SOHAIB KHAN
2022551
CYBER SECURITY

SSD WEEK : 1

1. Overview of the Secure Notes Application

This project is a Flask-based web app that allows users to create, store, and manage encrypted notes. The primary focus is on security, ensuring that user data is protected from unauthorized access through authentication, encryption, and secure session management.

---

2. Features Implemented

(i) User Authentication (Registration & Login)

• Users need to register before they can store notes.
• Passwords are securely hashed using bcrypt before being stored in the database.
• Users log in using their username and password.
• Authentication is handled via Flask-Login, which ensures that only registered users can access the application.

(ii) Encrypted Notes Storage

• When users create a new note, the content is encrypted before being stored in the database.
• Encryption is done using Fernet from the cryptography library.
• Only the user who created the note can decrypt and view it.
• When displaying notes on the dashboard, they are decrypted on the server before being sent to the frontend.

(iii) User Dashboard

- After logging in, users are directed to their personal dashboard.
- Features of the dashboard:
- Create New Notes: Users can write new notes, which are encrypted before saving.
- View Existing Notes: Users can see all their saved notes (decrypted before display).
- Future Improvements (editing & deleting notes) can be added.

(iv) Logout Functionality

- Users can log out to end their session, preventing unauthorized access.
- Session management is handled via Flask-Login to track whether a user is logged in.

---

3. Security Measures

Security is a critical aspect of the application. Below are the key security implementations:

(i) Password Security

- bcrypt is used for password hashing.
- Plain-text passwords are NEVER stored.
- During login, the stored hash is compared securely with the entered password.

(ii) Note Encryption

- Notes are encrypted using Fernet encryption from the cryptography library.
- Even if someone accesses the database, they cannot read the notes without the encryption key.

(iii) CSRF Protection

- Flask-WTF is used to protect against Cross-Site Request Forgery (CSRF).
- Every form in the app includes a CSRF token, preventing attackers from making unauthorized form submissions.

(iv) Secure Session Management

- The application uses Flask-Login to manage user sessions.

• If a user logs out, their session is destroyed, and they must log in again.

---

4. UI & User Experience Enhancements

(i) Bootstrap Integration

• The app uses Bootstrap 4 for a responsive and mobile-friendly design.

• The interface is modern, clean, and user-friendly.

(ii) Error Handling

• Flask flash messages provide feedback to users (e.g., “Invalid credentials”, “Account created successfully”).

• Errors like incorrect logins or missing fields are properly handled.

(iii) Custom Styling

• CSS is used to enhance the look and feel of the app.

• Buttons, forms, and layout are designed for a smooth user experience.

---

5. Next Steps & Improvements

While the basic app is functional, several enhancements can be made:

(i) Additional Features

• Password Reset: Allow users to reset their password if forgotten.

• Edit & Delete Notes: Users should be able to modify or delete their notes.

• Search Notes: Implement a search bar to quickly find specific notes.

(ii) Security Enhancements

• Store Fernet encryption keys securely (e.g., in environment variables).

• Use HTTPS to secure communications between the client and server.

- Implement stronger password policies (e.g., minimum length, special characters).

(iii) Performance & Database Improvements

- Upgrade from SQLite to PostgreSQL for better scalability.
- Implement caching or pagination for better performance with large numbers of notes.

(iv) Deployment Readiness

- Use Gunicorn for production deployment instead of Flask's built-in server.
- Deploy the app on Heroku, AWS, or DigitalOcean.
- Store secrets (e.g., Flask secret key, database URI) in environment variables.

---

6. Technologies & Libraries Used

The following libraries were used to build the application:

Technology	Purpose
Flask	Web framework for building the app
Flask-Login	User session management
Flask-WTF	Secure web forms with CSRF protection
Flask-Bcrypt	Password hashing
Flask-SQLAlchemy	Database interaction (currently using SQLite)
Cryptography (Fernet)	Secure encryption for notes
Bootstrap 4	Responsive UI design
CSS	Custom styling for improved user experience

THE END