Project Report – Week 1 Task

- Aryan Sengar

Python CLI Application: To-Do List Manager

1. Project Title

To-Do List Manager (Command Line Interface)

2. Objective

The objective of this project is to design and implement a Python-based Command Line Interface (CLI) application that functions as a simple, user-friendly, and efficient To-Do List Manager. The project demonstrates modular design, clean code practices, error handling, and user interaction through terminal-based inputs.

3. Tools & Technologies Used

Technology	Description
Python 3.x	Main programming language
JSON	Data storage format
VS Code	Code editor
Terminal / CMD	For running the CLI application

4. Folder Structure

```
To-Do List Manager/
├── data.json
├── main.py
├── README.md
├── todo.py
├── utils.py
├── .vscode/
│   └── settings.json
├── lib/
└── src/
```

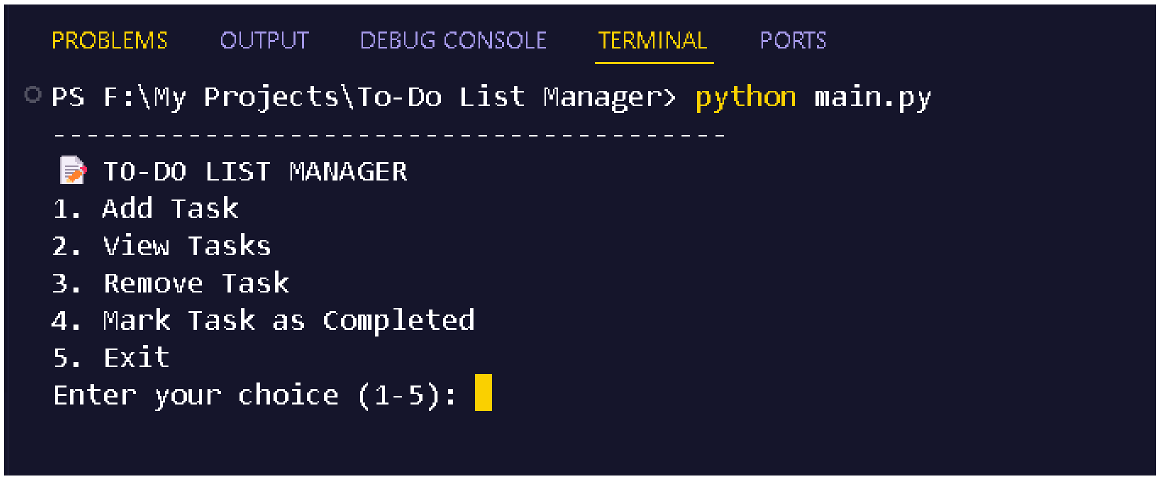
5. Application Architecture

- main.py: Entry point of the program. Displays menu, takes user input, and calls methods accordingly.
- todo.py: Contains the Task class and ToDoList class which encapsulate all task management operations.
- utils.py: Helper module containing methods to load/save tasks and clear the screen.
- data.json: Stores task data persistently between sessions using JSON.
- diagram.txt: Contains basic pseudocode and flow of the application logic.
- README.md: Contains project overview, how to run instructions, and testing guide.

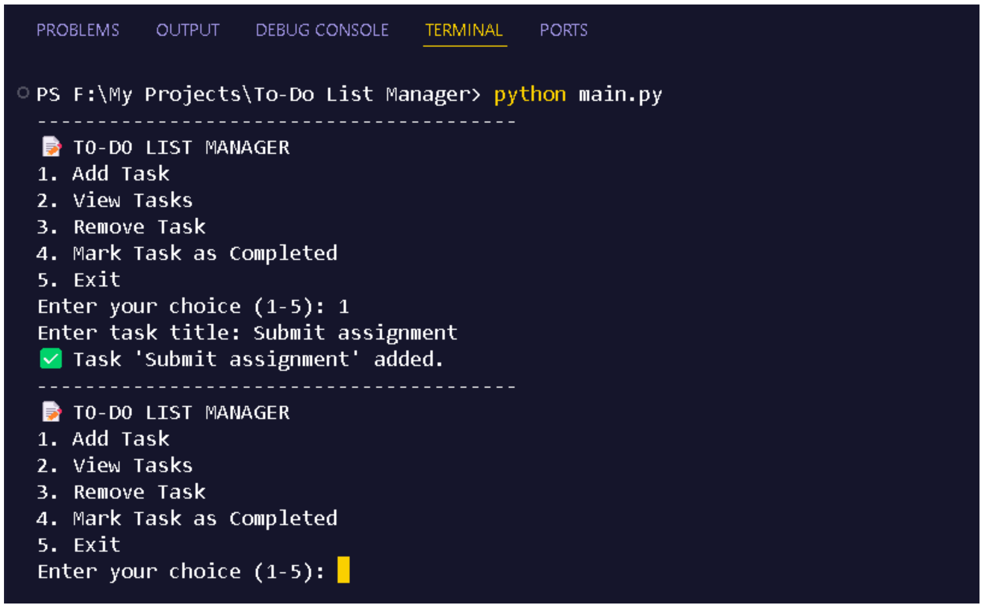
6. Key Features Implemented

- Add a new task
- View all tasks
- Remove a task by index
- Mark a task as completed
- Save/load tasks using JSON
- Error handling for invalid inputs

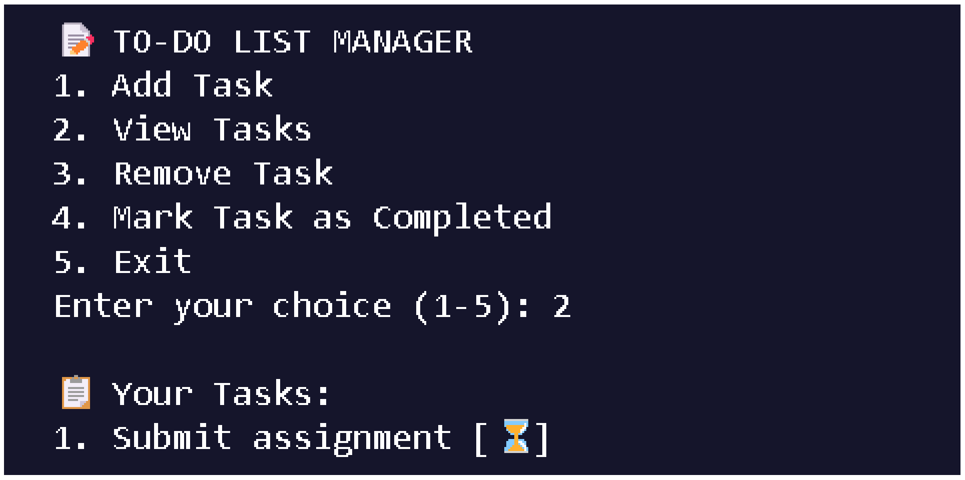
7. Sample Output



```
PS F:\My Projects\To-Do List Manager> python main.py
----------------------------------------
📝 TO-DO LIST MANAGER
1. Add Task
2. View Tasks
3. Remove Task
4. Mark Task as Completed
5. Exit
Enter your choice (1-5):
```

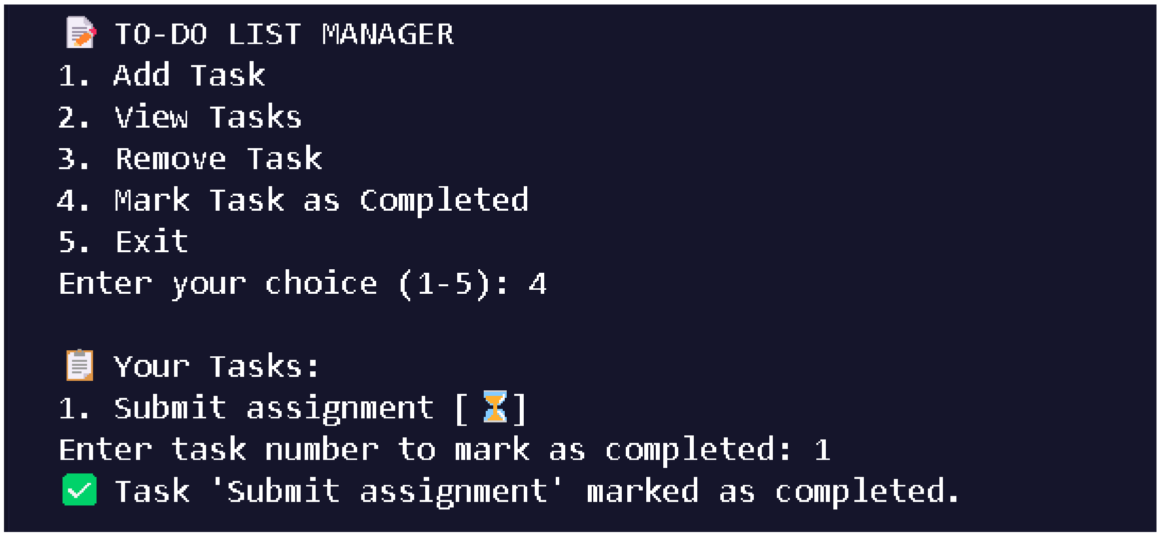


```
PS F:\My Projects\To-Do List Manager> python main.py
----------------------------------------
📝 TO-DO LIST MANAGER
1. Add Task
2. View Tasks
3. Remove Task
4. Mark Task as Completed
5. Exit
Enter your choice (1-5): 1
Enter task title: Submit assignment
✅ Task 'Submit assignment' added.
----------------------------------------
📝 TO-DO LIST MANAGER
1. Add Task
2. View Tasks
3. Remove Task
4. Mark Task as Completed
5. Exit
Enter your choice (1-5):
```



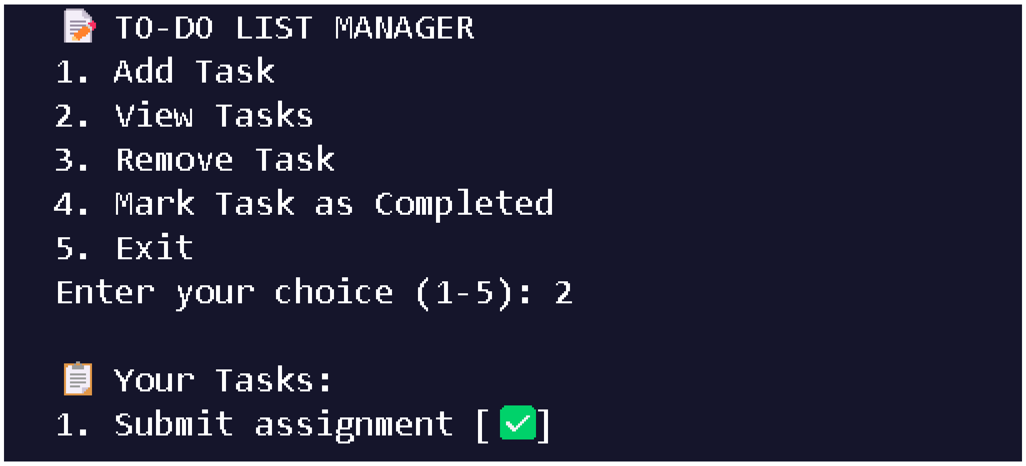
```
📝 TO-DO LIST MANAGER
1. Add Task
2. View Tasks
3. Remove Task
4. Mark Task as Completed
5. Exit
Enter your choice (1-5): 2

📋 Your Tasks:
1. Submit assignment [⏳]
```



```
📝 TO-DO LIST MANAGER
1. Add Task
2. View Tasks
3. Remove Task
4. Mark Task as Completed
5. Exit
Enter your choice (1-5): 4

📋 Your Tasks:
1. Submit assignment [⏳]
Enter task number to mark as completed: 1
✅ Task 'Submit assignment' marked as completed.
```



```
 📝 TO-DO LIST MANAGER
1. Add Task
2. View Tasks
3. Remove Task
4. Mark Task as Completed
5. Exit
Enter your choice (1-5): 2

 📋 Your Tasks:
1. Submit assignment [✅]
```

8. Error Handling

- Checks for invalid menu options
- Prevents blank task entries
- Catches out-of-range task indices
- Handles invalid numerical inputs with try-except

9. Testing

Manual Test Scenarios:

Scenario	Expected Result
Add a task	Task appears in list
Remove task using correct index	Task is deleted
Mark a task as completed	Status changes to ✅
Enter invalid menu option	Shows error and re-prompts
Enter invalid index to remove/mark	Shows invalid input warning

10. Pseudocode Summary

```
START
↓
DISPLAY MENU
↓
GET USER INPUT
↓
IF 1 → Add Task
IF 2 → View Tasks
IF 3 → Remove Task
IF 4 → Mark as Completed
IF 5 → Save & Exit
REPEAT UNTIL EXIT
```

11. Conclusion

This CLI-based To-Do List Manager project serves as a strong foundational exercise in Python programming.
It demonstrates practical skills in modular coding, data handling, input validation, and interactive CLI development.
The codebase is clean, commented, and designed to be scalable for future extensions such as due dates, priorities, or file syncing.