Smart Student Task Manager

Project Report

Author: Nitin Kumar
Registration No.: 24BCE10864

1. Introduction

This document is the project report for Smart Student Task Manager. The application is a Javabased console program intended to help students manage tasks, deadlines, and priorities. It demonstrates object-oriented programming, file handling, and modular design.

2. Problem Statement

Students often struggle to track multiple assignments and deadlines. The Student Task Manager provides a simple, reliable system to add tasks, set priorities, manage due dates, and persist data so that information is retained across sessions.

3. Objectives

- Implement a functional task manager in Java using OOP principles.
- Provide persistent storage through file handling.- Offer search, update, delete and completion features.

4. Functional Requirements

The system includes the following modules:

1. Task Management Module: Add, update, delete tasks.
2. Priority Handling Module: Set and sort by priority.
3. Deadline Management Module: Set due dates and list upcoming tasks.
4. Search & Filter Module: Search tasks by keywords.5. File Handling Module: Save and load tasks from a persistent file.

5. Non-Functional Requirements

- Usability: Simple console interface.
- Performance: Fast for up to hundreds of tasks.
- Reliability: Safe file operations and error handling.- Maintainability: Modular design and clear code structure.

6. System Architecture

The architecture follows a layered approach: Main (UI) -> TaskManager (business logic) -> FileHandler (persistence) -> Task (model).

7. Design Diagrams (Text)

Use Case (textual):

- Student: Add task, Update task, Delete task, Search tasks, Mark completed.

Component diagram (text):

Main (Menu) --calls--> TaskManager --uses--> FileHandler --reads/writes--> tasks.txt

Class diagram (summary):

Task {id, title, description, priority, dueDate, completed}
TaskManager {add, update, delete, search, list}
FileHandler {loadTasks, saveTasks}

8. Implementation Details

The code is written in Java and follows OOP. Files included:

- Task.java: Model class with parsing and serialization methods.
- TaskManager.java: Business logic for managing tasks.
- FileHandler.java: Handles reading and writing tasks to a file.

- Main.java: Console-based user interface and entry point.

9. Data Storage

Tasks are saved in a plain text file located in the user's home directory under .smarttask/tasks.txt. Each line stores a task as: id|title|description|priority|dueDate|completed

10. Testing

Manual testing was performed for: adding tasks, updating tasks, deleting tasks, searching, marking completed, and persistence across program runs.

Example test case: Add task 'Finish lab' due 2025-12-01 with HIGH priority; verify it appears in list and persists after program exit.

11. Challenges and Learnings

Working on this project improved understanding of Java file I/O, object-oriented design, and modular coding practices. Error handling for user input was an important learning area.

12. Future Enhancements

- GUI using JavaFX or Swing. - Export to CSV, calendar integration, user authentication.

13. References

- Project guidelines provided by course (BuildYourOwnProject.pdf).

14. Output: