**Step 1: -**

1. **Broken Access Control**

Potential Impact: -

Broken access control can affect companies severely by hurting them financially as well as damaging their reputation and business relationships. To prevent broken access control, it's important to implement and validate that your access controls are working properly on a continuous basis.

1. **Cryptographic Failures**

Potential Impact: -

Attackers can get hold of a complete database having thousands of sensitive information, data theft, public listing, breaches, and many critical problems with business-related data.

1. **Injection**

Potential Impact: -

Injections into the circular flow of income are a result of money borrowed by households and firms from different external sources, like financial institutions. However, this additional income does not result in an immediate expenditure.

1. **Insecure Design**

Potential Impact: -

Insecure design vulnerabilities result from non-adherence to security best practices during the design process. Today, it is one of the leading causes of functionality failures, data breaches, broken policies, and tarnished reputations.

1. **Security Misconfiguration**

Potential Impact: -

Security misconfigurations can allow attackers to gain unauthorized access to the networks, systems and data which in turn can cause significant monetary and reputational damage to your organization.

1. **Vulnerable and Outdated Components**

Potential Impact: -

A vulnerable and outdated component is a software component that is no longer being supported by the developer, making it susceptible to security vulnerabilities. Many times, a component has known vulnerabilities that don't get fixed due to a lack of maintainer.

1. **Identification and Authentication Failure**

Potential Impact: -

The failure of a system to identify and/or authenticate leaves the application susceptible to attacks and leaves user accounts/data at risk. Authentication failure poses a stringent and profound threat to an organization's security.

1. **Software and Data Integrity Failure**

Potential Impact: -

Addressing software and data integrity failures is crucial for maintaining the security of web applications. Failure to address these vulnerabilities can result in significant financial losses, reputational damage, legal liabilities, loss of customer trust, etc.

1. **Security Logging and Monitoring Failure**

Potential Impact: -

Insufficient logging and monitoring of systems can impact visibility, incident alerting, login failures, system failures and breaches. This makes it essential to have a fully operational logging and monitoring system to collect logs and give out alerts to Security Operation Center (SOC) staff and administrators.

1. **Server Side Request Forgery (SSRF)**

Potential Impact: -

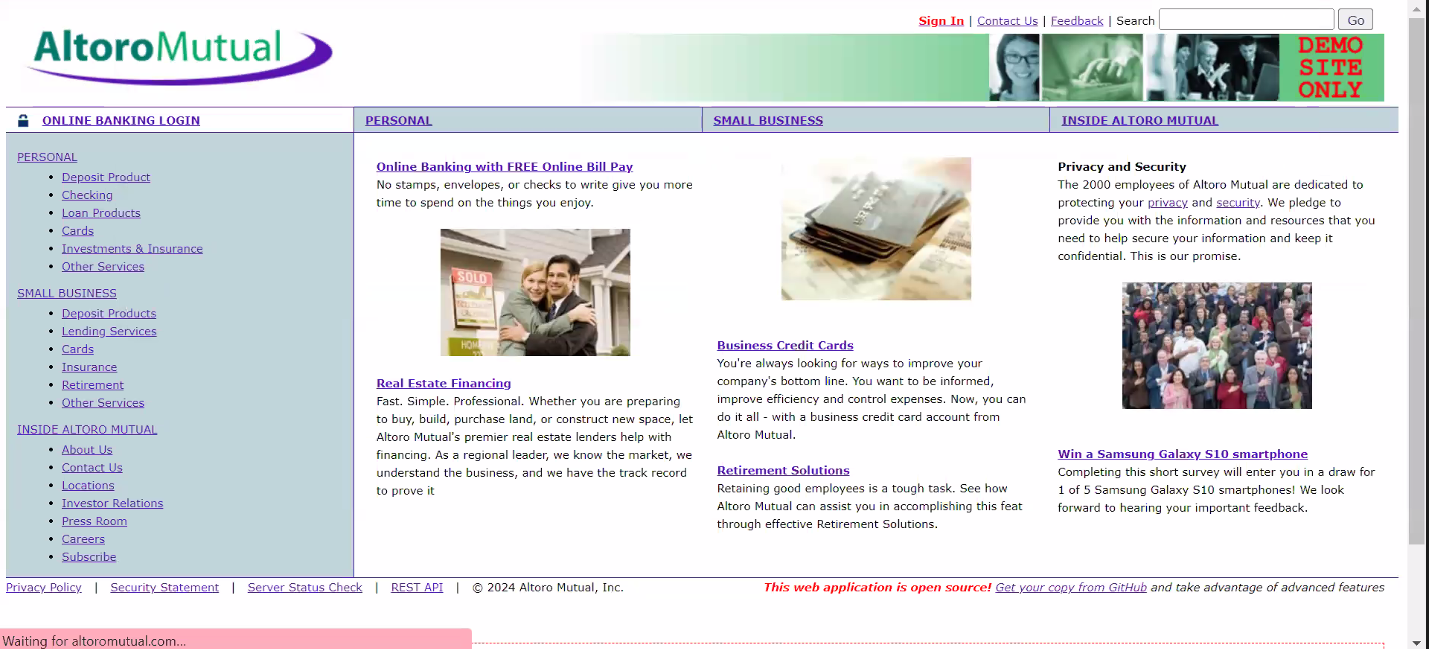
successful SSRF attack can often result in unauthorized actions or access to data within the organization. This can be in the vulnerable application, or on other back-end systems that the application can communicate with.

**Importance of addressing them to prevent exploitation by attackers.**

Addressing OWASP vulnerabilities is crucial to prevent exploitation by attackers and ensure the security of software applications. OWASP (Open Web Application Security Project) identifies and categorizes common security risks in web applications, providing guidelines and best practices for mitigating these vulnerabilities. Failing to address OWASP vulnerabilities leaves applications susceptible to attacks such as SQL injection, cross-site scripting (XSS), and authentication bypass, which can lead to data breaches, financial losses, and reputational damage. By proactively addressing OWASP vulnerabilities through secure coding practices, regular security assessments, and timely patching, organizations can significantly reduce the risk of exploitation and protect sensitive data from malicious actors.

**Step 2: -**

Altro Mutual website

****

**LoginPage**:Allowsregistereduserstosecurelyaccesstheiraccountsbyenteringtheircredentials,typicallyusername/emailandpassword.

**UserRegistration**:Newuserscancreateaccountsbyprovidingpersonalinformationlikenameandemail,andsettingupapassword,oftenwithoptional additionaldetails.

**PaymentPortal**:Enablesuserstomakesecurepaymentsforservicesorproducts,requiringinputofpaymentdetailslikecredit/debitcardinformationinaprotectedenvironment.

**ContactForms**:Userscansendinquiriesorfeedbackbysubmittingtheirname,email,andmessagedetailsthroughaform,sometimeswithoptionalfieldsforadditionalinformation.

**Identification of Vulnerabilities**

**SQL Injection -** SQL injection, also known as SQLI, is a common attack vector that uses malicious SQL code for backend database manipulation to access information that was not intended to be displayed.

**Cross Site Scripting (XSS)** - Cross-site scripting (XSS) is an attack in which an attacker injects malicious executable scripts into the code of a trusted application or website.

**Insecure Authentication Mechanism -**Insecure Authentication exploits vulnerable authentication schemes by faking or bypassing authentication.

**Insecure direct object references -** Insecure direct object references (IDOR) are a type of access control vulnerability that arises when an application uses user-supplied input to access objects directly.

**Step 3: -**

Altoro Mutual Website IP address is**65.61.137.117.** Vulnerability scan is done on Altoro Mutual Website

Starting Nmap 7.40 ( https://nmap.org ) at 2024-03-20 06:26 UTC

Nmap scan report for 65.61.137.117

Host is up (0.046s latency).

PORT STATE SERVICE

21/tcp filtered ftp

22/tcp filtered ssh

23/tcp filtered telnet

80/tcp open http

110/tcp filtered pop3

143/tcp filteredimap

443/tcp open https

3389/tcp filtered ms-wbt-server

**Website Structure and Functionality**

1. **Homepage**:ProvidesanoverviewofAltroMutual'sservices,promotions,andnews updates.
2. **AccountManagement**:Allowsuserstoregister,login,viewaccountdetails,andmanage theirprofiles.
3. **FinancialProducts**:ShowcasesAltroMutual'srangeoffinancialproductssuchassavingsaccounts,loans,andinvestmentoptions.
4. **PaymentandTransactions**:Enablesuserstomakepayments,transfers,andviewtransactionhistorysecurely.
5. **CustomerSupport**: Offers customerserviceoptions such as FAQs, contactforms,andlive chat forassistance.
6. **EducationalResources**:Providesresourceslikearticles,guides,andtoolstohelpusersmakeinformedfinancialdecisions.

**Potential Areas of Vulnerability**

* **InputValidation**:InsufficientvalidationinformsmayleadtoSQLinjectionorXSSvulnerabilities.
* **Authentication:**Weakauthenticationmeasuresorlaxpasswordpoliciescancompromiseaccounts.
* **SessionManagement**:Poorsessionhandlingmayresultinsessionhijacking.
* **DataProtection**: Inadequate encryptionor storagepractices can lead to databreaches.
* **Third-PartyIntegrations**:VulnerabilitiesinpluginsorAPIsposesecurityrisks.
* **AccessControl**:Weakaccesscontrolsmayallowunauthorizedaccesstosensitiveinformationorfunctionality.

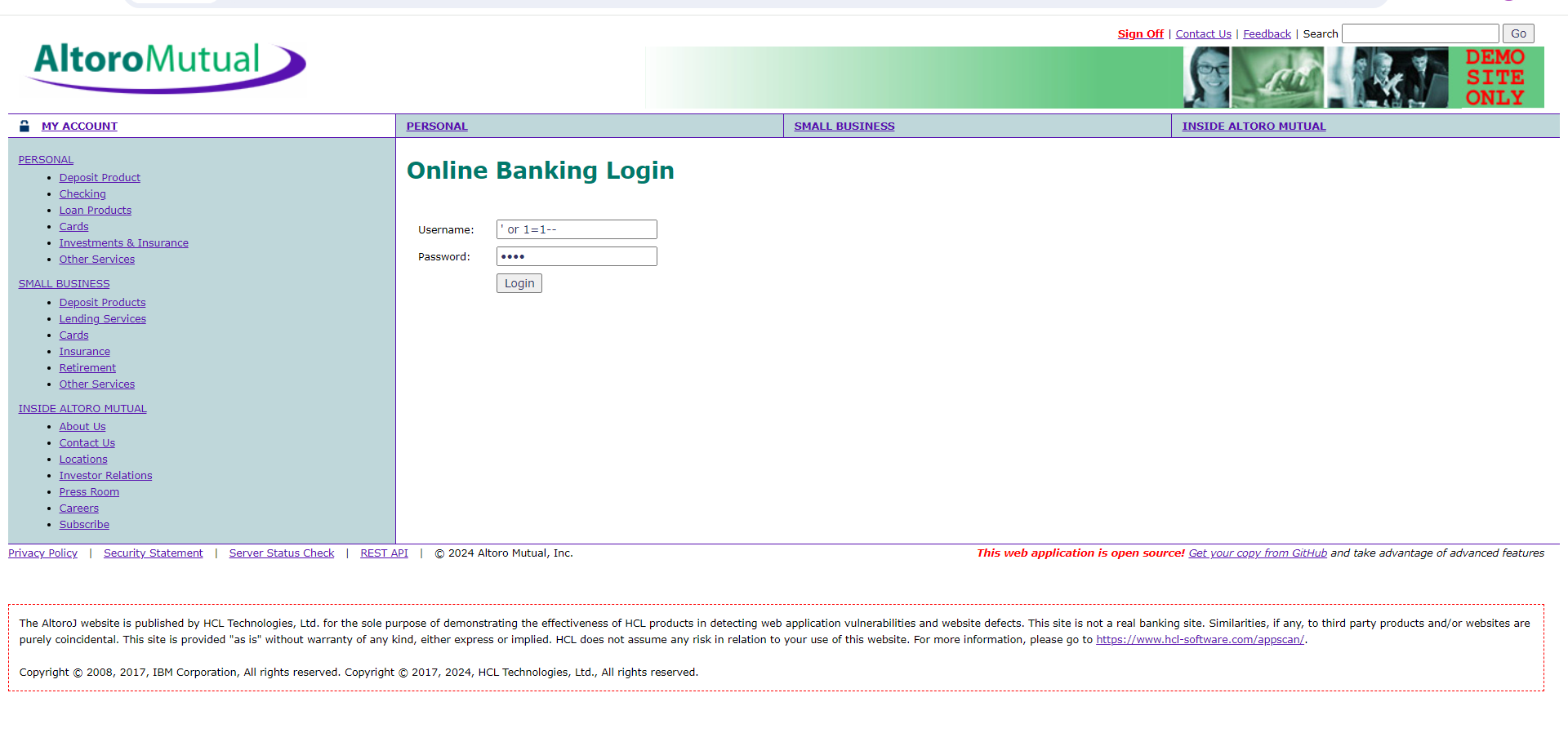
**Recommendations for mitigating Vulnerabilities**

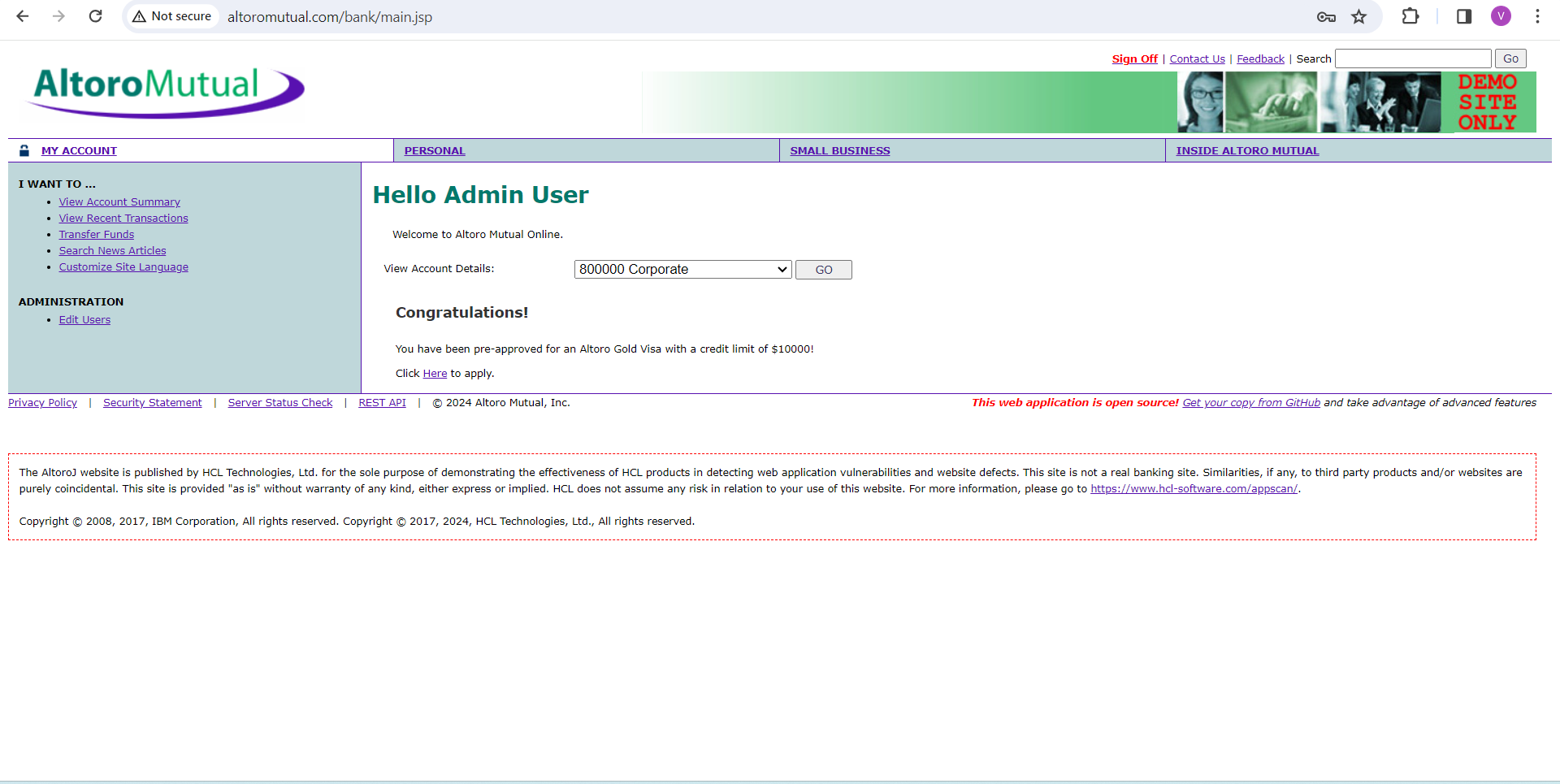
1. Secure Coding Practices: Follow secure coding guidelines such as OWASP's Secure Coding Practices Checklist, which covers topics like input validation, output encoding, authentication, and access control. Use secure coding frameworks and libraries that handle security concerns effectively.
2. Regular Security Assessments: Conduct regular security assessments, including penetration testing and code reviews, to identify and remediate vulnerabilities in your applications. Automated tools like OWASP ZAP (Zed Attack Proxy) can help scan for common vulnerabilities and provide actionable insights.
3. Patch Management: Keep your software and libraries up to date with the latest security patches and updates. Vulnerabilities in third-party components can expose your applications to attacks, so monitor vendor advisories and apply patches promptly.
4. Web Application Firewalls (WAF): Implement a WAF to filter and block malicious traffic, including attacks like SQL injection, XSS, and CSRF. Configure the WAF to enforce security policies and monitor for suspicious activity.
5. Secure Authentication and Session Management: Use strong authentication mechanisms (e.g., multi-factor authentication) to protect user accounts from unauthorized access. Implement secure session management practices to prevent session hijacking and fixation attacks.
6. Data Encryption: Encrypt sensitive data both at rest and in transit using strong encryption algorithms. Securely store and manage encryption keys to prevent unauthorized access to encrypted data.
7. Security Headers: Utilize security headers (e.g., Content Security Policy, X-Content-Type-Options, X-XSS-Protection) to enhance the security posture of your web applications. Properly configure these headers to mitigate common attack vectors.
8. Security Training and Awareness: Educate developers, testers, and other stakeholders about OWASP vulnerabilities and best practices for secure software development. Foster a security-aware culture within your organization to prioritize security throughout the software development lifecycle.

**Step 4:-**

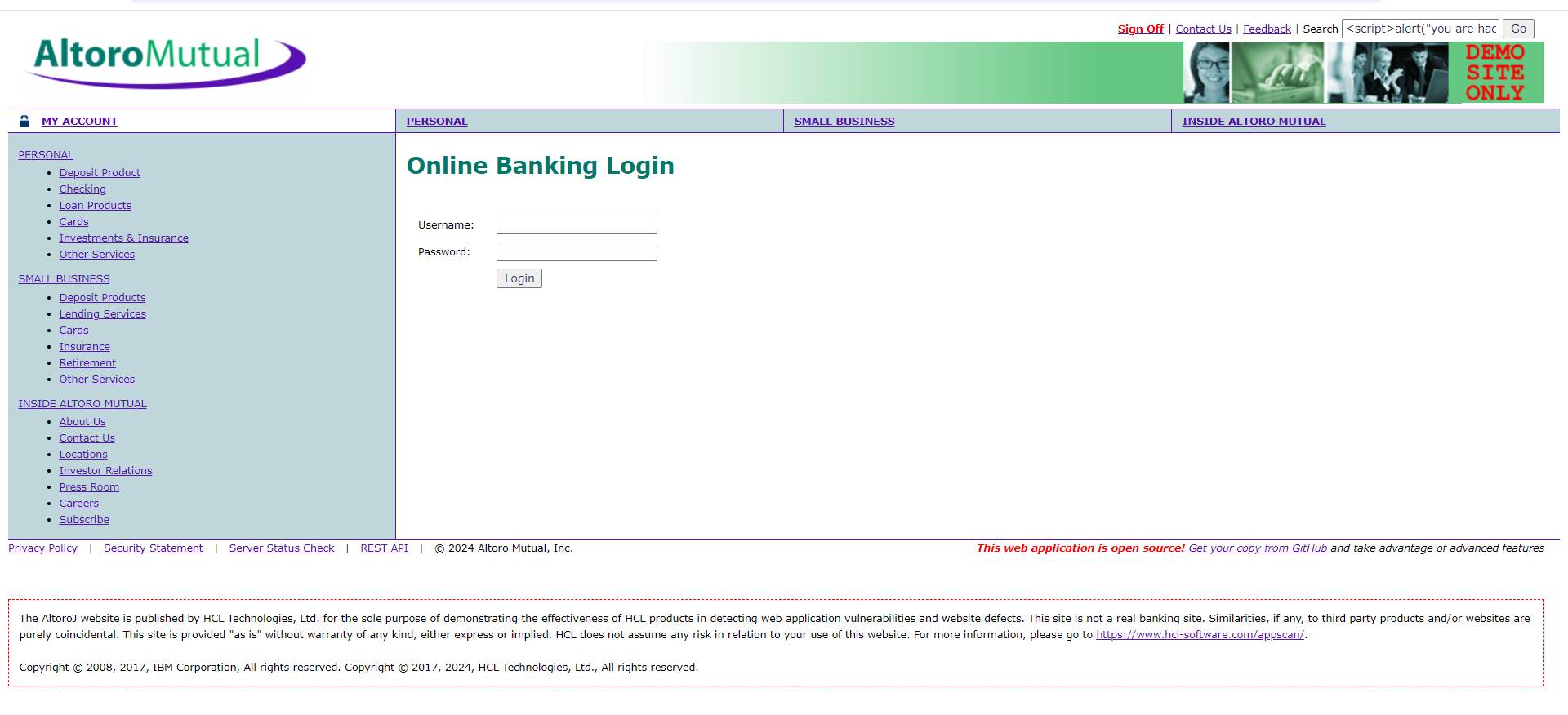
**Vulnerability Exploitation**

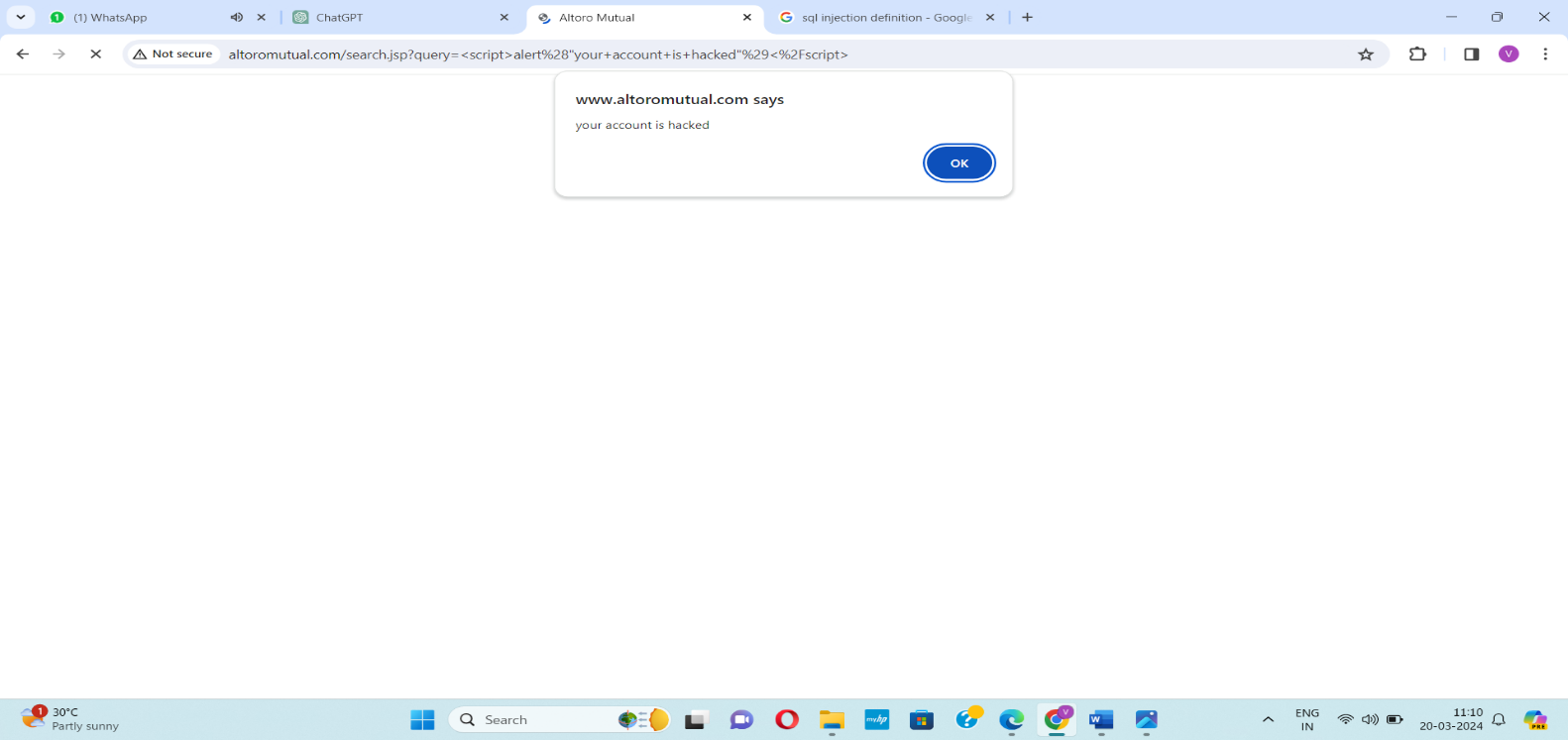
**SQL Injection -** SQL injection, also known as SQLI, is a common attack vector that uses malicious SQL code for backend database manipulation to access information that was not intended to be displayed. Performing SQL Injection on Altoro Mutual Website by giving username as **‘ or 1=1—** and password as **1234** in the login page.



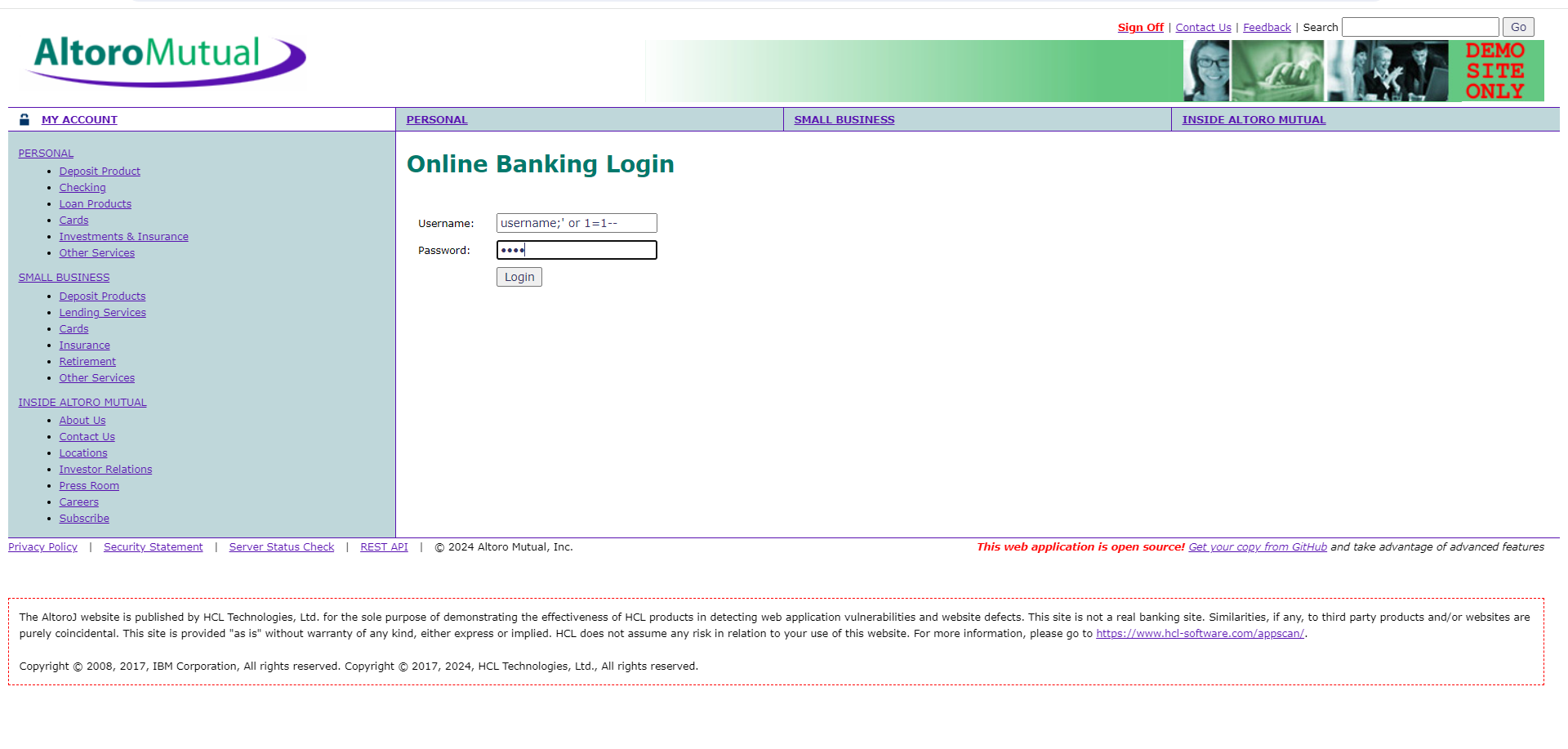


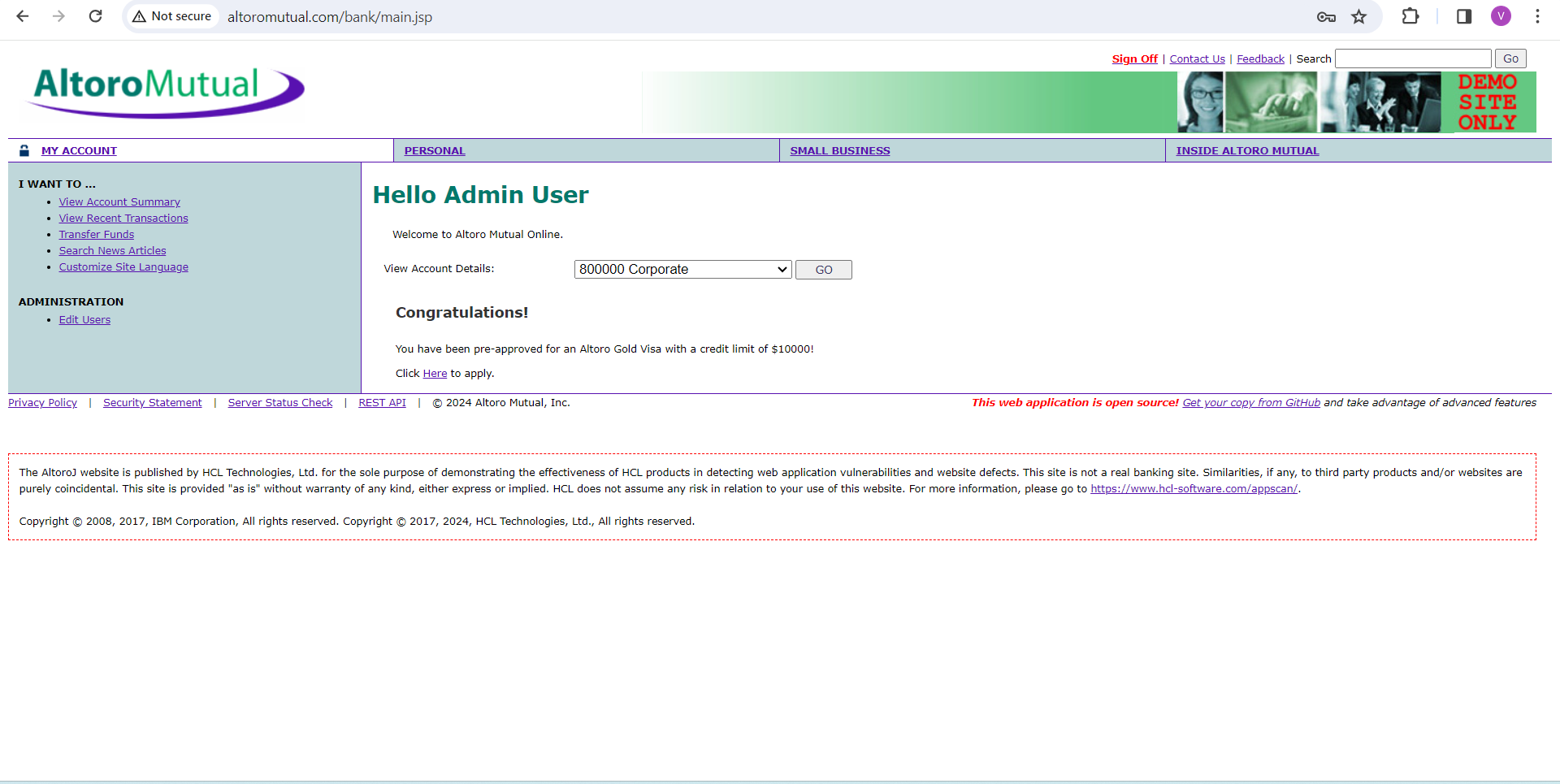
**Cross Site Scripting (XSS)** - Cross-site scripting (XSS) is an attack in which an attacker injects malicious executable scripts into the code of a trusted application or website. Performing XSS on Altoro Mutual Website by using the command **<script>alert(“you got hacked”)</script>** in search bar.



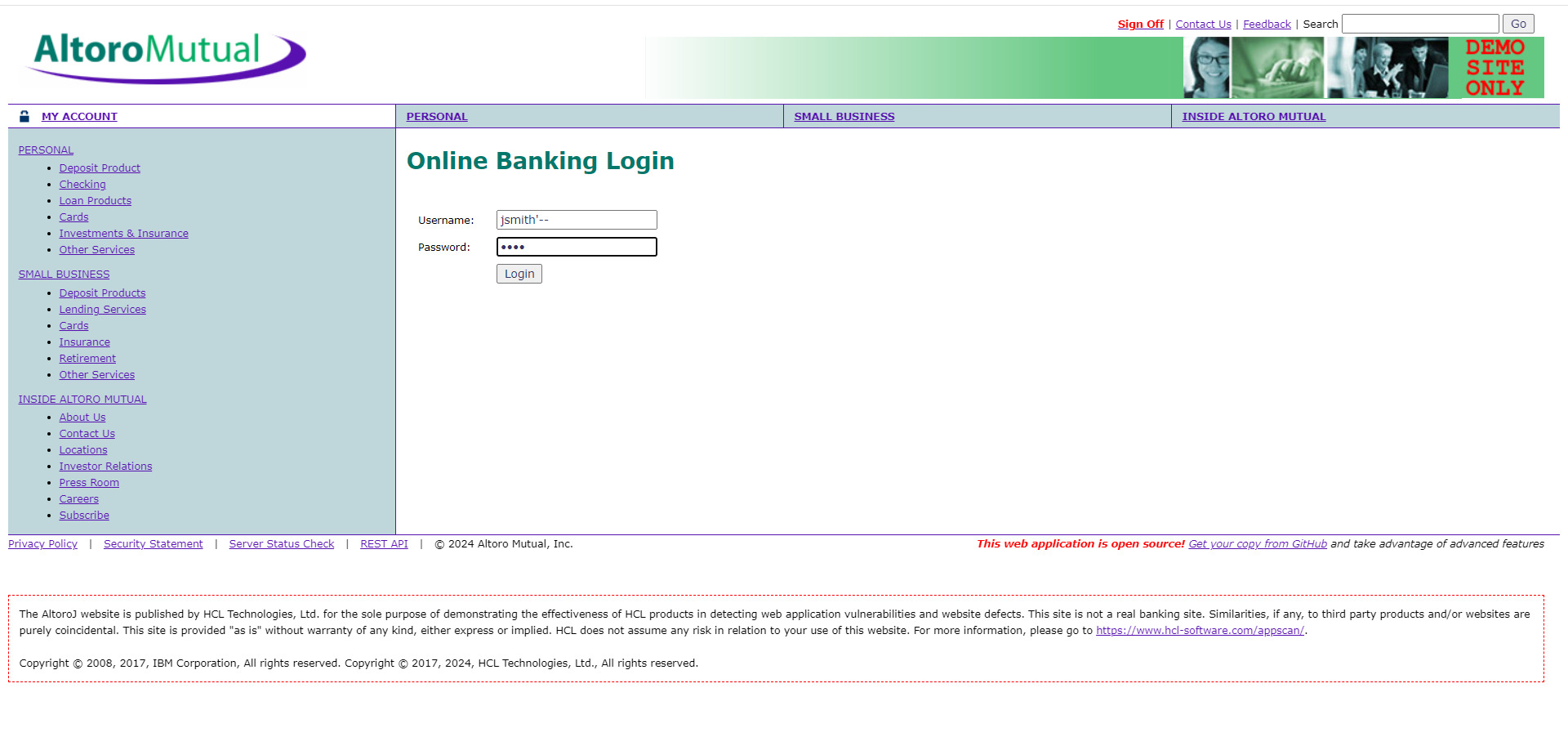


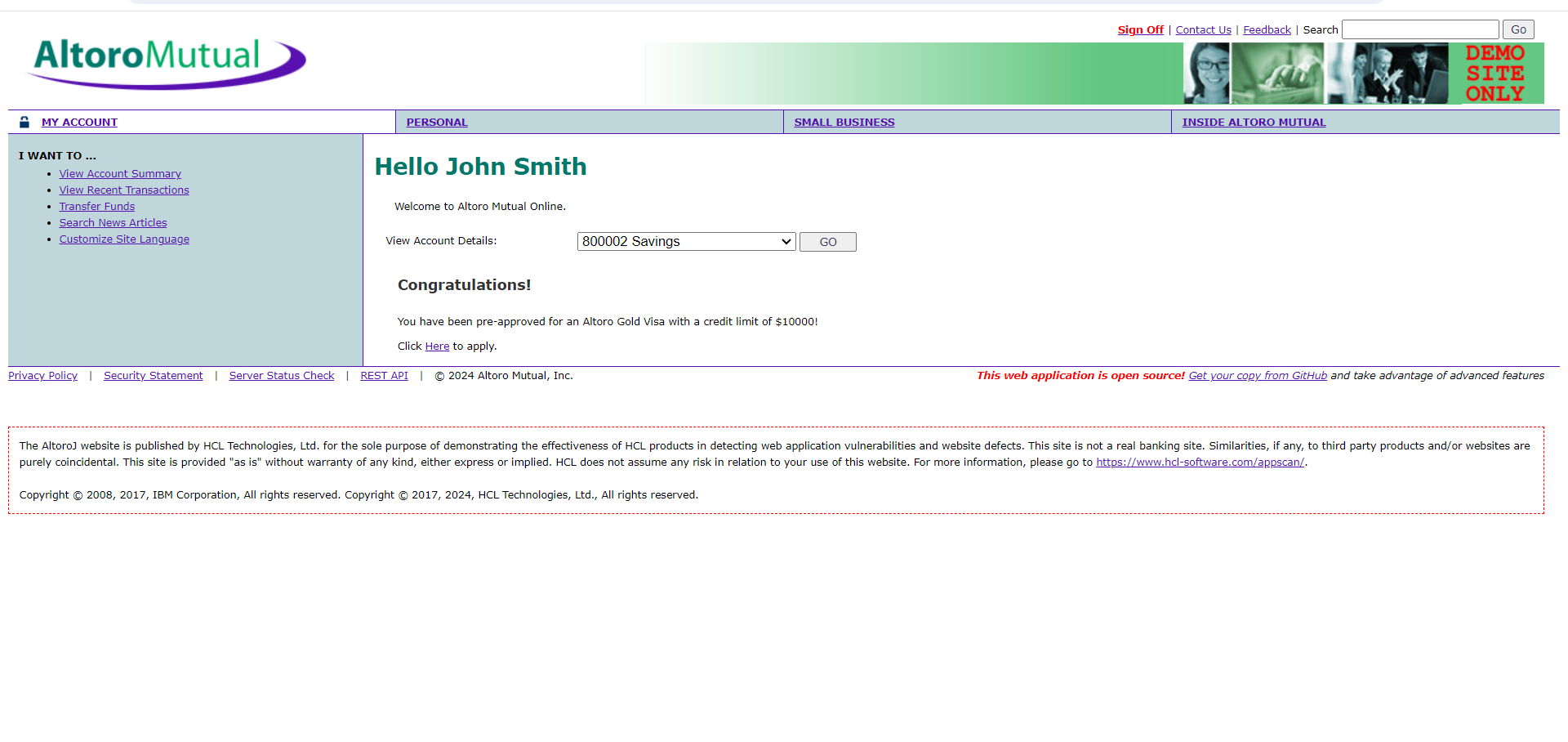
**Insecure Authentication Mechanism -**Insecure Authentication exploits vulnerable authentication schemes by faking or bypassing authentication. Performing Insecure Authentication Mechanism on Altoro Mutual Website by giving username as **username;’ or 1=1--** and password as **1234** in the login page.





**Insecure direct object references -** Insecure direct object references (IDOR) are a type of access control vulnerability that arises when an application uses user-supplied input to access objects directly. Performing IDOR on Altoro Mutual Website by giving username as **jsmith’--**and password as **1234**in the login page, with this attack we can enter into the public bank accounts and do whatever the transactions we want.



****

**Step 5: -**

**Mitigation Strategy Proposal**

1. **IdentifyHigh-RiskVulnerabilities:**Usevulnerabilityassessmentstofindcriticalvulnerabilities.
2. **Assess Risks:**Evaluatetheimpactandlikelihoodofexploitationforeachvulnerability.
3. **Prioritize based on Severity**: Useindustry-standardmetricslikeCVSStorankvulnerabilities.
4. **Establish Criteria**: Setclearcriteriaforprioritizingvulnerabilities,consideringfactorslikesystemcriticalityandpatchavailability.
5. **Patch Management:**Quicklydeploypatchesforhigh-riskvulnerabilitiesandregularlymonitoreffectiveness.
6. **Remediation Plan:**Developaplantosystematicallyaddresshigh-riskvulnerabilities,consideringoperationalconstraints.
7. **Continuous Monitoring:**Keepaneyeontheeffectivenessofmitigationeffortsandadaptstrategiesasneeded.
8. **Communication:**Maintainopencommunicationwithstakeholderstokeeptheminformed aboutprioritizedvulnerabilitiesandmitigationprogress.

**Step 6: -**

Documented the Exploitation process and the commands used in the exploitation and attached the output the respective exploitations.