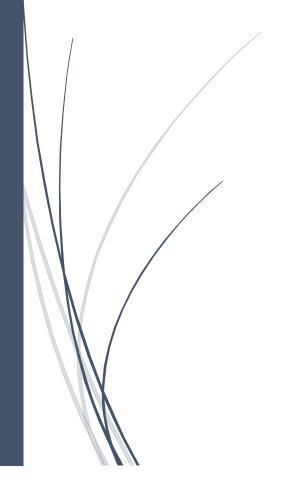2/19/2025

Pharmacy Management System - Development Documentation

Java MidTerm Sprint

stephen bad

SD12 GROUP 9 MIMYA HAFIZ BRAEDEN MERCER

Pharmacy Management System - Development Documentation

1. Introduction

This document provides an overview of the **Pharmacy Management System**, including source code structure, build process, dependencies, coding standards, theoretical database design, and instructions for accessing the project from GitHub.

---

2. Source Code Directory Structure

```
java_midtermm_sprint_sd12_group9/
│   ── src/
│   │   ── models/             # Business logic classes
│   │   │   ── Person.java       # Base class for Patient and Doctor
│   │   │   ── Patient.java      # Represents a patient
│   │   │   ── Doctor.java        # Represents a doctor
│   │   │   ── Medication.java     # Represents a medication
│   │   │   ── Prescription.java   # Represents a prescription
│   │   │   ── MedicationTrackingSystem.java # Core functionality
│   │   ── test/            # Test cases
│   │   │   ── TestMedicationTrackingSystem.java
│   ── README.md              # Project overview and instructions
│   ── .gitignore            # Git ignore file
│   ── Development_Documentation.pdf # This Documentation
```

---

3. Build Process

To compile and run the system, use the following commands:

- ◆ Compilation Command

javac -d bin src/models/**.java src/test/TestMedicationTrackingSystem.java

- ◆ Run the Test Program

java -cp bin src.test.TestMedicationTrackingSystem

4. Compiler-Time Dependencies

This project does not use external libraries and relies only on **Java SE**. Ensure you have **JDK 11 or higher** installed.

 Required:

- Java Development Kit (**JDK 11+**)
- A Java-compatible IDE (e.g., IntelliJ, VS Code)

---

5. Development Standards

- Java Naming Conventions:
    - Classes: PascalCase (e.g., MedicationTrackingSystem)
    - Methods/Variables: camelCase (e.g., addPatientToDoctor)
    - Constants: UPPER_CASE_WITH_UNDERSCORES
- Javadoc Documentation:
    - Each class and method include a proper **/** Javadoc comment **/**.
- Code Formatting:
    - Indentation: **4 spaces per level**.
    - Proper use of **getter/setter methods**.
    - Avoiding **magic numbers** and **hardcoded strings**.

---

6. Theoretical Database Design

Even though the project does **not use a real database**, this section outlines how data would be structured if implemented in a relational database.

Entity Relationship Diagram (ERD)

```
[DOCTOR]    1 ——— N [PATIENT]
[DOCTOR]    1 ——— N [PRESCRIPTION]
[PATIENT]   1 ——— N [PRESCRIPTION]
[MEDICATION] 1 ——— N [PRESCRIPTION]
```

Table Definitions

Doctor Table

Column	Data Type	Description
doctor_id	VARCHAR(10)	Unique ID for each doctor
name	VARCHAR(50)	Doctor's name
age	INT	Doctor's age
phone	VARCHAR(15)	Contact number
specialty	VARCHAR(50)	Field of expertise

Patient Table

Column	Data Type	Description
patient_id	VARCHAR(10)	Unique ID for each patient
name	VARCHAR(50)	Patient's name
age	INT	Patient's age
phone	VARCHAR(15)	Contact number

Medication Table

Column	Data Type	Description
med_id	VARCHAR(10)	Unique ID for medication
name	VARCHAR(50)	Medication name
quantity	INT	Available stock count
expiry_date	DATE	Expiration date of drug

Prescription Table

Column	Data Type	Description
presc_id	VARCHAR(10)	Unique ID for prescription
doctor_id	VARCHAR(10)	Linked to Doctor Table
patient_id	VARCHAR(10)	Linked to Patient Table
med_id	VARCHAR(10)	Linked to Medication Table
presc_expiry	DATE	Expiration date of prescription

7. Accessing the Source Code from GitHub

◆ Clone the Repository

To download the project, use the following command:

git clone https://github.com/stepbad/java_midtermm_sprint_sd12_group9.git

◆ Navigate into the Project Directory

cd pharmacy-management-system

◆ Compile and Run

Follow the **build process** mentioned in Section 3.

---

8. Conclusion

This documentation serves as a guide for **building, running, and understanding the Pharmacy Management System**. The project is structured for **easy scalability** and will be expanded to support a **front end menu** integration in the future.