Pharmacy Management System.

Overview

This project involves creating a comprehensive pharmacy system to manage patient drugs and information. The system will be built using object-oriented programming (OOP) principles and will include several classes representing different entities such as doctors, patients, medications prescriptions, and the overall medication tracking system.

Classes

1. Person (Super Class)

The Person class is the base class for both the Patient and Doctor classes. It contains the following attributes:

- ID: A unique identifier for each person.
- Name: The name of the person.
- Age: The age of the person.
- Phone Number: the persons contact information

2. Patient

Patient represents a patient. Each patient has a list of medications they are taking and a **list** of prescriptions active in the pharmacy.

3. Doctor

Doctor represents a doctor. Each doctor has a specialization and a **list** of patients they are managing.

4. Medication

The Medication class represents a medication. It contains the following attributes:

- ID: A unique identifier for each medication.
- Name: The name of the medication.
- Dose: The dosage of the medication.
- Quantity in stock: the amount of medication in stock
- ExpiryDate: The expiry date of the medication (try and set the date to be a random date inclusive of the past)

5. Prescription

The Prescription class represents a prescription issued by a doctor for a patient

- **ID**: A unique identifier for each prescription.
- **Doctor**: A reference to the prescribing doctor.
- **Patient**: A reference to the patient.
- **Medication**: A reference to the prescribed medication.
- **Prescription Expiry**: The expiration date of the prescription, which defaults to one year from the date it was issued.

6. Medication System

The MedicationTrackingSystem class manages the entire system. It contains lists of patients, medications, and doctors, and provides the following functionalities:

- Search for drugs, patients, and doctors by name and display relevant details for each
- Add a patient to a doctor's list.
- Accept a prescription (staff manually inputs the prescription details from the doctor), linking the prescription and drug to the patient.
- Edit and delete medications, patients, and doctors
- Generate a report containing all system data, including drugs, patients, doctors, and prescriptions.
- Check for expired medications and display a message if any are found.
- Print a list of all prescription’s issued by a specific doctor.
- Restock the drugs in the pharmacy in some capacity. You can just add a random number to the stock, you can add a specific number. It's up to you!

7. You can set up a class file to test your program. Along with setting up a menu if you are working as a team. There will be a menu example provided for you.

Functionality

The system provides the following functionalities:

The system provides the following functionalities:

1. Adding/deleting a patient, and medication, doctor to the system
2. Editing: Edit the details of a patient, medication or doctor
3. Searching: Search for a patient medication and doctor by name and display the relevant information to the console
4. Accept Prescription: Manually input a prescription from a doctor, linking the medication to the patient. (Prescriptions can be set up beforehand with just object calls. No need to create them from the menu unless you have time on the end)
5. Add Patient to Doctor: Assign a patient to a doctor's list.
6. Generate Report: Print a report summarizing all data, including patients, doctors, and medications in the system
7. Generate Report: Check to see if any drug is expired in the system. If they are generate a report showing the information of each drug expired
8. Generate Report: Print a list of all prescriptions for a specific doctor
9. Generate a report of all the patients' prescriptions for the past year, summarize the report to just the drug names
10. Restock all the drugs in the pharmacy can be to a set level or just a random number

TEAM BASED REQUIREMENT: If you are working as a team, it is required that you set up a menu using the Scanner class.

Documentation Requirements

The project should have a document that outlines ALL the following sections

1. **User Documentation** This includes a document stating what the application is about, explanation of all the classes and their working, and how to start it/access it. Also include the class diagram with the associations between them.
2. **Development Documentation** This includes at least the Javadocs, a description of the source code directory structure, the build process (i.e., how to compile the project), compiler time dependencies, development standards, how a database would look in theory for this project (It Does Not Have to Be Actually Setup!). Include the entity relationships with

your database design, and how to get the source code from the GitHub repository. **Make sure that it has lots of detail and is formatted well!**

Grading

The project will be graded based on the following criteria:

1. **Functionality (60 marks)**:
    - Correct implementation of all required features.

    Documentation (15 marks):
    - Completeness and clarity of documentation
2. **Submission (25 marks)**:
    - Proper submission via GitHub. (5)
    - Evidence of branching and PRs being used within the repo with regular commits from the whole team (5)
    - Well-structured and organized repository. (5)
    - **Video presentation of your project. Please keep to between 5-10 minutes, Make sure all group members are participating in the presentation. This is important to show us your understanding! (10)**

Submission Guidelines

The project must be submitted via GitHub. ZIP files will not be marked**.**

Project Start: October 17th

Project End: November 2nd