**To-Do List App**

**Problem Statement**

Students and busy individuals often struggle to stay organized and manage their daily tasks efficiently. Lack of a structured system can lead to missed deadlines, forgotten assignments, and increased stress.

**Objective**

Build a user-friendly To-Do List application using Python and Tkinter that allows users to:

* Add new tasks
* Edit existing tasks
* Delete tasks
* Mark tasks as complete/incomplete
* Optionally set priority and due date
* Save/load tasks from a JSON file

**Requirements**

* Python 3.x
* Tkinter (comes built-in with Python)

**Tools & Technologies Used**

* **Programming Language:** Python
* **GUI Library:** Tkinter
* **File Handling:** JSON

**Step-by-Step Process**

1. **Design the GUI:**
   * Use Tkinter to create the main window, task display, and button controls.
2. **Create a Task Dialog:**
   * Use simpledialog.Dialog to create a pop-up for adding/editing task details (description, priority, due date).
3. **Add Functionality:**
   * **Add Task:** Open dialog to input task data, then append to task list.
   * **Edit Task:** Select a task, modify its data using the dialog.
   * **Delete Task:** Confirm and remove task from the list.
   * **Toggle Complete:** Flip the task's completed status.
4. **Display Tasks:**
   * Use ttk.Treeview to show task details.
   * Update the view after any change using a refresh function.
5. **Handle File I/O:**
   * Load tasks from tasks.json at startup.
   * Save tasks to tasks.json on clicking Save or when exiting.
6. **Sort and Validate:**
   * Sort tasks by completion status and priority.
   * Validate user input for due dates and non-empty descriptions.

**How to Run**

1. Make sure Python is installed.
2. Save the Python file as todo\_app.py.
3. Run the file:
4. python todo\_app.py
5. Tasks are saved to and loaded from tasks.json in the same directory.

**Expected Outcome**

A fully functional desktop task manager that helps users prioritize, track, and manage daily responsibilities while practicing:

* GUI development using Tkinter
* CRUD operations
* File handling with JSON

**Challenges Faced & Solutions**

* **Validating user input:** Solved by using try-except for date parsing and input checks.
* **Task reordering:** Solved by sorting tasks before displaying them in the Treeview.
* **Persistent storage:** Used JSON file handling to save/load task data.

**Future Improvements**

* Task search and filters
* Color-code based on priority/due date
* Notifications for due tasks
* Cross-platform packaging using PyInstall.