# Project: Task Manager with User Authentication

**Problem Statement:**

In today's world, individuals often need to keep track of various tasks in a structured way. You are tasked with building a Task Manager that allows users to manage their tasks. The system should include user authentication, meaning 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 should be stored separately, and only the authenticated user can access their tasks.

**Objectives:**

1. Design and implement a user authentication system (login and registration)
2. Create a task management system that allows users to:

a. Add, view, mark as completed, and delete tasks

3. Use file handling to store user credentials and tasks persistently
4. Create an interactive menu-driven interface to manage tasks

**Steps**

**to Perform:**

**1. User Authentication:**

- **Registration:**
  o Create a function to prompt the user to enter a username and password
  o Ensure that the username is unique, and hash the password for security before storing it in a file
- **Login:**
  o Create a function to prompt the user for their username and password, validate the credentials by comparing them with the stored data, and grant access to the task manager upon successful login

2. **AddaTask:**
   - **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**
3. **ViewTasks:**

- **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)**

4. **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

**5. Delete a Task:**
• Create a function that allows the user to select a task by its **ID** and delete

it from their task list

**6. Create an Interactive Menu:**

• Build a menu that allows users to choose between: o Add a Task
o View Tasks
o Mark a Task as Completed

o Delete a Task

o Logout
For each option, call the corresponding function, and loop back to the menu until the user logs out.

**1. Set and Track Budget:**

• Create a function that allows the user to input a monthly budget. Prompt the user to:

o Enter the total amount they want to budget for the month (Example: $500)

• Create another function that calculates the total expenses recorded so far. o Compare the total with the user's monthly budget
o If the total expenses exceed the budget, display a warning (Example:

You have exceeded your budget!)
o If the expenses are within the budget, display the remaining balance

(Example: You have $150 left for the month)

**2. Save and Load Expenses:**

- Implement a function to save all expenses to a CSV file, with each row containing the date, category, amount, and description of each expense
- Create another function to load expenses from the CSV file. When the program starts, it should:

o Read the saved data from the file
o Load it back into the list of expenses so the user can see their previous

expenses and continue from where they left off

**3. Create an Interactive Menu:**

• Build a function to display a menu with the following options: o Add Expense

o View Expenses o Track Budget o Save Expenses o Exit

- Allow the user to enter a number to choose an option.
- Implement conditions for each menu selection:

o If the user selects option 1, call the function to add an expense
o If the user selects option 2, call the function to view expenses
o If the user selects option 3, call the function to track the budget
o If the user selects option 4, call the function to save expenses to the file o If the user selects option 5, save the expenses and exit the program