**CSS**

**Experiment 9**

**Aim:** Perform and Analyze at least two injection attacks

**Theory:**

**Injection Attacks:**

Injection attacks are a type of security vulnerability that occur when an attacker is able to inject malicious code or commands into a computer system or application. Injection attacks refer to a broad class of attack vectors. In an injection attack, an attacker supplies untrusted input to a program. This input gets processed by an interpreter as part of a command or query. In turn, this alters the execution of that program.

The oldest and most harmful attacks against web applications are injections. These may result in data loss, data integrity loss, data theft, service denial, and total system compromise. Insufficient user input validation is typically the main cause of injection vulnerabilities. The huge attack surface of injection vulnerabilities makes them extremely terrifying. Moreover, injection attacks are a form of vulnerability that is well known. This indicates that there are numerous trustworthy tools that are readily available for free that enable even novice attackers to automatically exploit these vulnerabilities.

**Types of Injection Attacks:**

* Code Injection
* CRLF Injection
* Cross-site Scripting (XSS)
* Email Header Injection
* Host Header Injection
* LDAP Injection
* OS Command Injection
* SQL Injection (SQLi)
* Blind XPath Injection

**SQL Injection:**

An injection attack known as SQL Injection (SQLi) enables the execution of malicious SQL commands. These commands manage a database server that sits in front of a web application. SQL Injection flaws allow attackers to get around application security safeguards. The full content of a SQL database can be retrieved by getting past authentication and authorisation of a web page or online application. They can also employ SQL Injection to add, change, and delete records in the database.

Every website or web application that makes use of a SQL database, such as MySQL, Oracle, SQL Server, or another one, may be vulnerable to a SQL Injection flaw. Your sensitive data, including customer information, personal information, trade secrets, intellectual property, and more, could be accessed by criminals without your permission. One of the oldest, most common, and most harmful online application vulnerabilities is SQL Injection attacks. There are several types of SQL Injection attacks: in-band SQLi, blind SQLi, and out-of-band SQLi.

For example, consider a login page that takes a user's username and password and checks them against a database of registered users. If the application uses the following SQL query to check the user's credentials:

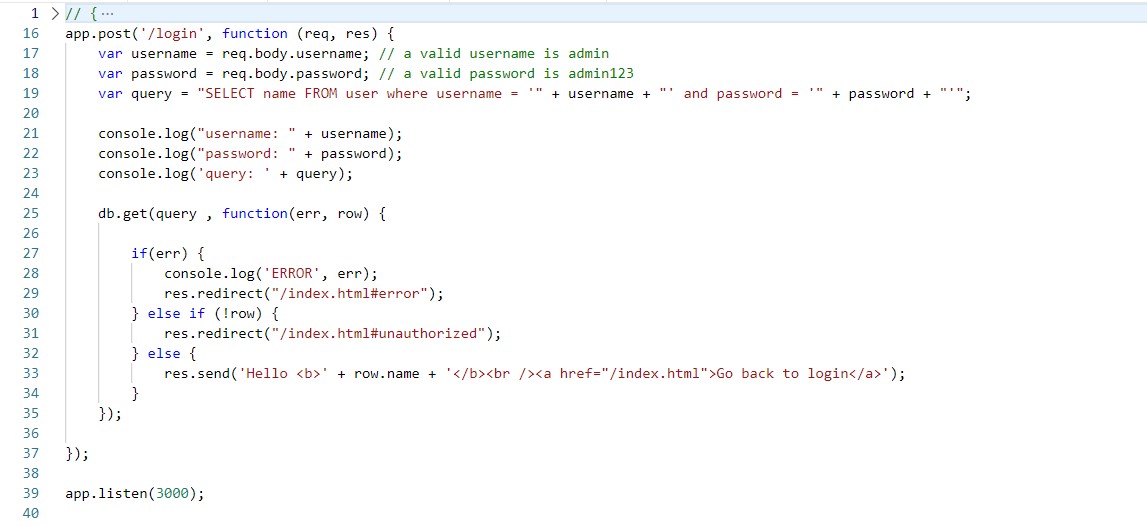
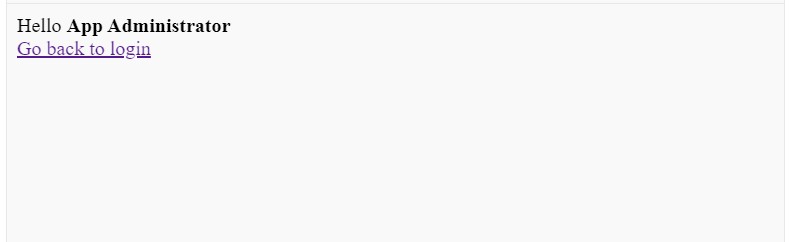
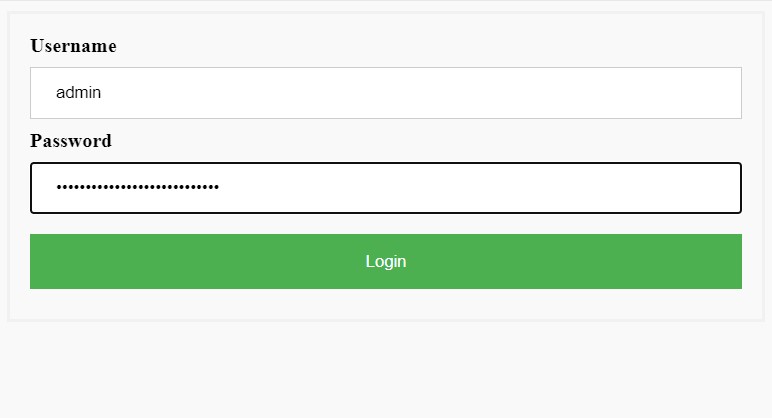
**SELECT \* FROM users WHERE username='$username' AND password='$password'**

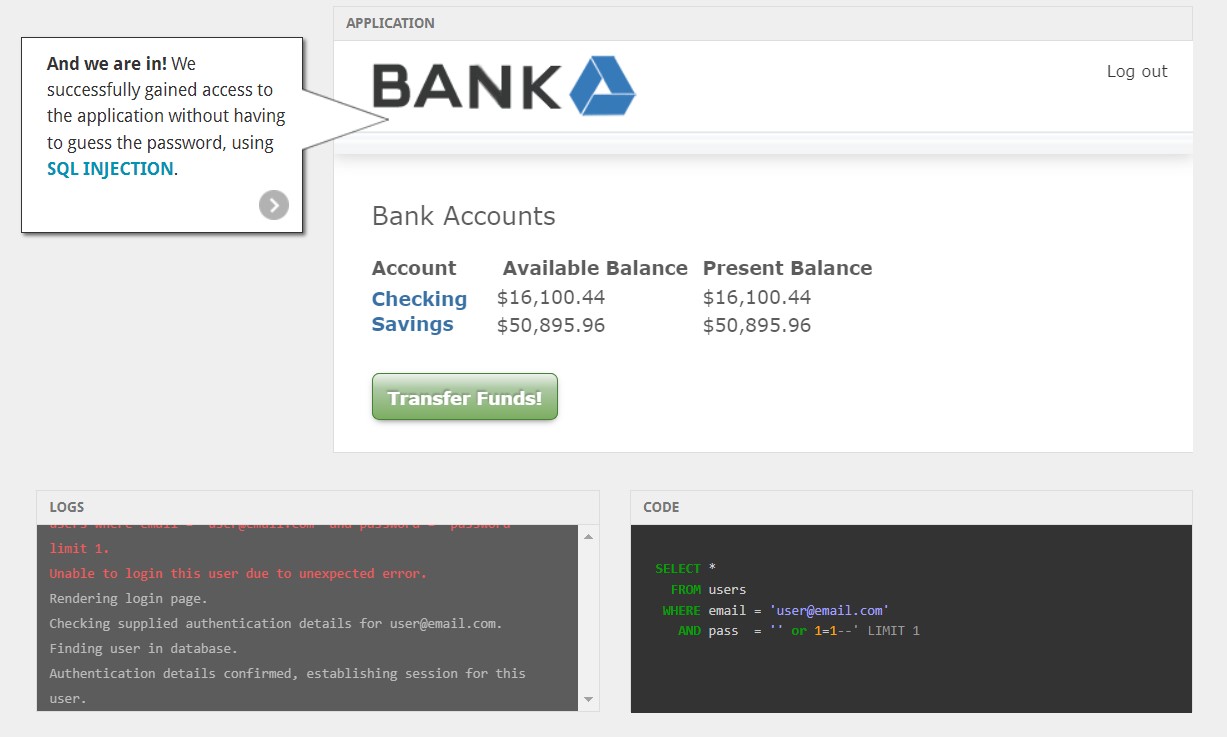
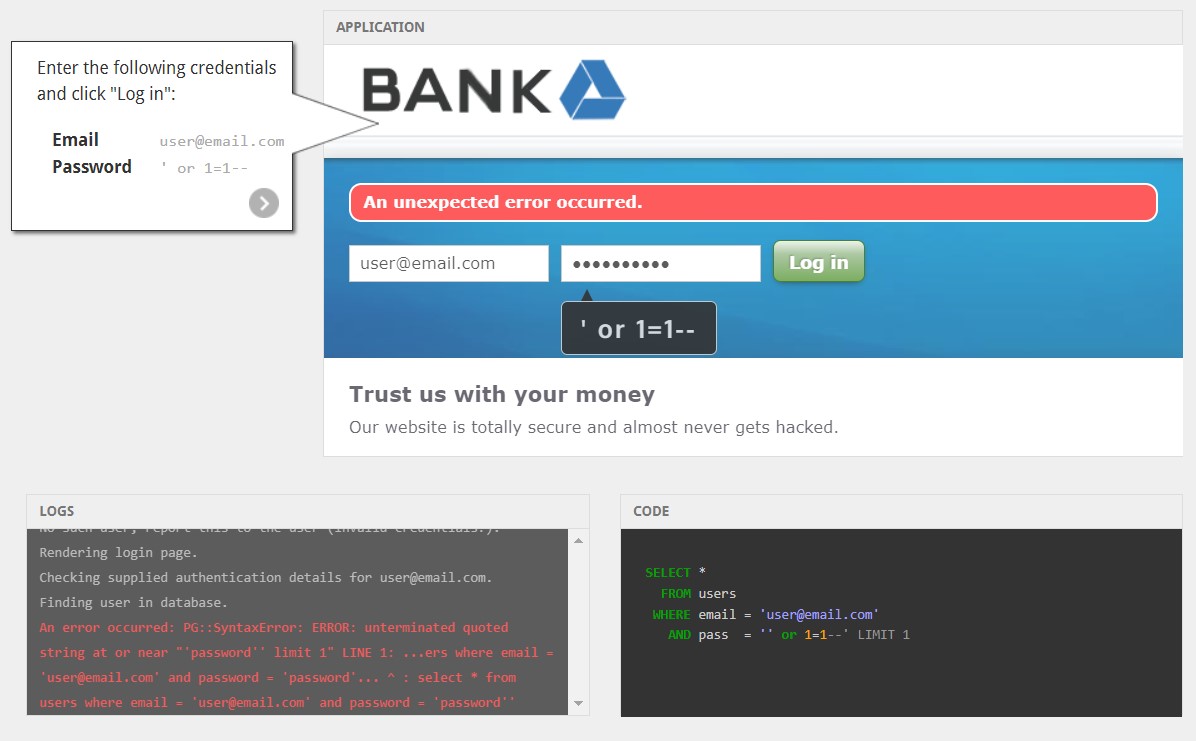
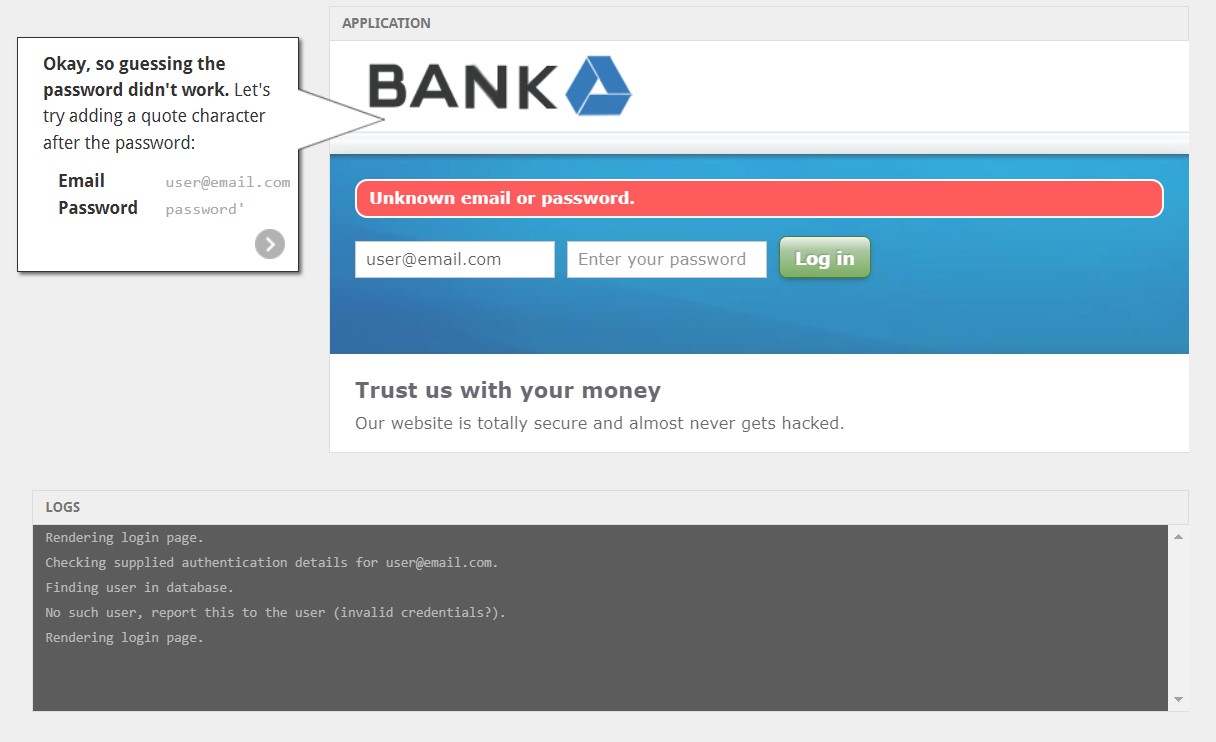
An attacker could inject malicious SQL code into the username or password fields, such as:

**' OR 1=1 --**

This would cause the SQL query to return all rows from the users table, effectively bypassing the login check and giving the attacker access to all user accounts.

Below given are two examples of SQL Injection attacks for two different websites which demonstrate how an attack can be carried out.





**HTML Injection:**

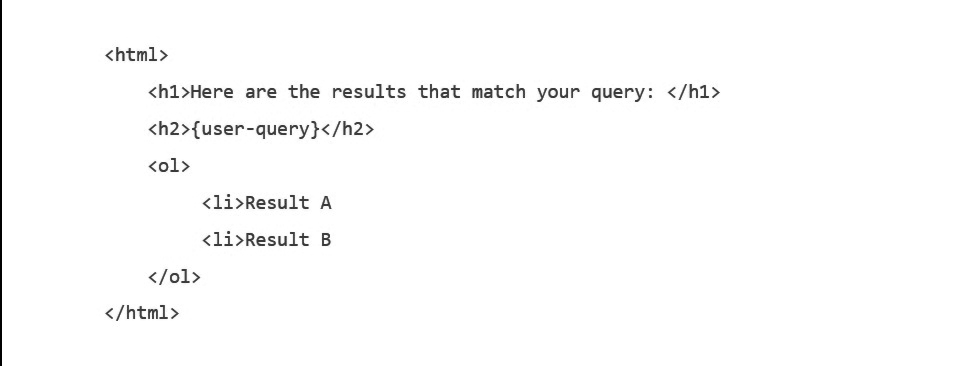
A technique known as Hypertext Markup Language (HTML) injection is used to alter a web page that a web application presents to its consumers by taking advantage of non-validated input. Attackers profit from the fact that a web page's content is frequently connected to past user interactions. When programmes fail to check user data, an attacker can send HTML-fomatted text to modify site content that gets presented to other users. A carefully constructed query can result in the addition of HTML elements under the control of the attacker in the web page, changing how the application content is made available on the internet.

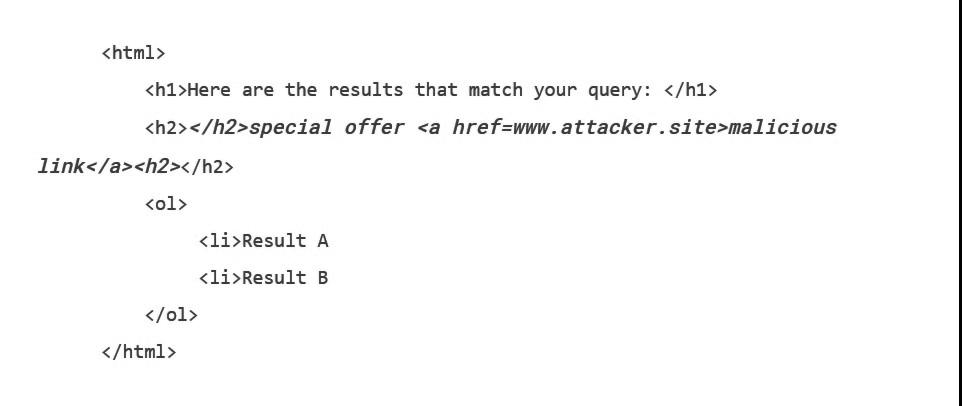
HTML injection attack is closely related to Cross-site Scripting (XSS). HTML injection uses HTML to deface the page. XSS, as the name implies, injects JavaScript into the page. Both attacks exploit insufficient validation of user input.

There are two major types of HTML injection: reflected and stored, similar to reflected XSS and stored XSS:

* In a reflected HTML injection, the payload must be delivered to each user individually (usually as a malicious link) and becomes part of the request.
* In a stored HTML injection, the payload is stored by the web server and delivered later, potentially to multiple users.

A results page often shows the original query text to let the user see the context of these results. If the embedded query text contains syntactically correct HTML, it may add attacker-controlled text, images and links to this generated response page. In the following example, if the application does not validate the user-query before embedding it in the simplified results page, the attacked can add content to the page by sending a query that contains appropriate HTML elements (tags to close and open <h2> context), producing a valid HTML after the injection:





**Conclusion:** We learned about the Injection Attacks and their types and how they can be used by personnel with bad intentions to exploit an organization and get access to important information. We also understood how two types of injection attacks work and analyzed the same. Several concepts related to security were revised while performing the experiment.

For Faculty Use

| **Correction Parameters** | **Formative Assessment [40%]** | **Timely completion of Practical [ 40%]** | **Attendance / Learning Attitude [20%]** |  |
| --- | --- | --- | --- | --- |
| **Marks Obtained** |  |  |  |