine.config file.

It is strongly recommended that you refer to your platform's documentation relating to this issue, and do not rely solely on the above remediation

x. Reference

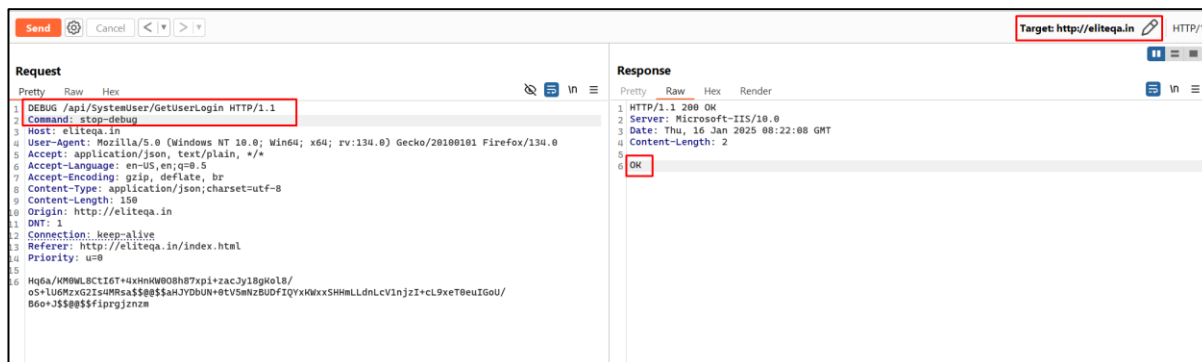
Case XV

xi. New or Repeat observation

New

xii. References to evidence / Proof of Concept

Step I: Debug method is enabled as mentioned in the snapshot below:



Note: Kindly, patch the vulnerability throughout the application.

Case XVI

i. Affected Asset i.e. IP/URL/Application etc.

http://eliteqa.in/index.html#!/index/WebsiteLogin

ii. Observation/ Vulnerability title

Lack of security Headers

iii. Detailed observation / Vulnerable point

Applications can unintentionally leak information about their configuration, internal workings, or violate privacy through a variety of application problems. Attackers use this weakness to steal sensitive data or conduct more serious attacks.

iv. CVE/CWE

CWE-644

v. Control Objective #

NA

vi. Control Name #

NA

vii. Audit Requirement #

NA

viii. Severity

Low

ix. Recommendation

Implement security headers such as X-XSS-Protection, Content-Security-Policy, Referrer Policy, X-Content-Type-Options, Permission Policy and Strict-transport-layer-protection

x. Reference

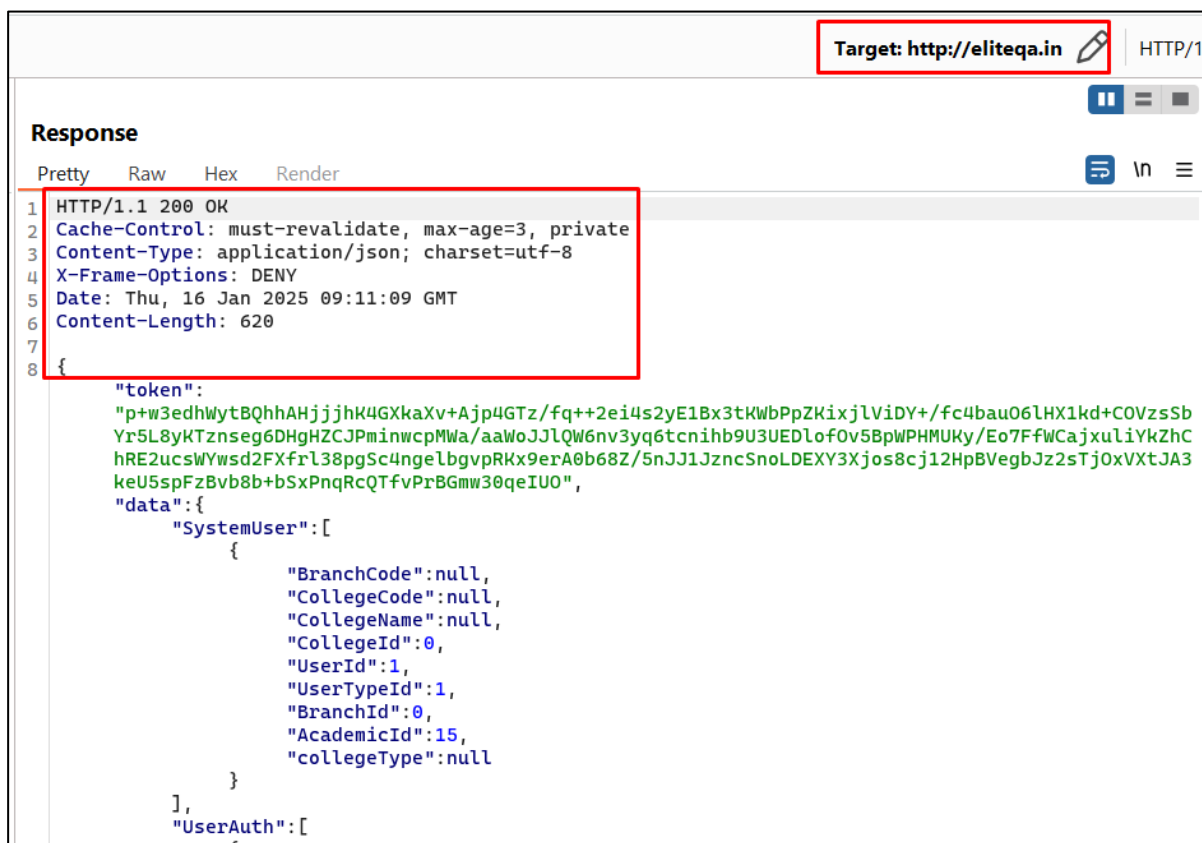
Case XVI

xi. New or Repeat observation

New

xii. References to evidence / Proof of Concept

Step I: The security headers are not implemented as shown in the snapshot below



Kindly Patch this vulnerability Throughout the Application

Case XVII

i. Affected Asset i.e. IP/URL/Application etc.

http://eliteqa.in/index.html#!/index/WebsiteLogin/

ii. Observation/ Vulnerability title

Clickjacking

iii. Detailed observation / Vulnerable point

Click Jacking (User Interface redress attack, UI redress attack, UI redressing) is a malicious technique of tricking a Web user into clicking on something different from what the user perceives they are clicking on, thus potentially revealing confidential information or taking control of their computer while clicking on seemingly innocuous web pages.

iv. CVE/CWE

CWE-1021

v. Control Objective

NA

vi. Control Name

NA

vii. Audit Requirement

NA

viii. Severity

Low

ix. Recommendation

Preventing the browser from loading the page in frame using the X-Frame-Options or Content Security Policy (frame-ancestors) HTTP headers.
Preventing session cookies from being included when the page is loaded in a frame using the SameSite cookie attribute.
Implementing JavaScript code in the page to attempt to prevent it being loaded in a frame (known as a "frame-buster").

x. Reference

Case XVII

xi. New or Repeat observation

New

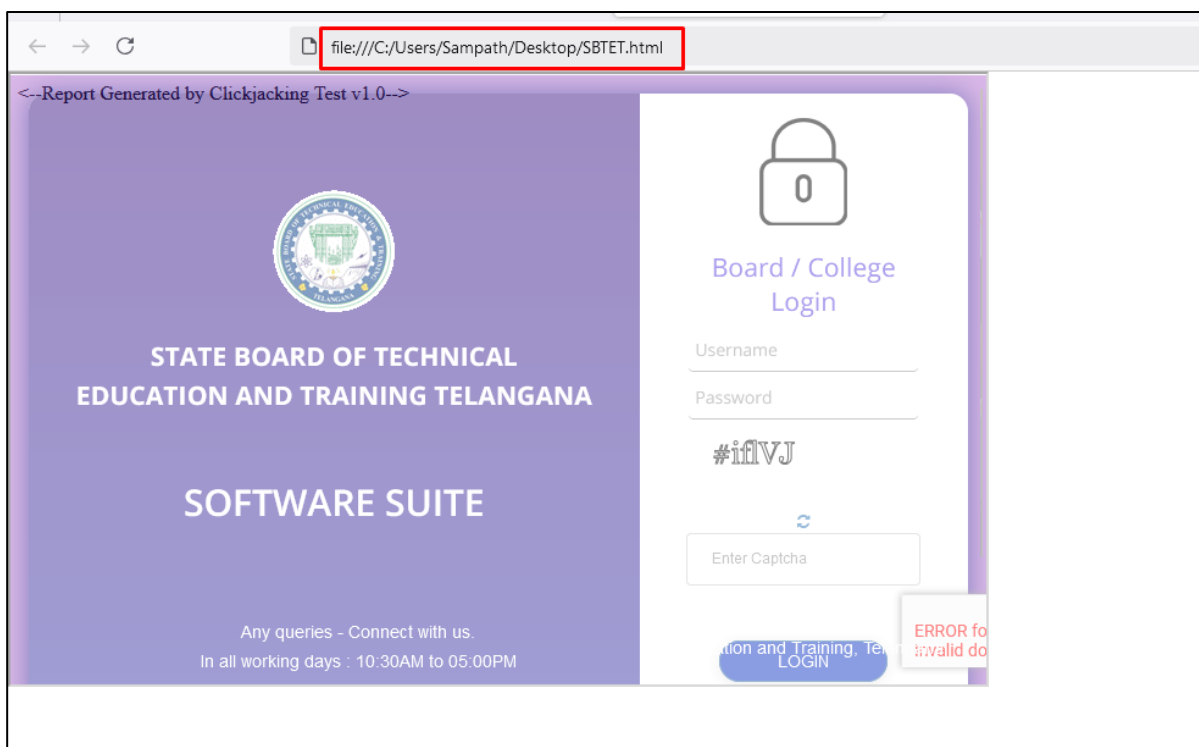
xii. References to evidence / Proof of Concept

Step I: Create a malicious html file which contains Clickjacking Code

```
File Edit View

<html>
  <!--Report Generated by Clickjacking Test v1.0-->
  <style>
    iframe {
      width: 800px;
      height: 500px;
      position: absolute;
      top: 0; left: 0;
      filter: alpha(opacity=50);
      opacity: 0.5;
    }
  </style>
  <iframe src="http://eliteqa.in/index.html#!/login">
</html>
```

Step II: Execute the file in web browser. Application can be injected in frame window.



Case XVIII

i. Affected Asset i.e. IP/URL/Application etc.

http://eliteqa.in/index.html#!/index/WebsiteLogin

ii. Observation/ Vulnerability title

Version Disclosure

iii. Detailed observation / Vulnerable point

Applications can unintentionally leak information about their configuration, internal workings, or violate privacy through a variety of application problems. Attackers use this weakness to steal sensitive data or conduct more serious attacks.

iv. CVE/CWE

CWE-200

v. Control Objective #

NA

vi. Control Name #

NA

vii. Audit Requirement #

NA

viii. Severity

Low

ix. Recommendation

Hide version details in the response

x. Reference

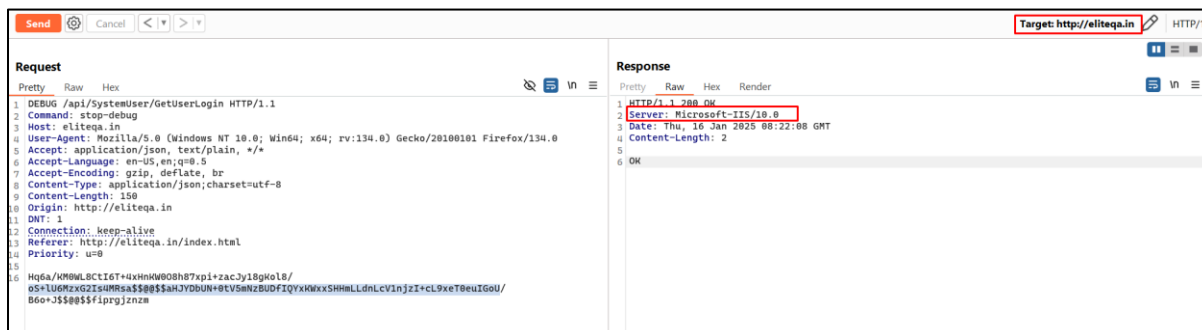
Case XVIII

xi. New or Repeat observation

New

xii. References to evidence / Proof of Concept

Step I: Intercept the any request and we observed that application disclosures the Versions in response as shown in the snapshot below



Case XIX

i. Affected Asset i.e. IP/URL/Application etc.

http://eliteqa.in/index.html#!/Dashboard/AdmissionDashboard/SearchStudent

ii. Observation/ Vulnerability title

Aadhaar Number in plain text

iii. Detailed observation / Vulnerable point

UID number is treated as highly sensitive information and it should not be leaked at any end-point and must not be kept longer than it is required. The attacker can steal the UID numbers at end-points and can use them for any personal use.

iv. CVE/CWE

CWE-319

v. Control Objective

NA

vi. Control Name

NA

vii. Audit Requirement

NA

viii. Severity

Low

ix. Recommendation

As per the Aadhaar Act and Regulations by UIDAI, Aadhaar number must not be stored in the database/files without the consent of the user. If the application requires to store Aadhaar number in permanent storage, Aadhaar seeding process must be followed.

Permanent storages for further delivery of services, Aadhaar seeding process must be followed.

Aadhaar number should be used as per the Aadhaar Act and Regulations.

x. Reference

Case XIX

xi. New or Repeat observation

New

xii. References to evidence / Proof of Concept

Step I: Login with Admin and Go Student Search and Enter the Details of student and click on search here We observed that aadhar number in plaintext as shown in the snapshot:

The screenshot shows the 'Student Search' page on the AP State Board of Technical Education website. The search bar contains the text '21001-EC-001'. Below the search bar, there are three tabs: 'Assessment', 'Attendance', and 'Admission'. The 'Admission' tab is selected, displaying a student's profile for EDE TEJA VARDHAN. The profile includes personal details, academic information, and a list of documents. The 'Admission Number' is highlighted with a red box.

Name	EDE TEJA VARDHAN		
Pin	21001-EC-001		
Adhaar/Verified	true		
Active/Log	true		
Attendeeid	1058-21023		
Id/ansferred	NO		
Admission Type	POLYTECH		
Father Name	E NAGU RAJU		
Mother's Name	E SRUJANTHI		
Date of Birth	27-12-2005		
Gender	Male		
Semester	SSEM		
Branch	ELECTRONICS & COMMUNICATION ENGINEERING	Tenth Year	2021
South Board	SBC	South Hallticket Number	2121124879
Polytech Hallticket Number	4920548	Minority Type	
Admission Number	598424100074	Region	
Admission Number	598424100074	Region	
Midtest (Marked) During			
SIC	1		
House No & Street (or)			
Village	HYDERABAD	Mandal	KESARA
District	MEDCHL	Email Id	ESBT191@GMAIL.COM
Pincode	501301	Parent Mobile Number	9392843730
Student Mobile Number	9392843730	Mother Adhaar Number	
Father Adhaar Number		Maxima Income	
Is Father Govt		Certificate Number	IC02212571807
Employee ?	No	Max Sava Caste	
Occupation	SE	Certificate Number	
Bank Name		Bank Account Number	
IF SC Code		Branch	

Case XX

i. Affected Asset i.e. IP/URL/Application etc.

http://eliteqa.in/index.html#!/index/WebsiteLogin

ii. Observation/ Vulnerability title

Application accepting same password

iii. Detailed observation / Vulnerable point

The application is accepting the same password issued by the administrator such as "Exl@123456". However the application shouldn't accept the same password.

iv. CVE/CWE

CWE-620

v. Control Objective

NA

vi. Control Name

NA

vii. Audit Requirement

NA

viii. Severity

Low

ix. Recommendation

Implement proper validation for disallowing the user to create same password.

x. Reference

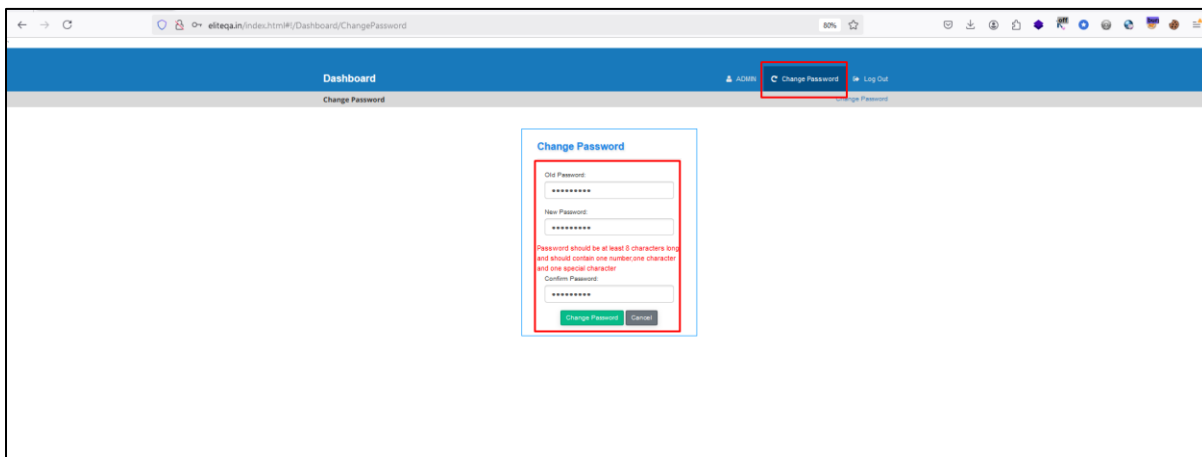
Case XX

xi. New or Repeat observation

New

xii. References to evidence / Proof of Concept

Step I: Login and Go to the User profile and Change Password Field and Give the same password in current, new and re-enter new password and click on change password as shown in the snapshot below:



The screenshot shows a web browser window with the URL `eltraqa.in/index.html#/Dashboard/ChangePassword`. The page has a blue header bar with the text "Dashboard" on the left and "ADMIN" on the right. In the center of the header bar, there is a "Change Password" link, which is highlighted with a red rectangle. Below the header bar, the main content area is titled "Change Password". In the center of this area, there is a "Change Password" form, also highlighted with a red rectangle. The form contains three input fields: "Old Password", "New Password", and "Confirm Password". Each field has a red asterisk below it, indicating a required field. Below the input fields, there is a red text message: "Password should be at least 6 characters long and should contain one number one character and one special character". At the bottom of the form, there are two buttons: "Change Password" (in green) and "Cancel" (in grey).

Case XXI

i. Affected Asset i.e. IP/URL/Application etc.

<http://eliteqa.in/index.html#!/index/WebsiteLogin>

ii. Observation/ Vulnerability title

Autocomplete Enabled on Password Field

iii. Detailed observation / Vulnerable point

Most browsers have a facility to remember user credentials that are entered into HTML forms. This function can be configured by the user and also by applications that employ user credentials. If the function is enabled, then credentials entered by the user are stored on their local computer and retrieved by the browser on future visits to the same application.

iv. CVE/CWE

CWE-200

v. Control Objective

NA

vi. Control Name

NA

vii. Audit Requirement

NA

viii. Severity

Low

ix. Recommendation

To prevent browsers from storing credentials entered into HTML forms, include the attribute autocomplete="off" within the FORM tag (to protect all form fields) or within the relevant INPUT tags (to protect specific individual fields).

x. Reference

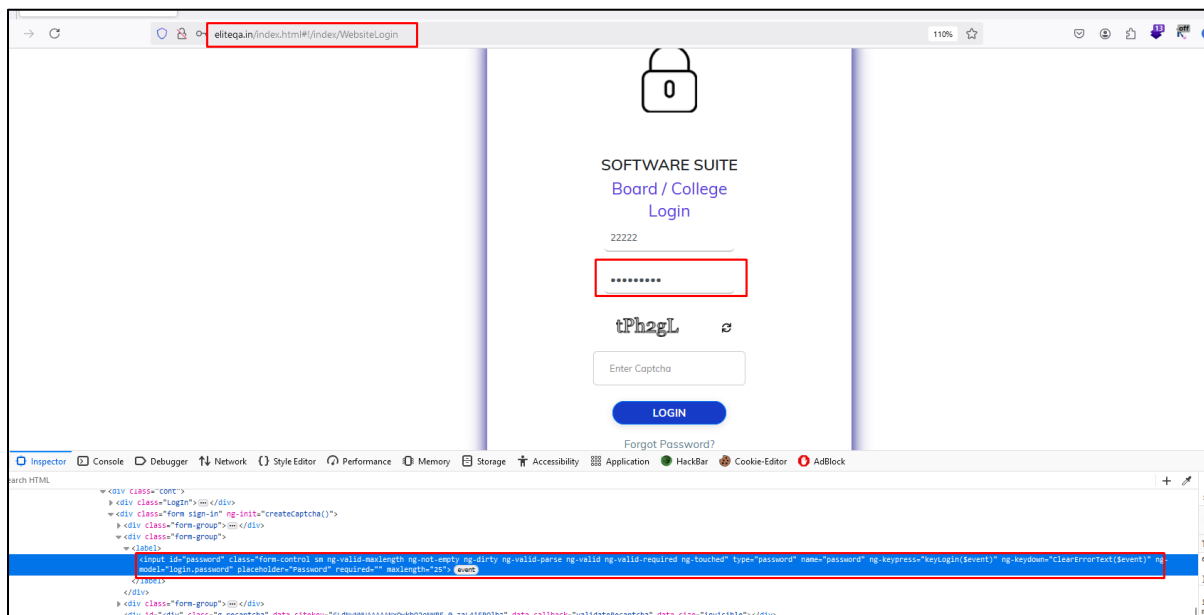
Case XXI

xi. New or Repeat observation

New

xii. References to evidence / Proof of Concept

Step I: It is observed that autocompleted for password is enabled in the application.



Case XXII

i. Affected Asset i.e. IP/URL/Application etc.

http://eliteqa.in/index.html#!/index/WebsiteLogin

ii. Observation/ Vulnerability title

Security Logging and Monitoring Failures

iii. Detailed observation / Vulnerable point

The application does not maintain any record of the application usage in the form of a report or audit trail. Any malicious activity cannot be monitored or traced back. In case of any misuse or attack, it may be difficult to trace and locate the origin.

iv. CVE/CWE

CWE-778

v. Control Objective

NA

vi. Control Name

NA

vii. Audit Requirement

NA

viii. Severity

Low

ix. Recommendation

Information to be logged includes the following: IP address of the originating Source, Date, Time, Username (No Password), session details, Referrer, Process id, URL, User Agent, Countries if any in addition to other details to be logged in the website. Logging of Authentication Process which includes number of successful and failed login attempts. To create audit logs, use auto numbering so that every logged entry has an un-editable log number. Then if one audit entry is deleted a gap in the numbering sequence will appear. Report of the website logs to be generated weekly by the administrator to keep track of the website activities.

x. Reference

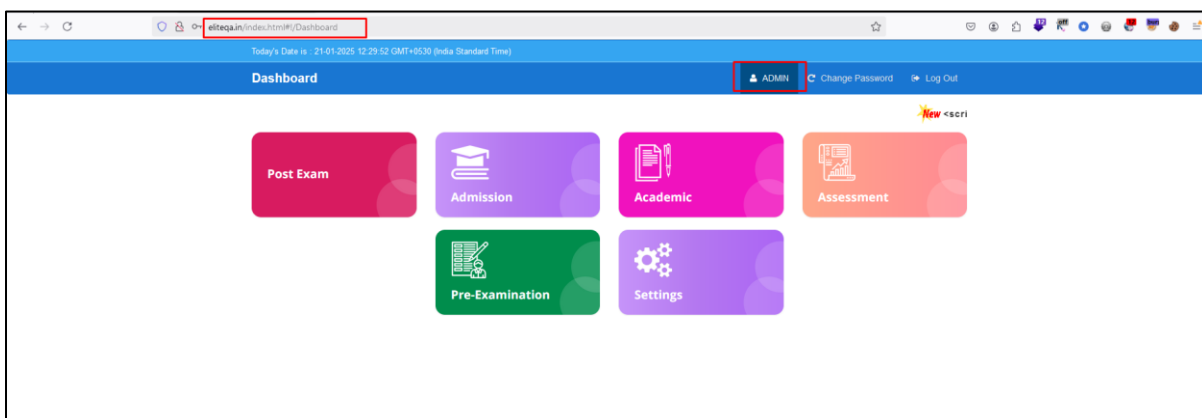
Case XXII

xi. New or Repeat observation

New

xii. References to evidence / Proof of Concept

Step I: Login with admin and Here we Observed that there is no logs in the application



Note: If the logs are being maintained on admin panel or server then kindly share screenshot with us in the email.

Appendix 'A'

OWASP TOP 10: 2021

The Open Worldwide Application Security Project (OWASP) is a non-profit foundation dedicated to improving the security of software. OWASP Top 10 is an online document on OWASP's Web Application that provides ranking of and remediation guidance for the top 10 most critical Web Application security risks. It represents a broad consensus about what are the most critical application security flaws. The risks are ranked and based on the frequency of discovered security defects, the severity of the vulnerabilities, and the magnitude of their potential impacts. The purpose of the report is to offer developers and Web Application security professionals insight into the most prevalent security risks so that they may incorporate the report's findings and recommendations into their security practices, thereby minimizing the presence of these known risks in their applications.

The following table summarizes the OWASP Top 10 2021 Most Critical Web Application Security Vulnerabilities:

S. No.	Vulnerability & Description	Impact
<u>A1</u>	<i>Broken Access Control</i> Restrictions on what authenticated users are allowed to do are often not properly enforced. Attackers can exploit these flaws to access unauthorized functionality and/or data, such as access other users' accounts, view sensitive files, modify other users' data, change access rights, etc.	Impact can be a vertical or a horizontal privilege escalation. For e.g., attackers acting as users or administrators, or users using privileged functions, or creating, accessing, updating or deleting every record.

<u>A2</u>	<i>Cryptographic Failures</i> Many Web Applications and APIs do not properly protect sensitive data, such as financial, healthcare, and PII. Attackers may steal or modify such weakly protected data to conduct credit card fraud, identity theft, or other crimes. Sensitive data may be compromised without extra protection, such as encryption at rest or in transit, and requires special precautions when exchanged with the browser.	Compromise of all data that should have been protected. Typically, this includes sensitive personal information data such as health records, credentials, personal data, and credit cards, which often require protection.
<u>A3</u>	<i>Injection</i> Injection flaws, such as SQL, NoSQL, OS, and LDAP injection, occur when untrusted data is sent to an interpreter as part of a command or query. The attacker's hostile data can trick the interpreter into executing unintended commands or accessing data without proper authorization.	Injection can result in data loss or corruption, lack of accountability, or denial of access. Injection can sometimes lead to complete host takeover.
<u>A4</u>	<i>Insecure Design</i> An insecure design cannot be fixed by a perfect implementation as by definition, needed security controls were never created to defend against specific attacks. One of the factors that contribute to insecure design is the lack of business risk profiling inherent in the software or system being developed, and thus the failure	Insecure application design can have severe consequences for the business, as it may allow attackers interfere with the application logic and lead to sensitive information disclosure or Web Application compromise.

	to determine what level of security design is required.	
<u>A5</u>	<i>Security Misconfiguration</i> Security misconfiguration is the most commonly seen issue. This is commonly a result of insecure default configurations, incomplete or ad hoc configurations, open cloud storage, misconfigured HTTP headers, and verbose error messages containing sensitive information. Not only must all operating systems, frameworks, libraries, and applications be securely configured, but they must be patched/updated in a timely fashion.	Such flaws frequently give attackers unauthorized access to some system data or functionality. Occasionally, such flaws result in a complete system compromise.
<u>A6</u>	<i>Vulnerable and Outdated Components</i> Components, such as libraries, frameworks, and other software modules, run with the same privileges as the application. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server takeover. Applications and APIs using components with known vulnerabilities may undermine application defenses and enable various attacks and impacts.	While some known vulnerabilities lead to only minor impacts, some of the largest breaches to date have relied on exploiting known vulnerabilities in components.
<u>A7</u>	<i>Identification and Authentication Failures</i> Application functions related to authentication and session management are often implemented incorrectly,	Attackers have to gain access to only a few accounts, or just one admin account to compromise the system. Depending on the domain of the application, this may allow money

	allowing attackers to compromise passwords, keys, or session tokens, or to exploit other implementation flaws to assume other users' identities temporarily or permanently.	laundering, social security fraud, and identity theft, or disclose legally protected highly sensitive information.
<u>A8</u>	<i>Software and Data Integrity Failures</i> Software and data integrity failures relate to code and infrastructure that does not protect against integrity violations. An example of this is where an application relies upon plugins, libraries, or modules from untrusted sources, repositories, and content delivery networks (CDNs). An insecure CI/CD pipeline can introduce the potential for unauthorized access, malicious code, or system compromise.	Attackers could potentially upload their own updates to be distributed and run on all installations.
<u>A9</u>	<i>Security Logging and Monitoring Failures</i> Insufficient logging and monitoring, coupled with missing or ineffective integration with incident response, allows attackers to further attack systems, maintain persistence, pivot to more systems, and tamper, extract, or destroy data. Most breach studies show time to detect a breach is over 200 days, typically detected by external parties rather than internal processes or monitoring.	Most successful attacks start with vulnerability probing. Allowing such probes to continue can raise the likelihood of successful exploit to nearly 100%.

<u>A10</u>	<i>Server-Side Request Forgery (SSRF)</i> The application is vulnerable to server-side request forgery. Server-side request forgery occurs when an application submits a URL that contains user-controlled data to the server. An attacker can manipulate the URL and cause the server to make requests to internal resources that are not available on the Internet.	An attacker who can control requests made by a server can conduct a number of attacks against other servers. By manipulating a URL and having requests go through a server, the attacker may be able to abuse trust relationships to bypass firewall rules or IP whitelisting, for example. Services that are available internally may be accessible to an attacker through server-side request forgery. An attacker may be able to interact with an internal-only API by making HTTP requests through the manipulated URL. In other cases, the file URI scheme (file://) may be used to retrieve files from servers on the internal network.
-------------------	---	---

Table 1: OWASP Top 10 - 2021

Reference: <https://owasp.org/www-project-top-ten/>

Appendix 'B'

Tools Description

Burp Suite Professional

Portswigger's Burp Suite Professional is an advanced set of tools for testing web security. Burp Suite offers the features for both manual and automated scans. Through Burp Suite, a user can intercept HTTP traffic, find hidden attack surface, assess strength of tokens, perform brute-forcing and fuzzing, construct CSRF exploits, modify HTTP messages, scan for common vulnerabilities including the OWASP Top 10.

Acunetix Vulnerability Scanner

Acunetix is a web vulnerability scanner which is also a complete Web Application security testing solution that can be used both standalone and as part of complex environments. It offers built-in vulnerability assessment and vulnerability management, as well as many options for integration with market-leading software development tools.

Gobuster

Gobuster is a command line scanner that looks for existing or hidden web objects. It works by launching a dictionary attack against a web server and analyzing the response. Gobuster is used to brute-force URIs (directories and files) and DNS subdomains.

SQLMAP

SQLMAP is an open-source penetration testing tool that automates the process of detecting and exploiting SQL injection flaws and taking over of database servers. It comes with a powerful detection engine, many niche features for the ultimate penetration tester and a broad range of switches lasting from database fingerprinting, over data fetching from the database, to accessing the underlying file system and executing commands on the operating system via out-of-band connections.

Nmap

Nmap (Network Mapper) is a free and open-source network scanner that is used to discover hosts and services on a computer network by sending packets and analyzing the responses. Nmap provides a number of features for probing computer networks, including host discovery and service and operating system detection. These features are extensible by scripts that provide more advanced service detection, vulnerability detection, and other features.