

Accessing Webapplication using domain name:

Ex: <https://csye6225-fall2018-panchalk.me:443/ROOT-1/user/register>

Email:kp@gmail.com

The screenshot shows the Postman application interface. On the left, the 'History' tab is active, displaying a list of recent requests. The main workspace shows a POST request to `https://csye6225-fall2018-panchalk.me:443/ROOT-1/user/register`. The 'Authorization' tab is selected, showing 'Inherit auth from parent'. The 'Body' tab is also visible, showing a JSON response: `{"message": "User added successfully"}`. The status bar at the bottom indicates a 200 OK status, 1498 ms time, and 359 B size.

Trying to register the same user again:

Email:kp@gmail.com

The screenshot shows the Postman application interface. On the left, the 'History' tab is active, displaying a list of recent requests. The main workspace shows a POST request to `https://csye6225-fall2018-panchalk.me:443/ROOT-1/user/register`. The 'Body' tab is selected, showing a JSON response: `{"message": "User already exists"}`. The status bar at the bottom indicates a 200 OK status, 211 ms time, and 355 B size.

Creating Transaction:

<https://csye6225-fall2018-panchalk.me:443/ROOT-1/transaction>

The screenshot displays a REST client application with a dark theme. The top bar includes buttons for 'New', 'Import', 'Runner', and 'My Workspace', along with an 'Invite' button and a refresh icon. A search filter is present on the left. The main interface is divided into three sections: a history/collections pane on the left, a request editor in the center, and a response viewer on the right.

**History/Collections Pane:** Shows a list of requests under the 'Today' filter. The requests are:

- POST <https://csye6225-fall2018-panchalk.me:443/ROOT-1/user/register>
- POST <https://csye6225-fall2018-panchalk.me:443/ROOT-1/user/register>
- POST <https://csye6225-fall2018-panchalk.me:443/ROOT-1/transaction/1/attachment>
- POST <https://csye6225-fall2018-panchalk.me:443/ROOT-1/transaction/1/attachment>
- POST <https://csye6225-fall2018-panchalk.me:443/ROOT-1/transaction/1/attachment>
- GET <https://csye6225-fall2018-panchalk.me:443/ROOT-1/transaction>

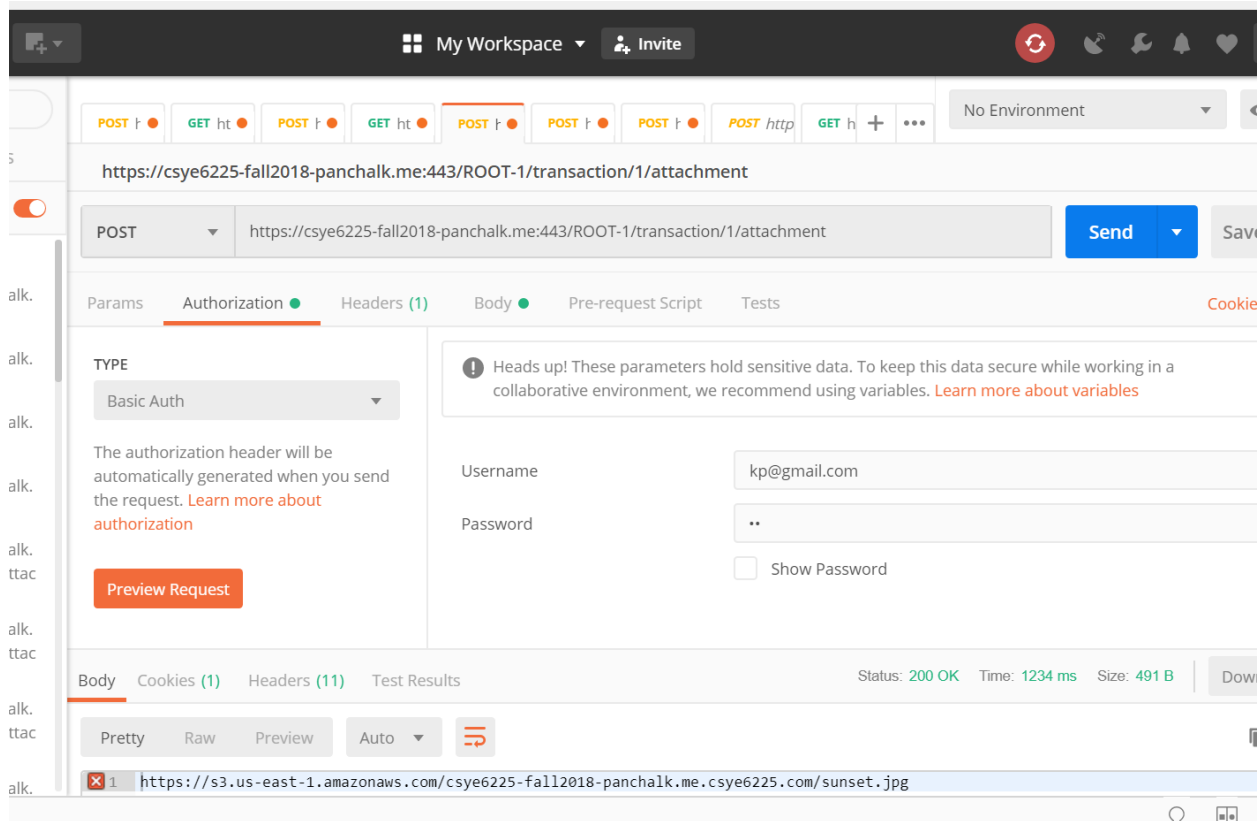
**Request Editor:** The selected request is a POST to <https://csye6225-fall2018-panchalk.me:443/ROOT-1/transaction>. It includes a 'Preview Request' button and a 'Show Password' checkbox.

**Response Viewer:** The response is displayed in the 'Body' tab. It shows a status of 200 OK, a time of 1916 ms, and a size of 447 B. The response body is a JSON object: `{"message": "Transaction added successfully"}`.

## Creating Attachment:

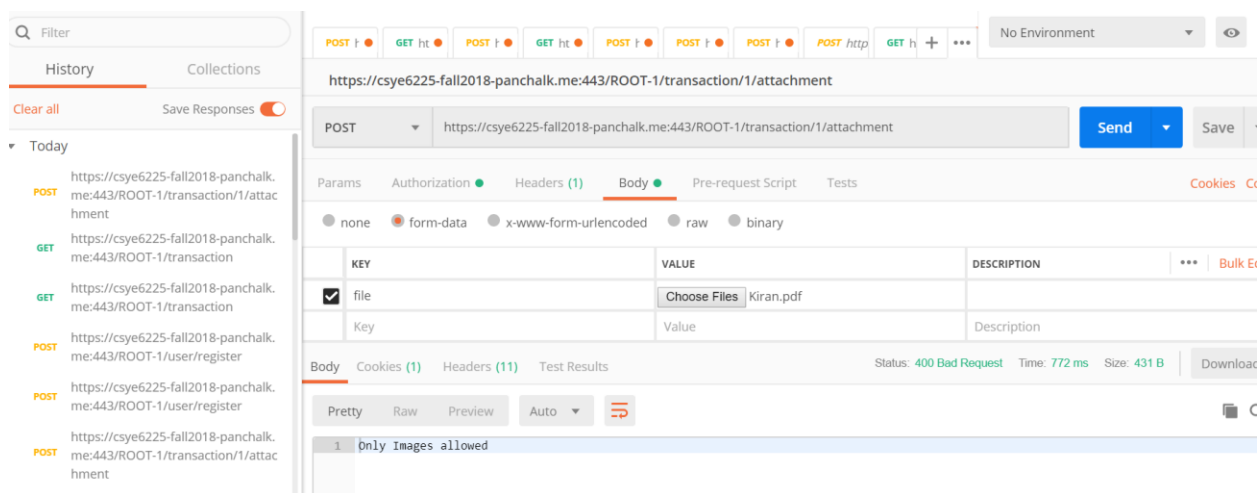
### 1. With the supported file formats.

<https://csye6225-fall2018-panchalk.me:443/ROOT-1/transaction/1/attachment>



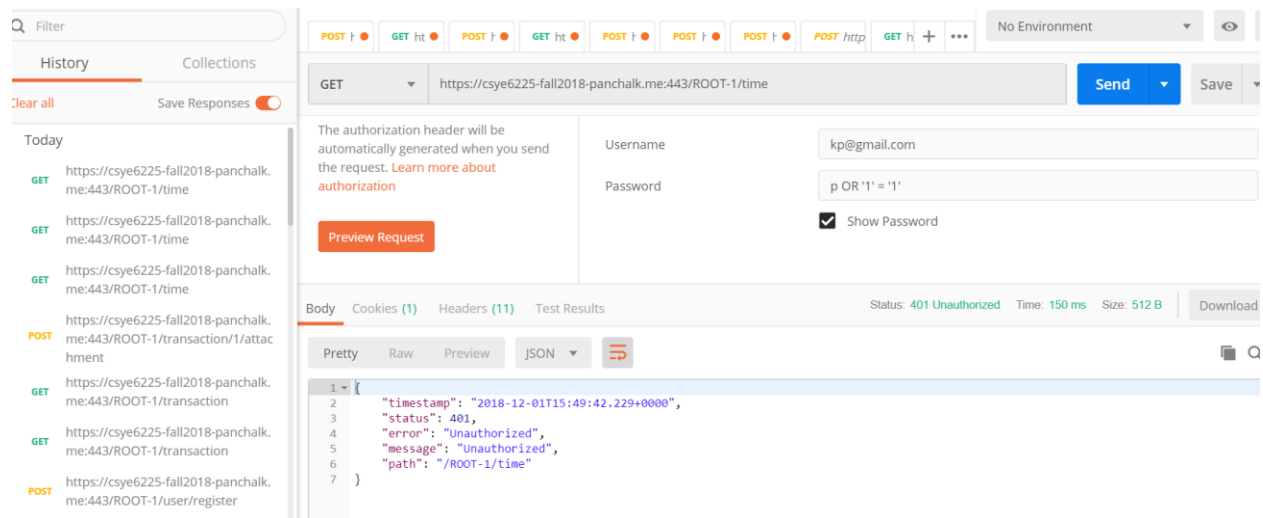
## 2. With the non-supported file formats.

`https://csye6225-fall2018-panchalk.me:443/ROOT-1/transaction/1/attachment`



## Attack Vector

### 1. SQL Injection:



It is a code injection technique for data driven applications in which we test the username and password authentication while in place of username or password when given  $1=1$  (which is always true) the code which is vulnerable to attacks displays the result as when authenticated username and password is passed.

## 2. BlackListed IP Address:


There are many attackers who are constantly looking to hack into applications and are constantly trying to send malicious data/requests to find out the vulnerability in the application. So these IP addresses should be monitored and blocked from doing any type of transaction with the application.

The malicious users can be blocked using AWS WAF, by setting IP addresses or ranges of IP addresses to the block list. Any request/transaction from these IPs are not processed.

Here, IP address is 73.69.60.203/32 which is put under blocked IP list

## Apache Tomcat/8.5.35

If you're seeing this, you've successfully installed Tomcat. Congratulations!

 [tomcat logo]

### Recommended Reading:

[Security Considerations HOW-TO](#)

[Manager Application HOW-TO](#)

[Clustering/Session Replication HOW-TO](#)

[Server Status](#)

[Manager App](#)

[Host Manager](#)

### Developer Quick Start

[Tomcat Setup](#)

[First Web Application](#)

[Realms & AAA](#)

[JDBC DataSources](#)

[Examples](#)

[Servlet Specifications](#)

[Tomcat Versions](#)

### IP match conditions

Create conditionDelete

Filter

US East (N. Virginia) ▾

Viewing 1 to 1

10 ▾

Name

☒ IPSet for blacklisted IP adresse

### IPSet for blacklisted IP adresse ?

Add IP addresses or rangesDelete IP address or range

Filter by IP address or

Viewing 1 to 2 of 2 IP descriptors

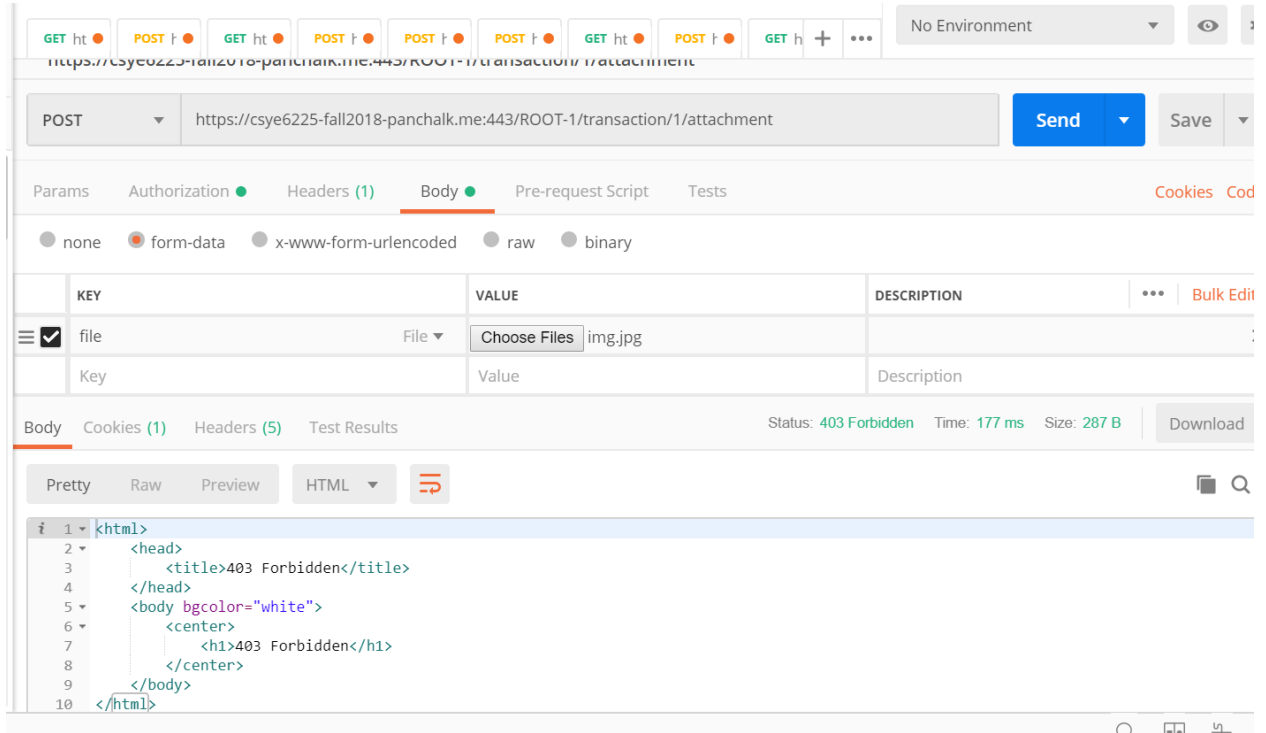
Results per page

10 ▾

| <input type="checkbox"/> IP addresses or range | IP version |
|--|------------|
| <input type="checkbox"/> 73.69.60.203/32       | IPV4       |
| <input type="checkbox"/> 127.0.0.1/32          | IPV4       |

### 3.Attachment maximum size

Attachment maximum size vulnerability test is done to verify set restrictions on the files being uploaded to our S3 bucket. In this test we pass heavy files and in a well written code, these files should not be posted as it induces considerable latency on our instance.



**Conclusion:** With the installation of AWS Web Application Firewall and SSL protocol implementation, we can conclude that the application is reasonably safe from attacks.