Internship Project Report

Secure Login System

Karthik P
Cybersecurity Intern

30 September 2025

Contents

Introduction

The Secure Login System (SLS) is an application running on the web that aims to provide a dependable, secure, and easy-to-use authentication system. In the era of digital communications, when the online exchange of sensitive data happens constantly, a strong login and user verification scheme become indispensable. The project is trying to meet this demand by bringing together different technologies that interoperate to guarantee security as well as to be efficient. The invention was made with a very clear concern for protecting user credentials, verifying identities, and at the same time delivering a smooth login through data integrity not compromised.

Objective

The main objective of this initiative is the creation of a secure user authentication platform that will be able to resist attacks such as password theft, brute-force, and intrusion by automated bots. Also, the team is aimed at developing a system that is not only scalable but also adaptable to real-life scenarios, guaranteeing as such that the organizations or the individuals who deploy the system can rely on it. Yet another target they had was the use of the standard technologies and frameworks for the ease of maintenance and flexible future development.

Project Features

1. User Registration with Google CAPTCHA.
2. Hashed passwords for user security.
3. User Login with JWT session handling.
4. Separate dashboard for admin and user.
5. Admin functionalities:
    - View all users.
    - Delete users.
    - Promote/Demote accounts.
    - Monitor user by login and logout time.

Technology Used

The Secure Login System makes use of a number of cutting-edge tools and frameworks, whereby each of them performs a different function in the project flow:

4.1 Use Python (Flask Framework)

Flask is the main backend framework that is selected on the basis of its light yet powerful nature. It can be very flexible when putting together web applications and also allows other libraries and modules to be integrated in a very easy way. While dealing with the server-side logic, routing and database communication, Flask keeps simplicity for developers.

4.2 MySQL Database

MySQL is the technology used for data persistence. The database primarily holds user details, login credentials, and other related data in a structured format. MySQL is preferred because it is very reliable, efficient in handling large datasets, and very compatible with Python.

4.3 JWT (JSON Web Tokens)

JWT is a system integrated to manage user sessions in a secure, stateless way. In contrast with conventional session storage, JWT authorizes users without keeping session data in memory, hence boosting scalability. At the same time, it guarantees that the tokens are not interfered with, as they are encrypted.

4.4 bcrypt

Security and the privacy of user passwords are on top of the list of priorities. That is why Bcrypt is utilized to hash passwords. Instead of storing plain passwords, Bcrypt employs a computationally costly hashing algorithm which makes it nearly impossible for the attackers to decrypt the passwords even in case of a data leak.

4.5 Google reCAPTCHA

To maintain the system safe from automated bots and brute-force attacks, Google reCAPTCHA is put in the login and registration forms. This makes sure that only real users get access, thereby effectively blocking the scripts or the malicious programs from exploiting authentication endpoints.

Day-wise Progress

Day 1: Project Setup

- Installed VS Code and configured Python environment.
- Created a GitHub repository.
- Initialized folder structure.

Day 2: Frontend Development

- Designed login and registration pages with HTML and CSS.
- Added form fields for username, email, password, and role.
- Implemented basic client-side validation.

Day 3: Backend Setup

- Installed Flask and Mysql.
- Created database anmed `vault`.
- Defined schema for `user` table.

Day 4: User Registration

- Implemented registration route in Flask.
- Used bcrypt for password hashing.
- Implemented duplicate email validation.

Day 5: User Login

- Implemented login route.
- Verified password hashes against stored values.
- Created JWT-based session handling.

Day 6: Role-Based Access Control

- Added role column to users table.
- Implemented admin dashboard and user dashboard.
- Restricted non-admin users from accessing admin routes.

Day 7: Security Enhancements

- Added input validation.
- Integrated Google reCAPTCHA.
- Implemented account lockout after 5 failed attempts.

Day 8: Testing and Debugging

- Tested duplicate registrations and invalid logins.
- Debugged dashboard redirection issues.
- Ensured lockout mechanism was effective.

Day 9: Documentation

- Created README with setup instructions and screenshots.
- Documented challenges faced (DB migrations, reCAPTCHA setup).

Day 10: Final Submission

- Finalized dashboards with proper alignment and styling.
- Uploaded full project to GitHub.
- Prepared final report with code, screenshots, and documentation.

Screenshots

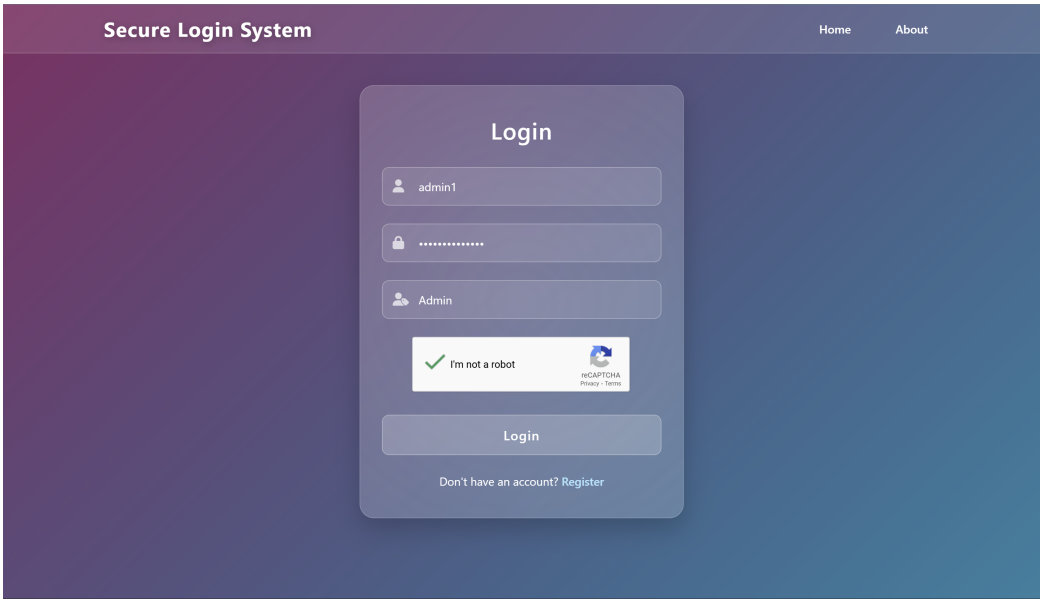

Figure 6.1: Login Page

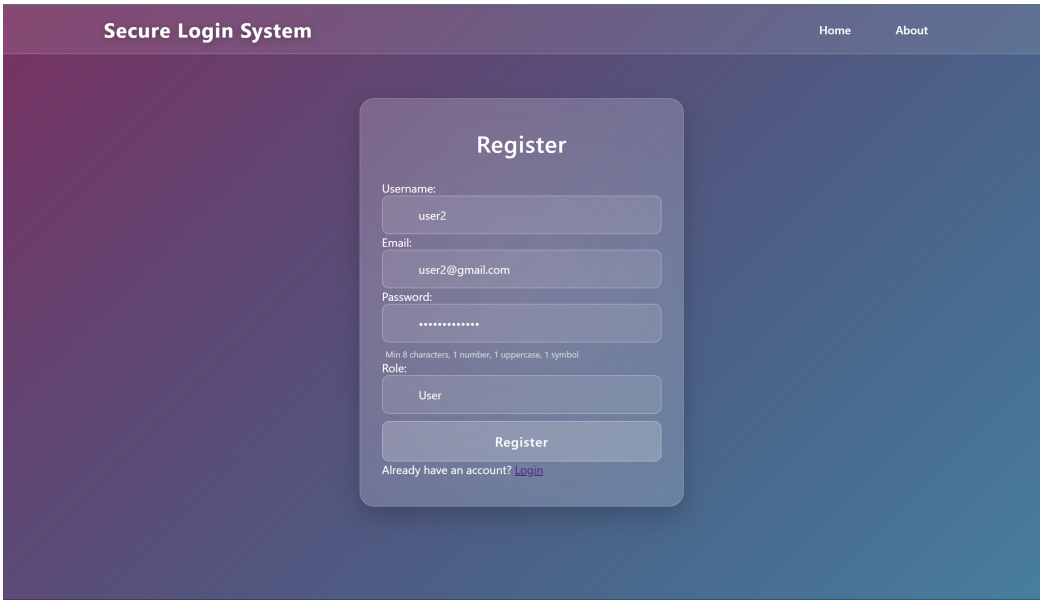

Figure 6.2: Register Page

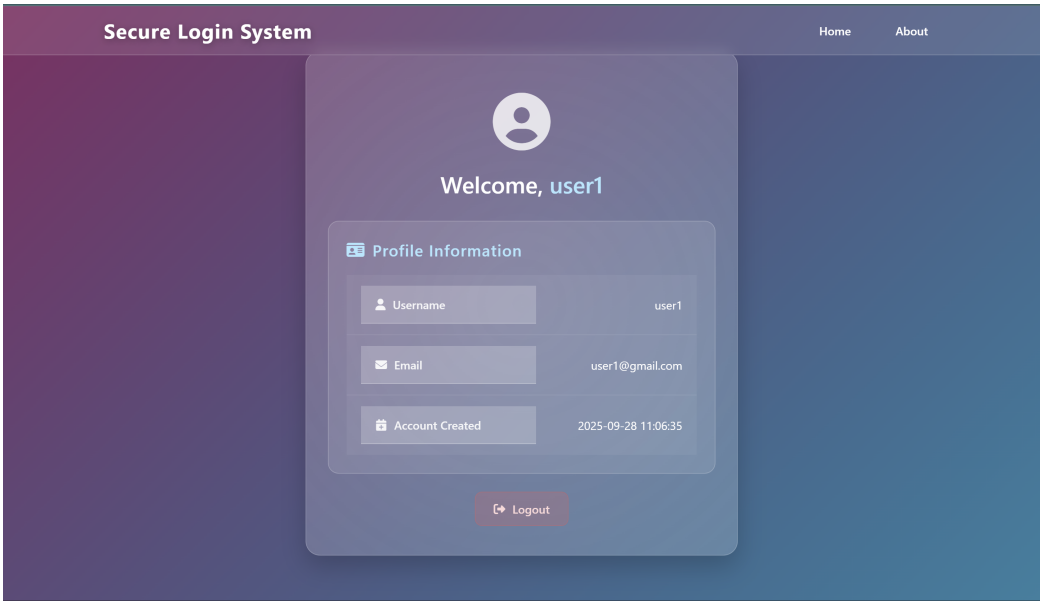

Figure 6.3: User Dashboard

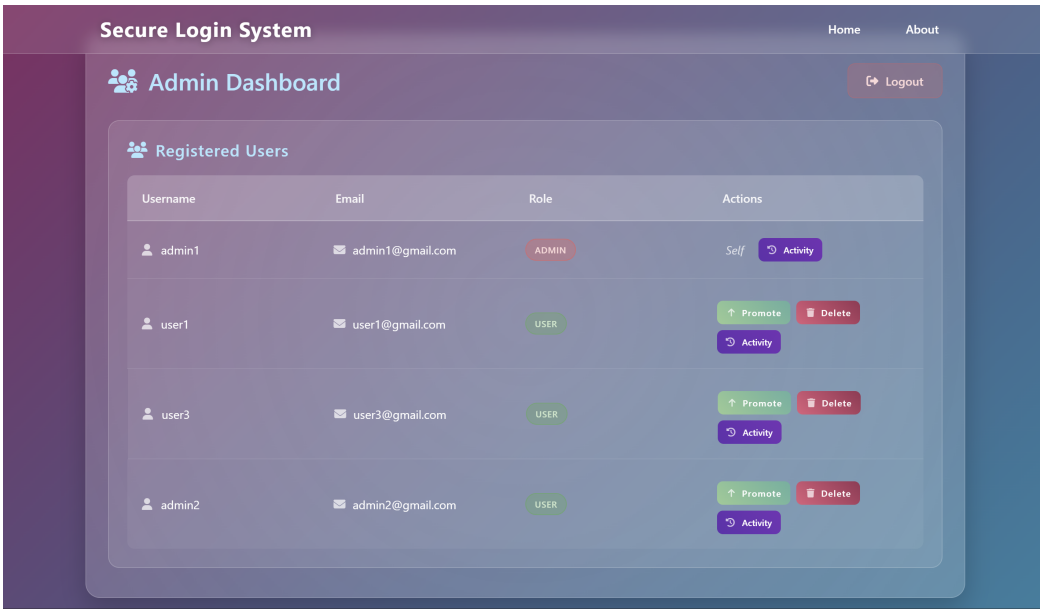

Figure 6.4: Admin Dashboard

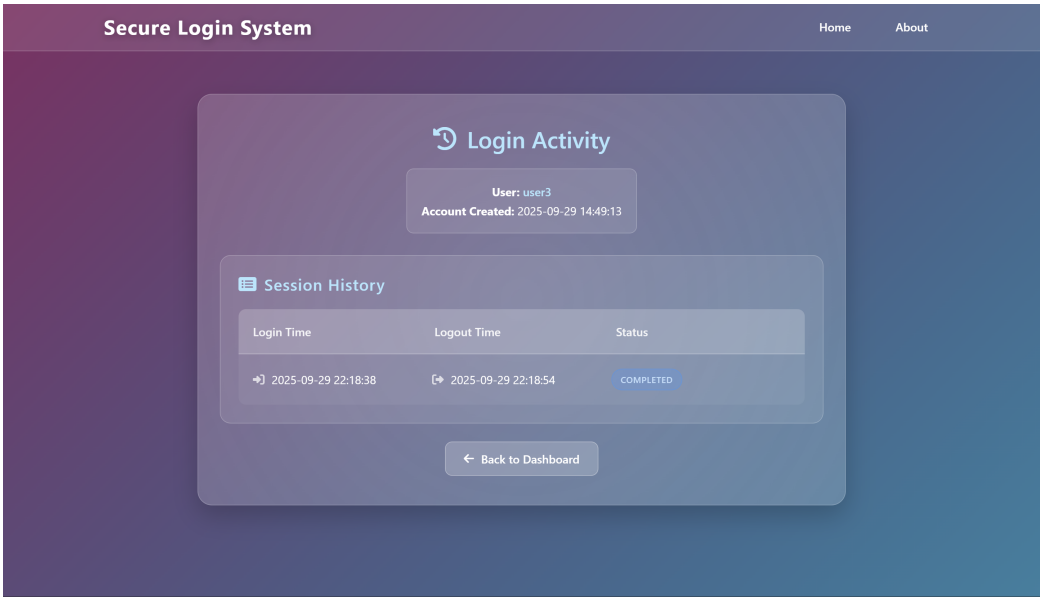

Figure 6.5: Activity Dashboard

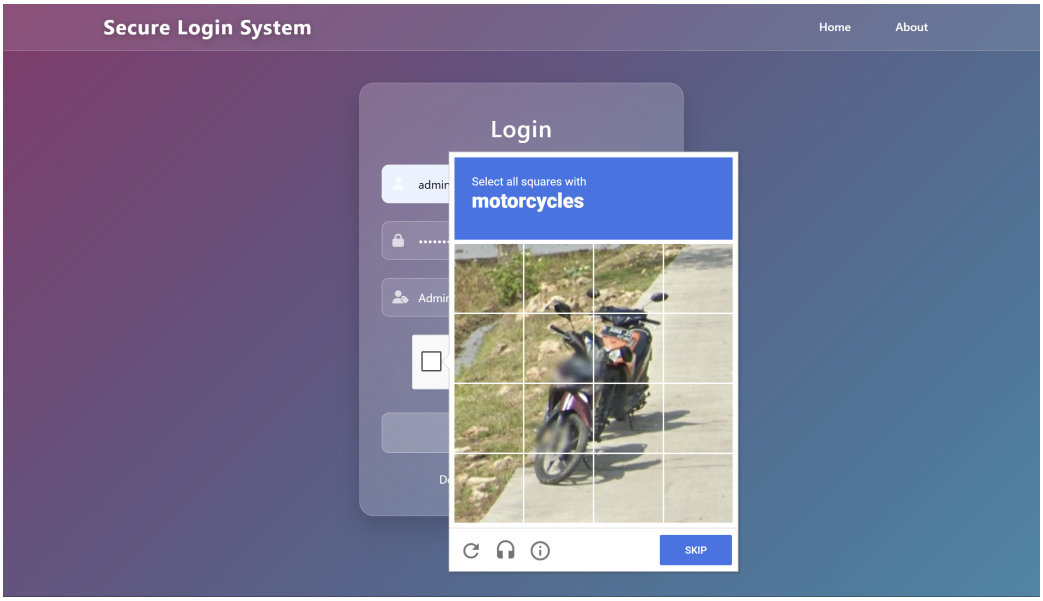

Figure 6.6: Login with CAPTCHA

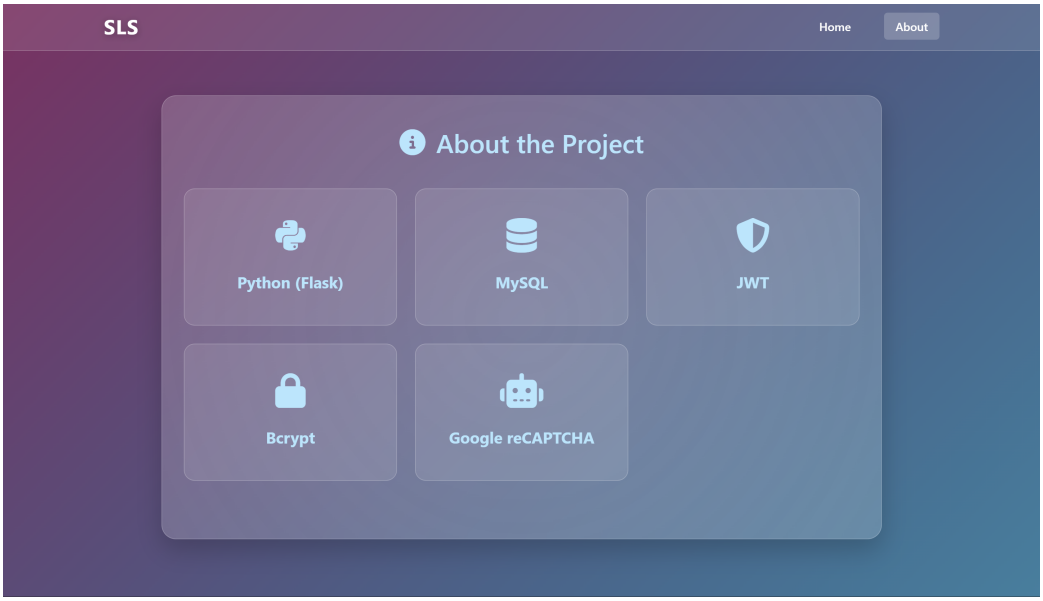

Figure 6.7: About Page

```
mysql> select * from user;
+----+----------+-----------------+--------------------------------------------------------------+---------+---------------------+
| id | username | email_id        | password                                                     | role_id | created_at          |
+----+----------+-----------------+--------------------------------------------------------------+---------+---------------------+
|  5 | admin1   | admin1@gmail.com | $2b$12$sGnGiQEnsaZDPOGO3B78D.UaSR2HItD2vd9Y77raf/7THtXkciLMe |       1 | 2025-09-28 11:06:07 |
|  6 | user1    | user1@gmail.com  | $2b$12$8N3eve/UIRAHAFL3oN.ZXuOzjL4EXRJZK9qPwCLLnznJsWRZfQLNO |       2 | 2025-09-28 11:06:35 |
|  7 | user3    | user3@gmail.com  | $2b$12$zBW0WjKoZb3AD1nG1v.i/uri8.6oN2K54XcK4KYyiQYJDE2aqFWDu |       2 | 2025-09-29 14:49:13 |
|  8 | admin2   | admin2@gmail.com | $2b$12$335bycEoSRgrX2e/IcMJ6OpJT/f5lgbkcMdhpv8kn9jneH39SPWVG |       2 | 2025-09-29 20:03:48 |
+----+----------+-----------------+--------------------------------------------------------------+---------+---------------------+
4 rows in set (0.00 sec)

mysql> select * from user_role;
+----+-------+
| id | name  |
+----+-------+
|  1 | admin |
|  2 | user  |
+----+-------+
```

Figure 6.8: User table

```
mysql> select * from login_activity;
+----+---------+-------------------+---------------------+---------------------+
| id | user_id | username_snapshot | login_time          | logout_time         |
+----+---------+-------------------+---------------------+---------------------+
|  1 |       5 | admin1            | 2025-09-29 20:22:47 | 2025-09-29 20:35:44 |
|  2 |       5 | admin1            | 2025-09-29 21:12:40 | NULL                |
|  3 |       5 | admin1            | 2025-09-29 21:13:43 | NULL                |
|  4 |       5 | admin1            | 2025-09-29 21:18:21 | 2025-09-29 21:55:13 |
|  5 |       5 | admin1            | 2025-09-29 22:03:18 | 2025-09-29 22:03:23 |
|  6 |       7 | user3             | 2025-09-29 22:18:38 | 2025-09-29 22:18:54 |
|  7 |       5 | admin1            | 2025-09-29 22:37:53 | 2025-09-29 22:38:17 |
|  8 |       5 | admin1            | 2025-09-29 22:44:26 | 2025-09-29 22:46:14 |
|  9 |       5 | admin1            | 2025-09-30 00:23:45 | 2025-09-30 00:23:57 |
| 10 |       6 | user1             | 2025-09-30 00:24:41 | 2025-09-30 00:24:47 |
| 11 |       5 | admin1            | 2025-09-30 00:33:35 | NULL                |
| 12 |       5 | admin1            | 2025-09-30 00:37:34 | NULL                |
| 13 |       5 | admin1            | 2025-09-30 01:08:06 | 2025-09-30 01:08:37 |
| 14 |       6 | user1             | 2025-09-30 01:09:38 | NULL                |
+----+---------+-------------------+---------------------+---------------------+
14 rows in set (0.00 sec)
```

Figure 6.9: Activity table

Conclusion

The Secure Login System (SLS) project provided me with valuable hands-on experience in designing and implementing a secure authentication platform. Through the development of this system, I gained practical knowledge in integrating backend technologies such as Flask with a MySQL database, while ensuring data security through the use of Bcrypt for password hashing and JWT for token-based authentication. The inclusion of Google reCAPTCHA further enhanced the reliability of the system by protecting against automated attacks and ensuring only genuine users could access the platform. This project strengthened my technical skills in web development and database management but also improved my understanding of security practices that are essential in real-world applications. It highlighted the importance of balancing usability with security, and how different tools and frameworks can be combined to create a robust solution.