# Secure File Upload System Documentation

This documentation provides details about the secure file upload system The system is designed to allow authenticated users to upload files securely while implementing robust security measures to prevent malicious uploads.

### 1. Database Setup

The system uses a MySQL database. Run the following SQL commands to set up the required tables:

```
-- User Table
CREATE TABLE user (
   id INT AUTO_INCREMENT PRIMARY KEY,
   username VARCHAR(255) UNIQUE NOT NULL,
   password VARCHAR(255) NOT NULL
);

-- Uploads Table
CREATE TABLE uploads (
   id INT AUTO_INCREMENT PRIMARY KEY,
   user_id INT,
   filename VARCHAR(255) NOT NULL,
    time TIMESTAMP DEFAULT CURRENT_TIMESTAMP);


-- Logs Table
CREATE TABLE logs (
   id INT AUTO_INCREMENT PRIMARY KEY,
```

```
user_id INT,
ip_address VARCHAR(45) NOT NULL,
filename VARCHAR(255) NOT NULL,
time  TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
status INT);
```

these are required tables:

**2. PHP File Upload System**

The PHP code is organized into separate files for clarity. Key features include user authentication, file type validation, and logging.

**3. Configuration**

Update the database connection details and file upload directory in the configuration file:

```
$connect=mysqli_connect("localhost","root","","machinetest");
$targetDir = "uploads/";
```

4. User Authentication

User authentication is implemented to ensure that only authenticated users can upload files.

5. Testing Scenarios

Testing scenarios cover successful file uploads and rejected attempts with different file types and sizes.

6. Logging

Logs record each file upload, including the uploader's IP address, file name, and upload timestamp.

7. Security Headers

The server is configured to send appropriate security headers in the HTTP response to mitigate potential attacks.

8. Testing the System

Instructions for testing the system, including verification of security measures.

=>Localhost/machinetest/register.php

=> login using registered username and password

=> only registered members can login here

=> authenticated users can upload files.

=> Testing scenarios cover successful file uploads and rejected attempts with different file types and sizes.

=> Logs record each file upload, including the uploader's IP address, file name, and upload timestamp.

=> The server is configured to send appropriate security headers in the HTTP response to mitigate potential attacks.

All the codes are attached to the php files with explanation.

Additional Notes

Please note that I have thoroughly tested the system based on the provided scenarios. However, if you have any specific testing requirements, please let me know, and I will address them promptly.

In case of any questions or clarifications needed regarding the documentation or the system itself, please feel free to reach out to me.

Thank you for your time and consideration. I appreciate the opportunity to work on this project and look forward to receiving your feedback.