Flask Web Application Project Report

contributor to this report: Xinyu Li, Tong Xu, ChatGPT

Introduction

This application provides a platform for user authentication, grade management, anonymous feedback submission, and regrade requests. The application is built using Flask, SQLAlchemy, and Flask-Bcrypt for functionality and security.

Project Structure

The project structure consists of various components, including database models, routes for different functionalities, and session management.

Database Models

The application uses SQLAlchemy to define four models: Person, Grade, anonFeedback, and RegradeRequest.

- **Person**: This model represents the users of the application, storing their identification number (utorid), hashed password, and identity (role).
- **Grade**: This model manages the grades of the students, including midterm, final, assignments, and labs.
- **anonFeedback**: This model handles anonymous feedback from students. It stores positive and negative feedback about teaching and assignments.
- **RegradeRequest**: This model manages regrade requests, including the student's identification number (utorid), request details, and the type of regrade (e.g., midterm, final, assignment, lab).

Routes

The application defines numerous routes to handle different functionalities. These routes can be logically grouped based on their functionality:

Web Page Rendering

These routes simply render the corresponding pages of the application:

- **Home, News, Calendar, Resources**: These routes render the home page, news page, calendar page, and resources page respectively.
- **Staff**: This route renders the staff page, which likely contains information relevant to staff members.
- **Lecture, Assignment_Lab**: These routes render the lecture and assignment/lab pages, which likely contain academic content for students.

User Management

These routes handle user authentication, registration, and session management:

- **Login**: This route handles user authentication, checking the provided credentials against those stored in the database.
- **Register**: This route manages user registration, adding new users to the database.
- **Logout**: This route handles user logout, clearing the session information.

Academic Management

These routes allow students to view their grades, submit regrade requests, and provide anonymous feedback. Staff members can view all students' grades, all feedback, and handle regrade requests:

- **Marks**: This route enables individual students to view their grades.
- **All_student_marks**: This route allows staff members to view all students' grades.
- **Submit_remark_request**: This route manages the submission of regrade requests by students.
- **AnonFeedback**: This route handles the submission of anonymous feedback by students.
- **GetFeedback**: This route allows staff members to view all the feedback submitted by students.
- **RegradeRequests**: This route allows staff members to view all regrade requests submitted by students.
- **Update_grades**: This route allows staff members to update grades based on regrade requests.

Session Management

The application uses Flask's session object for managing user sessions. A session lasts forever (until logout) and stores the user's utorid and identity (role). This information is used across various routes to provide role-based functionalities.

Key Features

The application provides several key features:

- **User Authentication**: The application uses Flask-Bcrypt for hashing passwords, providing secure user authentication.
- **Grade Viewing**: Students can view their grades, and staff can view all students' grades.
- **Anonymous Feedback Submission**: Students can submit anonymous feedback about teaching and assignments.
- **Regrade Request Submission**: Students can submit regrade requests if they believe there's an error in their grading.
- **Regrade Request Processing**: Staff can view all regrade requests and update grades accordingly.

Conclusion

This Flask web application serves as a comprehensive tool for managing academic and administrative tasks in a university setting. The use of Flask, SQLAlchemy,

and Flask-Bcrypt ensures a secure and efficient application. Future improvements could include the implementation of email notifications, role-based access control, and data visualization tools.