

Table of Contents:

1. Introduction
2. Project Overview
3. Case Study: Solving My Daily Life Problem
4. Implementation Details
5. Screenshots and Outputs
6. Conclusion

1. Introduction

In this project, I developed a To-Do List Application using Python. The application aims to provide me with a simple yet effective tool to manage my tasks, whether for personal or professional use.

2. Project Overview

The To-Do List Application allows me to:

Add tasks to my list

Remove tasks from the list

Mark tasks as completed

View the list of tasks, including their completion status

3. Case Study: Solving My Daily Life Problem

Problem Statement:

In my busy life, it's often challenging to keep track of all the tasks I need to accomplish. This can lead to missed deadlines and a sense of disorganization.

Solution: The To-Do List Application provides a solution by allowing me to input tasks and categorize them as necessary. I can mark tasks as completed, helping me keep track of my progress and ensuring that important tasks are not overlooked.

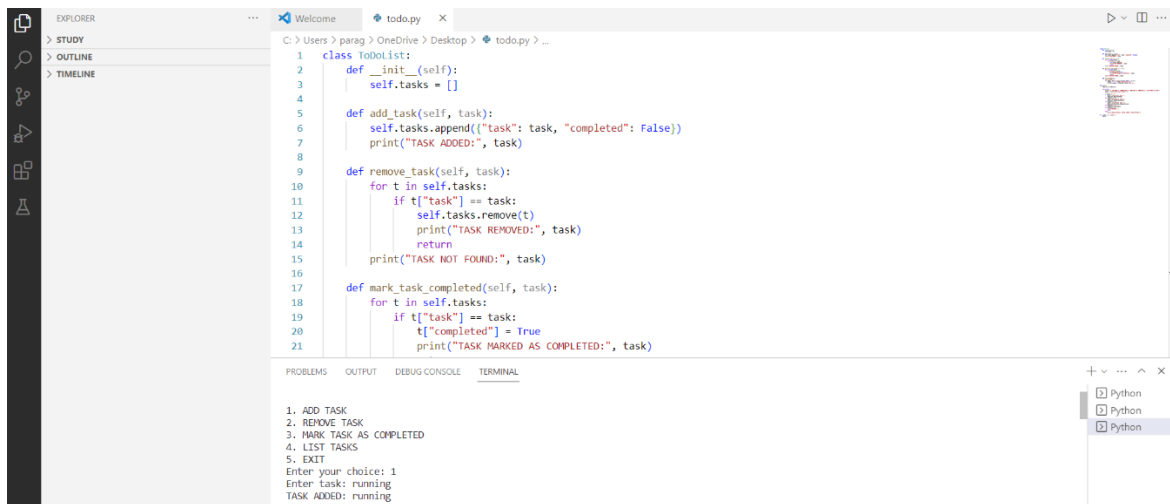
4. Implementation Details

I implemented the project in Python, featuring a command-line interface for easy interaction. I organized the code using a `ToDoList` class that encapsulates these functionalities.

5. Screenshots and Outputs

Adding a Task:

Add Task

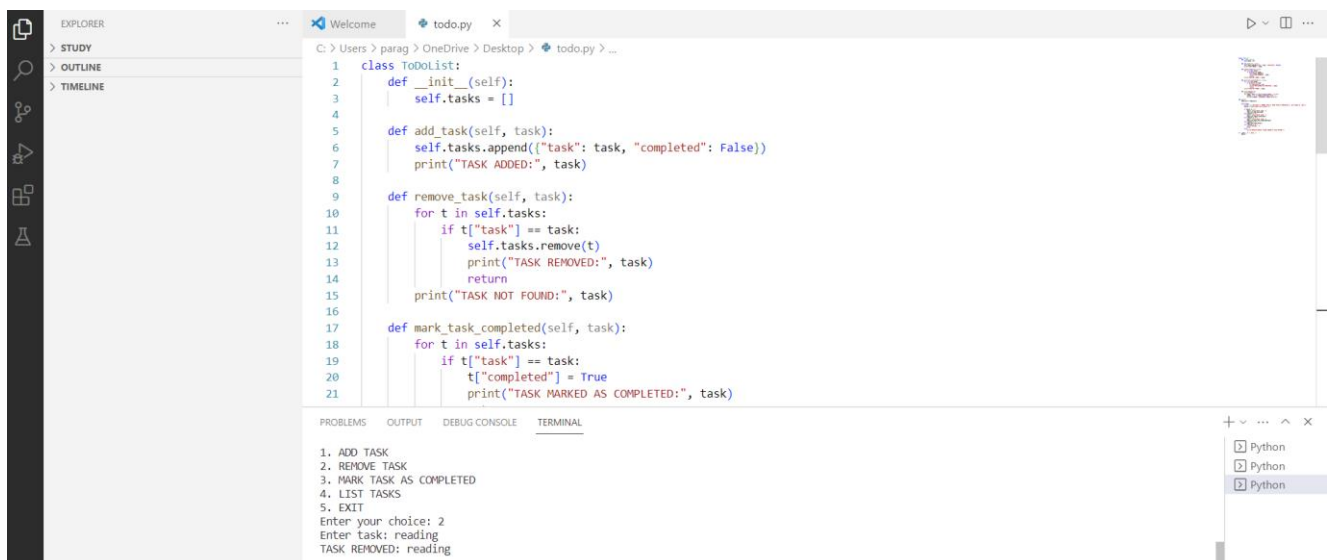


```
1 class ToDoList:
2     def __init__(self):
3         self.tasks = []
4
5     def add_task(self, task):
6         self.tasks.append({"task": task, "completed": False})
7         print("TASK ADDED:", task)
8
9     def remove_task(self, task):
10        for t in self.tasks:
11            if t["task"] == task:
12                self.tasks.remove(t)
13                print("TASK REMOVED:", task)
14                return
15            print("TASK NOT FOUND:", task)
16
17    def mark_task_completed(self, task):
18        for t in self.tasks:
19            if t["task"] == task:
20                t["completed"] = True
21                print("TASK MARKED AS COMPLETED:", task)
```

1. ADD TASK
2. REMOVE TASK
3. MARK TASK AS COMPLETED
4. LIST TASKS
5. EXIT
Enter your choice: 1
Enter task: running
TASK ADDED: running

Removing a Task:

Remove Task

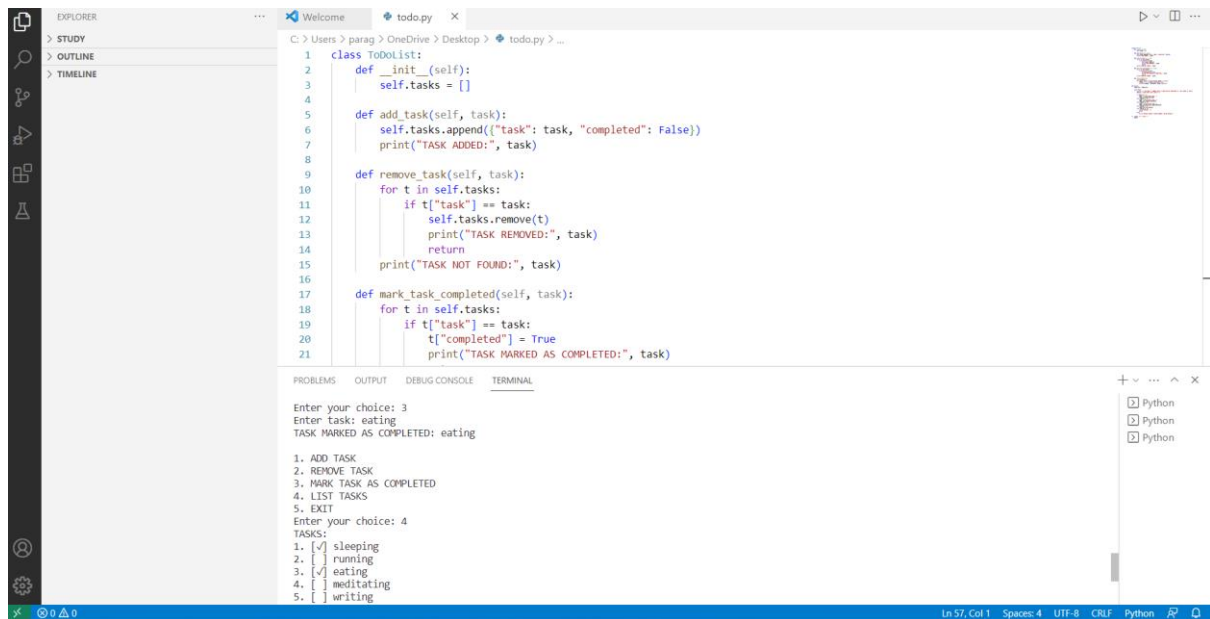


```
1 class ToDoList:
2     def __init__(self):
3         self.tasks = []
4
5     def add_task(self, task):
6         self.tasks.append({"task": task, "completed": False})
7         print("TASK ADDED:", task)
8
9     def remove_task(self, task):
10        for t in self.tasks:
11            if t["task"] == task:
12                self.tasks.remove(t)
13                print("TASK REMOVED:", task)
14                return
15            print("TASK NOT FOUND:", task)
16
17    def mark_task_completed(self, task):
18        for t in self.tasks:
19            if t["task"] == task:
20                t["completed"] = True
21                print("TASK MARKED AS COMPLETED:", task)
```

1. ADD TASK
2. REMOVE TASK
3. MARK TASK AS COMPLETED
4. LIST TASKS
5. EXIT
Enter your choice: 2
Enter task: reading
TASK REMOVED: reading

Marking Task as Completed:

Mark Task



The screenshot shows the Visual Studio Code editor with a file named `todo.py` open. The code defines a `ToDoList` class with methods `__init__`, `add_task`, `remove_task`, and `mark_task_completed`. The `mark_task_completed` method iterates through the `self.tasks` list and sets the `completed` attribute to `True` for the specified task. The terminal window at the bottom shows the program's execution: it prompts for a choice (3), then a task ('eating'), and prints 'TASK MARKED AS COMPLETED: eating'. It then displays the current list of tasks: sleeping, running, eating, meditating, and writing.

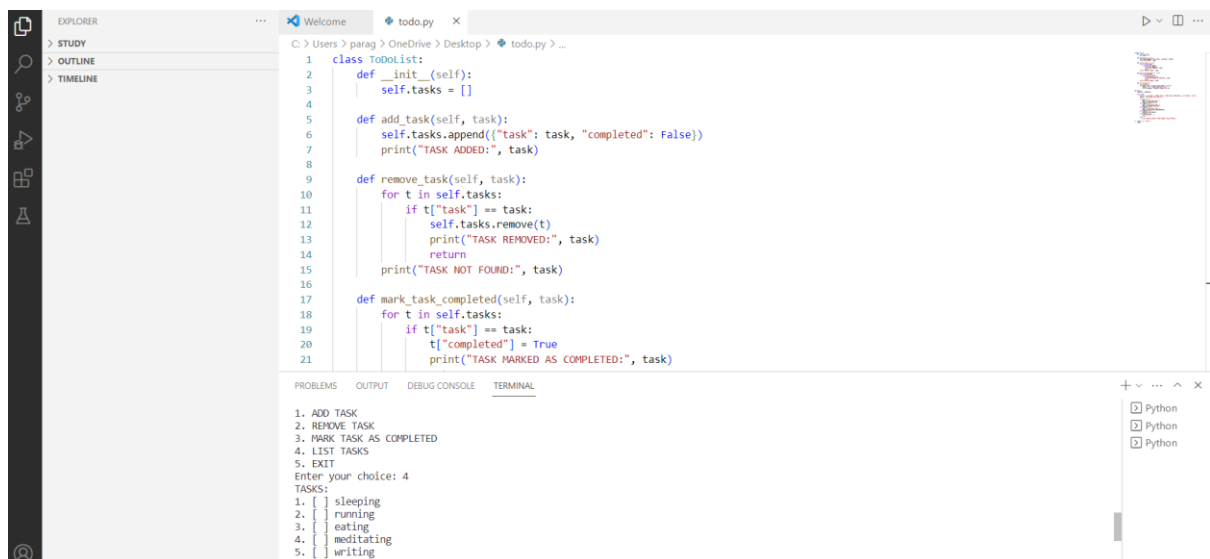
```
1 class ToDoList:
2     def __init__(self):
3         self.tasks = []
4
5     def add_task(self, task):
6         self.tasks.append({"task": task, "completed": False})
7         print("TASK ADDED:", task)
8
9     def remove_task(self, task):
10        for t in self.tasks:
11            if t["task"] == task:
12                self.tasks.remove(t)
13                print("TASK REMOVED:", task)
14                return
15        print("TASK NOT FOUND:", task)
16
17    def mark_task_completed(self, task):
18        for t in self.tasks:
19            if t["task"] == task:
20                t["completed"] = True
21                print("TASK MARKED AS COMPLETED:", task)
```

Enter your choice: 3
Enter task: eating
TASK MARKED AS COMPLETED: eating

1. ADD TASK
2. REMOVE TASK
3. MARK TASK AS COMPLETED
4. LIST TASKS
5. EXIT
Enter your choice: 4
TASKS:
1. [x] sleeping
2. [] running
3. [x] eating
4. [] meditating
5. [] writing

Listing Tasks:

List Tasks



This screenshot is identical to the previous one, showing the same code and terminal output. The terminal shows the program listing the tasks after the user enters choice 4.

6.The Code for the Project:

```
class ToDoList:
    def __init__(self):
        self.tasks = []

    def add_task(self, task):
        self.tasks.append({"task": task, "completed": False})
        print("TASK ADDED:", task)
```

```

def remove_task(self, task):
    for t in self.tasks:
        if t["task"] == task:
            self.tasks.remove(t)
            print("TASK REMOVED:", task)
            return
    print("TASK NOT FOUND:", task)

def mark_task_completed(self, task):
    for t in self.tasks:
        if t["task"] == task:
            t["completed"] = True
            print("TASK MARKED AS COMPLETED:", task)
            return
    print("TASK NOT FOUND:", task)

def list_tasks(self):
    print("TASKS:")
    for index, task in enumerate(self.tasks, start=1):
        status = "✓" if task["completed"] else " "
        print(f"{index}. [{status}] {task['task']}")

def main():
    todo_list = ToDoList()

    while True:
        print("\n1. ADD TASK\n2. REMOVE TASK\n3. MARK TASK AS COMPLETED\n4. LIST TASKS\n5. EXIT")
        choice = input("Enter your choice: ")

        if choice == "1":
            task = input("Enter task: ")
            todo_list.add_task(task)
        elif choice == "2":
            task = input("Enter task: ")
            todo_list.remove_task(task)
        elif choice == "3":
            task = input("Enter task: ")
            todo_list.mark_task_completed(task)
        elif choice == "4":
            todo_list.list_tasks()
        elif choice == "5":
            print("EXITING...")
            break
        else:
            print("INVALID CHOICE. PLEASE CHOOSE A VALID OPTION.")

if __name__ == "__main__":
    main()

```

7. Conclusion

The To-Do List Application project demonstrates my practical use of programming to solve everyday problems. It provides me with a simple and efficient way to manage tasks, helping me stay organized and focused on my goals. This project showcases the power of Python in creating functional and user-friendly applications.

By organizing my tasks and providing a clear overview of pending and completed items, the To-Do List Application empowers me to take control of my daily tasks and responsibilities.

In this PDF document, I've presented the project's case study, implementation details, screenshots of the code with outputs, and a conclusion summarizing the project's significance.