Flask Demo Account Registration App

Isaac

September 17, 2025

List of Group Members

- Narter-Tawiah Isaac Yohanes -01245954B
- Gideon Nana Ofosu-01246061B
- Osman Anwar-01244669B
- Charles Ebanyenle -01233698B
- Darko-Ameyaw Joel-01245244B

Abstract

This report documents the development of a simple Python web application using Flask. The app allows users to create a demo account with full name, email, and password. It includes server-side validation, password hashing, and SQLite database integration—all in a single file for simplicity and portability.

Technologies Used

- Python 3.x
- Flask (micro web framework)
- SQLite (lightweight database)
- Werkzeug (for password hashing)

Application Features

- User registration form
- Validation for email format and password strength

- Password confirmation check
- Secure password hashing
- Persistent storage using SQLite
- Clean routing and modular logic

Validation Rules

- Email must be valid and unique
- Password must be at least 8 characters, include one uppercase letter and one digit
- Passwords must match

Database Schema

```
CREATE TABLE users (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    full_name TEXT NOT NULL,
    email TEXT NOT NULL UNIQUE,
    password_hash TEXT NOT NULL
);
```

Setup Instructions

Install Dependencies

```
pip install flask werkzeug
```

Run the App

Save the code as `app.py` and run:

```
python app.py
```

Open your browser and visit:

```
http://127.0.0.1:5000
```

Code Overview

The application is structured as follows:

- `init_db()`: Initializes the SQLite database
- `is_valid_email()`: Validates email format
- `is_strong_password()`: Checks password strength
- `email_exists()`: Checks for duplicate email
- `save_user()`: Saves user to database
- `register()`: Main route for form handling

Extensibility Ideas

- Add login and session management
- Add image upload for profile pictures
- Deploy to Render, Replit, or Vercel
- Add email confirmation via Flask-Mail

Conclusion

This project demonstrates how to build a secure, modular, and user-friendly web application using Flask. It serves as a foundation for more advanced features such as authentication, file uploads, and deployment.