PROJECT REPORT

SIMPLE PRODUCTIVITY SYSTEM

BATCH: 2023-2027

CLASS & SECTION: CSE-A

(Fourth Semester)

SUBMITTED TO:

Dr. Vidushi

SUBMITTED BY:

- HARSHIT KAUSHIK (01717702723)
- TANMAY NEGI (02217702723)
- ADITHYA VENKATESWARAN (02317702723)
- ROHAN BHANDARI (02817702723)

Project Report: Simple Productivity App

1. Introduction

The Simple Productivity App is a Command-Line Interface (CLI) application designed to help users track their daily tasks and analyze how productively they use their 24-hour day. The application uses Java for the core logic and PostgreSQL as the backend database, offering features such as user authentication, task management, and daily productivity reports.

2. Objectives

- Enable users to log in securely.
- Allow users to add, view, update, and delete tasks.
- Classify tasks into “Productive” and “Non-Productive” categories.
- Calculate remaining productive hours in a day.
- Store all data persistently using PostgreSQL.

3. Technologies Used

Component	**Technology Used**
Programming Language	Java
Database	PostgreSQL
JDBC Driver	PostgreSQL JDBC
Development Environment	Any IDE or terminal with Java and PostgreSQL setup

4. System Architecture

The app follows a layered architecture:

- Models: Represent data structures for User and Task.
- DAOs (Data Access Objects): Handle communication with the PostgreSQL database.
- Services: Implement business logic like authentication and reporting.
- Main CLI Menu: Provides a text-based user interface.

5. Database Schema

Users Table

```
CREATE TABLE users (
    id SERIAL PRIMARY KEY,
    username VARCHAR(100) UNIQUE NOT NULL,
    password_hash VARCHAR(100) NOT NULL
);
```

Tasks Table

```
CREATE TABLE tasks (
    id SERIAL PRIMARY KEY,
    user_id INTEGER REFERENCES users(id),
    description TEXT NOT NULL,
    category VARCHAR(50) NOT NULL,
    hours DOUBLE PRECISION NOT NULL
);
```

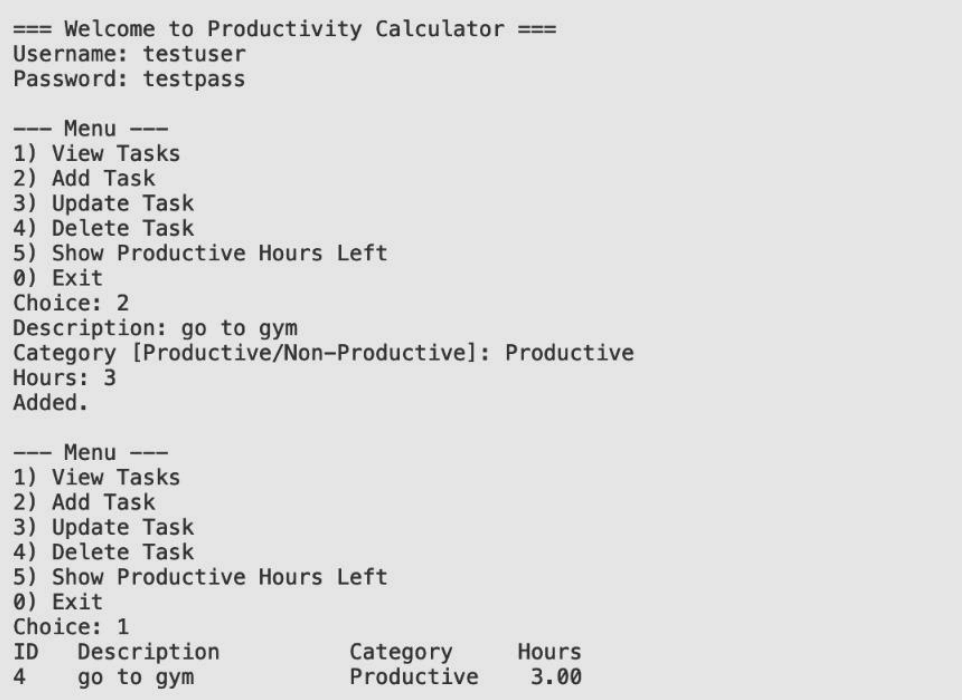
6. Key Features

User Authentication:
- Secure login based on username and password.
- Passwords are stored as plaintext (for now), but hashing is suggested for production.

Task Management:
- Add Task: Provide a description, category, and hours.
- View Tasks: List all tasks with proper formatting.
- Update Task: Modify task fields using ID.
- Delete Task: Remove a task by its ID.

Productivity Report:
- Calculates and displays remaining hours out of 24 based on non-productive task time.

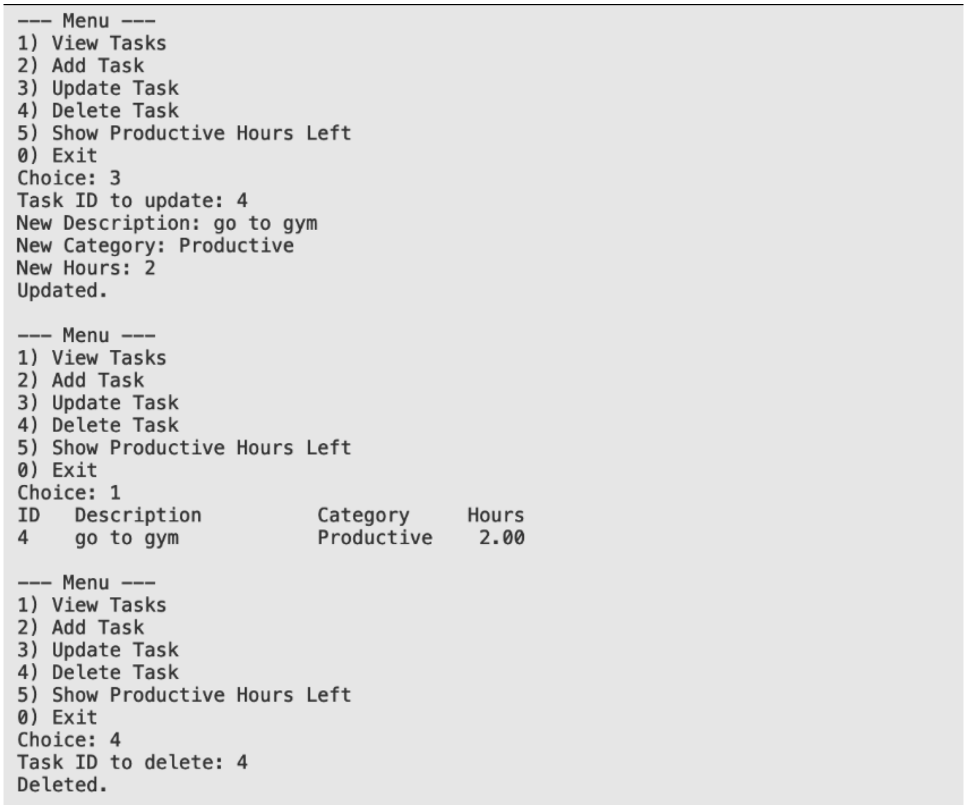
7. Screenshots



```
=== Welcome to Productivity Calculator ===
Username: testuser
Password: testpass

--- Menu ---
1) View Tasks
2) Add Task
3) Update Task
4) Delete Task
5) Show Productive Hours Left
0) Exit
Choice: 2
Description: go to gym
Category [Productive/Non-Productive]: Productive
Hours: 3
Added.

--- Menu ---
1) View Tasks
2) Add Task
3) Update Task
4) Delete Task
5) Show Productive Hours Left
0) Exit
Choice: 1
ID   Description          Category     Hours
4    go to gym            Productive    3.00
```



```
--- Menu ---
1) View Tasks
2) Add Task
3) Update Task
4) Delete Task
5) Show Productive Hours Left
0) Exit
Choice: 3
Task ID to update: 4
New Description: go to gym
New Category: Productive
New Hours: 2
Updated.

--- Menu ---
1) View Tasks
2) Add Task
3) Update Task
4) Delete Task
5) Show Productive Hours Left
0) Exit
Choice: 1
ID   Description          Category     Hours
4    go to gym            Productive    2.00

--- Menu ---
1) View Tasks
2) Add Task
3) Update Task
4) Delete Task
5) Show Productive Hours Left
0) Exit
Choice: 4
Task ID to delete: 4
Deleted.
```

Source Code Repository

The complete source code for the Productivity Calculator App is available on Github at the following link:
https://github.com/adithyanotfound/Java-Productivity-App

Demo Video: 🎬 demo-video-rno-17-22-23-24-28.mov

8. Strengths

- Clear code organization (separation of concerns using Models, DAOs, Services).
- Easy extensibility for GUI or web-based interface.
- Informative and concise user interaction.\
- Leverages standard JDBC practices for database operations.

9. Limitations & Suggestions

- Password Security: Currently using plain text comparison; should use hashed passwords.
- Input Validation: Limited error-checking for invalid input.
- Concurrency: No session or concurrency handling.
- Scalability: Meant for single-user CLI; not optimized for multiple users or cloud deployment.

10. Future Enhancements

- Integrate a password hashing mechanism.
- Implement a GUI (JavaFX or Swing).
- Export task reports to PDF/CSV.
- Introduce categories and tags for more granular productivity analysis.
- Add graph-based analytics using a web interface.

11. Conclusion

The Java Productivity App is a compact yet functional CLI-based application for personal productivity tracking. It demonstrates good use of Java object-oriented principles and database integration through JDBC. With further enhancements, it can evolve into a full-fledged productivity management tool.