**📝 Smart Task Scheduler with Priority Queues — Project Report**

**Introduction**

Time management is a crucial skill for professionals and students alike. The *Smart Task Scheduler* project aims to address this by providing a desktop application that helps users organize their tasks based on priority and deadlines. Built in Java using Swing, it offers a user-friendly interface for real-time task tracking and productivity enhancement.

**Abstract**

This project is a desktop-based task manager application that leverages a PriorityQueue to automatically order tasks based on urgency. Each task is associated with a title, a priority level (High, Medium, Low), and a deadline. The application supports core functionalities like adding, deleting, and editing tasks, with additional features such as real-time reminders, task filtering, and persistent storage via file I/O. The application is lightweight, intuitive, and caters to students, developers, and productivity enthusiasts.

**Tools Used**

* **Language:** Java (JDK 17)
* **UI Framework:** Java Swing (javax.swing)
* **Build Tool:** Manual compilation + Executable JAR
* **Data Handling:** Java I/O (Serialization)
* **IDE:** IntelliJ IDEA
* **Others:** Java Timer for scheduling task reminders

**Steps Involved in Building the Project**

1. Class Design:

* Task: Stores title, priority, and deadline; implements Comparable for sorting.
* TaskManager: Manages the task queue and performs I/O operations.
* TaskUI: Swing-based user interface for managing and interacting with tasks.

1. Priority Queue Logic:

* Tasks are automatically sorted by priority (1 = High, 3 = Low) and deadline using a PriorityQueue.

1. Graphical User Interface:

* Users can add, delete, save, and load tasks via interactive Swing components like JList, JButton, and JOptionPane.

1. Reminders:

* Java TimerTask is used to alert users if a task is due today (demo alert after 10 seconds).

1. Persistence:

* Tasks are serialized and saved to a local .dat file.
* Loading restores the queue and repopulates the task list.

1. Task Filtering:

* Tasks due today are retrieved using date comparison with LocalDate.now().

1. Packaging:

* Project compiled and exported as an executable .jar file with a custom manifest.txt.

**Conclusion**

The Smart Task Scheduler project meets its objective of managing daily tasks with enhanced clarity and efficiency. It demonstrates practical implementation of core Java concepts such as collections, Swing, serialization, and timers. The application is modular, easily extendable (e.g., adding calendar sync or notifications), and proves useful for personal time management. It showcases how traditional data structures can power simple yet effective productivity tools.