Final Report: Task Manager using OOP and File Management

Project Title: Task Manager using OOP and File Management
Course:Advanced Backend
Submitted By:

- Vanessa UWONKUNDA
- Thierry MARIDADE
- Gabriel
- Nelson Marvolous

Date: Friday, July 25th 2025
Instructor: Bernardy

Table of Contents

---

1. Introduction

In the modern digital workspace, staying organized and managing tasks efficiently is essential. While many applications exist for task management, most require a graphical interface or internet access. This project aims to build a simple, robust **command-line task manager** using **Object-Oriented Programming (OOP)** in **JavaScript (Node.js)**, with persistent storage using the file system.

---

2. Project Proposal

Problem Description

Many developers and students work in terminal-based environments where they need lightweight tools to manage tasks. Existing tools are either too complex or require online accounts. This creates a barrier for users who seek a lightweight, offline, and easy-to-use task tracking solution within the terminal.

Proposed Solution

We propose to build a Node.js command-line application that:

- Allows users to manage their to-do list via CLI commands.
- Implements core OOP principles for clean structure.
- Uses the file system to store tasks in a `tasks.json` file, ensuring data persistence.
- Incorporates user-friendly messages and basic error handling for reliability.

Technology Stack

Technology	Purpose
Node.js	CLI Logic
JavaScript	OOP implementation
File System (`fs`)	Persistent file storage
Git & GitHub	Version control and collaboration
Draw.io / Lucidchart	UML diagramming
Google Docs	Documentation

Timeline and Milestones

Timeline	Activities / Deliverables
Day 1	Define scope, create project repo, assign team roles
	Design class structure, write project proposal
	Build `Task` and `TaskManager` classes
	Implement file operations using `fs`, test saving/loading
	Finalize CLI features (add, update, delete)
Day 2	Complete documentation and UML diagrams
Day 3	Record video walkthrough, finalize report and contribution logs
	Submission and final review

---

3. System Architecture

The application has four main modules:

- **Task** – A class representing a single task.
- **TaskManager** – Manages a collection of Task instances.
- **FileService** – Reads/writes tasks to the file system (`tasks.json`).
- **CLI (app.js)** – Handles user input and routes commands.

Each module is designed to follow separation of concerns and reusability.

---

4. Implementation Details

- **Task Class**: Stores task info and provides an update method.
- **TaskManager Class**: Adds, lists, updates, deletes tasks from an array.
- **FileService**: Reads and writes the task array as JSON to `tasks.json`.
- **CLI (app.js)**: Uses `process.argv` to parse commands like `add`, `list`, `update`, `delete`.

---

5. Data Storage

Example Data in `tasks.json`:

```
[
  {
    "id": 1,
    "title": "Finish Report",
    "description": "Complete final documentation",
    "due Date": "2025-07-30",
    "status": "incomplete"
  }]
```

---

6. Setup Instructions

Requirements:

- Node.js
- Git

Steps:

```
# Clone the repository
git clone https://github.com/your-repo/task-manager.git
cd task-manager
npm install
node src/cli/app.js
```

---

7. Error Handling

Scenario	Handling
Missing arguments	Show helpful usage message
Invalid task ID	Return “Task not found”
JSON file not found	Create new file automatically
File write errors	Display and exit safely

---

8. Conclusion

This project demonstrated our ability to design and develop a complete CLI application using OOP in Node.js. We applied abstraction, modularity, and encapsulation effectively. We also learned collaboration through Git, good documentation practices, and testing with real-world use cases.

The application met its goals:

- Persistent local task storage
- Full CRUD support via CLI
- Clean, maintainable, object-oriented code

Future improvements may include:

- Tagging and filtering tasks
- Categorization or prioritization
- Converting to web/mobile interface

Overall, the project enhanced our understanding of software design and teamwork.