Team – 7

Review 2:

Project Title:

“Productivity Manager: Productivity Boosting through Tracking Activities.”

Team:

Admin	SABBIR AHMED KHAN
	Admission No: 24SCSE1180755
	Email: Sabbir.24scse1180755@galgotiasuniversity.ac.in
	Github ID:
Member	MITHUN CHANDRA ROY
	Admission No: 24SCSE1180749
	Email: raym23609@gmail.com
	Github ID:
Member	MRITTIC ROY
	Admission No: 24SCSE1180752
	Email: rmridul481@gmail.com
	Github ID:
Member	SUSHMITA GOSH
	Admission No: 24SCSE1180748
	Email: anughossusmita94@gmail.com
	Github ID:

1. Project Description

The Productivity Manager is a console-based Java application designed to help users manage, prioritize, and track their tasks. Users can add new tasks, assign priorities, and view the list of tasks in a structured format. It emphasizes core Java concepts such as object-oriented programming, file handling, and modular application design.

2. Objective

To build a user-friendly task tracking application using core Java without the use of databases. This system allows users to:

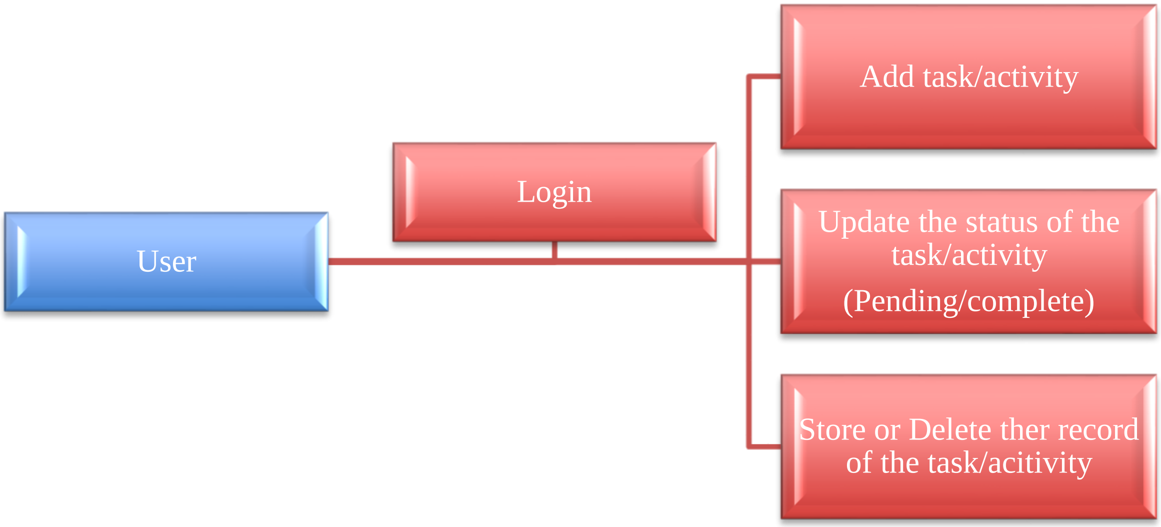
- Add tasks with description, deadline, and priority
- View all saved tasks
- Automatically persist tasks to a file
- Structure the application with DAO, model, and utility layers

3. Technologies and Tools Used

- Language: Java (JDK 17 or above)
- IDE: IntelliJ IDEA
- File Handling: Java I/O with `.txt` file storage
- Architecture: Layered – Main, DAO, Model, UI, Utility

4. Functional Requirements

- Add new task
- Validate task details (non-empty, valid date, priority range)
- Display all existing tasks
- Automatically load tasks from file on startup
- Save tasks on exit or addition


User
Login
Add task/activity
Update the status of the task/activity (Pending/complete)
Store or Delete ther record of the task/acitivity

5. Project Structure:

Package	Class Name	Purpose
model	Task, PriorityTask	Holds task data structure with optional priority
dao	TaskDAO, TaskFileDAO	Handles in-memory list and file operations
ui	ConsoleUI	Displays menu and handles user inputs
utils	TaskValidator	Utility functions for validation and formatting
main	MainApp	Application entry point with main loop

6. Sample File Format

tasks.txt
A simple text file that stores tasks line by line in a formatted representation like:

Task ID: 1, Description: Finish report, Due: 2025-06-01, Priority: HIGH

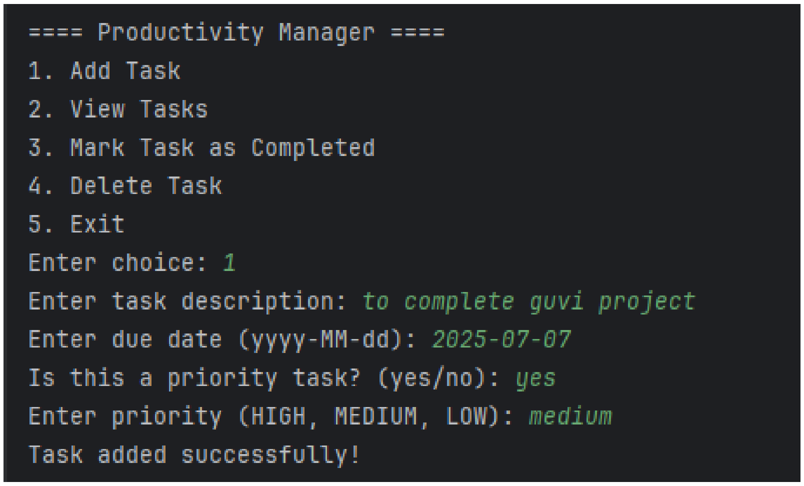
7. Known Issues

- Input Mismatch Exceptions:
  Using Scanner.nextInt() without validation may throw exceptions if non-numeric values are entered.

Future Scope:
- Use Scanner.hasNextInt() or try-catch for robust input handling.
- Include task status (e.g., completed/incomplete)
- Add support for editing and deleting tasks

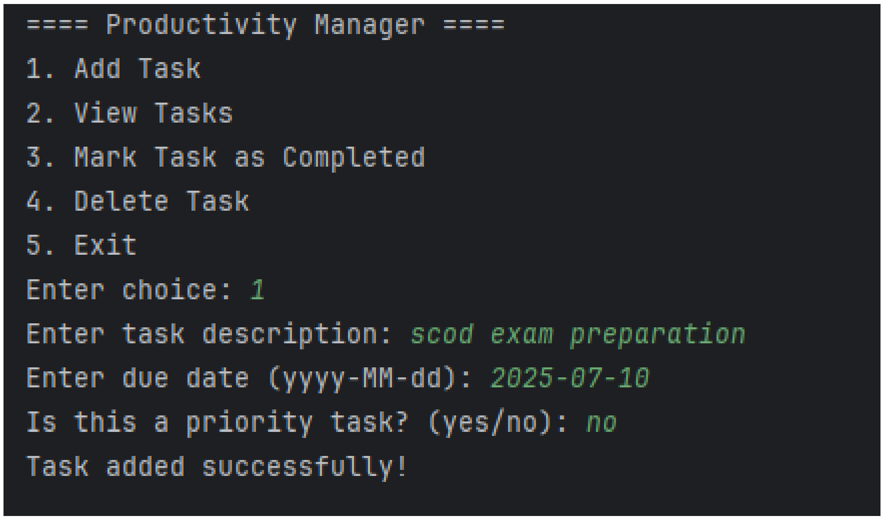
8. Screenshots

Adding task with date also the task has high/medium/low priority

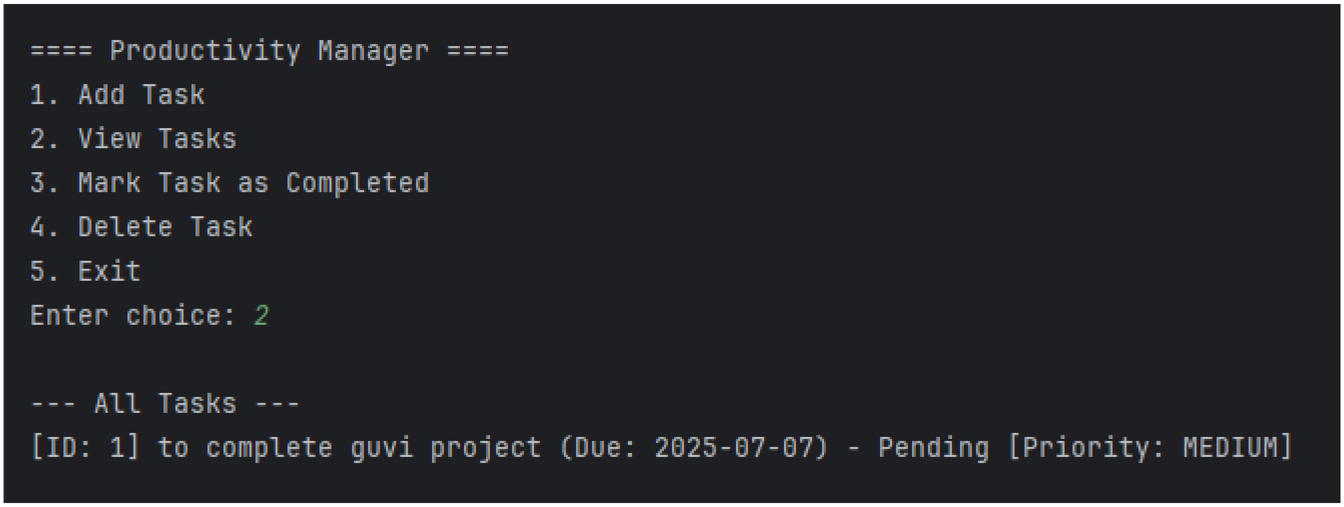


```
==== Productivity Manager ====
1. Add Task
2. View Tasks
3. Mark Task as Completed
4. Delete Task
5. Exit
Enter choice: 1
Enter task description: to complete guvi project
Enter due date (yyyy-MM-dd): 2025-07-07
Is this a priority task? (yes/no): yes
Enter priority (HIGH, MEDIUM, LOW): medium
Task added successfully!
```

Adding tasks

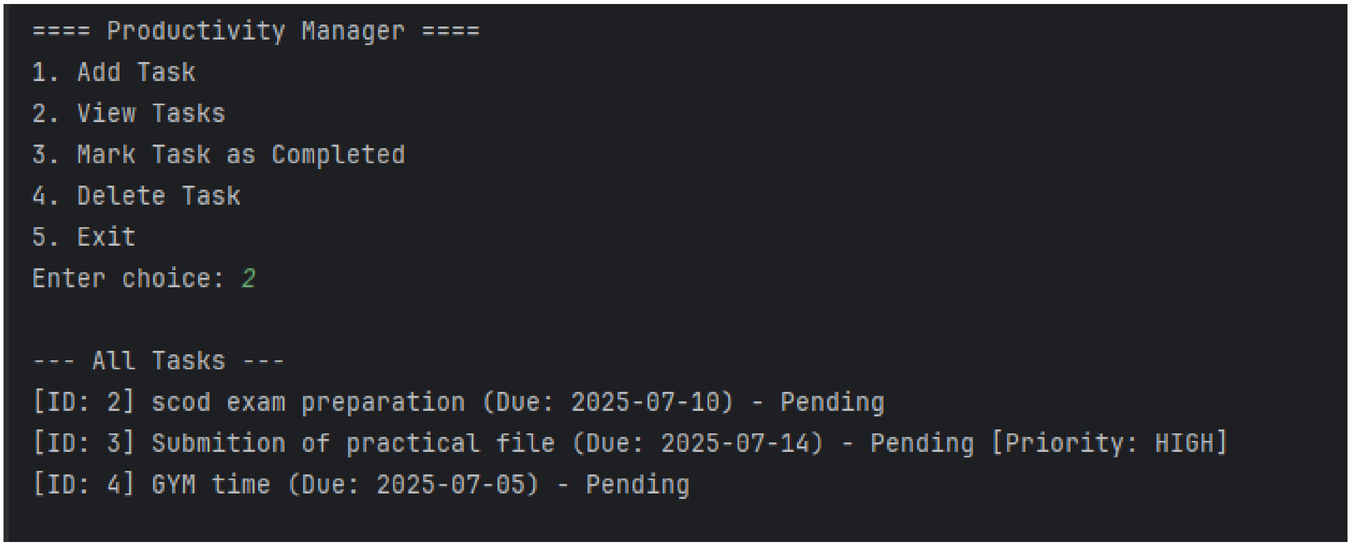


```
==== Productivity Manager ====
1. Add Task
2. View Tasks
3. Mark Task as Completed
4. Delete Task
5. Exit
Enter choice: 1
Enter task description: scod exam preparation
Enter due date (yyyy-MM-dd): 2025-07-10
Is this a priority task? (yes/no): no
Task added successfully!
```

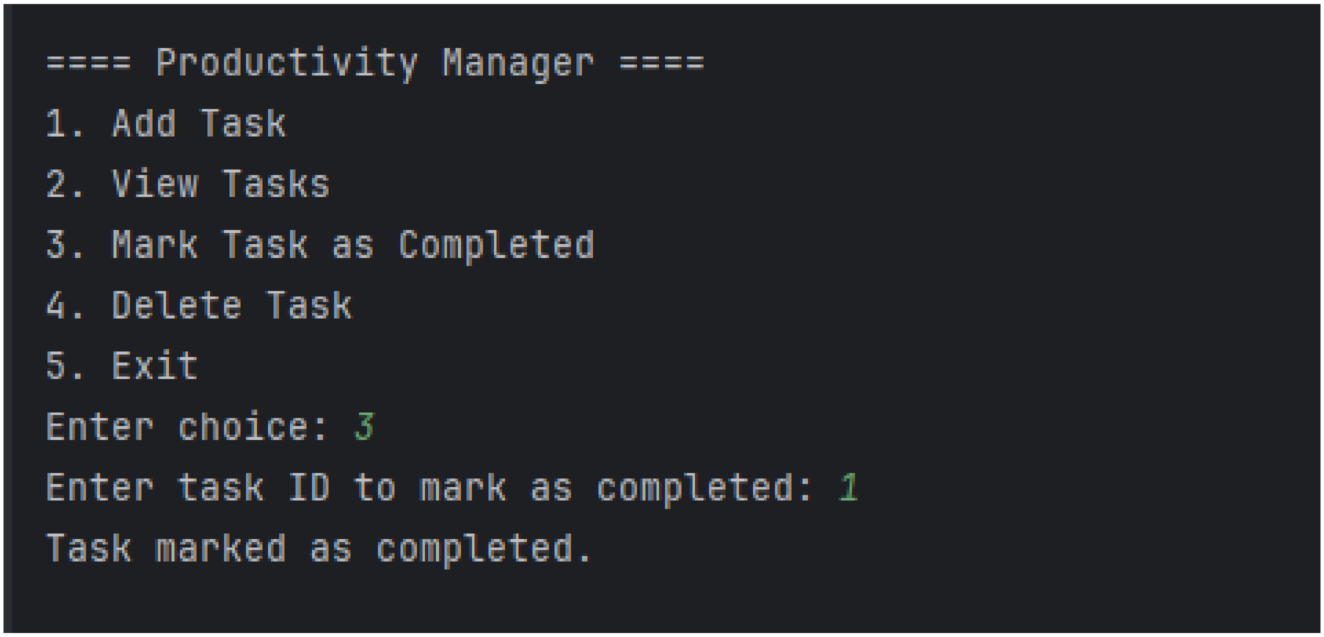


```
==== Productivity Manager ====
1. Add Task
2. View Tasks
3. Mark Task as Completed
4. Delete Task
5. Exit
Enter choice: 2

--- All Tasks ---
[ID: 1] to complete guvi project (Due: 2025-07-07) - Pending [Priority: MEDIUM]
```

Multiple tasks Added :

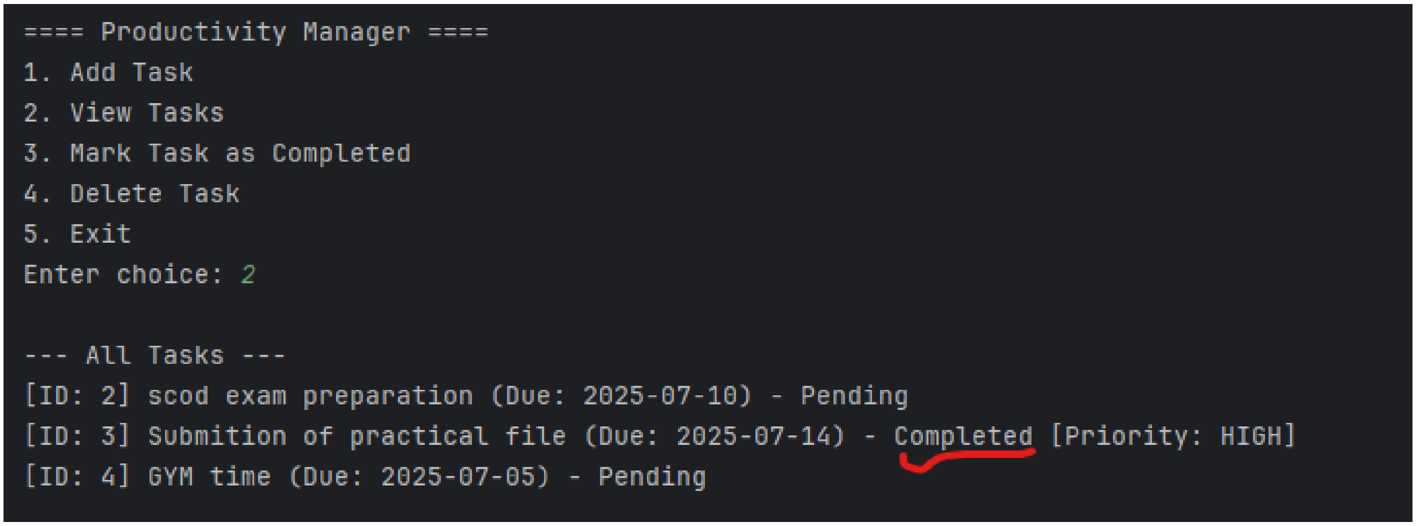


```
==== Productivity Manager ====
1. Add Task
2. View Tasks
3. Mark Task as Completed
4. Delete Task
5. Exit
Enter choice: 2

--- All Tasks ---
[ID: 2] scod exam preparation (Due: 2025-07-10) - Pending
[ID: 3] Submition of practical file (Due: 2025-07-14) - Pending [Priority: HIGH]
[ID: 4] GYM time (Due: 2025-07-05) - Pending
```



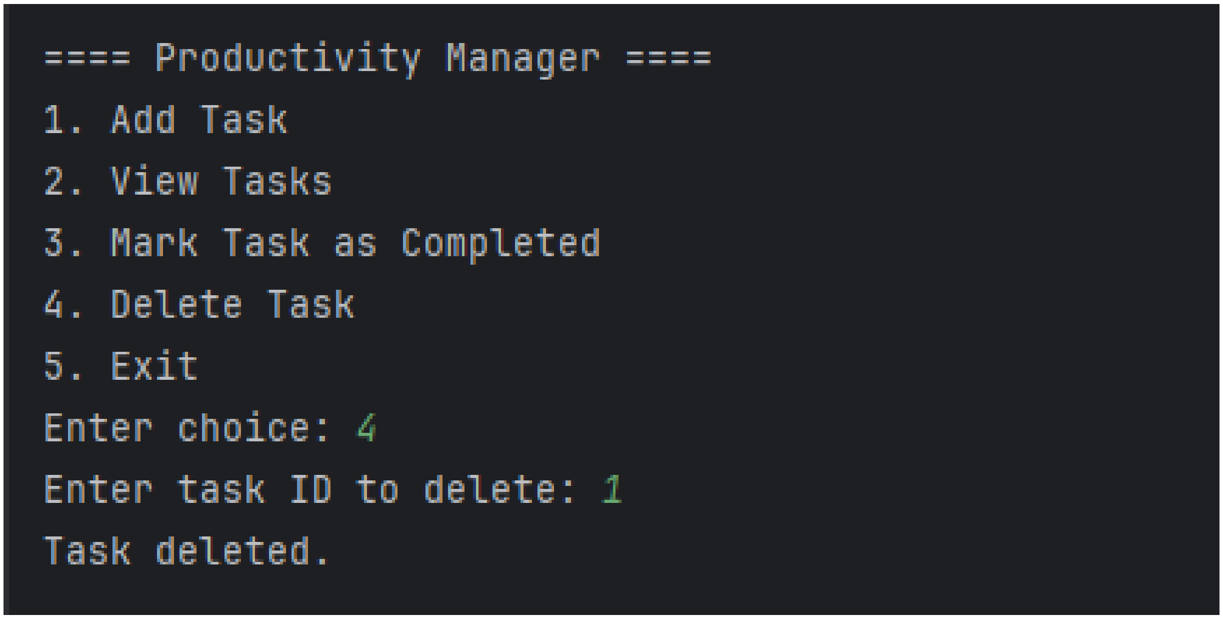
```
==== Productivity Manager ====
1. Add Task
2. View Tasks
3. Mark Task as Completed
4. Delete Task
5. Exit
Enter choice: 3
Enter task ID to mark as completed: 1
Task marked as completed.
```

If the task is completed it marked as completed. If it is not completed it shows as pending



```
==== Productivity Manager ====
1. Add Task
2. View Tasks
3. Mark Task as Completed
4. Delete Task
5. Exit
Enter choice: 2

--- All Tasks ---
[ID: 2] scod exam preparation (Due: 2025-07-10) - Pending
[ID: 3] Submition of practical file (Due: 2025-07-14) - Completed [Priority: HIGH]
[ID: 4] GYM time (Due: 2025-07-05) - Pending
```

We can delete the task if we want :



```
==== Productivity Manager ====
1. Add Task
2. View Tasks
3. Mark Task as Completed
4. Delete Task
5. Exit
Enter choice: 4
Enter task ID to delete: 1
Task deleted.
```

9. Challenges Faced

- Handling exceptions during input
- Managing file format consistency
- Ensuring unique task IDs
- Providing a clean and intuitive console menu

10. Conclusion

This project reinforced skills in:

- Java File I/O and OOP
- Application structure using modular layers
- User input handling and validation
- Real-world logic implementation through console apps
- Problem solving with patience

--- Thank You ---