Anthony Stock
PG AIML: Programming Refresher
Assessment Project

Task Manager with User Authentication

This Python-based Task Manager allows users to manage their tasks through an interactive, menu-driven interface. Each user has to authenticate using a username and password, ensuring only authenticated users can access their personal tasks. The project meets the following objectives:

1. User Authentication System (auth.py)

- **Objective Met**: Implements user registration and login using plain-text passwords (or hashed if needed). Credentials are stored persistently in a JSON file.
- **Code Breakdown**:
    - **register_user()**: Prompts the user to create a new account, saves the username and password in users.json.
    - **login_user()**: Allows users to log in by comparing entered credentials with stored credentials.

2. Task Management System (tasks.py)

- **Objective Met**: Provides functions to add, view, mark as completed, and delete tasks, ensuring tasks are only accessible to authenticated users.
- **Code Breakdown**:
    - **create_task()**: Prompts the user for a task description, generates a simplified task ID, and assigns a status of "Pending".
    - **view_tasks()**: Displays all tasks for the logged-in user, showing the task ID, description, and status.
    - **mark_task_completed()**: Allows the user to select a task by its ID and update its status to "Completed".
    - **delete_task()**: Deletes a task from the user’s task list by ID.

3. Persistent Storage (storage.py)

- **Objective Met**: Stores user credentials and tasks persistently in JSON format, ensuring data is saved between program executions.
- **Code Breakdown**:
    - **load_user_data()**: Loads the user data from the JSON file, ensuring that all user credentials and tasks are accessible.
    - **save_user_data()**: Saves any updates to user credentials or tasks back into the JSON file.

4. Menu-Driven Interface (main.py)

- **Objective Met**: The interface provides users with a clear, text-based menu to navigate the system and manage tasks.
- **Code Breakdown**:
    - **task_manager()**: Provides the main menu for login, registration, and task management.

    - **task_menu()**: After logging in, users can create, view, mark, or delete tasks through a simple menu interface.

How It Meets the Requirements:

- **File Handling**: Uses JSON to store user credentials and tasks, ensuring persistent storage between program sessions.
- **Authentication**: Each user’s tasks are stored separately, and only the logged-in user can view or modify their tasks.
- **Interactive Interface**: Provides a menu-driven approach that allows users to easily manage tasks after logging in.

This project demonstrates a well-structured task management system with user authentication, persistent storage, and an easy-to-use interface. Let me know if you'd like further details!