SECURE PAGE

Presented By : Vishal Naik

Self Project | 2024

INTRODUCTION

Objective: To build a simple and secure login page using Flask.

Key Features:

- **Secure login functionality (username & password).**
- **Password field hidden (input masking).**
- **Flash messages for login success or failure.**
- **Beautiful design with modern CSS.**

PROJECT STRUCTURE

File Structure:

```
/project
 /templates
  login.html
 app.py
```

app.py

Contains Flask app logic and routes.

login.html

Contains the login form and front-end design.

Vishal Naik | 2024

FLASK BACKEND (APP.PY)

Vishal Naik | 2024

Flask Setup:

```
from flask import Flask,
render_template, request,
redirect, url_for, flash
app = Flask(__name__)
app.secret_key =
'your_secret_key'
```

User Credentials:

Dummy user data: Username = 'user', Password = 'password123'.

Login Route:

Handles GET and POST requests.

Checks the entered credentials.

Redirects with a flash message on success or failure.

FRONT-END DESIGN (LOGIN.HTML)

- ## Login Form HTML:

  A form with username and password fields.
  A submit button to trigger login.

- ## Styling:

  Centered login form.
  Gradient background.
  Smooth hover effects on buttons and input fields.

- ## Flash Messages:

  Success and error messages displayed with appropriate colors.
  "Login Successful" or "Invalid Credentials".

CODE EXAMPLE – BACKEND LOGIC

Display the relevant part of app.py for handling form submissions and validation:

```
if request.method == 'POST':
    username = request.form.get('username')
    password = request.form.get('password')
    if username == USER_CREDENTIALS['username']
and password == USER_CREDENTIALS['password']:
        flash("Login Successful!", "success")
        return redirect(url_for('home'))
    else:
        flash("Invalid Credentials", "danger")
        return redirect(url_for('login'))
```

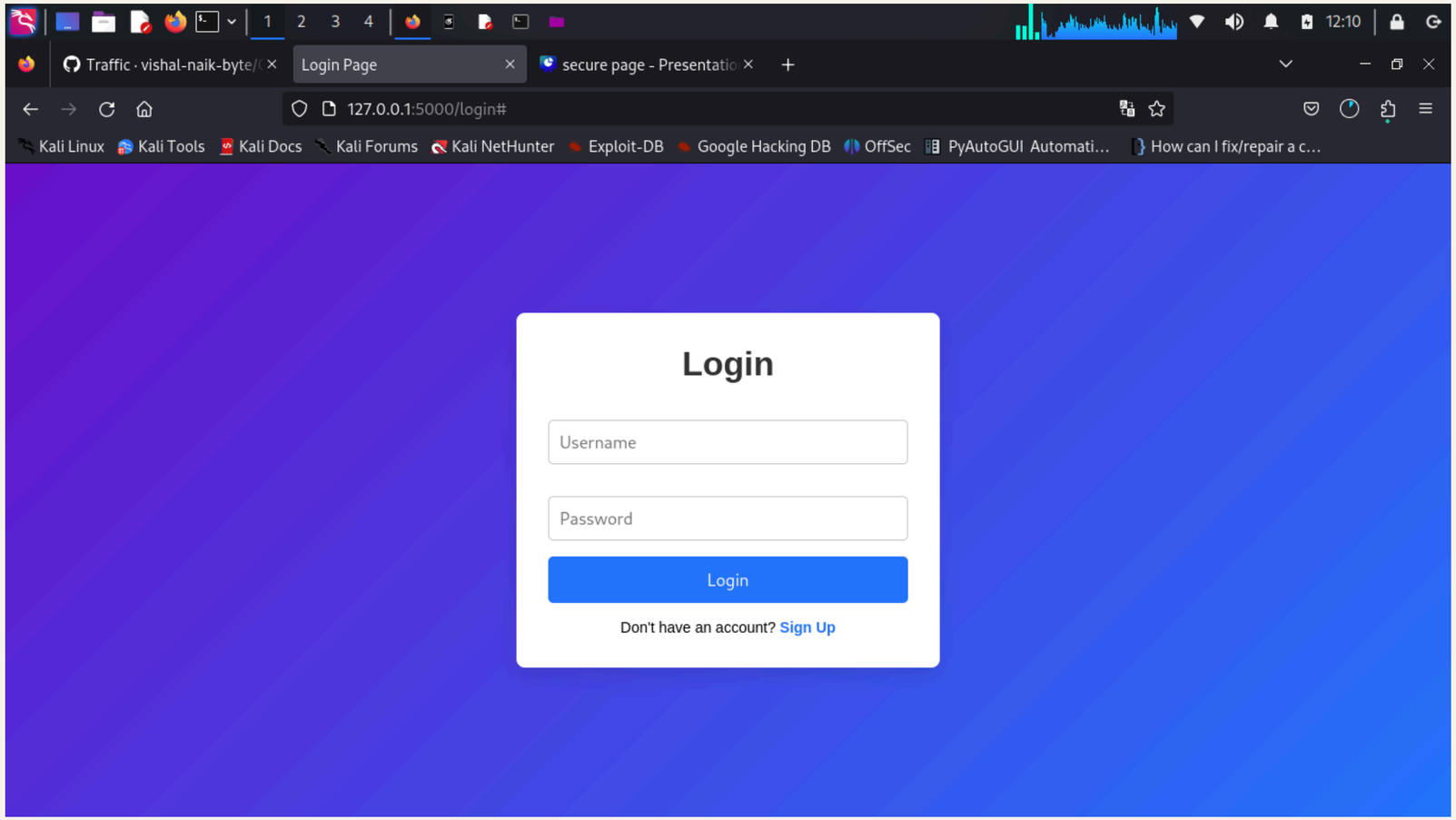
CODE EXAMPLE – FRONT-END (LOGIN.HTML)

Display the relevant part of login.html:

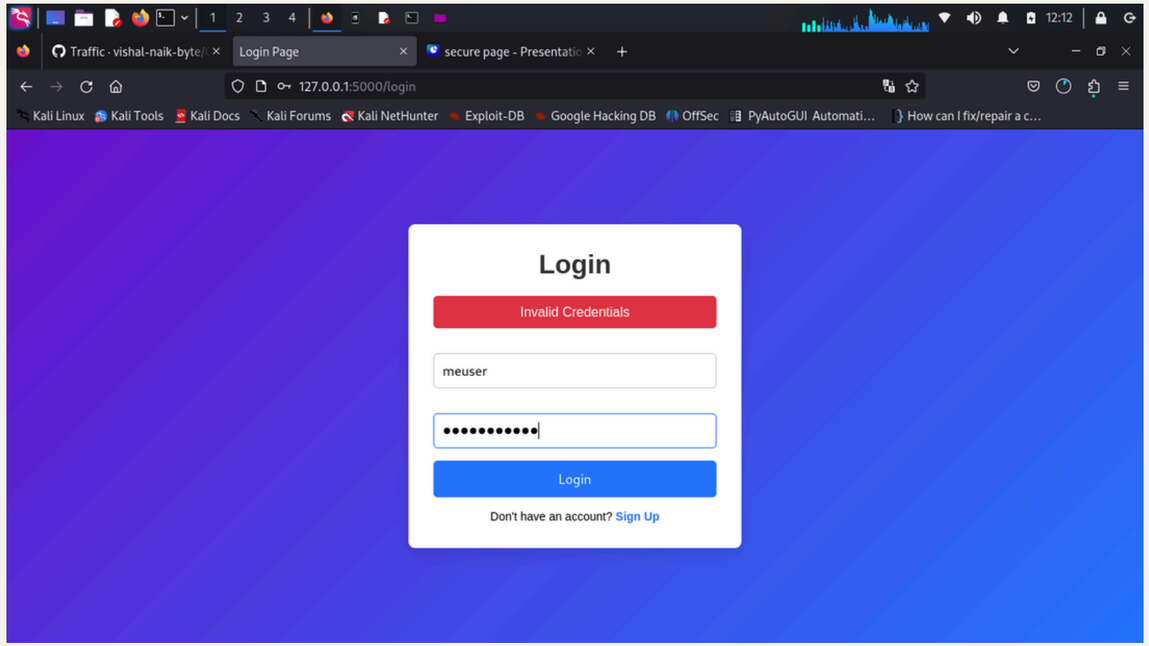
```
<input type="text" id="username"
name="username" class="input-field"
placeholder="Username" required>
<input type="password" id="password"
name="password" class="input-field"
placeholder="Password" required>
<button type="submit" class="btn-
submit">Login</button>
```

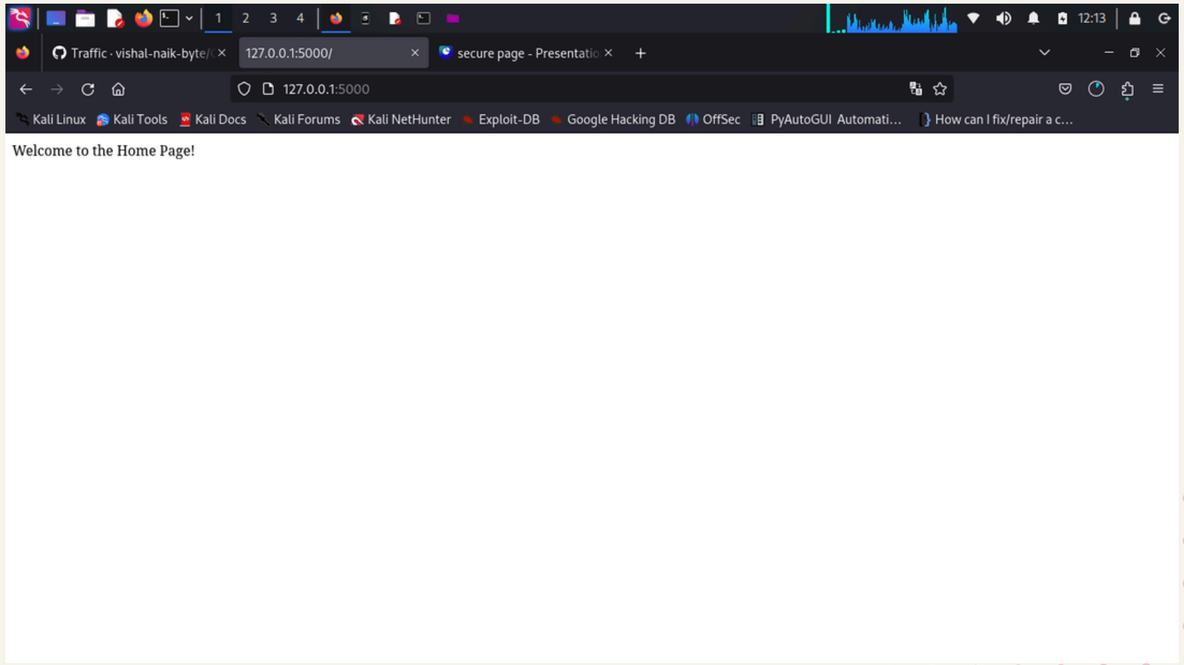
CSS for Design:
Display key styling snippets to show how the form is made visually appealing.

VISUAL


Traffic · vishal-naik-byte/
Login Page
secure page - Presentatio
127.0.0.1:5000/login#
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
OffSec
PyAutoGUI Automati…
How can I fix/repair a c…
Login
Username
Password
Login
Don't have an account? Sign Up

TESTING AND RESULTS





SECURITY CONSIDERATIONS

- Password Masking: Password is hidden when typed.
- Session Management: Flash messages are used for feedback after form submission.
- Storing Passwords: In production, use hashed passwords (e.g., bcrypt).

FUTURE ENHANCEMENTS

- Integrate a database to store user credentials securely.
- Add email verification and password reset functionality.
- Improve security with CAPTCHA to prevent bot logins.
- Style improvements for mobile responsiveness.

CONCLUSION

Summary: We have built a simple yet effective login system using Flask with a modern design and basic authentication.

Next Steps: Continue adding features and security enhancements.

Self Project | 2024

THANK YOU

Presented By : Vishal Naik