Introduction

This project is a simple Python-based console application that helps users manage their daily tasks. It allows users to add tasks, view them, mark tasks as complete, and delete tasks. The application stores data in a JSON file, ensuring that tasks persist even after the program is closed.
The goal of this project is to practice modular programming, file handling, error validation, and simple data storage in Python.

PROBLEM STATEMENT

People often forget their daily tasks because they don’t have a lightweight offline tool for tracking them. Most task apps require accounts, internet access or are too complex.
This project solves the problem by providing a simple, offline, menu-driven To-Do application that works entirely locally.

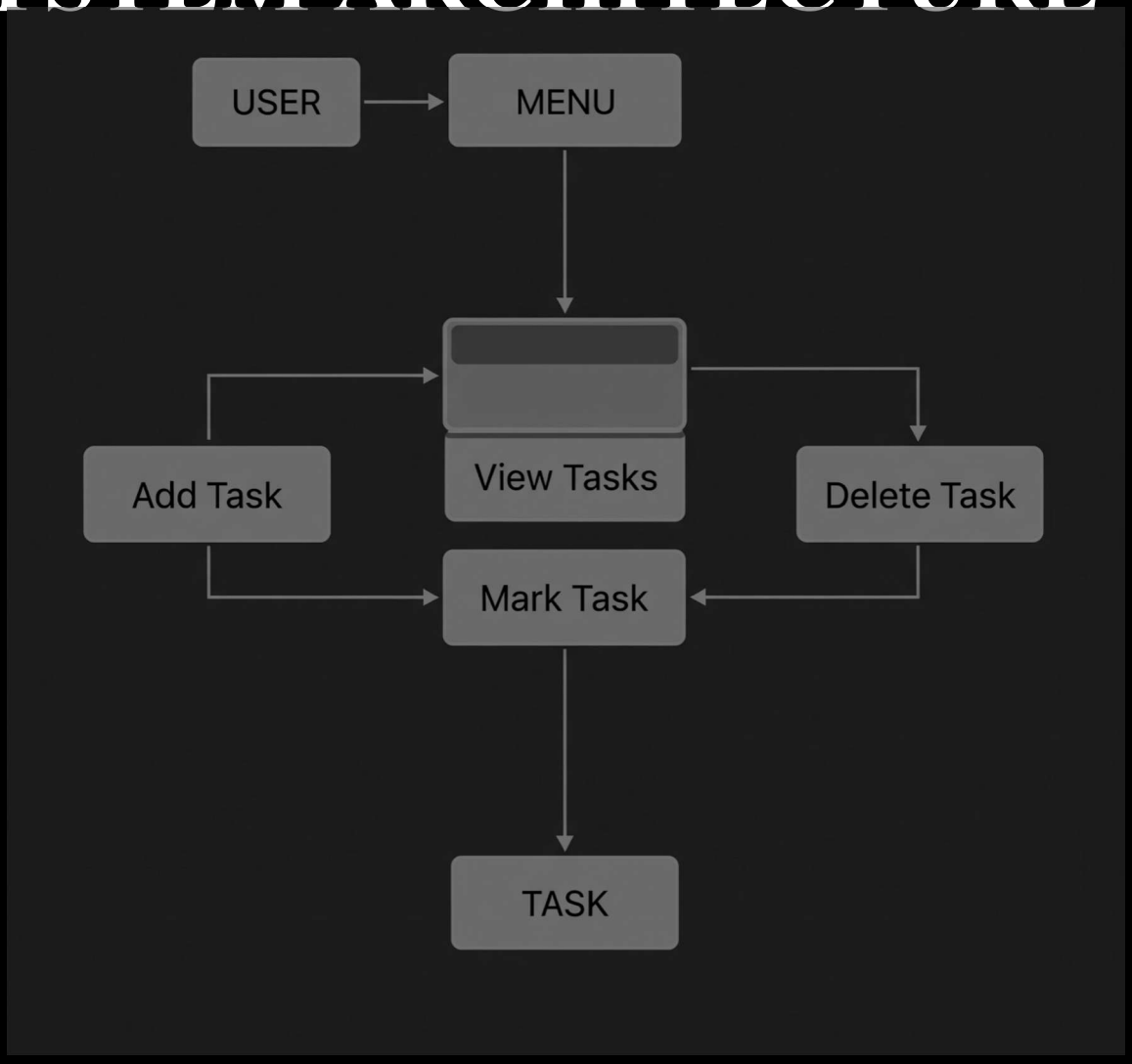
OBJECTIVES

- To design a console-based task manager.
- To implement basic CRUD operations (Create, Read, Update, Delete).
- To store tasks persistently using JSON.
- To ensure error-free and user-friendly input handling

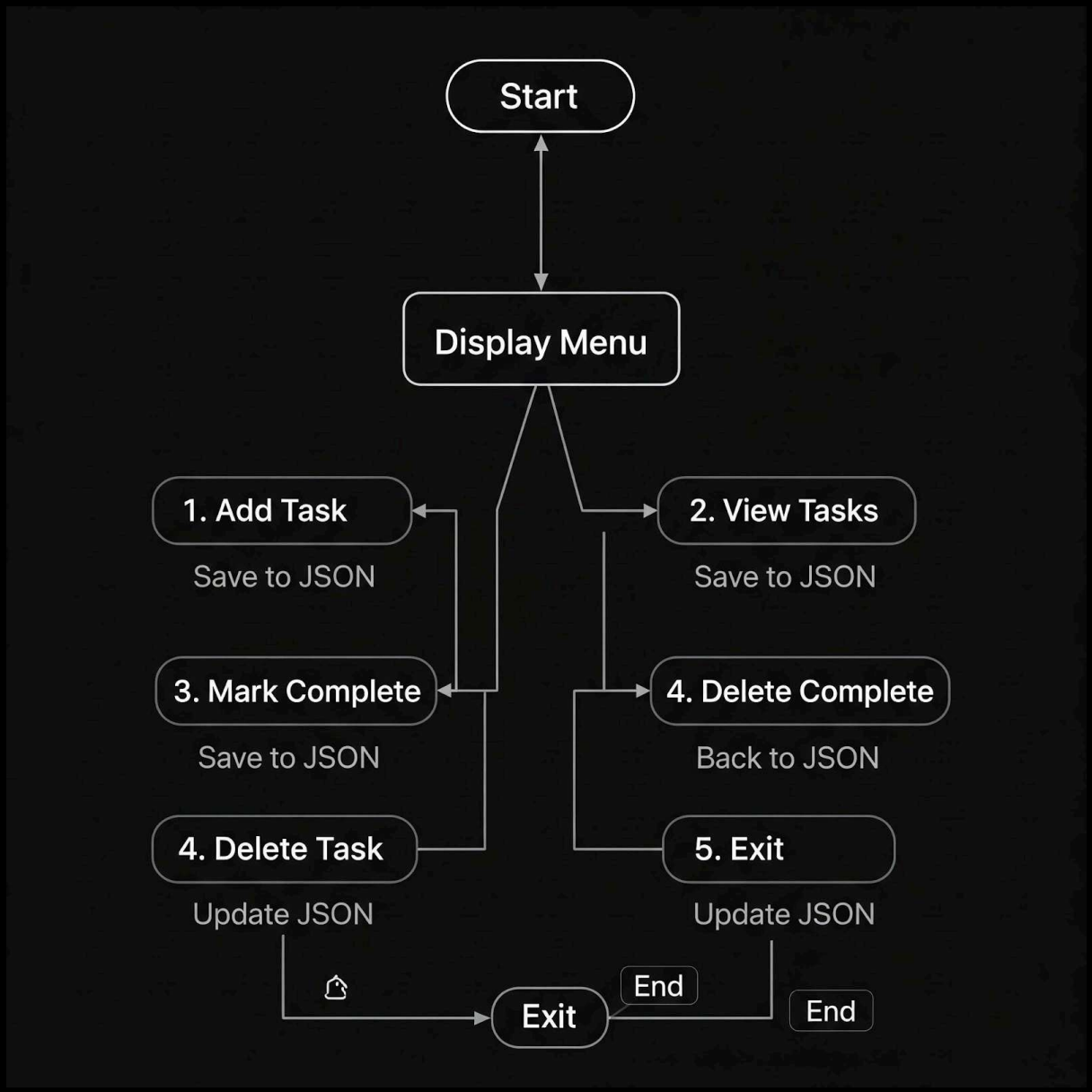
FUNCTIONAL REQUIREMENTS

- **Usability** – Simple, menu-driven interface.
- **Reliability** – Tasks are safely stored in JSON.
- **Maintainability** – Code is modular with reusable functions.
- **Error Handling** – Handles invalid dates, invalid inputs, empty list checks.

SYSTEM ARCHITECTURE

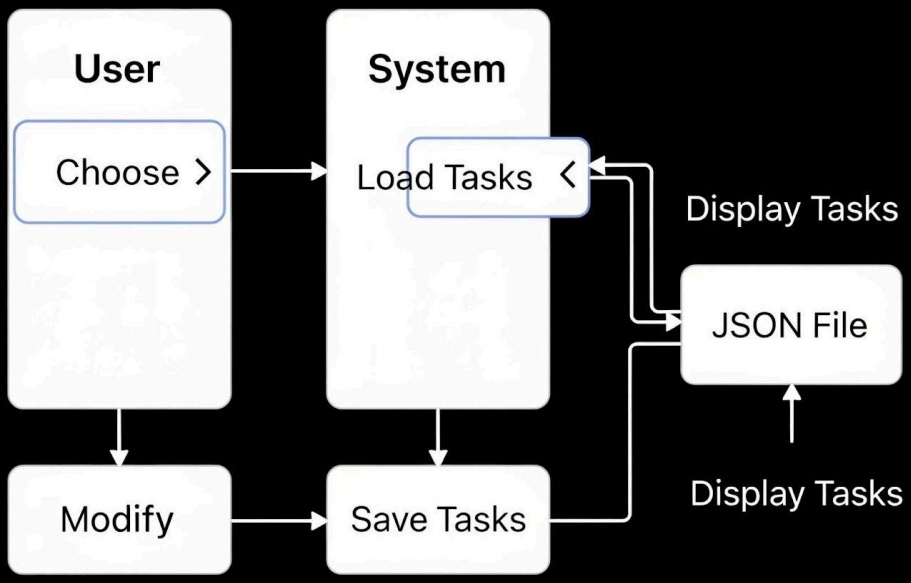


WORKFLOW DIAGRAM


Start
Display Menu
1. Add Task
Save to JSON
2. View Tasks
Save to JSON
3. Mark Complete
Save to JSON
4. Delete Complete
Back to JSON
4. Delete Task
Update JSON
5. Exit
Update JSON
Exit
End
End

USE CASE DIAGRAM

SEQUENCE DIAGRAM



DESIGN DECISIONS

- JSON was chosen for storage because it is lightweight and easy to parse.
- Modular functions used for better code organization.
- Date validation ensures reliability and avoids user mistakes.

IMPLEMENTATION DETAILS

- Python file structure

  Functions used

- How JSON is loaded and saved

  Validation using

TESTING

- Invalid date input
- Empty task description
- Marking/deleting non-existing tasks
- JSON corruption check
- Sorting tasks properly

CHALLENGES FACED

- Preventing JSON decode errors
- Sorting tasks by valid dates
- Ensuring menu does not crash on invalid input

LEARNINGS

- Use of file handling in Python
- Data persistence
- Modular programming
- Input validation
- Basic software architecture & diagrams

FUTURE ENHANCEMENTS

- Adding categories (work/study/personal)
- Adding a GUI using Tkinter
- Notifications/reminders
- Exporting tasks to CSV

REFERENCES

- Python Documentation

- StackOverflow
- Class notes
- JSON Reference