PharmaSync

A Pharmacy Management System

== Project Report ==

Supervisor:

“ Md. Bakhtiar Hasan ”

Assistant Professor, CSE Department , Islamic University of Technology(IUT)

Submitted By:

{ Team no. 12 }

Ahnaf Shahriar Pias [220042146]
Sameur Rahman [220042144]
Irfan Shafee [220042164]

Course:

SWE- 4304
(Software Project Lab)

Github Repository :

https://github.com/A-Piyas-04/PHARMASYNC

Table of Contents

Project Overview

PharmaSync is a comprehensive pharmacy management system designed to streamline inventory management, sales processing, and supplier tracking for pharmacies. The system provides an intuitive console-based interface that allows pharmacists to efficiently:

- Manage medicine stocks
- Process sales
- Track supplier performance
- Receive timely notifications about low stock and expiring medicines

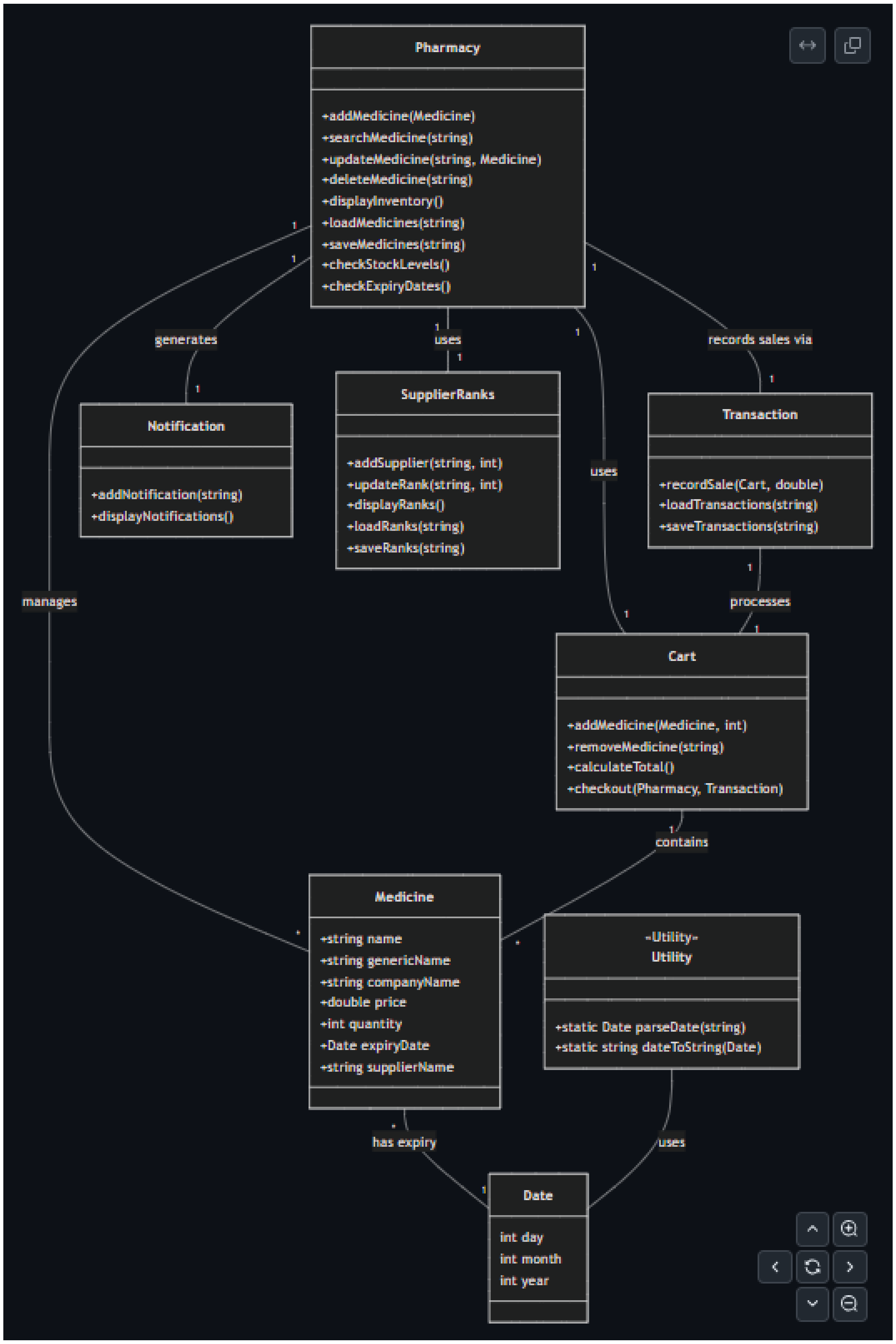
Motivation Behind the Project

The traditional pharmacy management process often involves manual record-keeping, leading to:

- Inventory discrepancies and stockouts
- Expired medicines remaining on shelves
- Inefficient supplier management
- Time-consuming sales processes
- Lack of timely notifications for critical events

PharmaSync addresses these challenges by providing an automated solution that enhances operational efficiency, improves inventory accuracy, and contributes to better patient care.

Class Diagram



Key Features of the Project

1. Medicine Inventory Management

- Add, update, and delete medicines with detailed attributes.
- Search, filter, and sort medicines by name, price, quantity, and supplier.

2. Sales Processing

- Shopping cart system with quantity validation.
- Receipt generation and transaction recording.
- Real-time inventory updates post-sale.

3. Supplier Ranking

- Monitor supplier performance based on sales volume.
- Maintain persistent supplier records.

4. Notification System

- Generate alerts for expiring medicines and low stock.
- Centralized notification viewing system.

5. Transaction History

- Record and view sales transactions for auditing and analysis.

Use-Case Descriptions

Use Case 1: Managing Medicine Inventory

Actor: Pharmacist
Description: Add, update, or delete medicines in the inventory.

Flow:

- Select "Manage Stock"
- Choose operation: Add / Update / Delete
- Enter necessary details
- System updates the inventory accordingly

Use Case 2: Processing a Sale

Actor: Pharmacist
Description: Handle customer purchases and generate receipts.

Flow:

- Select "Sell Medicine"
- Add medicines to the cart
- Validate available quantities

- Confirm checkout and generate receipt
- Update inventory and supplier ranking

<u>Use Case 3: Monitoring Notifications</u>

Actor: Pharmacist
Description: View notifications related to inventory status.

Flow:

- View notification indicator
- Select "View Notifications"
- Review categorized alerts

<u>Use Case 4: Analyzing Supplier Performance</u>

Actor: Pharmacist
Description: Review supplier rankings.

Flow:

- Select "Supplier Rankings"
- View ranked list based on sales metrics

Tools and Technologies Used

Programming Language

- C++

Development Environment

- Visual Studio Code
- g++ Compiler

Data Storage

- *Text Files:*
    - medicine_data.txt
    - supplier_ranks.txt
    - transaction.txt

Custom Utilities

- String manipulation functions
- Console color formatting
- Date utilities for expiry tracking

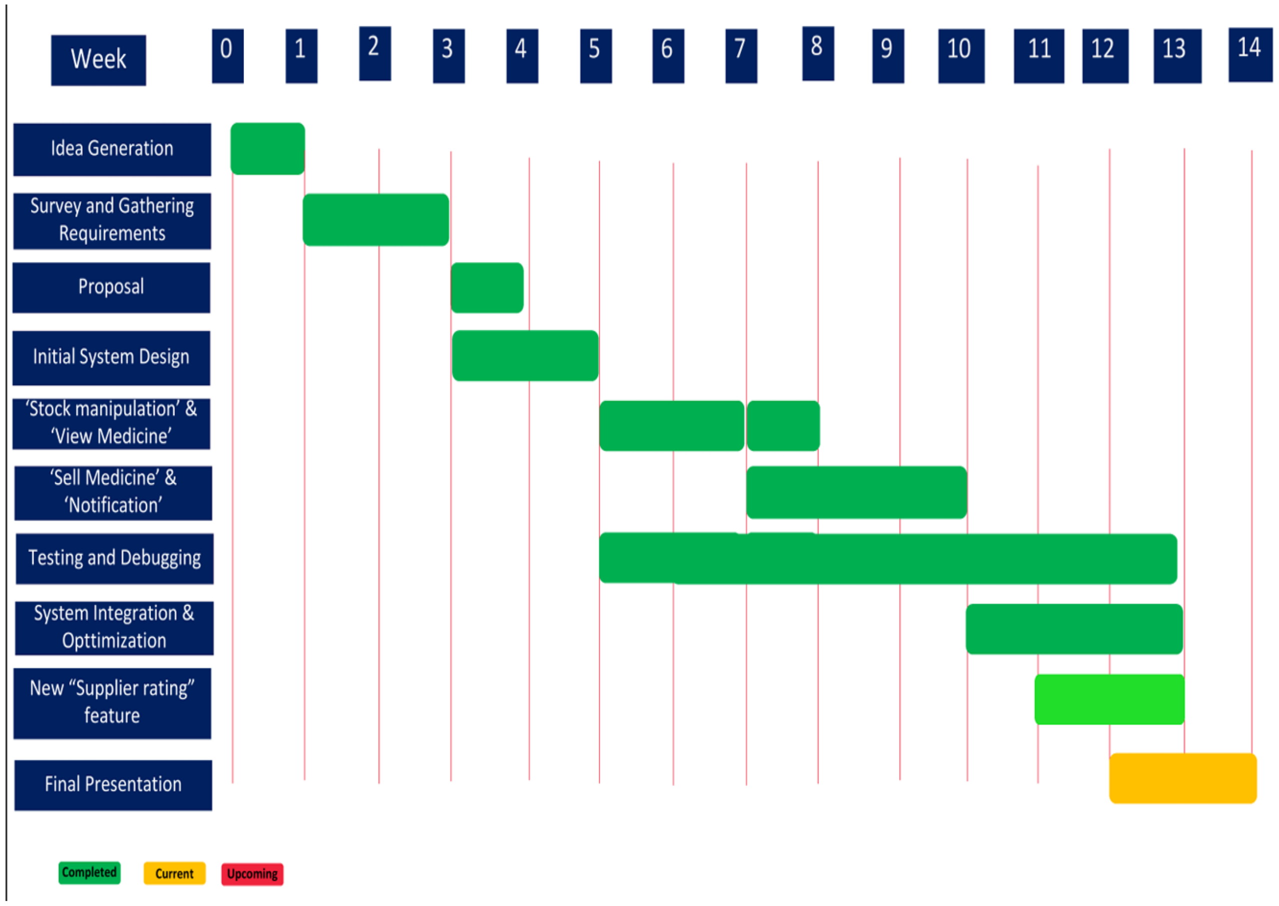
Alternative Tools and Technologies

- **Java:** For platform independence
- **Python:** For rapid development
- **SQLite:** For lightweight database management
- **Qt Framework:** For GUI-based solutions

Reason for Final Choice:

To meet project constraints, a C++ console-based application using file storage was selected for better alignment with learning objectives.

Proposed Timeline



Detailed Contribution of Each Member

[Ahnaf Shahriar Pias]

- Designed and implemented the Medicine and Pharmacy modules.
- Developed features for adding, updating, deleting, and searching medicines.
- Managed file operations for inventory persistence.
- Enhanced user experience with custom console formatting.

[Sameur Rahman]

- Built the Cart and Transaction modules for smooth sales processing.
- Implemented transaction recording and receipt generation.
- Handled inventory updates following sales.
- Focused on ensuring data integrity during checkout processes.

[Irfan Shafee]

- Developed Supplier Ranking and Notification modules.
- Created the dynamic alert system for expirations and low stock.
- Worked on system architecture and ensured data consistency.
- Optimized overall system responsiveness and usability.

Challenges Faced During Development

1. Custom String Handling

- Developed safe and efficient string operations to replace standard libraries.

2. File Format Design

- Structured data storage for easy parsing and scalability.

3. Date Management

- Implemented expiry tracking through custom date calculations.

4. Memory Management

- Ensured safe dynamic memory usage to avoid leaks.

5. Robust User Input Handling

- Built comprehensive validation to enhance system stability.

Solution Approach for Top Features

Inventory Management

- Utilized linked lists for dynamic and flexible data storage.
- Implemented manual file read/write operations for data persistence.

Supplier Ranking

- Dynamically updated supplier performance metrics after each sale.
- Applied custom sorting algorithms for ranking suppliers.

Notification System

- Regularly checked for expiration dates and stock levels.
- Provided centralized access to all alerts and notifications.

Learning Outcomes

- **Modular Programming:** Gained experience in writing maintainable and scalable code.
- **Data Structures:** Applied linked lists and other structures effectively.
- **File Management:** Strengthened understanding of file input/output operations.
- **Memory Management:** Practiced dynamic memory allocation and deallocation.
- **Console UI Design:** Focused on creating a user-friendly console experience.
- **Testing and Debugging:** Implemented a test-driven approach to development.
- **Team Collaboration:** Improved teamwork skills through task division and integration efforts.

Through **PharmaSync**, we demonstrated how effective planning, modular design, and teamwork can deliver a practical pharmacy management solution.

Individual Reflection:

[Ahnaf Shahriar Pias]

As a key contributor to the **Medicine and Pharmacy modules** in the PharmaSync project, I was responsible for designing and implementing the functionalities for adding, updating, deleting, and searching medicines. I also managed the file operations that ensured inventory persistence. The main challenge was ensuring that the data was handled efficiently and consistently, particularly with text files being used for storage. My work focused heavily on maintaining data integrity during each transaction, ensuring that no data was lost or corrupted.

In terms of teamwork, our collaboration worked well when it came to debugging and solving issues related to file I/O and database consistency. However, we faced challenges in agreeing on the structure of the data files early on. We overcame this by holding regular discussions to align our approach and ensure consistency across modules.

One key lesson I learned is the importance of early alignment in project design, particularly when it comes to shared data structures and file formats. Moving forward, I would advocate for clearer initial discussions to avoid confusion during later stages of development. I also learned the value of continuous communication in a team environment, ensuring everyone is informed of any changes to the overall design.

[Sameur Rahman]

My main responsibilities in the PharmaSync project included developing the **Cart and Transaction modules**, which were crucial for smooth sales processing. I was involved in implementing transaction recording, receipt generation, and ensuring inventory updates following sales. A critical aspect of my work was ensuring data integrity during the checkout process and ensuring that sales data was consistently stored in the correct format.

The team was able to work well together in terms of debugging, especially when troubleshooting issues related to the cart system and transaction handling. However, managing multiple dependencies across the Cart, Inventory, and Supplier modules proved challenging. To resolve this, I took the initiative to engage with other team members frequently, making sure the changes I was implementing didn’t conflict with other features, which ensured smooth integration.

A key lesson I learned was the importance of cross-functional collaboration. It became clear that effective communication between the Cart and Inventory teams was critical to maintain synchronization. Moving forward, I plan to prioritize modular testing and have more frequent check-ins to ensure that all modules work cohesively and that potential conflicts are addressed early.

[Irfan Shafee]

In the PharmaSync project, I was responsible for developing the **Supplier Ranking** and **Notification modules**, which played an essential role in providing pharmacists with real-time updates on supplier performance and inventory status. I focused on creating a dynamic alert system that notified users about expirations and low stock levels. Additionally, I worked on ensuring that the system architecture was well-organized and that data consistency was maintained throughout the development process.

While the team worked well in terms of integrating features, we encountered challenges when it came to optimizing the responsiveness of the system. Specifically, managing real-time notifications and keeping the system's performance efficient was more difficult than anticipated. To overcome this, I optimized the system's back-end architecture by implementing more efficient data structures and algorithms. This resulted in faster response times and more reliable alerting.

A key lesson I learned from this project was the importance of scalability in system design. Ensuring that the system could handle growing data and continued functionality without sacrificing performance was crucial. In future projects, I will emphasize the need for performance optimizations early in the design process, ensuring that the system can scale efficiently as it evolves.