User Management System (Flask and SQLite)

Objective, Structure, and Explanation

Objective of User Management System

- Build a simple User Management System using Flask and SQLite.
- Implement user registration and login functionality.
- Use a RESTful API approach with JSON responses for client-server communication.

System Structure

- Server (Flask):
- - Handles database operations with SQLite.
- - Provides endpoints for registration and login.

- Client:
- - Sends requests to the server.
- - Prints JSON responses, including success or error messages.

Explanation of Server Code (User Management System)

- init_db(): Creates 'users' table in SQLite database.

- @app.route('/register', methods=['POST']):
- - Handles registration by adding new user details to the database.
- - Validates data and responds with JSON success or error messages.

- @app.route('/login', methods=['POST']):
- - Checks user credentials, returning a success message or error.

Explanation of Client Code (User Management System)

- register(username, password): Sends a POST request with registration data.
- login(username, password): Sends a POST request to verify login credentials.
- Main program flow:
- - Prompts user to choose registration or login.
- - Calls respective function and displays server response.

To-Do List Program (Flask and SQLite)

Objective, Structure, and Explanation

Objective of To-Do List Program

- Create a To-Do List management system with Flask and SQLite.
- Allow users to add, view, update, and delete tasks.
- Demonstrate CRUD operations using RESTful API with JSON responses.

System Structure (To-Do List Program)

- Server (Flask):
- - Uses SQLite to manage task data.
- - Provides endpoints for task CRUD operations.

- Client:
- - Interacts with the server to manage tasks.
- - Sends requests for adding, viewing, updating, and deleting tasks.

Explanation of Server Code (To-Do List Program)

- init_db(): Creates 'tasks' table in SQLite database.
- @app.route('/tasks', methods=['POST']): Adds a new task.
- @app.route('/tasks', methods=['GET']): Retrieves all tasks.
- @app.route('/tasks/<task_id>', methods=['PATCH', 'DELETE']):
- - Updates task completion or deletes a task by ID.

Explanation of Client Code (To-Do List Program)

- get_tasks(): Retrieves and displays all tasks.
- add_task(description): Sends a request to add a task.
- update_task(task_id, completed): Updates task completion status.
- delete_task(task_id): Sends a request to delete a task by ID.