**Task Management System**

Table of Contents

1. Introduction
2. System Overview
3. Main Features of the System

* **Load Task Function**
* **Save Task Function**
* **Show Task Function**
* **Add Task Function**
* **Mark Completed Function**
* **Delete Task Function**
* **Prioritize Task Function**
* **Set Due Date Function**
* **Categorize Task Function**
* **Search Task Function**
* **Main Function**

1. Function Testing
2. Further Enhancement
3. Conclusion

**Introduction**This task management system is a Python-based program that offers customers an easy-to-use and effective way to organize and manage their activities. The purpose of this report is to examine all facts of the system, such as its architecture, features, and the testing procedure used to guarantee its dependability and efficiency.

**System Overview**

The Task Management System is fundamentally constructed with basic Python features. The system stores task data in a tasks.txt file and uses a file-based method for task persistence. Tasks for loading and storing, displaying, adding, removing, marking as done, prioritizing, assigning deadlines, classifying, and keyword-based task searches are among the main features of the system.

**Main Features of the System**

The goal of the task management system is to provide users with an extensive range of features so they can effectively handle their jobs. The system's main features include the ability to load and save tasks, view tasks, add tasks, mark tasks as finished, delete tasks, prioritize tasks, establish due dates, classify tasks, and search for tasks using keywords.

1. Load Task Function

The load\_tasks() function in the Task Management System fulfils the vital purpose of reading task data from the tasks.txt file. This function uses file handling to retrieve tasks that have already been saved, turning the contents of the file into a list of tasks using the eval function. The main goal is to make sure that tasks continue to exist between sessions. If there is a FileNotFoundError, the function will return an empty list politely, ensuring that the system can handle situations in which task data is missing.

1. Save Task Function

Writing tasks to the tasks.txt file is how the save\_tasks() function stores them permanently. To ensure that tasks added or updated during a session are kept for later access, this function is a complement to load\_tasks(). Through the function's conversion of the tasks list into a string representation, the file handling mechanism is smoothly integrated, offering users a consistent and seamless task management experience.

1. Show Task Function

The purpose of the show\_tasks() method is to display tasks in an approachable manner. The function shows important information such task name, completion status, due date, priority, and category by looping over the tasks list. This feature improves the user's comprehension and task management by contributing to an intuitive and clear user interface. When there are no tasks to complete, the function shows a message to make sure the user is comfortable.

1. Add Task Function

Adding tasks to the Task Management System is made easier with the help of the add\_task(name) function. This feature makes it easy for users to add new tasks to the system by asking them for a task name, generating a new task dictionary with the necessary information, and adding it to the tasks list. The added task's durability is guaranteed by the ensuing call to the save tasks() method, preserving a consistent and dependable task management experience.

1. Mark Completed Function

Users can mark certain jobs as done with the mark\_completed(name) method. This feature helps users recognize and keep track of completed activities by looking for tasks with the same name and changing their completion status to True. The revised completion status is saved permanently, reflecting the user's activities over sessions, thanks to the connection with the save\_tasks() method.

1. Delete Task Function

Removing tasks from the Task Management System is the sole purpose of the delete\_task(name) function. Users can effectively reduce their task list by looking for tasks with matching names and eliminating them from the list. The removal of tasks is guaranteed to be permanently preserved by the ensuing call to save\_tasks(), preserving the task management system's accuracy and consistency.

1. Prioritize Task Function

The option to allocate priority levels (High, Medium, Low) to individual activities is introduced via the prioritize\_task(name, priority) function. This feature improves task organisation by looking for tasks with the same name and adjusting the tasks' priority according to user input. The incorporation of save\_tasks() guarantees the permanent storage of the set priority, enabling users to efficiently prioritise and manage tasks.

1. Set Due Date Function

Users can specify deadlines for certain tasks by using the set\_due\_date(name, due\_date) method. Users can make a visual task management timeline by looking for tasks with the same name and giving them a deadline. Users may efficiently track and manage their tasks thanks to the subsequent call to save\_tasks(), which guarantees the persistent storing of due dates.

1. Categorize Task Function

Task organization capabilities are introduced by the categorize\_task(name, category) function. Jobs can be categorized by users, making it easier to group and arrange jobs according to their characteristics or purposes. The function looks for tasks whose names match and puts them in the designated category. The task classification is preserved permanently thanks to the interaction with save\_tasks(), which enhances the efficiency and organization of task management.

1. Search Task Function

The function search\_tasks(keyword) allow users to search for tasks based on keywords, which improves task retrieval capabilities. When no matching tasks are identified, the function displays a corresponding notice. Otherwise, it searches the tasks list for tasks whose names contain the supplied keyword. This feature makes it easier to find particular activities inside the system, which makes it more effective and user-friendly.

1. Main Function

The main() function provides a menu-driven task management interface to users, acting as the core hub of the Task Management System. The main() method coordinates user interactions by providing users with a variety of alternatives, each of which corresponds to a particular functionality. By invoking the appropriate functions in response to user selections, it guarantees a smooth and straightforward user experience while providing a thorough and approachable method of job management.

**Functions Testing**

Software testing is a critical stage that guarantees the functionality, performance, and dependability of the system. A thorough testing procedure was implemented for the Task Management System in order to detect and resolve possible problems, verify features, and provide a smooth user experience.

Verifying the "Mark Task as Completed" feature's capacity to modify a task's completion status was the main goal of testing. To verify that the system appropriately changed a task's completion status from "Incomplete" to "Completed" when the user chose this option, a unit test was created. The test made use of a dummy task whose initial completion status was known, and it checked that the completion status was changed correctly after using the "Mark Task as Completed" feature. Unit tests were used to verify the "Search Tasks" feature, which examined the system's capacity to look up tasks using keywords entered by the user. The durability and accuracy of the search capability were tested. To make sure it could delete tasks from the system, the "Delete Task" functionality was put through a rigorous testing process. To make sure the system appropriately recorded and maintained the priority information, unit tests were designed to accept a task name and a priority level. Tests covering a range of priority levels, including "High," "Medium," and "Low," validated the prioritization feature's accuracy and versatility. To make sure it could correctly assign due dates to tasks, the "Set Due Date" functionality was tested. To ensure that the task was updated correctly with the given due date information, tests were created to allow users to submit a task name and a specified date. The system's response was analyzed.

**Further Enhancement**

Although the Task Management System has features that are appropriate for personal use, there are a few improvements that might be made to increase its functionality and user experience:

* *Task Remainders and Notifications:* Adding a notification system for approaching deadlines or unfinished work could help increase task management effectiveness.
* *Task Dependencies:* Users would be able to comprehend the relationships between various activities more easily if a feature to indicate dependencies between tasks was included.
* *Export and Import Feature:* It might be easier to move data between systems and act as a backup if tasks had the option to export and import.

**Conclusion**

As is, the Task Management System provides a good option for those looking for a simple task management application. The system's efficacy and dependability were confirmed throughout the testing phase by comparing each functionality to pre-established test cases. Even if the system achieves its main goals, the additional improvements found point to potential directions for further work. The system could be improved to better serve customers' changing needs by adding a graphical user interface, user authentication, and other features like task reminders. All things considered, the system demonstrates the ease of use and flexibility of implementing task management with Python.