# Task Scheduler Project Documentation

## 1. Project Overview

The Task Scheduler project is a Python-based application designed to manage tasks efficiently. Users can add, edit, delete, and execute tasks while receiving notifications for approaching deadlines. The application uses a graphical user interface (GUI) built with Tkinter and leverages data structures like min-heaps for task prioritization. Notifications are provided via system alerts and sound prompts.

## 2. Objectives

1. To develop a task management system that allows users to organize their tasks effectively.  
2. To integrate a user-friendly GUI for task input, display, and interaction.  
3. To implement deadline-based notifications with sound alerts for user convenience.  
4. To ensure data persistence for tasks using JSON file handling.

## 3. Technology Stack

1. Programming Language: Python  
2. GUI Framework: Tkinter  
3. Notifications: Plyer for system alerts  
4. Sound Alerts: Pygame  
5. Data Structures: Min-Heap (Heapq)  
6. File Handling: JSON for persistent storage

## 4. Implementation Details

### Backend Development:

The backend is built around a `TaskScheduler` class that manages tasks using a min-heap structure. It provides functionality for adding, editing, deleting, and executing tasks. Tasks are saved and loaded from a JSON file to ensure persistence.

### GUI Development:

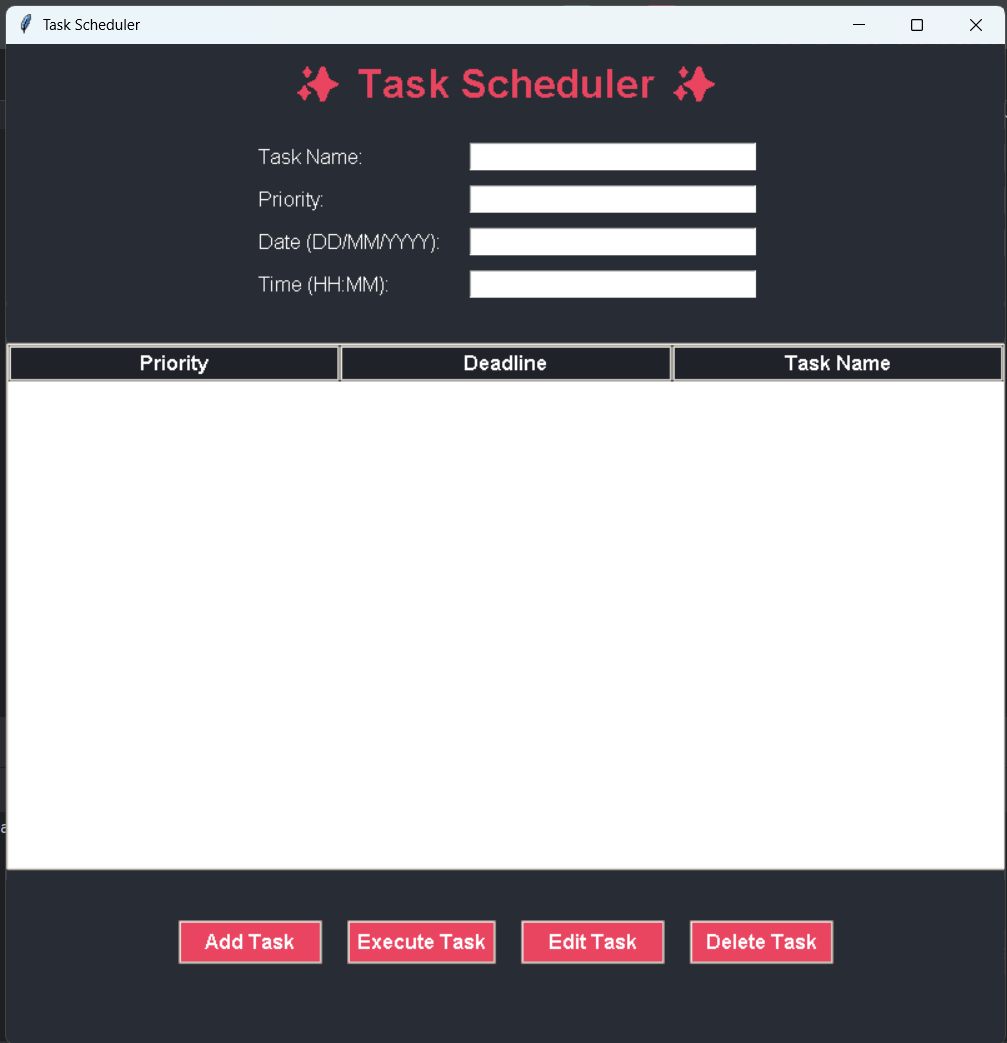
The GUI, built with Tkinter, includes input fields for task name, priority, date, and time, a Treeview for task display, and buttons for user actions such as adding, editing, and deleting tasks.

### Notifications:

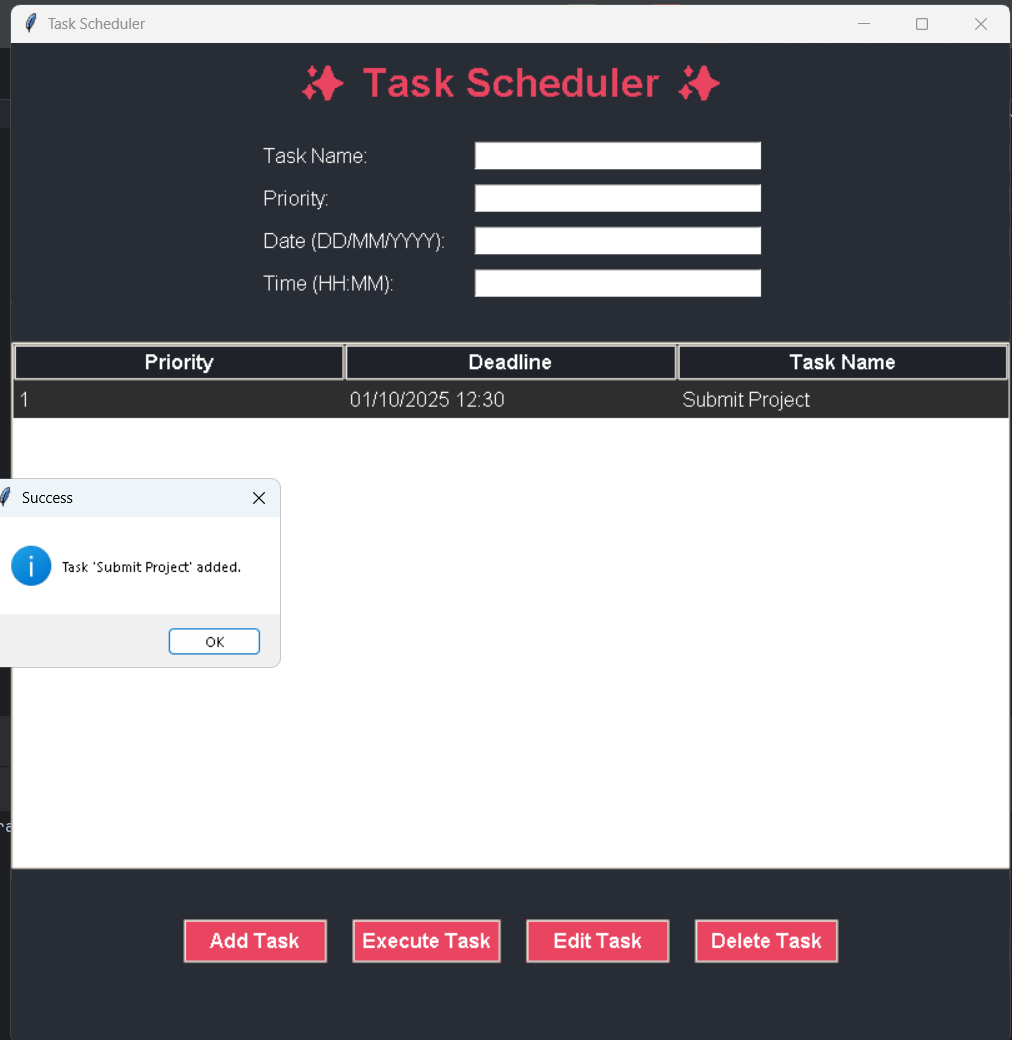
A notification system was implemented using Plyer for system alerts and Pygame for sound notifications. Background threading ensures the GUI remains responsive while monitoring deadlines.

## 5. Screenshots

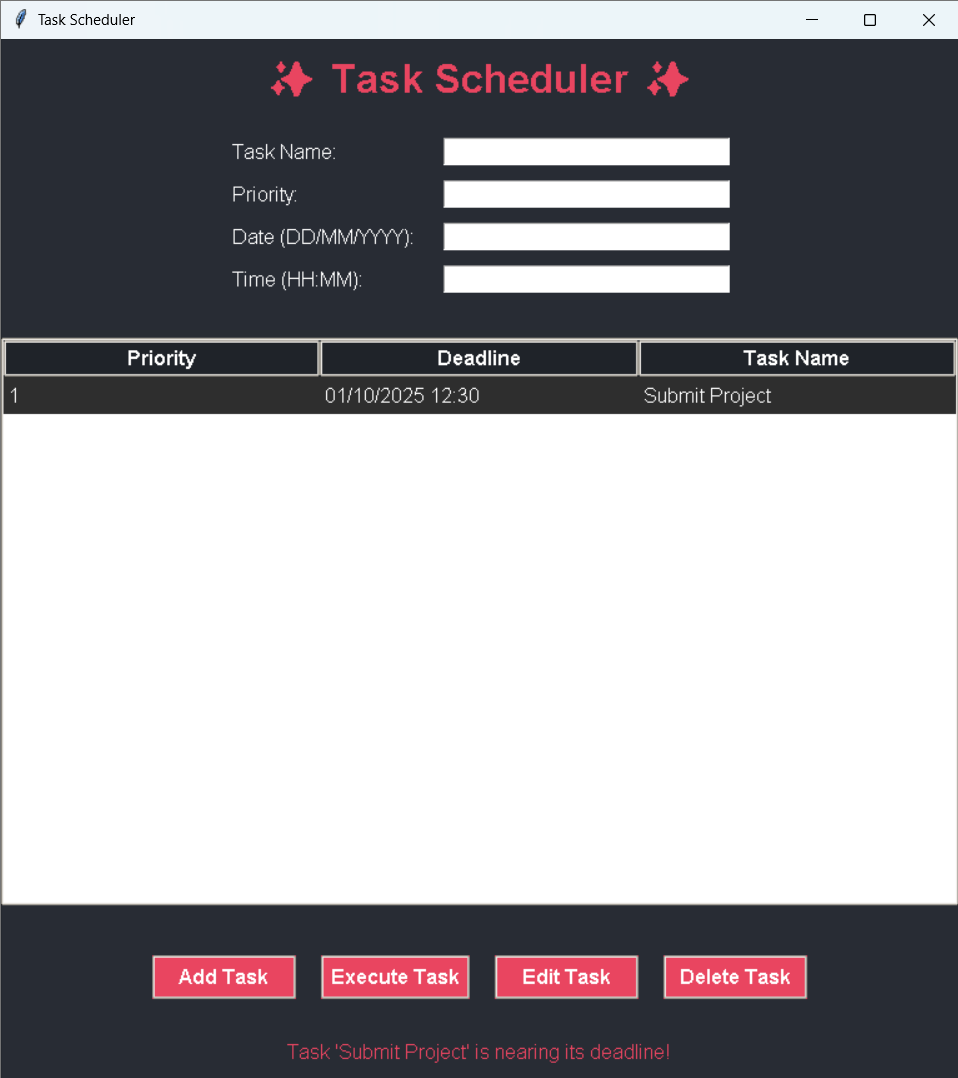
Include screenshots of the application here. Screenshots should showcase:  
1. The main GUI layout.



2. Adding a task.

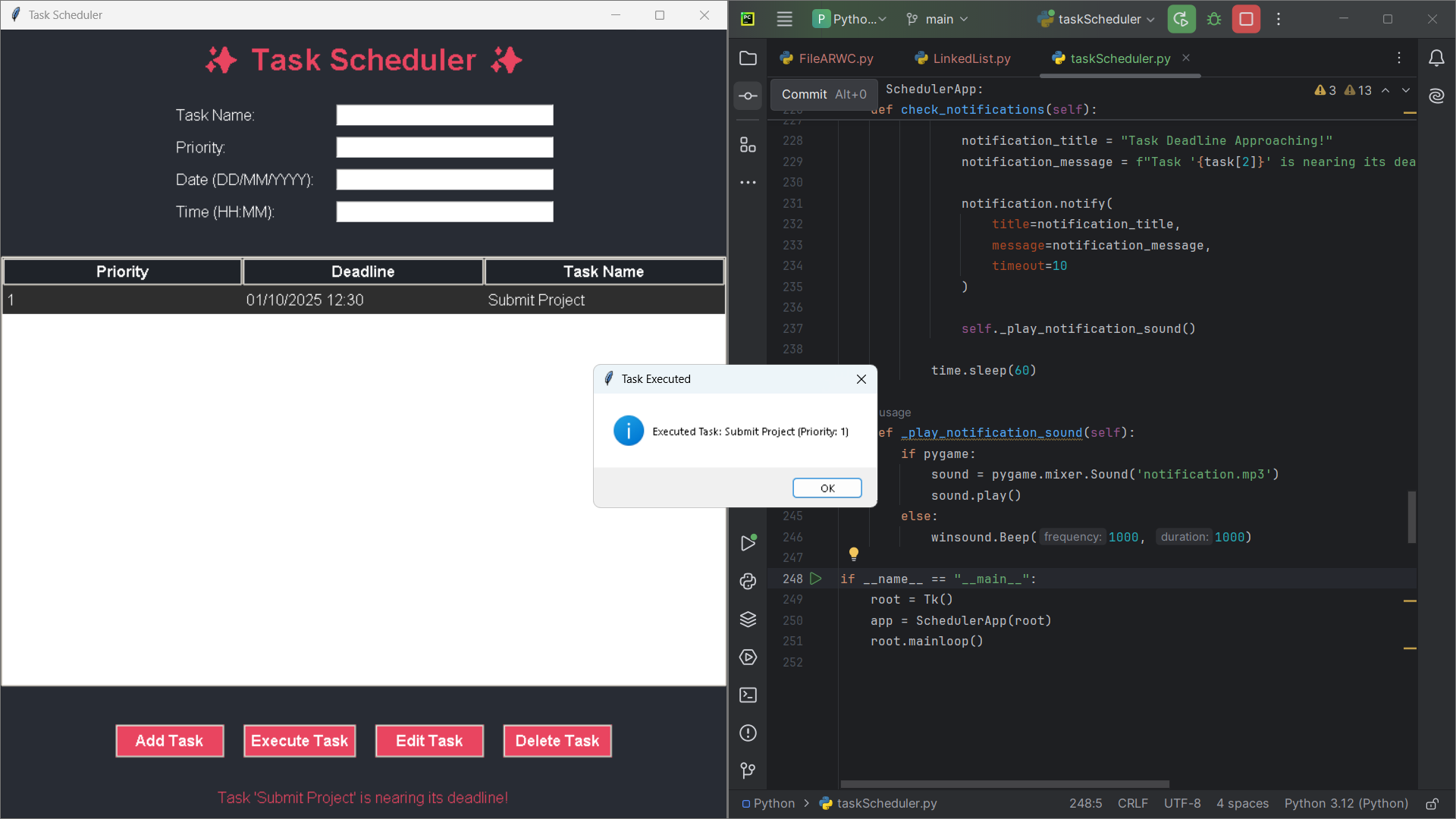


3. Notifications (both system and sound alerts).



Notification

4. Task execution.



## 6. Conclusion

The Task Scheduler project demonstrates the use of Python for creating functional and user-friendly applications. It integrates efficient data structures, persistent storage, and a responsive GUI to provide a seamless experience. The project serves as a foundation for building more advanced scheduling and management tools.