# GROUP 7 PROJECT PROPOSAL

## Task Manager – Simple Command-Line App

## 1. Mission

### Description of the concept of Our Mission:

Our mission is to make it easier for students and workers to organize their daily activities. We aim to develop a tool that is easy to use and does not require an internet connection. By making this app, we hope to help people become more productive and avoid forgetting important tasks. We chose to make the app offline because it matches our assignment requirements and is easy for anyone to use, even if they do not have a good internet connection.

- Who: Students and workers in Africa
- What: Need a simple way to manage daily tasks
- When: Every day, for school or work
- Where: In places with slow or no internet
- Why: To help people remember and finish their tasks on time
- How: By making a command-line app that saves tasks on the computer

# 2. Problem Statement:

Many students and workers in Africa have trouble keeping track of what they need to do. Most task manager apps need the internet or are too hard to use. Some people cannot afford to buy these apps. Because of this, people forget tasks or miss deadlines. We want to solve this problem by making a simple task manager that works offline and is easy for anyone to use.

# 3. The hypothesis of the Solution:

Suppose we create a simple, offline task manager that works from the command line. In that case, students and workers will be able to organize their daily activities better, remember important tasks, and meet their deadlines, even if they do not have internet access or cannot afford expensive software.

## Outline of the Proposed Solution:

- We will develop a command-line application that allows users to add, view, edit, and delete tasks.
- The app will store all tasks locally on the user's computer in a file (like JSON), so it works without the internet.
- The interface will be simple, with clear instructions, so anyone can use it, even if they are not tech-savvy.
- There will be options to mark tasks as completed and to back up tasks.
- The app will be lightweight and run on most computers.

# 4. List of Technologies to be Used

- Node.js (JavaScript runtime for building the app)
- JavaScript (main programming language)
- JSON (for storing task data)
- Git (for version control)
- Visual Studio Code (for coding)
- Windows or Linux (for testing)

# 5. Preliminary Project Timeline with Milestones:

| **Date** | **Task** |
|---|---|
| July 20, 2025 | Group meeting, discuss project, assign roles, and plan |
| July 21, 2025 | Write and complete the Project Proposal |
| July 22, 2025-July 25,2025 | Design the app structure and plan features & Start coding: implement add, view, update, and delete task features. |
| July 26, 2025 | Test the app, fix bugs, write the software documentation, plus record presentation video |
| July 27, 2025 | Final review, polish documentation, and submit the project |

# 6. Division of Responsibilities Among Group Members

| Members | Responsibilities |
|---|---|
| Belinda Larose | Implemented the **core object-oriented task logic** including class properties, methods like `markComplete()`, `isOverdue()`, and the base class functionality<br><br>Developed `TaskManager` operations such as **add**, **update**, **delete**, and **search**<br><br>Co-authored the **initial proposal and functional requirements** |
| Plamedi Mayala | Built the **interactive command-line interface** using Commander.js, Inquirer.js, and Chalk<br><br>Designed the **menu system**, task flows, and CLI user experience<br><br>Wrote and executed **unit tests** for task logic, file operations, and validation<br><br>Assured **integration across modules** and ensured consistent interaction throughout the app |
| Grace Munezero | Created and maintained **file storage** logic using the `fs` module<br><br>Implemented **data validation** methods for task input, including date and ID checks<br><br>Helped ensure **data integrity** and **error handling** across the application |
| Kevine Umutoni | Developed the **entry point logic** (`index.js`) for initializing and running the CLI<br><br>Authored **software documentation**, including the system architecture, UML diagrams, README, and |

| | |
|---|---|
| | usage guide<br><br>Wrote and maintained content for the `docs/` directory and supported team communication of technical decisions |

## 7. Software Development Model:

We will use the Waterfall model for this project. This means we will finish one stage before moving to the next. For example, we will complete the planning and design before we start coding. This model is good for our project because our requirements are clear and not likely to change.

## 8. References

1. Node.js Foundation. (n.d.). About Node.js. https://nodejs.org/en/about/

2. Mozilla Developer Network. (n.d.). JavaScript Guide.https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide

3. . W3Schools. (n.d.). Node.js File System Module. https://www.w3schools.com/nodejs/nodejs_filesystem.asp
4. African Leadership University, Tunde Isiaq Gbadamosi. (2024). Project Proposal Template. Introduction to Software Engineering. African Leadership University. https://alueducation.instructure.com/courses/1628

AFRICAN LEADERSHIP UNIVERSITY

# [PROGRAMME]

[Software Engineering] FORMATIVE ASSIGNMENT:
[Software Development Project Proposal]