G.LOKESHREDDY
721128805325
ASSIGNMENT-4

# STEP- : OWASP Top 10 Vulnerabilities

## Overview

The OWASP (Open Web Application Security Project) Top 10 vulnerabilities list is a widely recognized resource in the field of web application security. It identifies and ranks the most critical security risks faced by web applications. Here's an overview of some of the key vulnerabilities listed and their potential impacts on web application security:

Injection Attacks: 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. This can lead to unauthorised access to sensitive data, data loss, or even complete system compromise.
Broken Authentication: Weaknesses in authentication mechanisms can allow attackers to compromise user accounts, impersonate users, and gain unauthorised access to sensitive data or functionality.
Sensitive Data Exposure: When sensitive data is not adequately protected, it can be exposed to unauthorised parties. This can lead to identity theft, fraud, and other malicious activities.
XML External Entities (XXE): XXE vulnerabilities occur when an application parses XML input from untrusted sources. Attackers can exploit these vulnerabilities to read sensitive files, perform server-side request forgery (SSRF) attacks, and execute arbitrary code.ri
Broken Access Control: Inadequate access controls can allow unauthorised users to access restricted functionality or data. This can result in unauthorised data modification, information disclosure, or privilege escalation.
Security Misconfigurations: Incorrectly configured security settings, such as default credentials, unnecessary services, or overly permissive access controls, can expose web applications to various attacks and compromise their security posture.
Cross-Site Scripting (XSS): XSS vulnerabilities occur when attackers inject malicious scripts into web applications, which are then executed in the browsers of unsuspecting users. This can lead to session hijacking, defacement, and theft of sensitive information such as cookies or credentials.
Insecure Deserialization: Insecure deserialization vulnerabilities can be exploited to execute arbitrary code, perform denial-of-service (DoS) attacks, or tamper with application logic. This can result in data loss, system compromise, or service disruption.
Using Components with Known Vulnerabilities: Including third-party components with known vulnerabilities in web applications can expose them to exploitation by attackers. Regularly updating and patching these components is essential to mitigate this risk.

Insufficient Logging and Monitoring: Inadequate logging and monitoring mechanisms can make it difficult to detect and respond to security incidents effectively. This can prolong the duration of attacks, increase the severity of their impact, and hinder forensic analysis and incident response efforts.

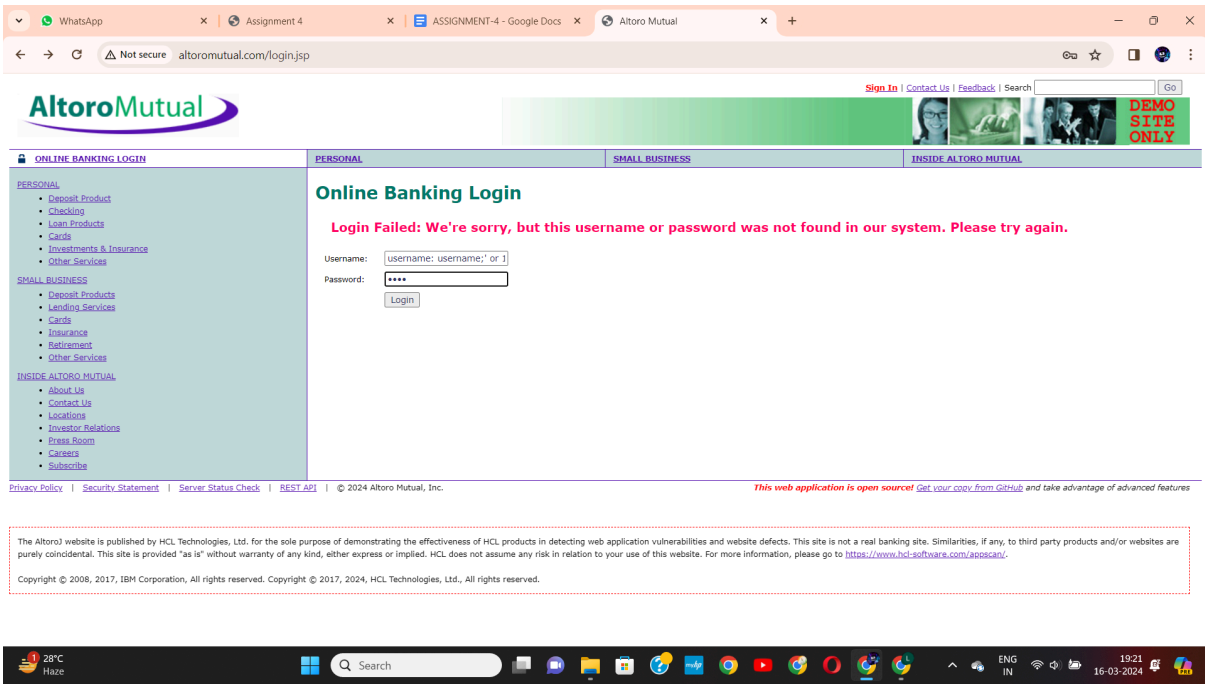
Addressing these vulnerabilities is crucial for preventing exploitation by attackers and maintaining the security and integrity of web applications. This involves implementing secure coding practices, conducting regular security assessments, and adopting robust security controls and monitoring mechanisms. Additionally, fostering a security-conscious culture within organisations and staying informed about emerging threats and best practices are essential for effectively mitigating web application security risks.

# STEP-2 : Altro Mutual Website Analysis

Analyzing the Altro Mutual website for potential vulnerabilities based on the OWASP Top 10 list requires a systematic approach to identify and assess various sections and functionalities. Here's a breakdown of the analysis:

Login Page:

- Check for SQL injection vulnerabilities by attempting to inject SQL code into the login form fields.



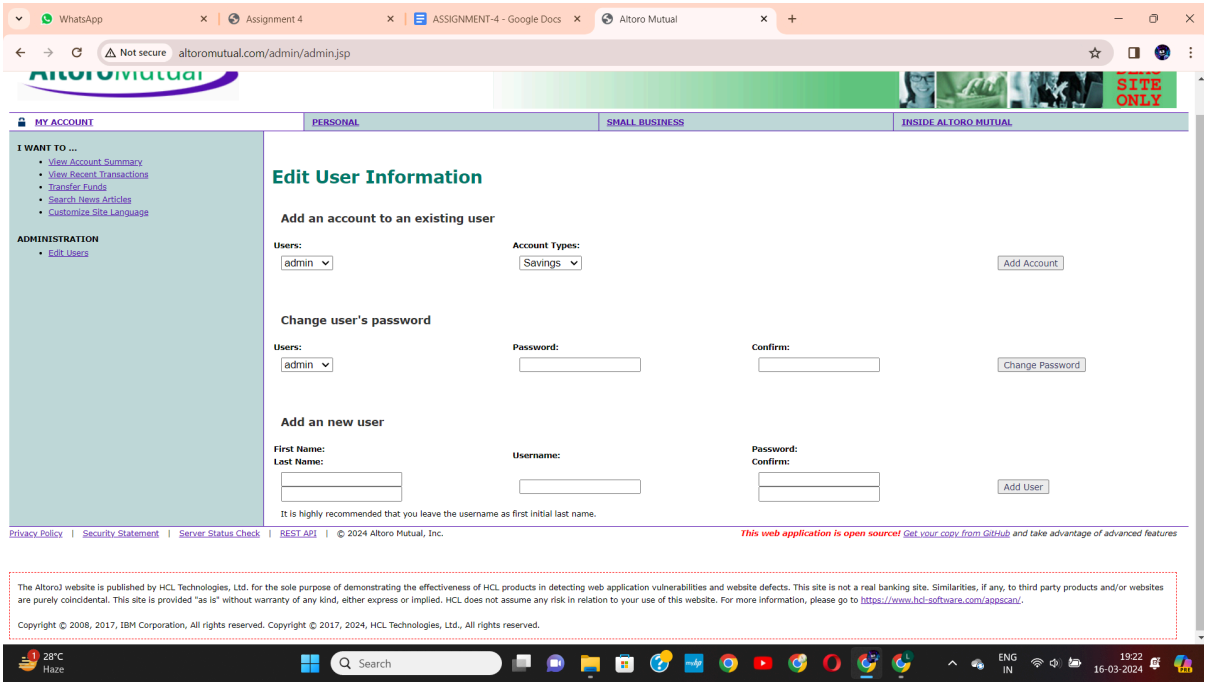

- 


- Look for XSS vulnerabilities by entering script tags or other HTML/JavaScript code into the login form fields.

- Assess the authentication mechanism for weaknesses such as weak password policies, lack of multi-factor authentication, or predictable login credentials.

User Registration:

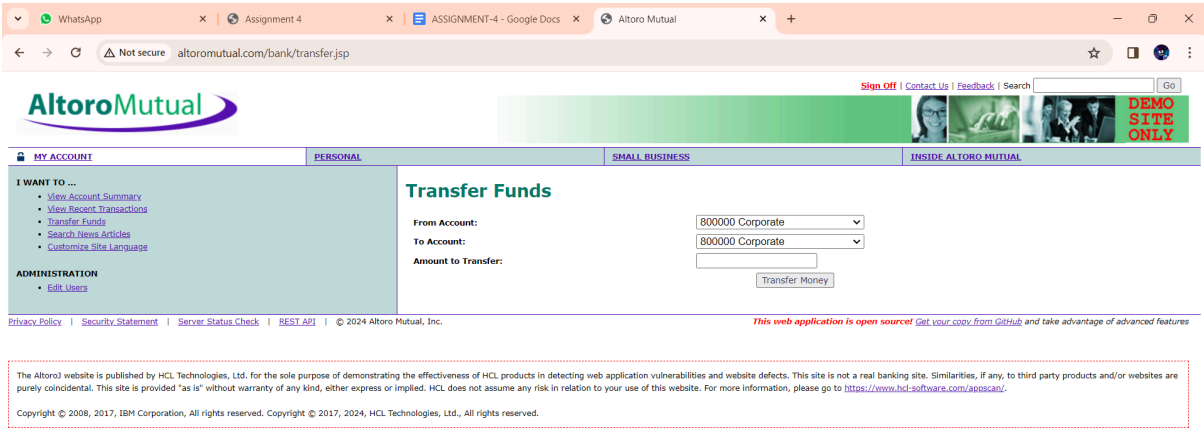
- Verify the registration form for SQL injection vulnerabilities by submitting malicious input.




- Test for XSS vulnerabilities by entering script tags or other HTML/JavaScript code into the registration form fields.
- Evaluate the strength of password policies and ensure secure storage of user credentials.
- Check for insecure direct object references by manipulating parameters or URLs to access unauthorised resources.

Payment Portal:

- Examine the payment portal for SQL injection vulnerabilities by attempting to inject SQL code into payment-related form fields.




- 

- Test for XSS vulnerabilities by injecting script tags or other HTML/JavaScript code into payment-related form fields.
- Assess the security of payment transactions, including the use of encryption and adherence to PCI DSS (Payment Card Industry Data Security Standard) guidelines.
- Verify the authorization mechanisms to prevent unauthorised access to payment-related functionalities or data.

Contact Forms:

- Check for SQL injection vulnerabilities by submitting malicious input into contact form fields.
- Test for XSS vulnerabilities by injecting script tags or other HTML/JavaScript code into contact form fields.
- Assess the validation and sanitization of user input to prevent injection attacks and other forms of malicious input.

Other Interactive Features:

- Explore any additional interactive features on the website, such as forums, chat systems, or file upload functionalities.
- Test these features for common vulnerabilities such as SQL injection, XSS, CSRF (Cross-Site Request Forgery), etc., based on their respective functionalities.

After identifying potential vulnerabilities in each section, prioritise them based on severity and likelihood of exploitation. Then, develop a plan to address and mitigate these vulnerabilities, which may include implementing secure coding practices, applying security controls, conducting regular security assessments, and providing security awareness training for developers and users. Regular monitoring and

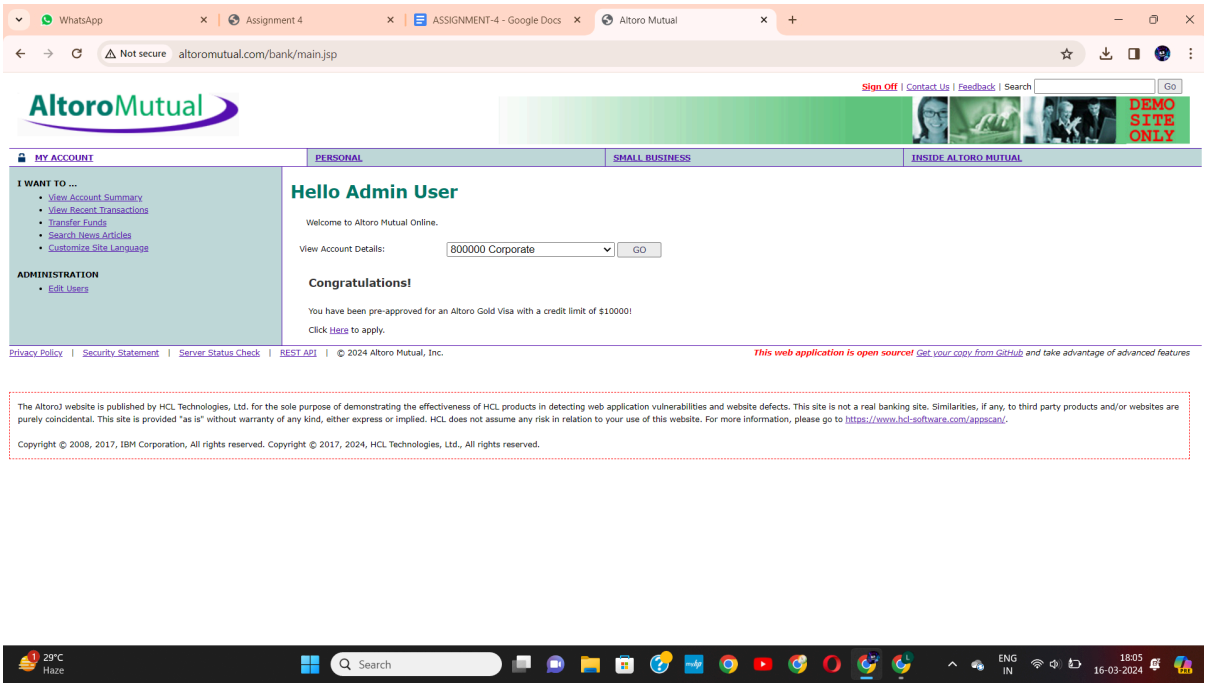
maintenance are also essential to ensure ongoing security of the Altro Mutual website.

# STEP-3 : Vulnerability Identification Report

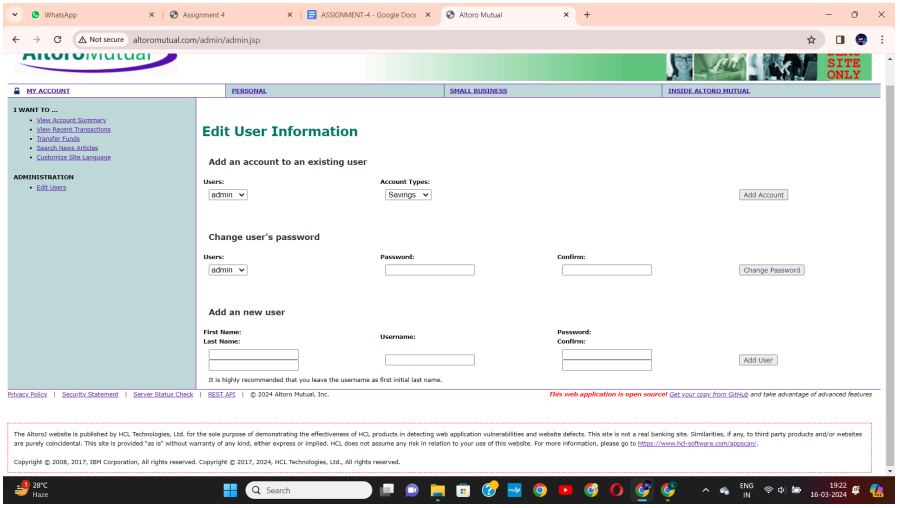
Altro Mutual's website is a comprehensive platform designed to provide various financial services and resources to its users. The website structure typically includes the following components:

Homepage: The homepage serves as the main landing page for the website, providing an overview of Altro Mutual's services, promotions, and important announcements. It may feature links to key sections such as account login, registration, services offered, contact information, and more.
Account Management Section:

- Login Page: This page allows registered users to log in to their accounts using their credentials.

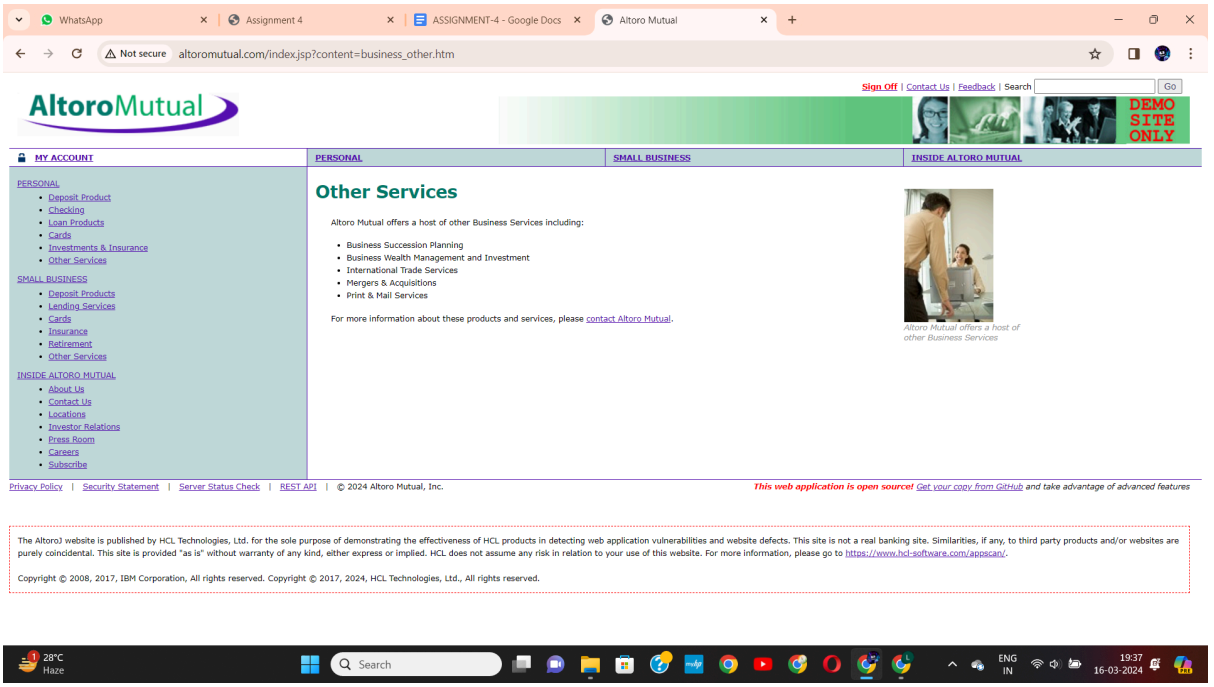


- User Registration: New users can create accounts by providing necessary information such as name, email address, password, and other relevant details.




- Profile Management: Once logged in, users can manage their profiles, update personal information, change passwords, and configure account settings.

Financial Services:

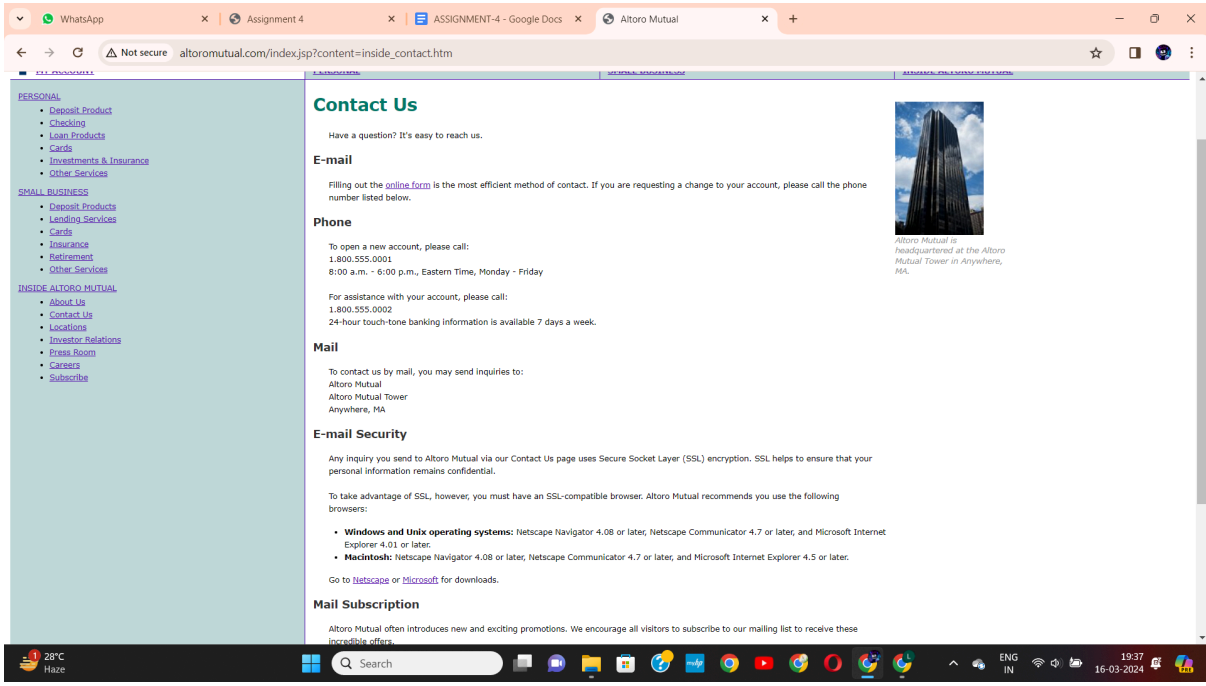





- Payment Portal: This section enables users to make payments, transfer funds, pay bills, or manage transactions.
- Investment Management: Users may have access to tools and resources for managing investments, viewing account balances, analysing market trends, and making investment decisions.
- Loan Applications: Users can apply for loans, view loan terms, calculate loan repayments, and track the status of their loan applications.

Customer Support:

- Contact Forms: Users can reach out to customer support or submit inquiries through contact forms available on the website.




- FAQs: Frequently Asked Questions (FAQs) section may provide answers to common queries related to services, accounts, policies, and procedures.
- Live Chat: Some websites offer live chat support to assist users in real-time with their questions or concerns.

Security and Privacy:

- Privacy Policy: Altro Mutual may provide a privacy policy outlining how user data is collected, used, and protected.
- Security Measures: Information about security measures implemented by Altro Mutual to protect user data and transactions.

Potential Areas of Vulnerability:

Insecure Authentication Mechanisms:

- Weak password policies may make user accounts susceptible to brute-force attacks.
- Lack of multi-factor authentication (MFA) could increase the risk of unauthorised access.

Injection Attacks:

- Input fields such as login forms, search bars, or contact forms may be vulnerable to SQL injection or cross-site scripting (XSS) attacks if input validation and sanitization are not implemented properly.

Sensitive Data Exposure:

- Inadequate encryption of sensitive data such as login credentials or financial information during transmission could expose users to data interception attacks.

Insecure Direct Object References (IDOR):

- Improperly configured access controls may allow users to access restricted resources or perform unauthorised actions by manipulating parameters or URLs.

Insufficient Logging and Monitoring:

- Inadequate logging and monitoring mechanisms may hinder detection and response to security incidents, such as unauthorized access attempts or suspicious activities.

Identifying and addressing these potential vulnerabilities is essential for Altro Mutual to ensure the security and integrity of its website and protect user data and assets. Regular security assessments, vulnerability scans, and adherence to security best practices are crucial for mitigating risks and maintaining a secure online environment for users.

# STEP-4 : Vulnerability Exploitation Demonstration

**Insecure Authentication Mechanisms:**

Scenario: A brute-force attack is simulated using Hydra against the login page of Altro Mutual.

- Demonstration: Configure Hydra to systematically guess passwords until it successfully logs in to a user account. Show how weak passwords or the absence of account lockout mechanisms can lead to successful authentication bypass.

**Injection Attacks (SQL Injection, XSS):**

- Scenario: A SQL injection attack is simulated using SQLMap against the search bar on Altro Mutual's website.
- Demonstration: Run SQLMap against the search functionality, injecting malicious SQL queries to retrieve sensitive data from the database. Show how improper input validation can lead to SQL injection vulnerabilities and potential data exposure.
- Scenario: An XSS attack is simulated using BeEF against the contact form on Altro Mutual's website.
- Demonstration: Inject malicious JavaScript payloads into the contact form using BeEF, which, when executed by an unsuspecting user, can hijack their session or steal their cookies. Show how inadequate output encoding can lead to XSS vulnerabilities and compromise user security.

**Sensitive Data Exposure:**

- Scenario: Network sniffing using Wireshark is simulated to intercept unencrypted login credentials transmitted over the network.
- Demonstration: Set up Wireshark to capture network traffic while logging in to Altro Mutual's website. Show how sensitive information,

such as usernames and passwords, can be intercepted if transmitted without encryption, highlighting the importance of implementing HTTPS.

**Insecure Direct Object References (IDOR):**

- Scenario: Manipulating URLs to access unauthorised resources is simulated on Altro Mutual's website.
- Demonstration: Modify the URL parameters to access restricted resources or view other users' account information. Show how improper access controls and direct object references can lead to unauthorised data access and potential breaches of user privacy.

**Insufficient Logging and Monitoring:**

- Scenario: Unauthorised activities, such as brute-force attacks, are simulated against Altro Mutual's website.
- Demonstration: Conduct a brute-force attack against the login page while monitoring the server logs. Show how insufficient logging and monitoring can result in undetected security incidents and make it difficult to identify and respond to malicious activities effectively.

By simulating these scenarios and demonstrating the exploitation of each vulnerability using proof-of-concept attacks or simulation tools, students can gain practical insights into the potential risks posed by these vulnerabilities and the importance of implementing robust security measures to mitigate them effectively.

# STEP-5 : MITIGATION STRATEGY PROPOSAL

Insecure Authentication Mechanisms:

- Priority: High
- Mitigation Strategy: Implement multi-factor authentication (MFA) immediately to enhance authentication security. Enforce strong password policies with regular password expiration and account lockout mechanisms. Conduct security awareness training for users to encourage the use of strong, unique passwords.

Injection Attacks (SQL Injection, XSS):

- Priority: High
- Mitigation Strategy: Conduct a comprehensive code review and implement input validation and output encoding to prevent SQL injection and XSS attacks. Utilise parameterized queries and prepared

statements to mitigate SQL injection vulnerabilities. Implement content security policies (CSP) to mitigate XSS vulnerabilities.

Sensitive Data Exposure:

- Priority: High
- Mitigation Strategy: Encrypt sensitive data both at rest and in transit using strong encryption algorithms. Implement HTTPS (SSL/TLS) to encrypt data transmission between clients and servers. Minimise data retention by only storing necessary data and securely disposing of any data no longer needed.

Insecure Direct Object References (IDOR):

- Priority: Medium
- Mitigation Strategy: Implement proper access controls, including role-based access control (RBAC) and access control lists (ACLs), to restrict access to sensitive resources. Use indirect object references and validate user access rights before serving requested resources.

Insufficient Logging and Monitoring:

- Priority: Medium
- Mitigation Strategy: Enhance logging and monitoring capabilities by implementing comprehensive logging of security-relevant events. Set up intrusion detection systems (IDS) and intrusion prevention systems (IPS) to monitor network traffic for suspicious activities. Conduct regular security audits and reviews to ensure the effectiveness of logging and monitoring mechanisms.