Secure PHP Authentication Web Server - Project Summary

# Project Name

Secure PHP Authentication Web Server

# Name

Matthew Collins

# Objective of the Project

The goal of this project was to create a simple and secure web server using PHP and Apache. The system allows users to log in with a username and password. Passwords are stored safely using hashing, and only correct logins can access the protected page. The server runs on a virtual machine and is protected with a firewall using iptables.A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

# Implementation Design and Explanation

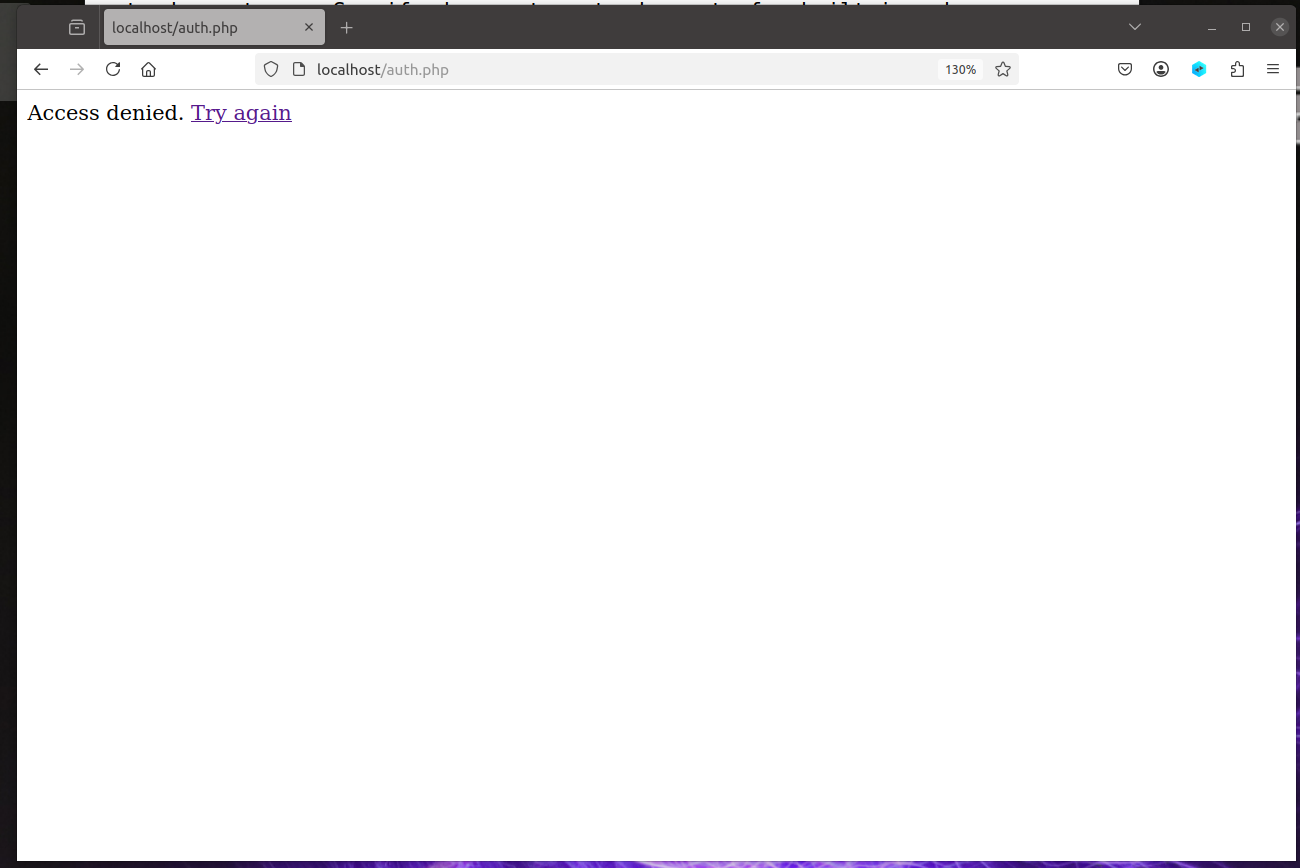
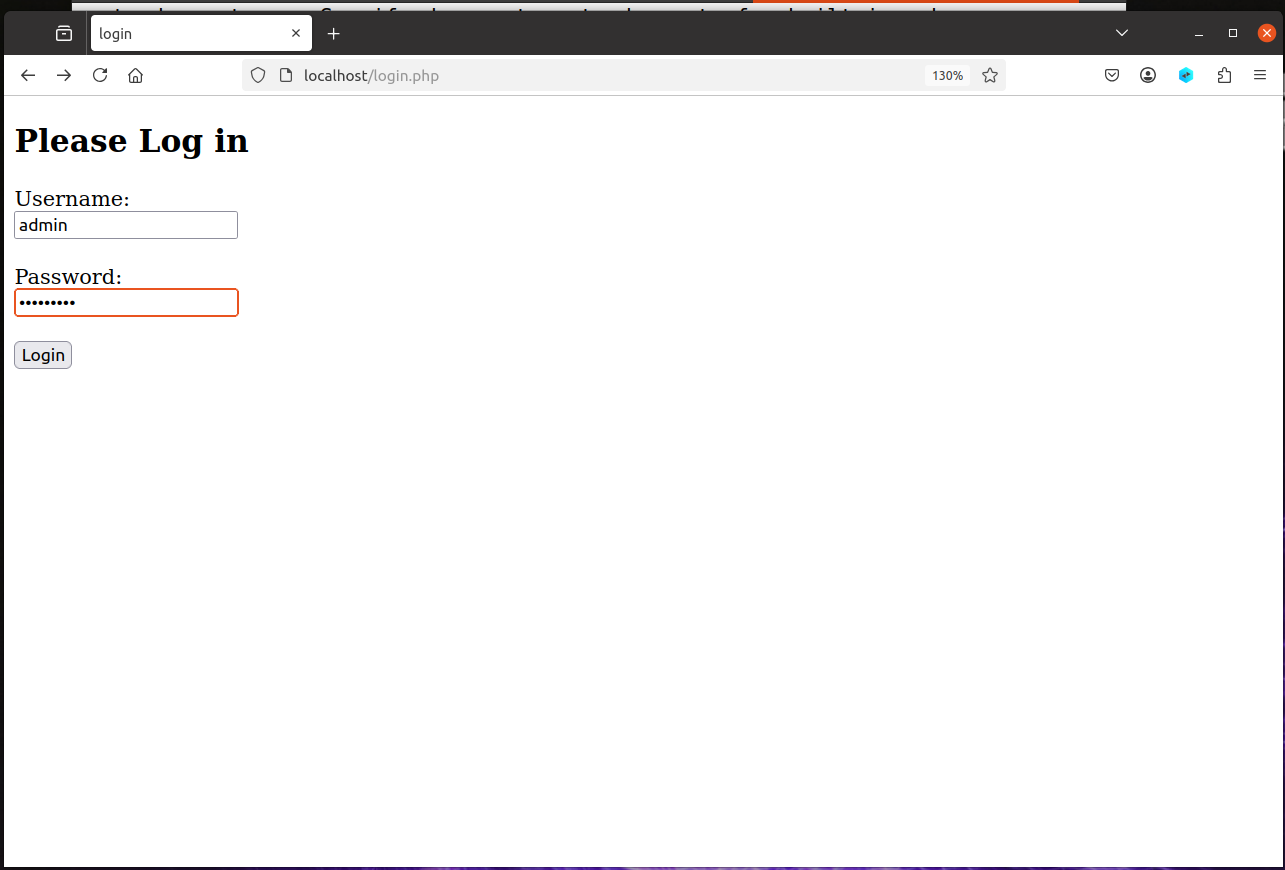
This project uses PHP for handling logins and Apache as the web server. Usernames and hashed passwords are stored in a file called users.txt. Users.txt only allows permission from the webpage and can only be accessed from the root. A login form (login.php) collects user input and sends it to a script (auth.php), which checks if the password is correct using password\_verify(). Password\_verify() is the most important section of code in auth.php. It is responsible for confirming or denying the username and password that is input into the web server. If it is, the user is taken to a private page (index.php).

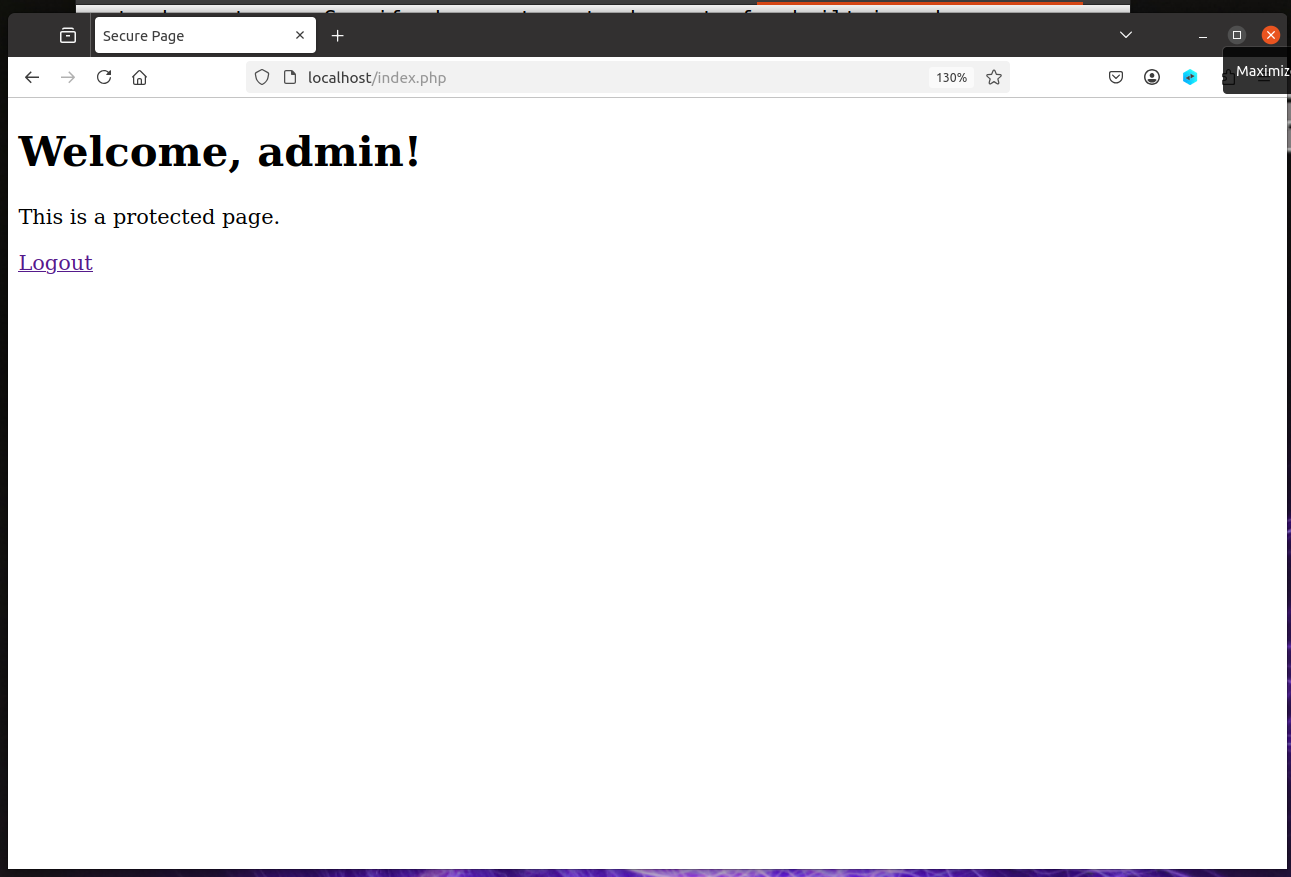
The system runs on a Linux virtual machine, Ubuntu. Apache was configured to work with PHP by switching from mpm\_event to mpm\_prefork. The firewall was set up with iptables to only allow web (HTTP) and SSH traffic. Passwords are hashed using password\_hash(), and the login system uses sessions to keep track of who is logged in.

A computer code on a white background

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.



# Challenges and Limitations

One challenge was getting Apache to work with PHP 7.4. It took many hours of work to complete this portion. I had update the PHP verions in order to correct the issues I had authorizing the logging in to the server. By default, Apache uses a setup (mpm\_event) that doesn't work with PHP. I had to switch it to mpm\_prefork and enable the PHP module to complete the task. This required restarting Apache and installing the correct PHP version. This seemed like a simple task at first, but took much assistance from Chatgpt to finish this issue.

A limitation of this project is that it uses a simple text file to store user’s names and passwords. The hashes could be broken if they were discovered. This works for small projects, but it would not work well for larger systems. It doesn't have features like user registration, password reset, or automatic logouts. A real system would use a database and have more security features.