Task Manager with User Authentication

Problem Statement:

In today's fast-paced world, individuals often need to keep track of various tasks in a structured way. This project involves building a Task Manager that allows users to manage their tasks efficiently. The system includes user authentication, ensuring that each user has to log in with a username and password. Once logged in, users can create, view, update, and delete their tasks. Each user's tasks are stored separately, and only the authenticated user can access their tasks.

Objectives:

1. **User Authentication System**:
    - **Registration**: Users can register with a unique username and a hashed password.
    - **Login**: Users log in with their credentials to access the task manager.
2. **Task Management System**:
    - **Add Tasks**: Users can add tasks with a description and a unique ID.
    - **View Tasks**: Users can view all their tasks, showing the ID, description, and status (Pending or Completed).
    - **Mark Tasks as Completed**: Users can update the status of tasks to Completed.
    - **Delete Tasks**: Users can delete tasks by their ID.
3. **Persistent Storage**:
    - User credentials and tasks are stored in files for persistence.
4. **Interactive Menu**:
    - A menu-driven interface allows users to manage tasks and log out.

Implementation Steps:

1. **User Authentication**:
    - **Registration**:
        - Create a function to prompt the user to enter a username and password.
        - Ensure that the username is unique, and hash the password for security before storing it in a file.
    - **Login**:
        - Create a function to prompt the user for their username and password.
        - Validate the credentials by comparing them with the stored data.

        - Grant access to the task manager upon successful login.

2. **Task Management**:
    - **Add a Task**:
        - Create a function that prompts the user for a task description.
        - Assign a unique task ID and set the status to Pending.
        - Store the task in a file and confirm that the task was added.
    - **View Tasks**:
        - Create a function to retrieve and display all tasks for the logged-in user.
        - Each task should show the task ID, description, and status (Pending or Completed).
    - **Mark a Task as Completed**:
        - Create a function that allows the user to select a task by its ID and update its status to Completed.
    - **Delete a Task**:
        - Create a function that allows the user to select a task by its ID and delete it from their task list.
3. **Interactive Menu**:
    - Build a menu that allows users to choose between:
        - Add a Task
        - View Tasks
        - Mark a Task as Completed
        - Delete a Task
        - Logout
    - For each option, call the corresponding function and loop back to the menu until the user logs out.

Additional Features:

1. **Budget Tracking**:
    - Users can set a monthly budget and track expenses.
    - The system warns users if they exceed their budget and shows the remaining balance.

2. **Expense Management**:
    - Users can save and load expenses from a CSV file.
    - An interactive menu allows users to add expenses, view expenses, track the budget, save expenses, and exit the program.

Conclusion:

This project combines user authentication with task and budget management, providing a comprehensive tool for personal organization. It ensures that users can securely manage their tasks and track their expenses, making it a valuable application for daily use.