SWE

PROJECT PROPOSAL

Presented to:

Md. Sefatullah

Proposal by:

Efaz Ahmmed Asfi
Nazmus Sakib Somudro
Khondaker Hasib Hossain

PRESENTED BY:
TEAM HAWK

PREPARED FOR:
MD. SEFATULLAH

DAILY TASK PLANNER

SWE 41 (N), DAFFODIL INTERNATIONAL UNIVERSITY

PROJECT INFIRMATION

The Daily Task Planner project is a simple, text-based program built using the C programming language. Its purpose is to help users organize their daily tasks, allowing them to conveniently manage, view, and modify tasks. This project involves core C programming concepts, including character arrays, loops, functions, and conditional logic.
The application primarily uses:

- **Global Declarations:** Variables such as task, task count, recentCompletedTask, completedTaskList, listedDeletedTask, and counters for tracking tasks.

- **Task Management Functions:**
  viewTask() – Displays the list of current tasks.
  addTask() – Adds a new task to the list.
  markCompletedTask() – Marks a task as completed and updates completedTaskList.
  delete task () – Deletes a specified task from the list.
  listOfAllCompletedTask() and listOfAllDeletedTask() – Show lists of completed and deleted tasks.

- **2D Character Arrays:** Used to store task information with char task[100][100] for all tasks, and separate arrays for tracking completed and deleted tasks.
  Control Statements and Loops: To handle task iteration, selection, and user prompts.

THE INTERNAL TEAM

Efaz Ahmmed Asfi

Nazmus Sakib Somudro

Khondaker Hasib Hossain

PROBLEMS ADDRESSED

Many users lack a straightforward tool to keep track of daily tasks without complicated software or databases. This Daily Task Planner addresses the need for a simple, portable, and effective solution to:

- Organize tasks with clear actions like add, view, delete, and mark as completed.
- Provide a straightforward text interface that runs on any C-compatible system.
- Offer a lightweight solution for managing tasks without reliance on external data storage.

BENEFITS

- **Practical Skill Development:** Helps users practice C programming concepts like 2D arrays, functions, loops and control statements through functions such as viewTask, addTask, markCompletedTask, and deleteTask.
- **Task Organization:** The system allows users to efficiently manage tasks
- **Portability and Simplicity:** The program runs on multiple operating systems
  and requires no external dependencies or storage.
- **Efficiency:** Core task management features are provided in a lightweight, user-friendly interface.

DELIVERABLES

- **Source Code:** Complete and functional C code implementing all task management features. Functions are used to structure task management actions, contributing to readability and modularity.
- **Documentation:** Instructions on using the Daily Task Planner.
- **Test Cases and Results:** Test cases demonstrate that all functions (add, view, delete, complete) work as intended.

SUCCESS CRITERIA

The project will be considered successful if:

- All specified features, including task creation, deletion, marking as completed, and displaying tasks, work as expected.
- The program provides an intuitive user interface and meets the password security criteria (e.g., minimum of 8 characters with uppercase, lowercase, and a digit).
- The planner runs smoothly across platforms, using standard C libraries and control structures.

DEADLINE / PLAN / APPROACH

Project Timeline:

- Week 1: Implement the login system and basic functions such as addTask and viewTask.
- Week 2: Add markCompletedTask functionality and test task display.
- Week 3: Test and refine, adding validation for functions like deleteTask.
- Week 4: Final testing and documentation.

Approach:
Development will focus on incremental functionality, starting with the login and moving through task management actions. Arrays and control statements are central to the program's design for tracking task status. The final week will focus on testing, debugging, and documentation.

COST / BUDGET

This project requires minimal costs since it uses only C programming and basic development tools (GCC or Clang compilers).
Estimated budget: $0 - $50.