**Lab Project Report**

**OOP II: PYTHON**

**Semester: Spring 2024**

**Student Name: Sazzad Mahmud Joy**

**Student ID: 221-15-5777**

**Section: 61-A2**

**Title of the Project: Task Manager**

**Date of Submission: -05-2024**

**Project Title: Task Manager**

**Introduction:**

In a time when the World is running fast, managing our tasks efficiently is imperative to become successful person. A lifetime to-do list is great, and an annual plan has its place as well, but the more responsibilities we have — especially for teams of people working together — depth of thought around organization, prioritization, and task tracking becomes what separates successful organizations. Since traditional methods, like pen on paper or simple to do lists are not structured and flexible enough for managing the complexity of tasks with your schedule requirements.

**Motivation:**

There are many reasons I want to work on this project, but I just need a good-ol Task Manager.

* In all: A beautifully architected task management system helps users to manage their time and resources for far more productivity, both in a personal and professional setting.
* Common Needs: Everyone and every organization struggles with managing multiple projects, deadlines, and events. The goal of this system is to deliver an all-inclusive answer for automating these procedures.
* OOP, File Handling and User Interface Design Learning Opportunity: I can show my skills in this project because here we have the chance to use an OOP language for file handling purposes. It also allows me to explore more advanced features such as how to serialize and deserialize tasks with pandas.
* User-Driven Design: Made flexible allows for a wide variety of users by including multiple task types (Todo, Deadline, Event)

**Project Objectives:**

Objective of TMS (Task Management System) project To design and build a comprehensive yet user-friendly app that serves the multitude of requirements for users. The following are the goals:

Objective 1: Flexible Task Management Systems:

Develop a system for different kinds of tasks such as Todo type tasks, Deadline type tasks and Event

type tasks. Ensure that the system can support key characteristics and attributes of each type of work

such as start-to-end times for event tasks or deadlines for deadline activities.

Objective 2: Boost User Participation and Usability:

A user-friendly interface for users to add, remove and list tasks. Implement processes to update task

descriptions and statuses so users know the current status of each Task.

Objective 3: Encourage the Reuse and Maintainability of Code:

Apply object-oriented programming principles in a modular and maintainable way codebase. Leverage

inheritance and abstract base classes to reduce duplicated code, making it easier for future additions or

changes.

Objective 4: Add Task Management Features

Allow users to set a task as "Completed", which encourages the real completion and tracking of tasks.

We allow users to look up each task with detailed information. Let them delete tasks from the list, so

that it remains up-to-date.

**Project Description:**

To help users manage and plan their tasks were created a full software program called the Task Management System. This project addresses the issues that many individuals and teams, need to keep track of; tasks, due dates events etc. Providing a structured and easy interface, in addition to organization of the daily activities that ensures the total control about the time spent on any task brought productivity gains and reduction of stress for users.

Key Features:

* Type of task : The system supports three main types of tasks (Todo, Deadline and Event) Each type of task is created to serve a particular user and the same has unique features for its category (start/end times for Event tasks or deadlines for Deadline tasks).
* Easy-To-Use Interface : Users can instantly add new tasks with a user-friendly interface by providing information about the task such as title, description, priority, and due date. With this system, we can also view, update and delete tasks which are the basic functions of any task management so everything is easy to difficult.
* Status Management : The user can update task status according to its current state (ex : Pending, Completed, etc )
* More Information on each task : allows users to see every detail of the tasks, this way they have all the information required for them to understand and manage their duties.

Workflow:

* Add Tasks: This allows users to create new tasks by inputting the task's title, description, priority and due date. It also stores additional specifics for those tasks with deadlines or events such as when it starts and ends.
* Manage Tasks : Users have visibility to all the tasks and can also update their status or delete them.It helps in enabling accessed tasks provide users view of all their duties, at the same time contributors can easily browse through its clean sheet to see and manage some of their expectations.
* Save and Load Tasks : To guarantee data persistence, the system automatically saves tasks to an Excel file. The system loads tasks from the Excel file upon startup, enabling users to carry on with their task management without interruption.

Benefits:

Productivity Boost: Enables users to focus on more meaningful higher-value work by organizing their tasks, allowing them to prioritize and complete high-priority ones.

Improved Organization: The system organizes different types of duties, so the chances to miss work or important events are reduced.

Empowered Users: Keep track of each scheduled item and what's in your to-do list with ease, now you can easily manage your responsibilities due its user-friendly interfaces and detailed tasks.

In the end, a task management system is designed to provide users with a reliable, efficient and user-friendly platform for managing tasks effectively in order to help them reach their goals and maintain an organized workflow.

**Project User:**

The Task Management System is specifically targeted at people who are diverse but all want a streamlined and organised way to save time managing their tasks. Users of this system are primarily:

Individuals:

* Students: Managing homework, assignments, study schedules and project deadlines.
* Professionals: Agendas of work tasks, deadlines, meetings and personal errands.
* Freelancers : where they needs to track project deadlines, client meetings and also their personal tasks.

Teams:

* Project Teams: Organizing by activities and deadlines for running projects to make clear what needs to be done, by whom.
* Divisions: Handling interdepartmental goals, distensions and games, facilitating cross-team coercion and efficacy.

Organizations:

* Small Businesses :Maintaining a track of business tasks, employee schedules, deadlines and company events.
* Non-Profit Organizations :Scheduling volunteer activities, fundraising events & deadlines to complete various efforts.

Event Planners:

* Event Coordinators: They organize and plan everything for the events (from deadlines to schedules of anything).

As educators and Academic Institutions as:

* Teachers & Professors: Teachers and professors plan the lessons, assignments, exams and other academic activities.
* School Administrator : Set School events, deadlines for administrative tasks, and meetings for Administrators.

Project Managers:

* Project Leaders: Monitoring project timelines, assignment of tasks and review status updates to achieve the agreed upon project milestones.

**Project Features:**

The capabilities of the Task Management System are specifically tailored to meet both its wide-reaching customer base and a wealth of other senses, guaranteeing smooth task processes. Here are the key features

* Task Creation: Includes adding a new task with title, description, due date and priority.
* View Tasks: Show details about each task.
* Update Task Status: Administrative data (e.g., "Pending" to "Completed").
* Task Deletion: Delete the ones you do not need anymore
* Save and Load tasks: Tasks are automatically saved to an Excel file, which is loaded when the system boots.

**Addressing OOP Concepts:**

I have created the Task Management System using Object-Oriented Programming (OOP) to following modularity, reusability, maintainable code. How The Project Follows of OOP Concepts

* **Encapsulation:**

The task-related data (title, description, due date, priority, status) is encapsulated within

Task classes. This ensures that the task's internal state can only be modified through

defined methods, promoting data integrity.

* **Inheritance:**

A base abstract class Task is created, from which specific task types (TodoTask,

DeadlineTask, EventTask) inherit. This reduces code duplication and allows for extending

functionality easily.

* **Polymorphism:**

Polymorphism is achieved through method overriding. The derived classes (TodoTask,

DeadlineTask, EventTask) override the display\_details and to\_dict methods from the Task

base class to provide specific implementations.

* **Abstraction:**

The Task class uses abstract methods (display\_details and to\_dict), ensuring that any class

inheriting from Task must implement these methods. This enforces a consistent interface

for all task types.

**Conclusion:**

For the convenience of various user groups (people, teams and organization) to effectively organize and manage their tasks, Task Management System is an important weapon. Which enhances the work productivity and management with a user friendly interface to create, manage & track Todo, Deadline tasks and even events. It employ features of Strong Object-Oriented Programming principles which make the system more robust and easier to maintain, reusable and modular as well. This is done by embedding Excel fluently. And future additions — including notifications, collaboration features, and a mobile app will make it more effective and useful than just about any other task manager on the planet with a complete customizable solution that streamlines workflow.

**Coding:**

import pandas as pd

from abc import ABC, abstractmethod

class Task(ABC):

    def \_\_init\_\_(self, title, description, due\_date, priority):

        self.title = title

        self.description = description

        self.due\_date = due\_date

        self.priority = priority

        self.\_\_status = "Pending"

    def get\_status(self):

        return self.\_\_status

    def update\_status(self, new\_status):

        self.\_\_status = new\_status

    @abstractmethod

    def display\_details(self):

        pass

    @abstractmethod

    def to\_dict(self):

        pass

class TodoTask(Task):

    def display\_details(self):

        print(f"Title: {self.title}")

        print(f"Description: {self.description}")

        print(f"Due Date: {self.due\_date}")

        print(f"Priority: {self.priority}")

        print(f"Status: {self.get\_status()}")

    def to\_dict(self):

        return {

            "type": "Todo",

            "title": self.title,

            "description": self.description,

            "due\_date": self.due\_date,

            "priority": self.priority,

            "status": self.get\_status()

        }

class DeadlineTask(Task):

    def \_\_init\_\_(self, title, description, due\_date, priority, deadline):

        super().\_\_init\_\_(title, description, due\_date, priority)

        self.deadline = deadline

    def display\_details(self):

        print(f"Title: {self.title}")

        print(f"Description: {self.description}")

        print(f"Due Date: {self.due\_date}")

        print(f"Priority: {self.priority}")

        print(f"Status: {self.get\_status()}")

        print(f"Deadline: {self.deadline}")

    def to\_dict(self):

        return {

            "type": "Deadline",

            "title": self.title,

            "description": self.description,

            "due\_date": self.due\_date,

            "priority": self.priority,

            "status": self.get\_status(),

            "deadline": self.deadline

        }

class EventTask(Task):

    def \_\_init\_\_(self, title, description, due\_date, priority, start\_time, end\_time):

        super().\_\_init\_\_(title, description, due\_date, priority)

        self.start\_time = start\_time

        self.end\_time = end\_time

    def display\_details(self):

        print(f"Title: {self.title}")

        print(f"Description: {self.description}")

        print(f"Due Date: {self.due\_date}")

        print(f"Priority: {self.priority}")

        print(f"Status: {self.get\_status()}")

        print(f"Start Time: {self.start\_time}")

        print(f"End Time: {self.end\_time}")

    def to\_dict(self):

        return {

            "type": "Event",

            "title": self.title,

            "description": self.description,

            "due\_date": self.due\_date,

            "priority": self.priority,

            "status": self.get\_status(),

            "start\_time": self.start\_time,

            "end\_time": self.end\_time

        }

class TaskManager:

    def \_\_init\_\_(self):

        self.tasks = []

        self.load\_tasks()

    def add\_task(self, task):

        self.tasks.append(task)

    def remove\_task(self, task):

        if task in self.tasks:

            self.tasks.remove(task)

            print("Task removed successfully.")

        else:

            print("Task not found.")

    def list\_tasks(self):

        if not self.tasks:

            print("No tasks.")

        else:

            for task in self.tasks:

                task.display\_details()

                print()

    def save\_tasks(self):

        tasks\_dict\_list = [task.to\_dict() for task in self.tasks]

        df = pd.DataFrame(tasks\_dict\_list)

        df.to\_excel("/content/drive/MyDrive/Colab Notebooks/tasks.xlsx", index=False)

        print("Tasks saved to /content/drive/MyDrive/Colab Notebooks/tasks.xlsx")

    def load\_tasks(self):

        try:

            df = pd.read\_excel("/content/drive/MyDrive/Colab Notebooks/tasks.xlsx")

            for i, row in df.iterrows():

                task\_type = row["type"]

                if task\_type == "Todo":

                    task = TodoTask(row["title"], row["description"], row["due\_date"], row["priority"])

                elif task\_type == "Deadline":

                    task = DeadlineTask(row["title"], row["description"], row["due\_date"], row["priority"], row["deadline"])

                elif task\_type == "Event":

                    task = EventTask(row["title"], row["description"], row["due\_date"], row["priority"], row["start\_time"], row["end\_time"])

                else:

                    continue

                task.update\_status(row["status"])

                self.tasks.append(task)

        except FileNotFoundError:

            print("/content/drive/MyDrive/Colab Notebooks/tasks.xlsx not found")

def create\_task():

    title = input("Enter task title: ")

    description = input("Enter task description: ")

    due\_date = input("Enter due date (YYYY-MM-DD): ")

    priority = input("Enter priority: ")

    task\_type = input("Enter task type (Todo/Deadline/Event): ").capitalize()

    if task\_type == "Todo":

        return TodoTask(title, description, due\_date, priority)

    elif task\_type == "Deadline":

        deadline = input("Enter deadline (YYYY-MM-DD): ")

        return DeadlineTask(title, description, due\_date, priority, deadline)

    elif task\_type == "Event":

        start\_time = input("Enter start time: ")

        end\_time = input("Enter end time: ")

        return EventTask(title, description, due\_date, priority, start\_time, end\_time)

    else:

        print("Invalid task type.")

        return None

task\_manager = TaskManager()

while True:

    print("\n1. Add Task")

    print("2. Remove Task")

    print("3. List Tasks")

    print("4. Exit")

    choice = input("Enter your choice: ")

    if choice == "1":

        task = create\_task()

        if task:

            task\_manager.add\_task(task)

            print("Task added successfully.")

    elif choice == "2":

        title = input("Enter task title to remove: ")

        task\_to\_remove = None

        for task in task\_manager.tasks:

            if task.title == title:

                task\_to\_remove = task

                break

        if task\_to\_remove:

            task\_manager.remove\_task(task\_to\_remove)

        else:

            print("Task not found.")

    elif choice == "3":

        print("\nAll tasks:")

        task\_manager.list\_tasks()

    elif choice == "4":

        task\_manager.save\_tasks()

        print("Exiting program.")

        break

    else:

        print("Invalid choice. Please try again.")