JAVA SPRINT 1
PHARAMACY SYSTEM

Michael Barney, Joey Thomas, Brandon Pike

Date: June 29,2025

Contents

User Documentation

Person (Superclass)

Purpose: Base class for Doctor and Patient.

Id, which is a string. Description: Unique identifier.

Name which is a string. Description: Name of a person.

Age which is an int. Description: this is the person's Age.

phoneNumber which is a string. Description: Contact number to reach them.

Doctor extends Person

Purpose: Represents a doctor and their list of patients.

Specialization is a string. Description: Doctor’s medical field

Patients is a List. Description: Associated Patients

Patient extends Person

Purpose: Represents a patient and their medications/prescriptions

Medications is a List. Description: shows current medications

Prescriptions is a List. Description: shows list of issued prescriptions

Medication

Purpose: Represents drugs in stock.

Id is an Int. Description: Auto-incremented ID

Name is a String. Description: Drug name

Dose is a String. Description: Dosage information

quantityInStock is an Int. Description: number of drugs in inventory

ExpiryDate is a LocalDate. Description: Expiration date of drugs

Prescription

Purpose: Links a doctor, patient, and medication.

Id is a string. Description: Unique prescription ID

Doctor is an Object of the doctor class which has a relation to this class

Patient is an object of the patient class which has a relation to this class

Medication is an object of the medication class which has a relation to this class

quantityPrescribed is an int. Description: Number of units prescribed

Instructions is a String. Description: Usage instructions

issueDate is a LocalDate. Description: Date of prescription

prescriptionExpiry is a LocalDate. Description: Expiration date of medication.

MedicationTrackingSystem

A static class that maintains all lists and provides utility methods for:

Searching

Prescription linking

Reports and restocking

Menu

This is the main entry point of the application. It displays a console-based menu and interacts with the user through:

Adding Doctors, Patients, Medications.

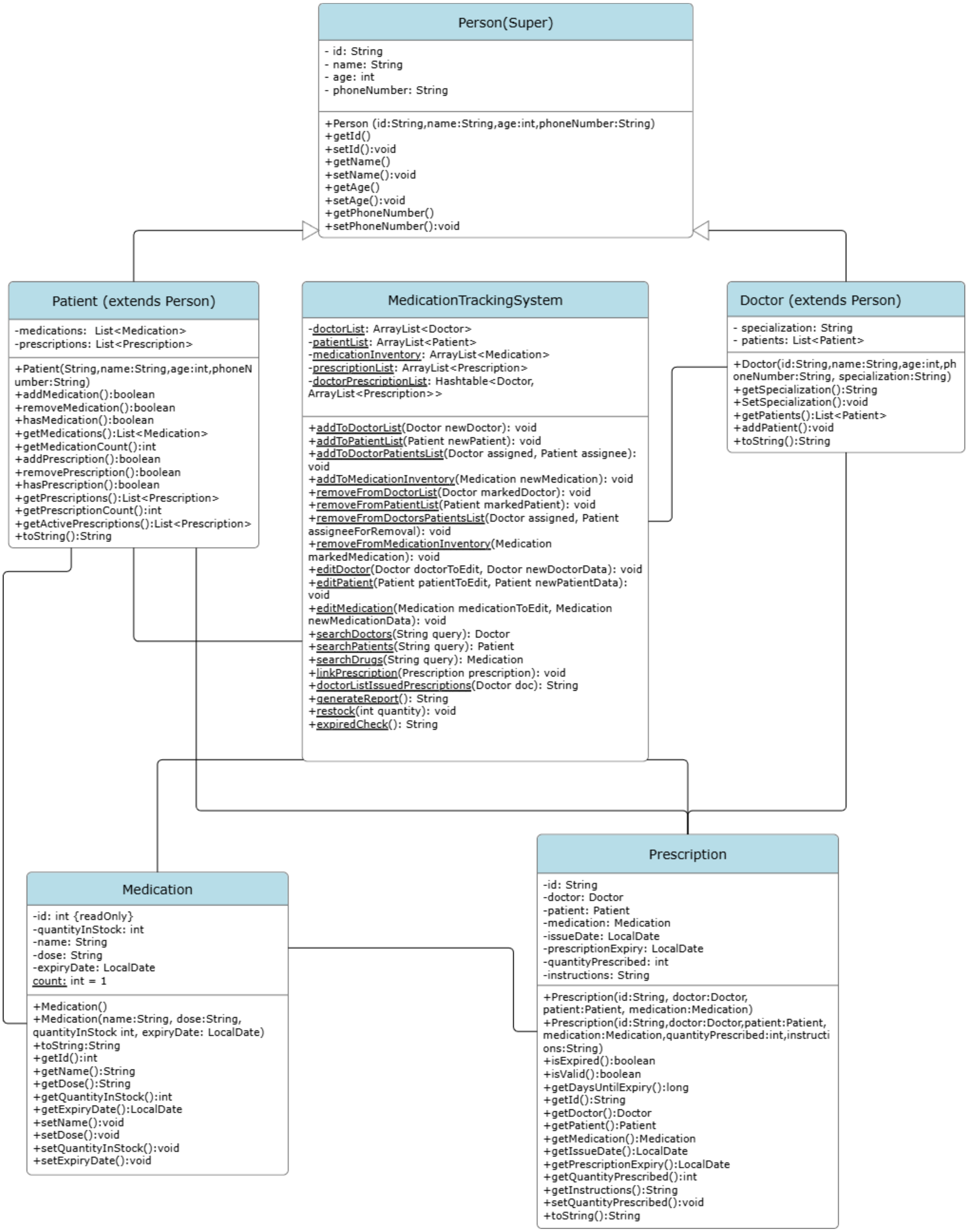
Creating prescriptions

Listing entities

Restocking

Checking expired drugs

UML Class Diagram



Person(Super)

- id: String
- name: String
- age: int
- phoneNumber: String

+Person (id:String,name:String,age:int,phoneNumber:String)
+getId()
+setId():void
+getName()
+setName():void
+getAge()
+setAge():void
+getPhoneNumber()
+setPhoneNumber():void

Patient (extends Person)

-medications: List<Medication>
-prescriptions: List<Prescription>

+Patient(String,name:String,age:int,phoneNumber:String)
+addMedication():boolean
+removeMedication():boolean
+hasMedication():boolean
+getMedications():List<Medication>
+getMedicationCount():int
+addPrescription():boolean
+removePrescription():boolean
+hasPrescription():boolean
+getPrescriptions():List<Prescription>
+getPrescriptionCount():int
+getActivePrescriptions():List<Prescription>
+toString():String

MedicationTrackingSystem

-doctorList: ArrayList<Doctor>
-patientList: ArrayList<Patient>
-medicationInventory: ArrayList<Medication>
-prescriptionList: ArrayList<Prescription>
-doctorPrescriptionList: Hashtable<Doctor, ArrayList<Prescription>>

+addToDoctorList(Doctor newDoctor): void
+addToPatientList(Patient newPatient): void
+addToDoctorPatientsList(Doctor assigned, Patient assignee): void
+addToMedicationInventory(Medication newMedication): void
+removeFromDoctorList(Doctor markedDoctor): void
+removeFromPatientList(Patient markedPatient): void
+removeFromDoctorsPatientsList(Doctor assigned, Patient assigneeForRemoval): void
+removeFromMedicationInventory(Medication markedMedication): void
+editDoctor(Doctor doctorToEdit, Doctor newDoctorData): void
+editPatient(Patient patientToEdit, Patient newPatientData): void
+editMedication(Medication medicationToEdit, Medication newMedicationData): void
+searchDoctors(String query): Doctor
+searchPatients(String query): Patient
+searchDrugs(String query): Medication
+linkPrescription(Prescription prescription): void
+doctorListIssuedPrescriptions(Doctor doc): String
+generateReport(): String
+restock(int quantity): void
+expiredCheck(): String

Doctor (extends Person)

- specialization: String
- patients: List<Patient>

+Doctor(id:String,name:String,age:int,phoneNumber:String, specialization:String)
+getSpecialization():String
+SetSpecialization():void
+getPatients():List<Patient>
+addPatient():void
+toString():String

Medication

-id: int {readOnly}
-quantityInStock: int
-name: String
-dose: String
-expiryDate: LocalDate
count: int = 1

+Medication()
+Medication(name:String, dose:String, quantityInStock int, expiryDate: LocalDate)
+toString:String
+getId():int
+getName():String
+getDose():String
+getQuantityInStock():int
+getExpiryDate():LocalDate
+setName():void
+setDose():void
+setQuantityInStock():void
+setExpiryDate():void

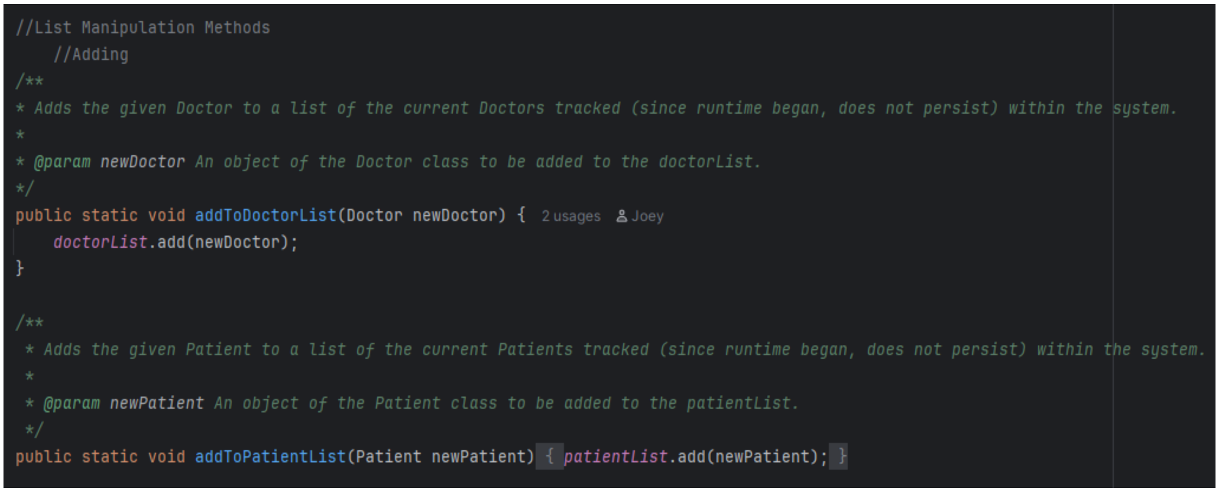
Prescription

-id: String
-doctor: Doctor
-patient: Patient
-medication: Medication
-issueDate: LocalDate
-prescriptionExpiry: LocalDate
-quantityPrescribed: int
-instructions: String

+Prescription(id:String, doctor:Doctor, patient:Patient, medication:Medication)
+Prescription(id:String,doctor:Doctor,patient:Patient, medication:Medication,quantityPrescribed:int,instructions:String)
+isExpired():boolean
+isValid():boolean
+getDaysUntilExpiry():long
+getId():String
+getDoctor():Doctor
+getPatient():Patient
+getMedication():Medication
+getIssueDate():LocalDate
+getPrescriptionExpiry():LocalDate
+getQuantityPrescribed():int
+getInstructions():String
+setQuantityPrescribed():void
+toString():String

Java Docs

We have updated all our code with the Java docs here is an example of the code.



```
//List Manipulation Methods
    //Adding
/**
* Adds the given Doctor to a list of the current Doctors tracked (since runtime began, does not persist) within the system.
*
* @param newDoctor An object of the Doctor class to be added to the doctorList.
*/
public static void addToDoctorList(Doctor newDoctor) {
    doctorList.add(newDoctor);
}

/**
 * Adds the given Patient to a list of the current Patients tracked (since runtime began, does not persist) within the system.
 *
 * @param newPatient An object of the Patient class to be added to the patientList.
 */
public static void addToPatientList(Patient newPatient) { patientList.add(newPatient); }
```

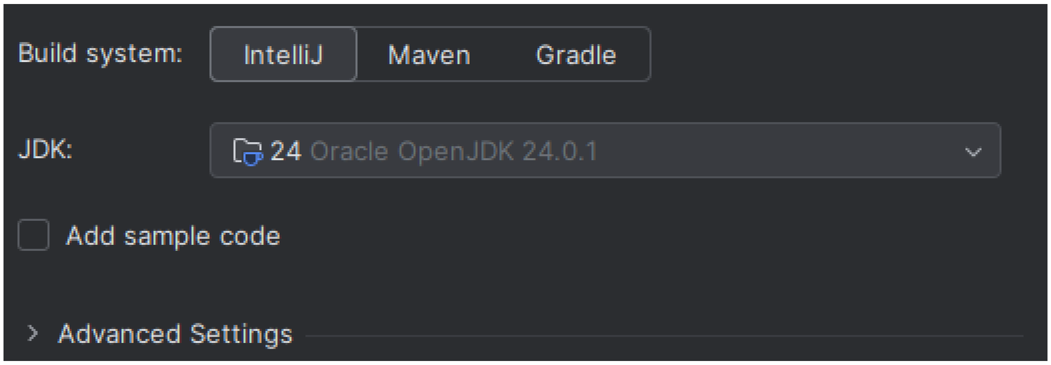
Source Code Directory

Midterm-sprint-JT-MB-BP

```
 ├── src/
│  │     └── docs/
│              ├── Dev Document.docx
│  │     └── PharmacySystem/
│  │        ├── Person.java              # Base class for all people
│  │        ├── Patient.java             # Patient entity extending Person
│  │        ├── Doctor.java              # Doctor entity extending Person
│  │        ├── Medication.java          # Medication entity with inventory
│  │        ├── Prescription.java        # Prescription linking patients/doctors/meds
│  │        ├── MedicationTrackingSystem.java  # Main system controller
│  │        ├── Menu.java                # User interface controller
│  │       └── TestPharmacySystem.java      # Main testing class
 ├── .gitignore
```

Build Process

We used IntelliJ with Oracle OpenJDK 24 for our project with no dependencies.



Build system: IntelliJ

JDK: 24 Oracle OpenJDK 24.0.1

☐ Add sample code

Development Standards

Code Style Guidelines

Naming Conventions

- **Classes**: PascalCase
- **Methods**: camelCase
- **Variables**: camelCase

Code Formatting

- **Indentation**: 4 spaces (no tabs)
- **Line Length**: Maximum 120 characters
- **Braces**: Opening brace on same line
- **Imports**: Organized and unused imports removed

Documentation Standards

- All public classes must have class-level Javadoc
- All public methods must have method-level Javadoc
- Use @param and @return tags appropriately

Error Handling

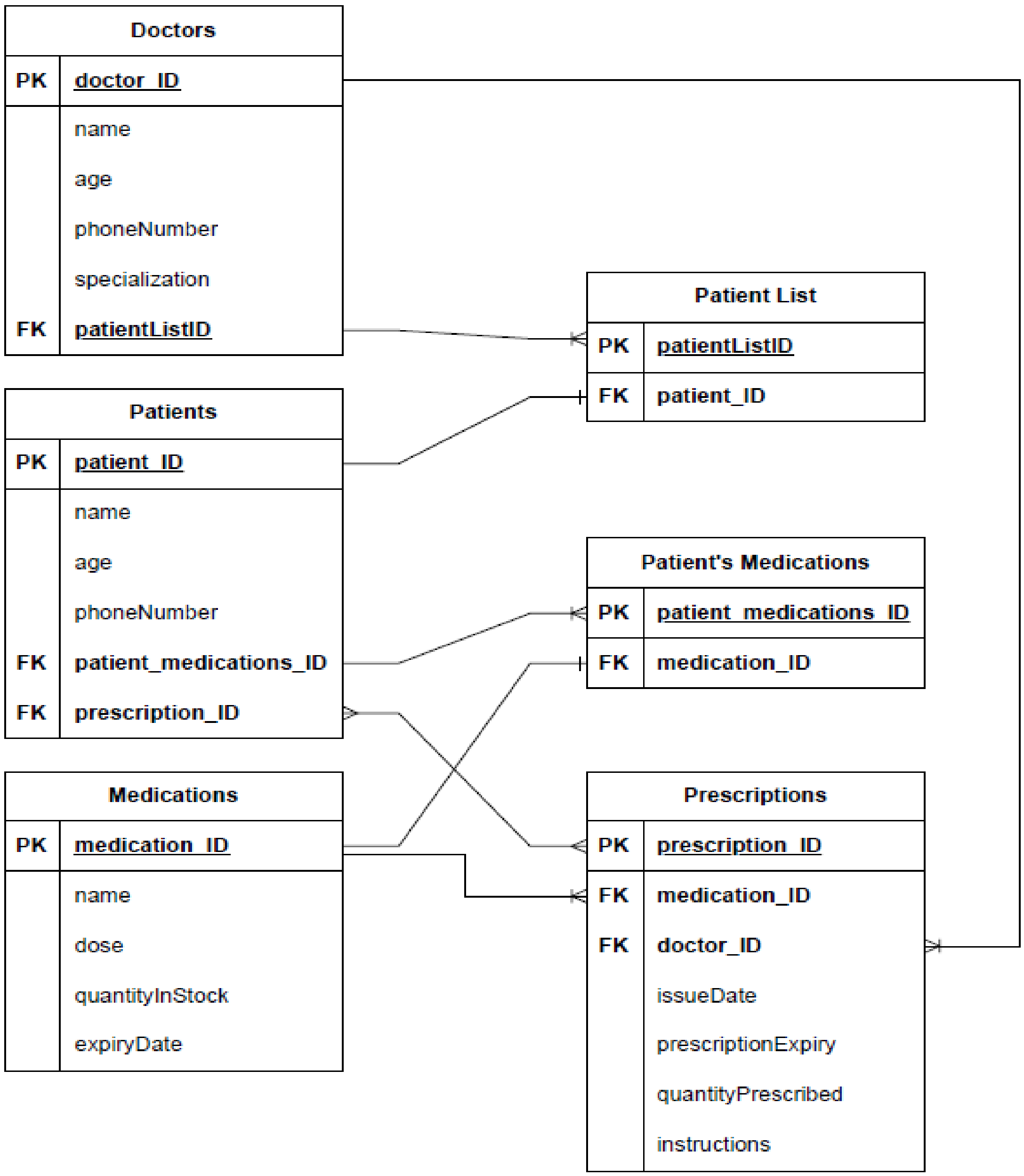
- Provide descriptive error messages

Log errors appropriately

Time Dependencies

We didn’t use any dependencies.

Database Design (entity relationships)



GitHub Repository

1. Download GitHub Desktop App.
2. Sign-In to your account.
3. Clone repository using https://github.com/JoeyJowhi/midterm-sprint-JT-MB-BP.git
4. Open in respective IDE.