NANYANG TECHNOLOGICAL UNIVERSITY

Software Engineering Report

CE2006 SOFTWARE ENGINEERING AY20/21 SEM 2

Darryl Tan Kah Heng
Chew Loh Seng
Chockalingam Kasi
Lim Jun Wei
Royce Tan Chun Wei

Lab Group: SE1

Table of Contents

Introductory Notes

Description of all documents submitted

1. Readme.txt
   Contains all of the descriptions of all documents submitted and how to run the application.

2. Software Requirements Specifications
   Contains:
   a. Functional and Non-functional Requirements
   b. Use Case Model Diagram
   c. Use Case Descriptions Report
   d. Requirement Analysis – Class Diagrams
   e. Requirement Analysis – Sequence Diagrams
   f. Requirement Analysis – Dialogue Map

3. Software Engineering Report
   Contains:
   a. System Architecture Diagram
   b. Object Design – Class Diagrams
   c. Object Design – Sequence Diagrams
   d. Software Engineering Practices Used

4. Use Case Model Diagram
5. Use Case Descriptions Report

6. Requirement Analysis – Class Diagrams
7. Requirement Analysis – Sequence Diagrams
8. Requirement Analysis – Dialogue Map

9. System Architecture Diagram
10. Object Design – Class Diagrams
11. Object Design – Sequence Diagrams

12. HealthZoneApplication.zip
    This is where our actual product lies.
13. HealthZone Demonstration Video

How to run the HealthZone Application

Steps to run the application:

1) Unzip the ZIP with folder name: "medicine_reminders"
2) Open Visual Studio Code (VS Code)
3) Open the project folder
4) You might have to install Flutter and Dart in VS Code, as well as set up Android Studio.
   Details on setting up Android Studio: https://flutter.dev/docs/get-started/install
   Details on installing Flutter and Dart in VS Code:
   https://flutter.dev/docs/development/tools/vs-code
5) Inside terminal, type in "flutter clean". This is to make sure dependencies are not based on the developer's PC.
6) After cleanup, open Android Phone Emulator (in Android Studio) or plug in Android phone in USB debugging mode.
   Details of how to set up Android Phone Emulator in Android Studio:
   https://developer.android.com/studio/run/emulator
   Details of how to set up USB debugging mode:
   https://developer.android.com/studio/debug/dev-options
7) Inside terminal, type in "flutter pub get". This is to get all the external packages set up in your system in VS Code.
8) Inside terminal, type in "flutter run". This is to set up the file structures based on your system in VS Code.
9) After building, you might need to run the application if it is not done automatically.
   Type in F5 (debugging mode) or Ctrl+F5 (Run without debugging) to install the apk into the Android emulator or phone.

In case the flutter build fails, these steps might need to be done:

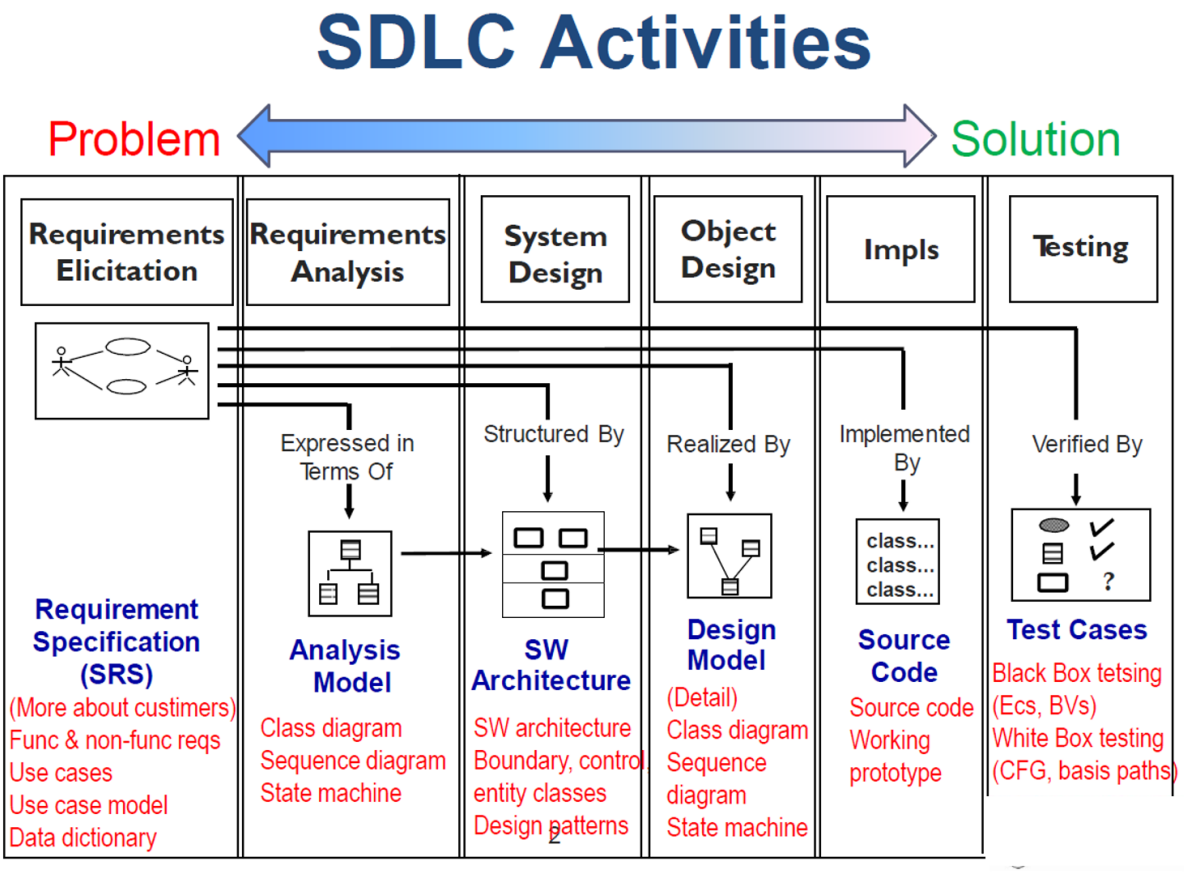
1) Unzip the ZIP with folder name: "medicine_reminders".
2) Open Visual Studio Code (VS Code).
3) You might have to install Flutter and Dart in VS Code, as well as set up Android Studio.
   Details on setting up Android Studio: https://flutter.dev/docs/get-started/install
   Details on installing Flutter and Dart in VS Code:
   https://flutter.dev/docs/development/tools/vs-code
4) Create new project under the package name "medicine_reminders"
5) Copy and replace the following files from the downloaded (and upzipped) "medicine_reminders" to the newly created "medicine_reminders":
   - All folder and files in the lib folder
   - All folder and files in the assets folder
   - "pubspec.yaml" file in the root folder
   - All folder and files in "android/app/src" folder
6) Open Android Phone Emulator (in Android Studio) or plug in Android phone in USB debugging mode.
   Details of how to set up Android Phone Emulator in Android Studio:
   https://developer.android.com/studio/run/emulator
   Details of how to set up USB debugging mode:
   https://developer.android.com/studio/debug/dev-options

7) Inside terminal, type in "flutter pub get". This is to get all the external packages set up in your system in VS Code.
8) Inside terminal, type in "flutter run". This is to set up the file structures based on your system in VS Code.
9) After building, you might need to run the application if it is not done automatically. Type in F5 (debugging mode) or Ctrl+F5 (Run without debugging) to install the apk into the Android emulator or phone.

Please submit a ticket to Group Leader Darryl Tan (dtan103@e.ntu.edu.sg) or Assistant Group Leader Loh Seng (chew0398@e.ntu.edu.sg) if you need help.

Purpose of this Software Engineering Report

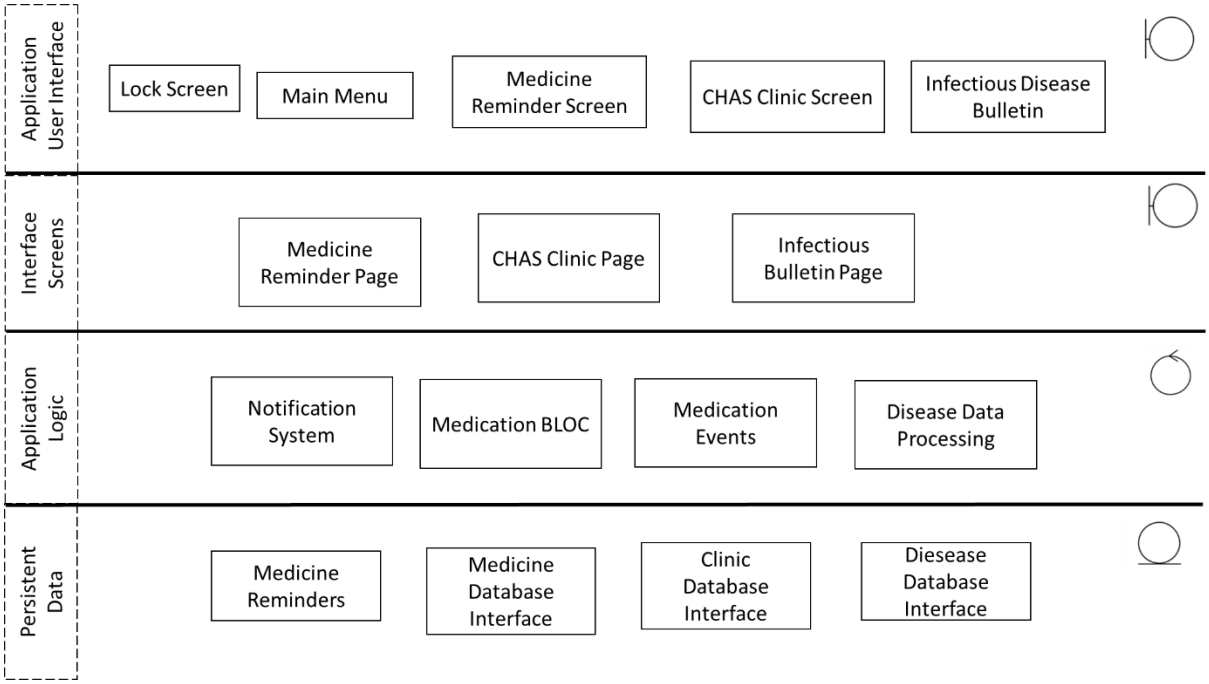
This report is made to document the Software Development Lifecycle (SDLC) and the state of the changing documents throughout this project. These changes are shown to highlight the effort made in making sure that we stick to the SDLC as much as possible, given the restrictions of this project.



The Software Requirements Specifications (SRS) only covers the Requirements Elicitation and Requirements Analysis part of the SDLC as it is only meant for communicating with the clients and detailed implementation is not necessary when talking to cilents. This report will cover the rest of the stages.

System Design

Software Architecture



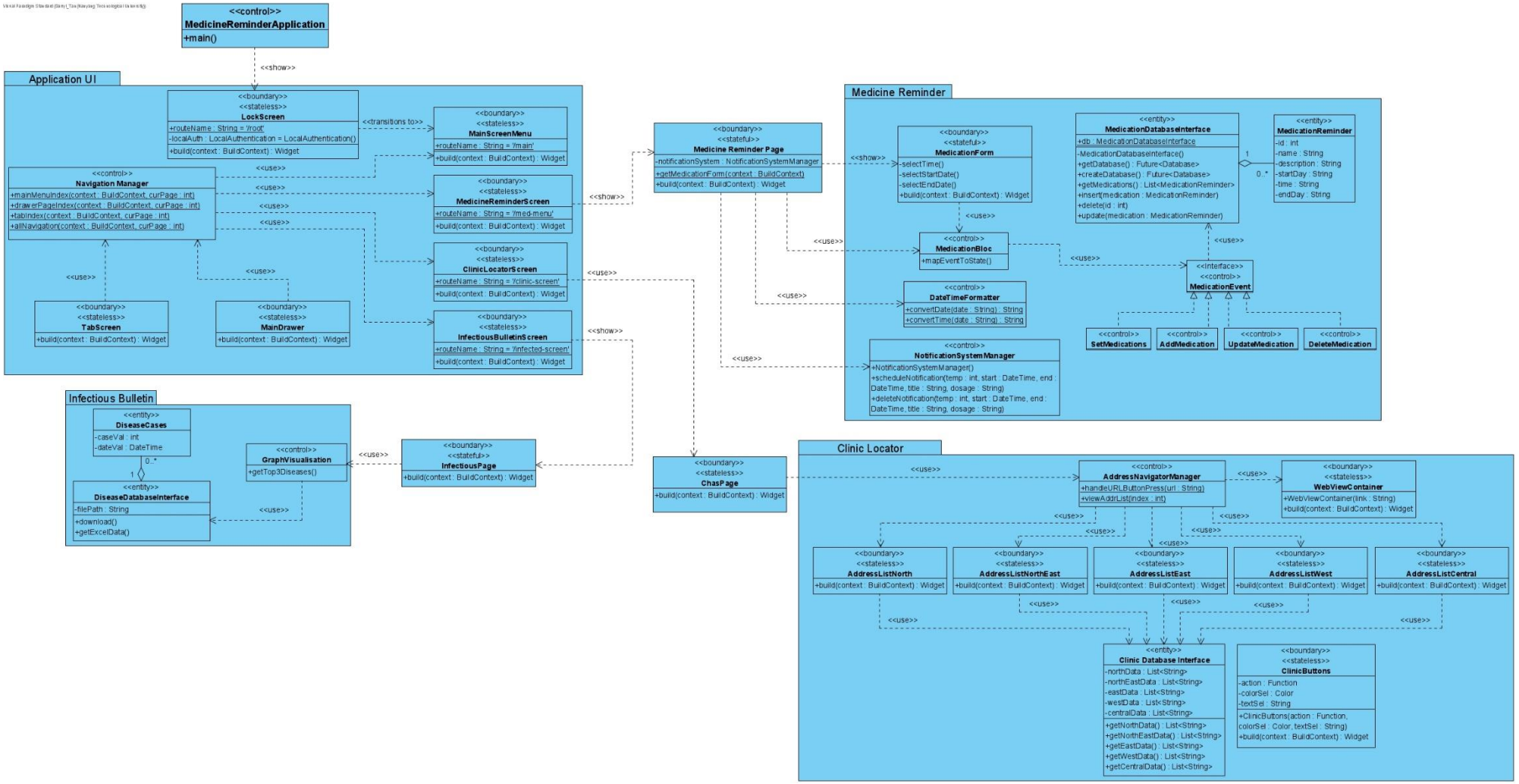
During the system design phase, our team started to analyse the flutter framework and tried to understand how we could translate our class and sequence diagrams from the SRS into the flutter framework. From this, we have come up with this layered system architecture. The layered system architecture is simple to understand and is simple to divide classes according to the boundary, control, and entity class stereotypes. It also represents the composition of our system and how some of the components are abstracted.

For example, the medicine reminder page is abstracted from the various components of the medicine reminder component. More information is shown in the class diagram below.

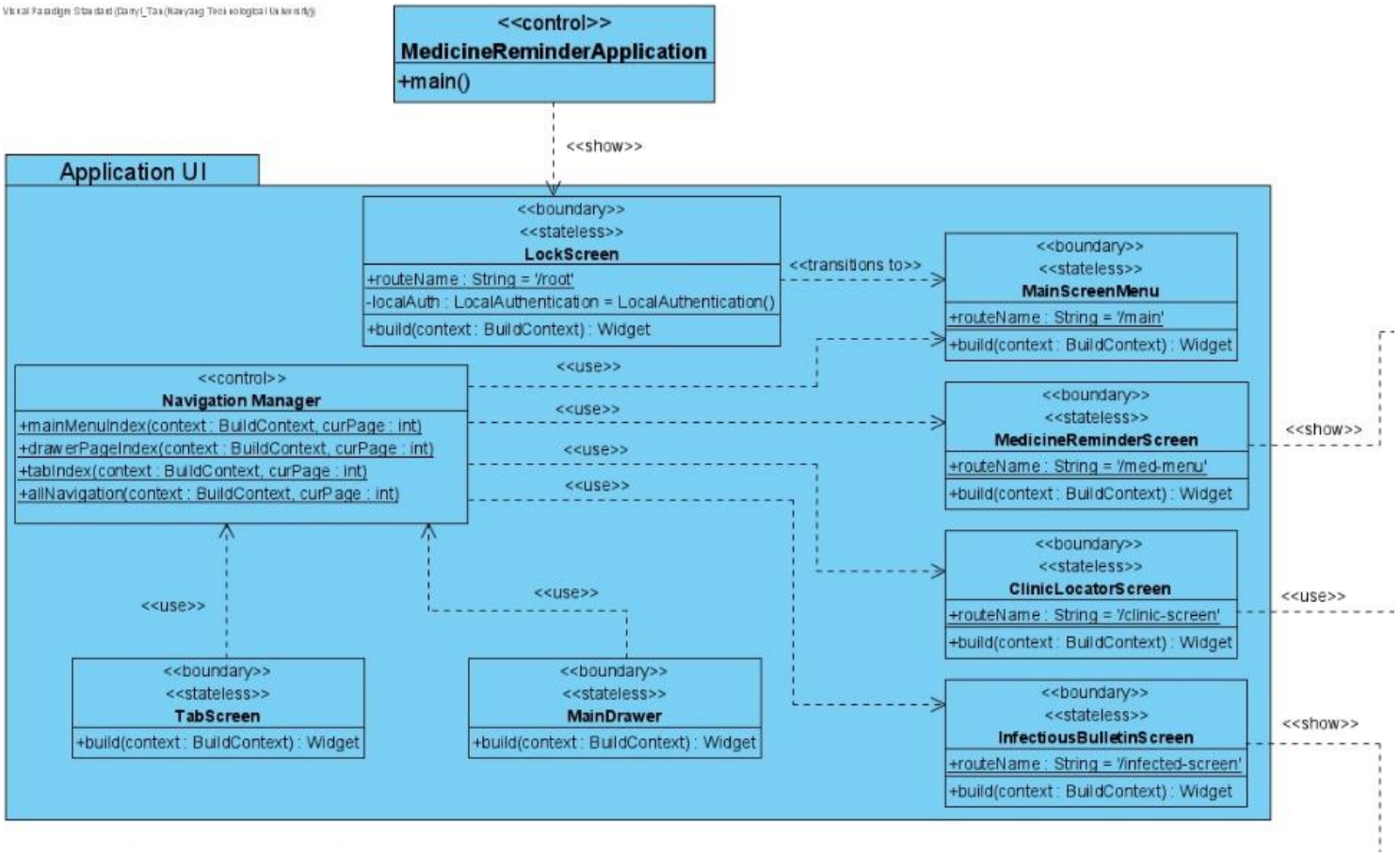
Object Design

Updated Class Diagram

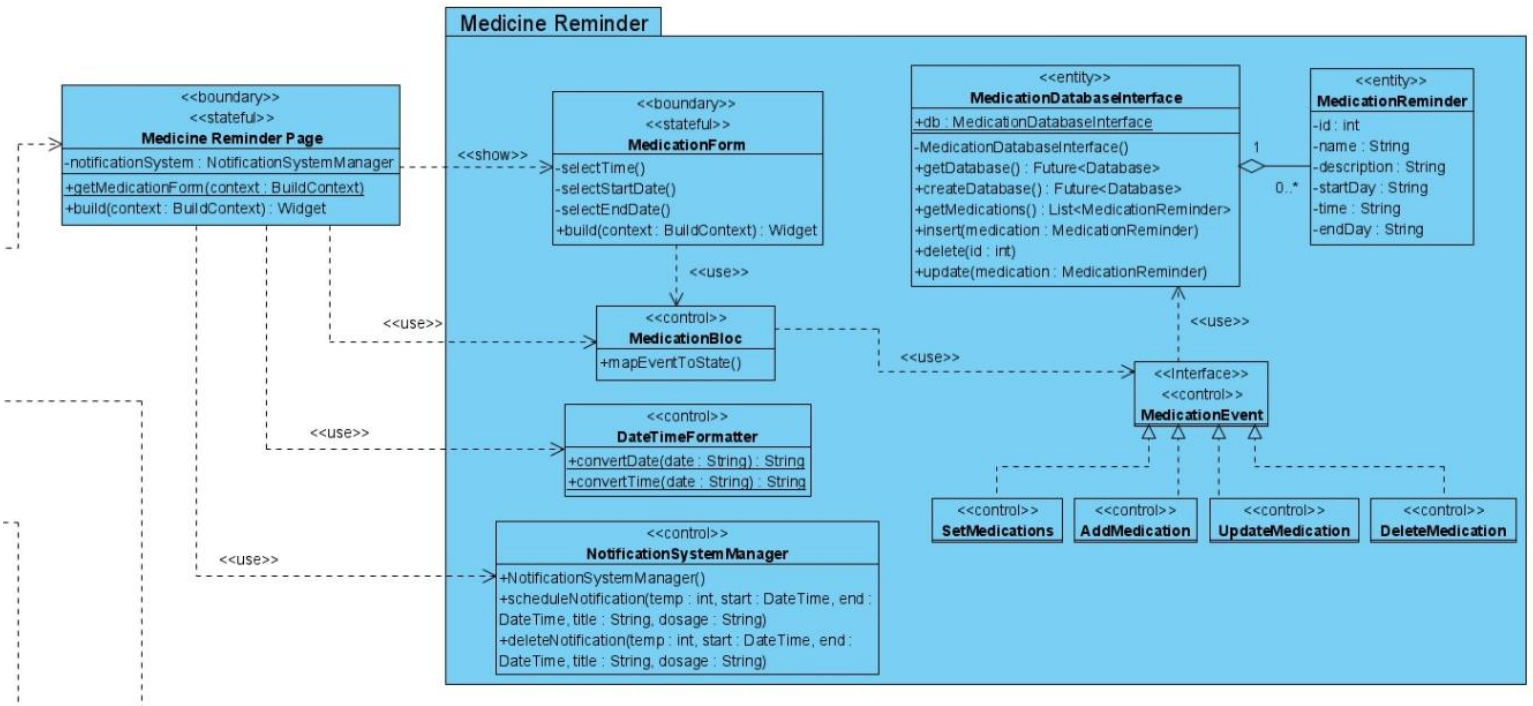
Overall Class Diagram



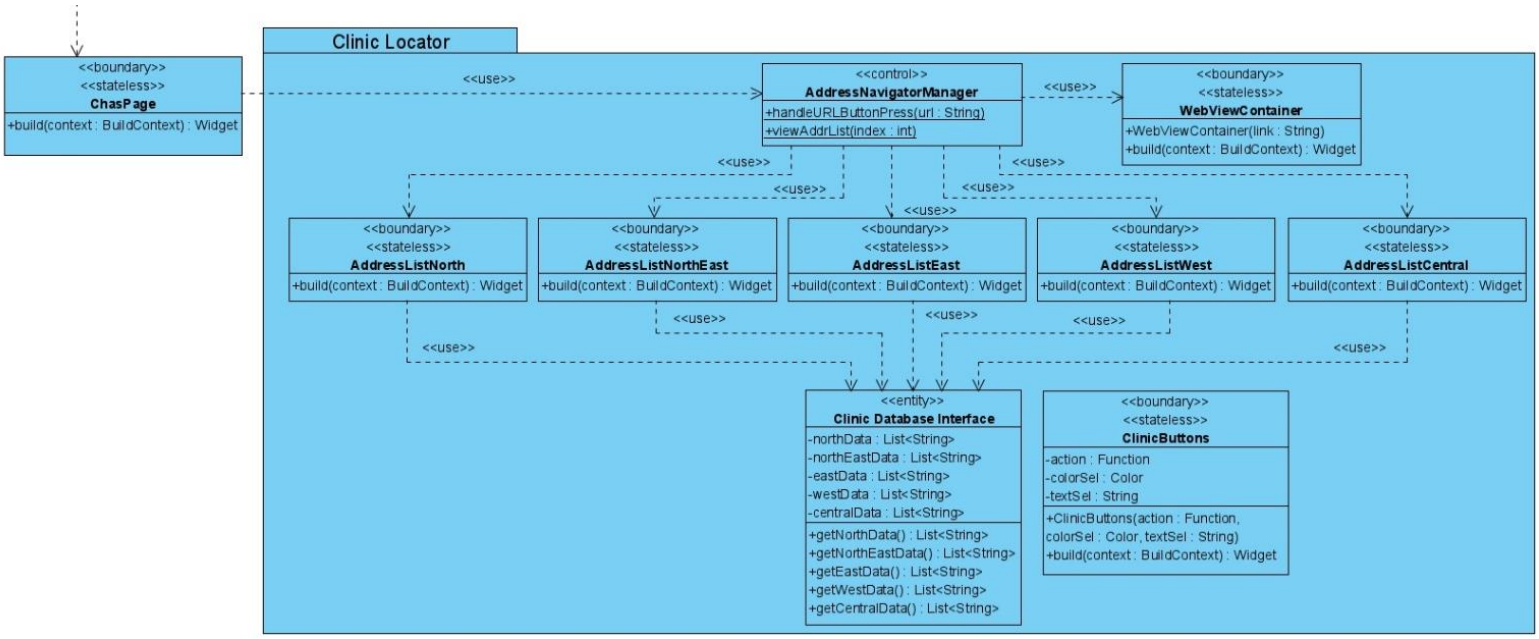
Application User Interface Component



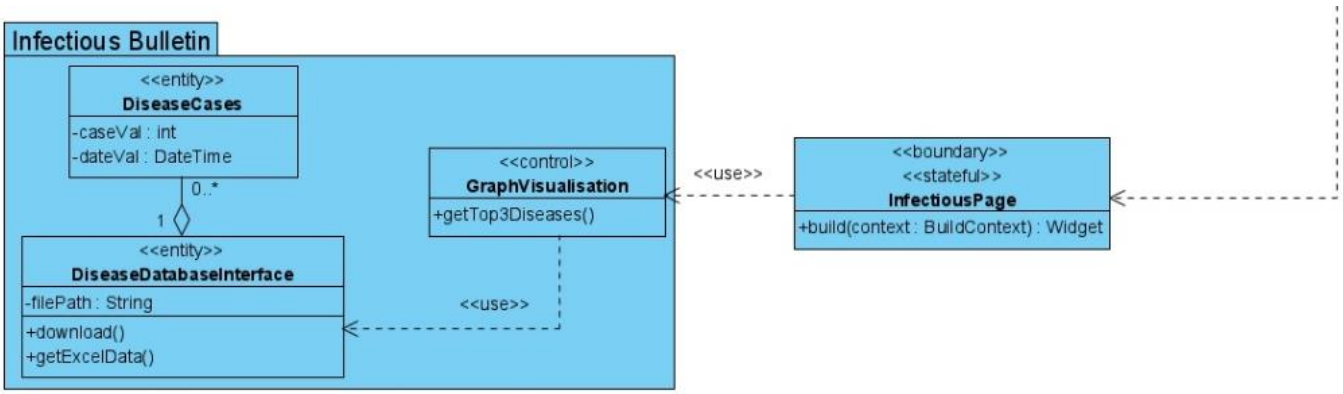
Medicine Reminder Component



Clinic Locator Component



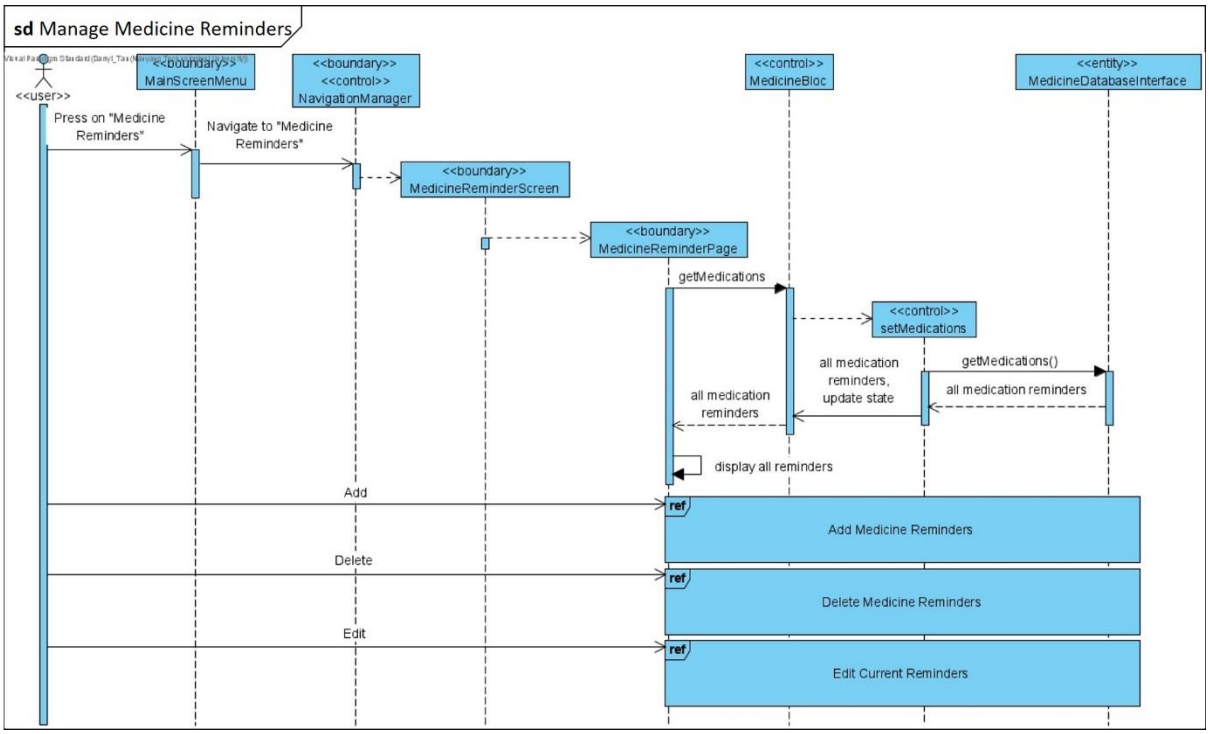
Infectious Bulletin Component



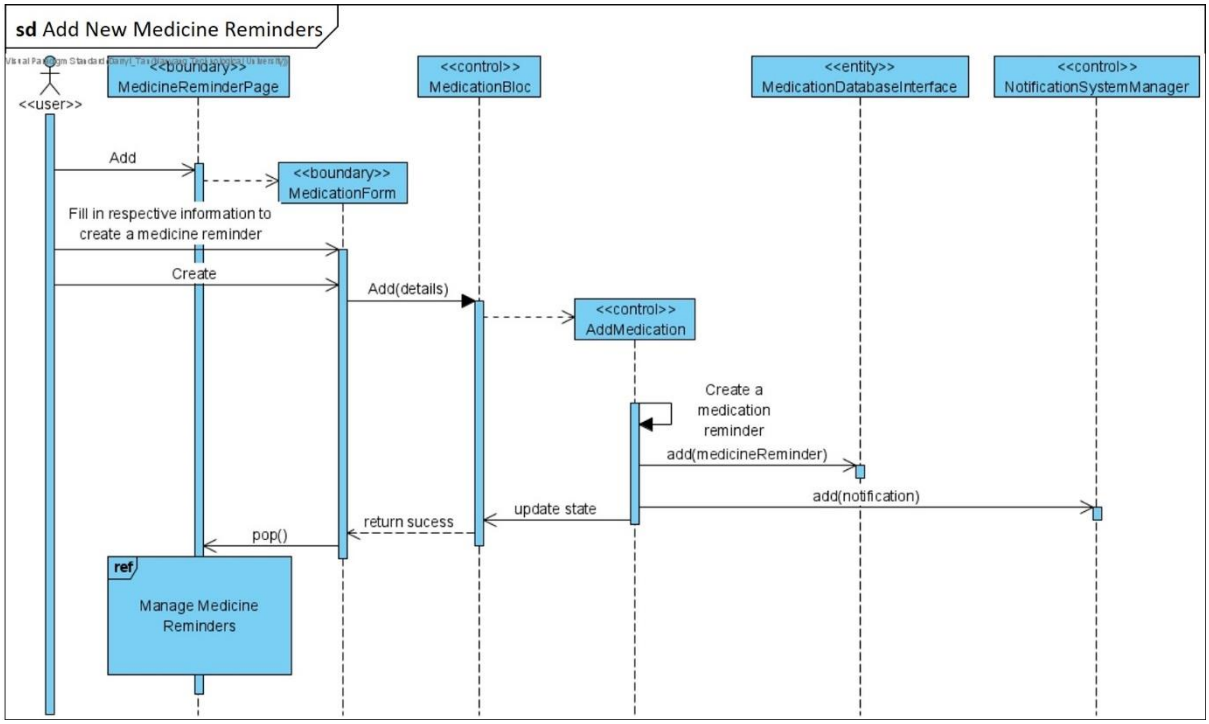
As you can see, the application is split into 4 components – Application User Interface, Medicine Reminder, Clinic Locator, and Infectious Bulletin. There is also an interface for each component, corresponding to the Façade Software Engineering Design Principle taught by the Gang of Four.

Updated Sequence Diagram

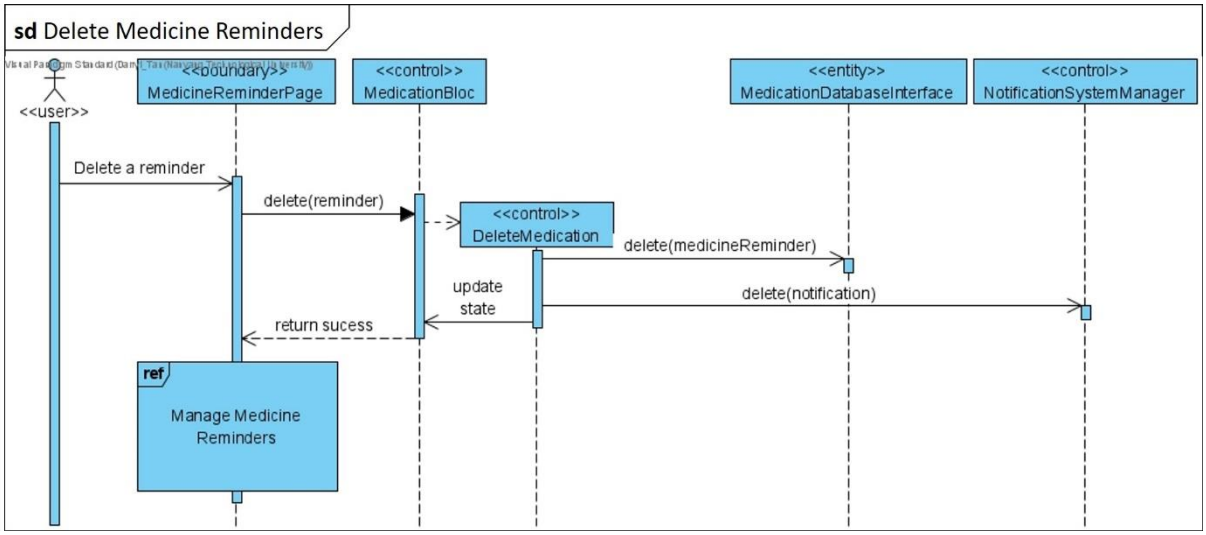
Use Case 01 – Manage Medicine Reminders



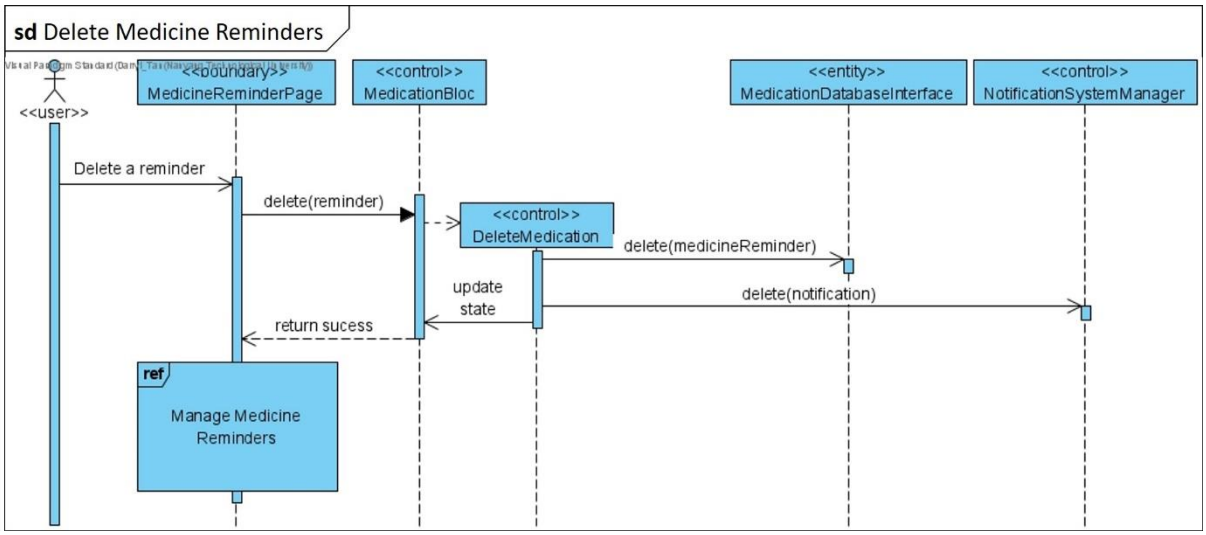
Use Case 02 – Add New Medicine Reminder



Use Case 03 – Edit Current Medicine Reminder



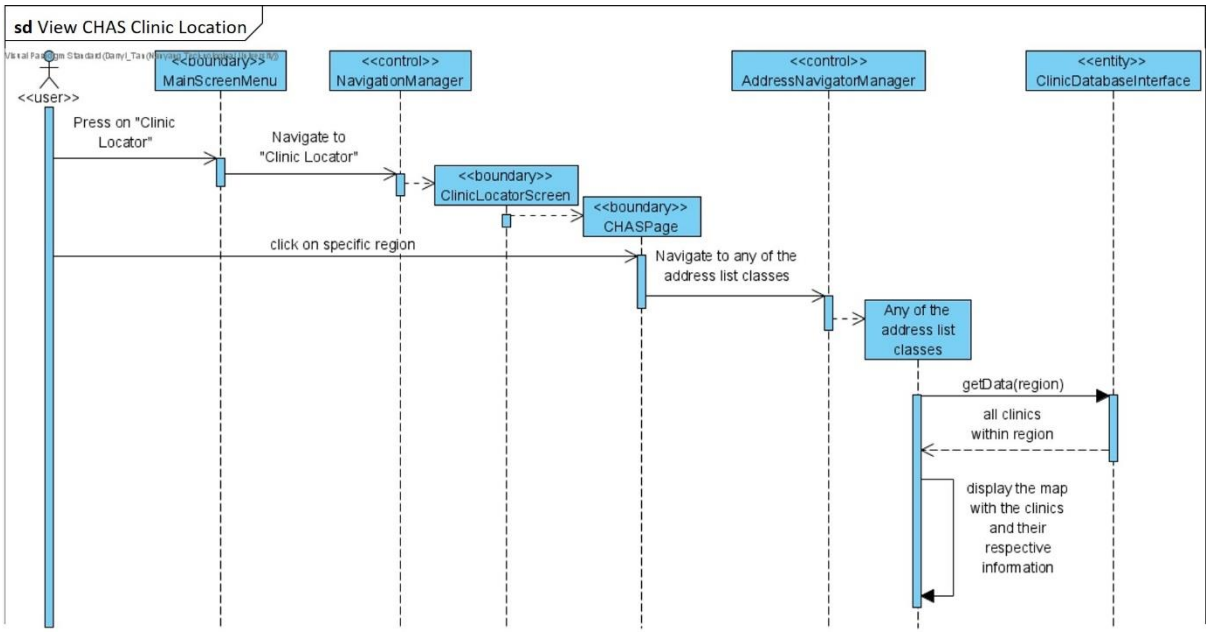
Use Case 04 – Delete Medicine Reminders



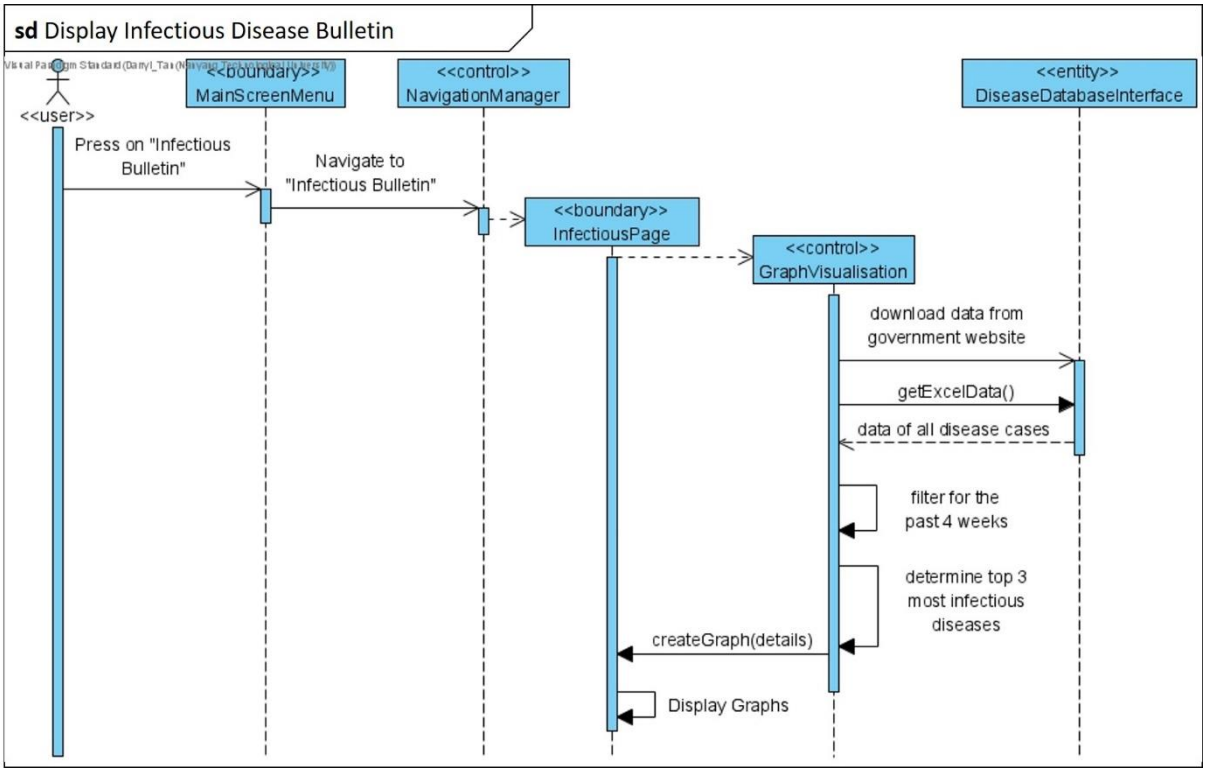
Use Case 05 – Send Medicine Reminder & Notify User

There is no sequence diagram for this use case as we realized that this use case will be achieved by the Android Notification System.

Use Case 06 – View CHAS Clinic Information

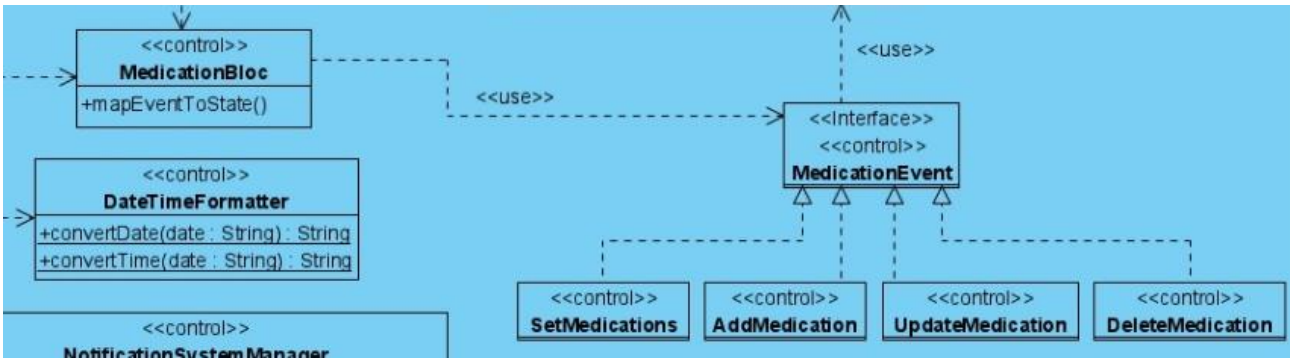


Use Case 07 – Display Infectious Disease Bulletin



Software Engineering Patterns Used

Strategy Pattern



MedicationEvent is an interface class, realised by 4 concrete classes – SetMedications, AddMedication, UpdateMedication and DeleteMedication, which are the 4 events that we would want to update the state of MedicationBloc. It is utilising the strategy pattern as these 4 events are interchangeable by MedicationBloc.

Façade Pattern

As you can see from the class diagram, the application is split up into 4 components – Application User Interface, Medicine Reminder, Clinic Locator and Infectious Bulletin. There are 3 interface classes that helps to interface the 3 main functions of the application with the application UI – Medicine Reminder Page, Clinic Locator Page and Infectious Bulletin Page. This layout and design corresponds to the Façade Design Pattern shown by the Gang of Four.

This design helps to reduce coupling among classes and is easier for our team to further develop sperate components of the application without fear of affecting other components of the application.