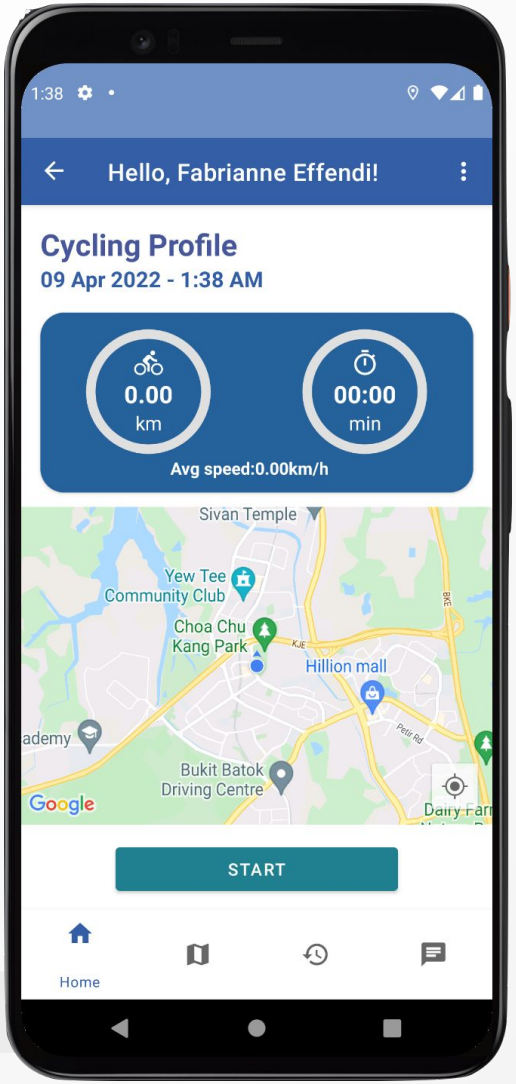
BIKER-X

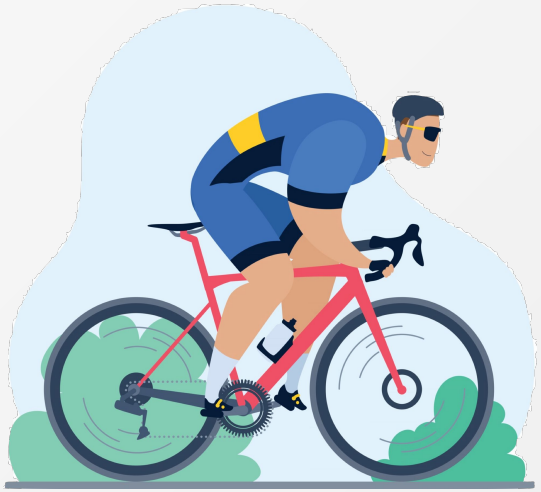
PROJECT DEMO

Lee Xuan Hua
Liau G Wayne
Lek Jie Kai
Loh Yi Ze
Fabrianne Effendi



BIKER-X

A mobile cycling buddy that tracks your cycling session and connects you with like-minded bikers



Target market: Local biking enthusiasts

Competitive Landscape

Cycling apps in the local and international landscape

Strava
App for cyclists & runners

MapMyRide
By Under Armour

RidenJoy (Local)
SG mobility lifestyle app

KEY FEATURES OF BIKER-X

Get Popular Local Biking Routes

View & Search
Recommended Routes

Track Cycling Session

Embark along cycling routes or cycle on your own route with live GPS tracking

Plan Own Routes

View Full Map to aid planning
View, Search & Filter Amenities

Track Cycling History and Goals

View Cycling History
View & Edit Goals

Chat with Biker-X Community

View Forum Threads
Send Chat Messages

Personal Account

Register & Login for a personalised experience

SYSTEM ARCHITECTURE

Android Fragment

- ➔ Frontend
- ➔ Populates UI

Android View Model

- ➔ Backend
- ➔ Handles data interaction with database

SYSTEM ARCHITECTURE

Gov.sg API

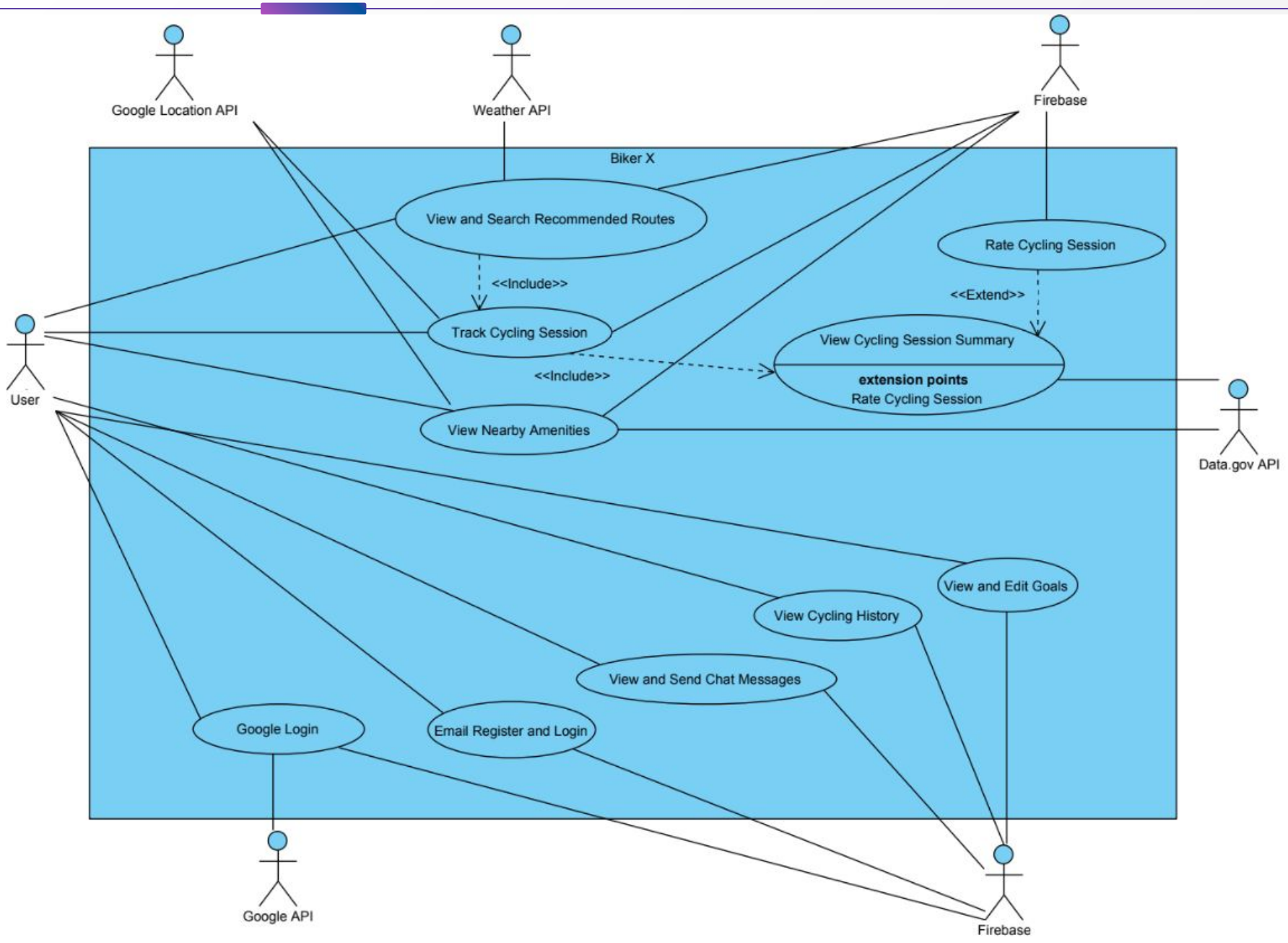
- Retrieve relevant data to enhance user experience
- E.g. Park Amenities, Cycling Routes, Weather

Firebase

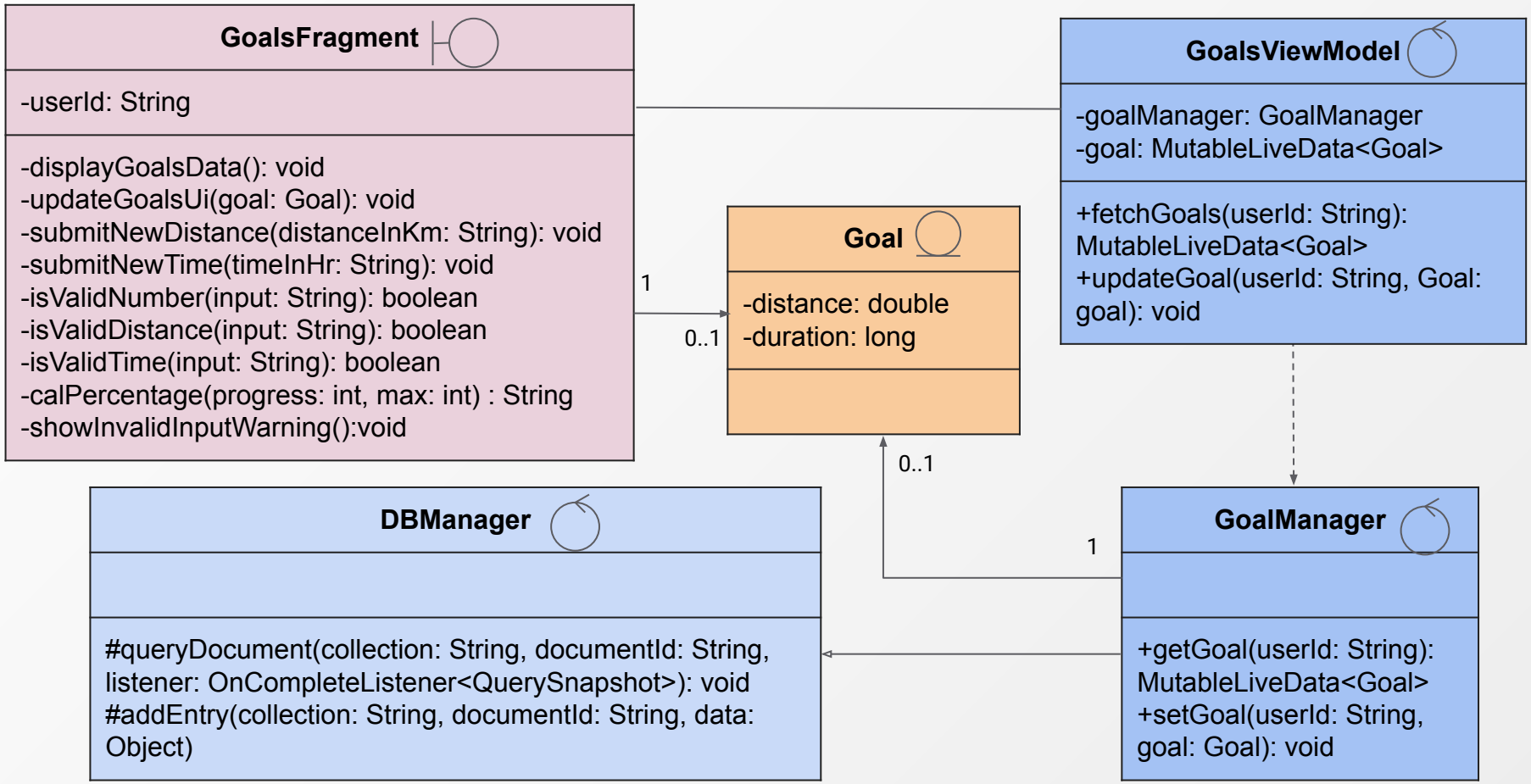
- Authenticate users (email)
- Non-relational database

Google API

- Authenticates users (Gmail)
- Handles geolocation tracking


Google Location API
Weather API
Firebase
Biker X
View and Search Recommended Routes
Rate Cycling Session
<<Include>>
<<Extend>>
Track Cycling Session
View Cycling Session Summary
<<Include>>
extension points
Rate Cycling Session
User
View Nearby Amenities
Data.gov API
View and Edit Goals
View Cycling History
View and Send Chat Messages
Google Login
Email Register and Login
Google API
Firebase

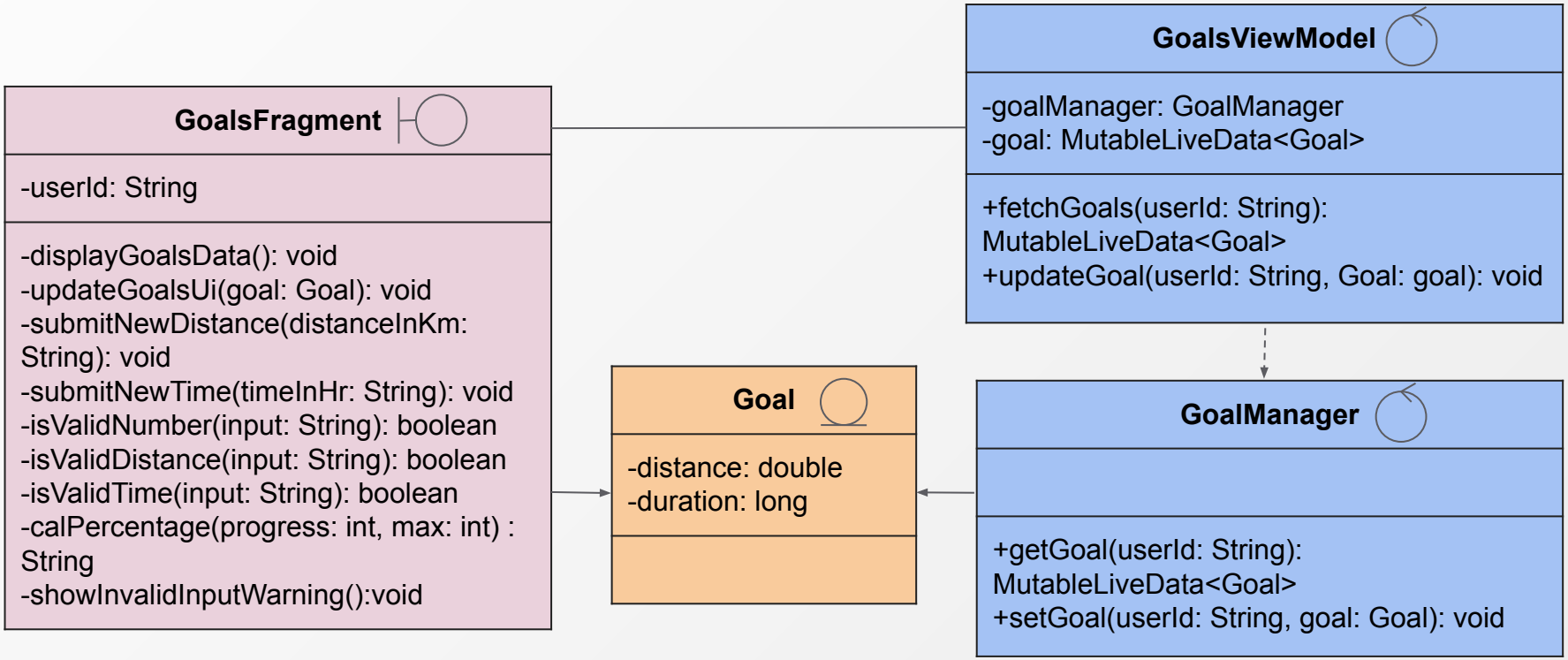
BIKER-X USE CASES - View & Edit Goals



BIKER-X USE CASES - View & Edit Goals

Design Principles

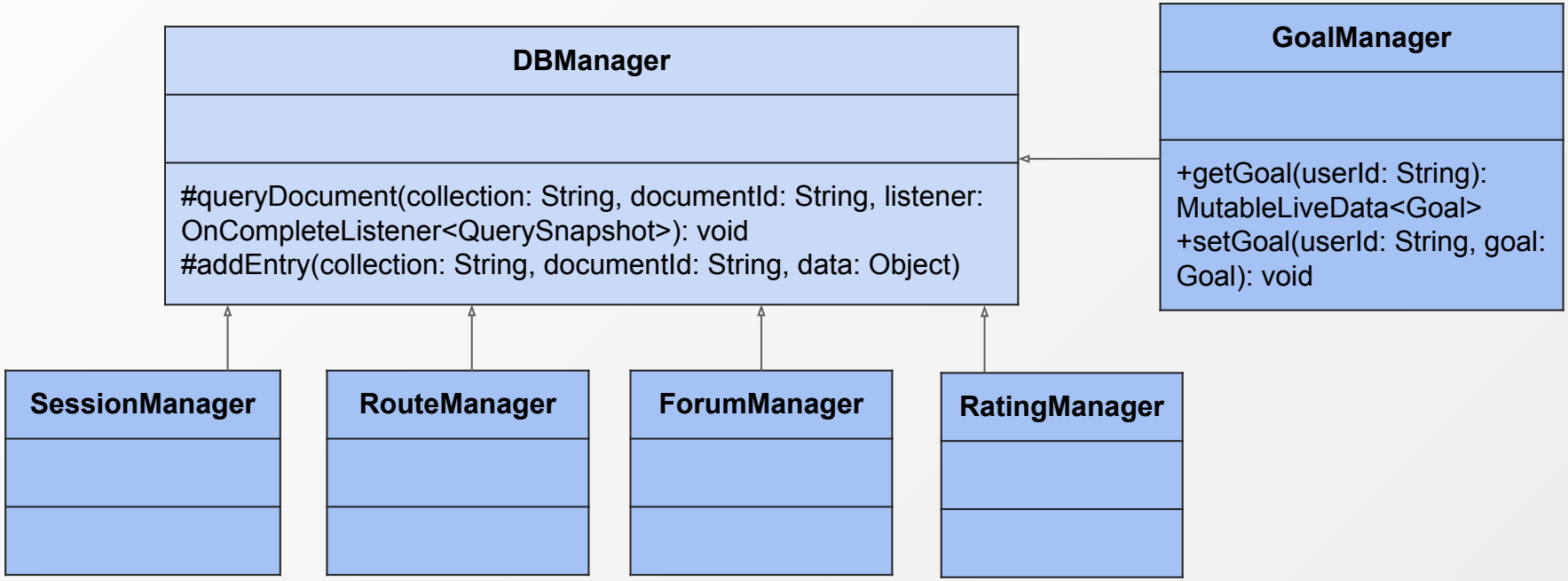
- Single Responsibility Principle

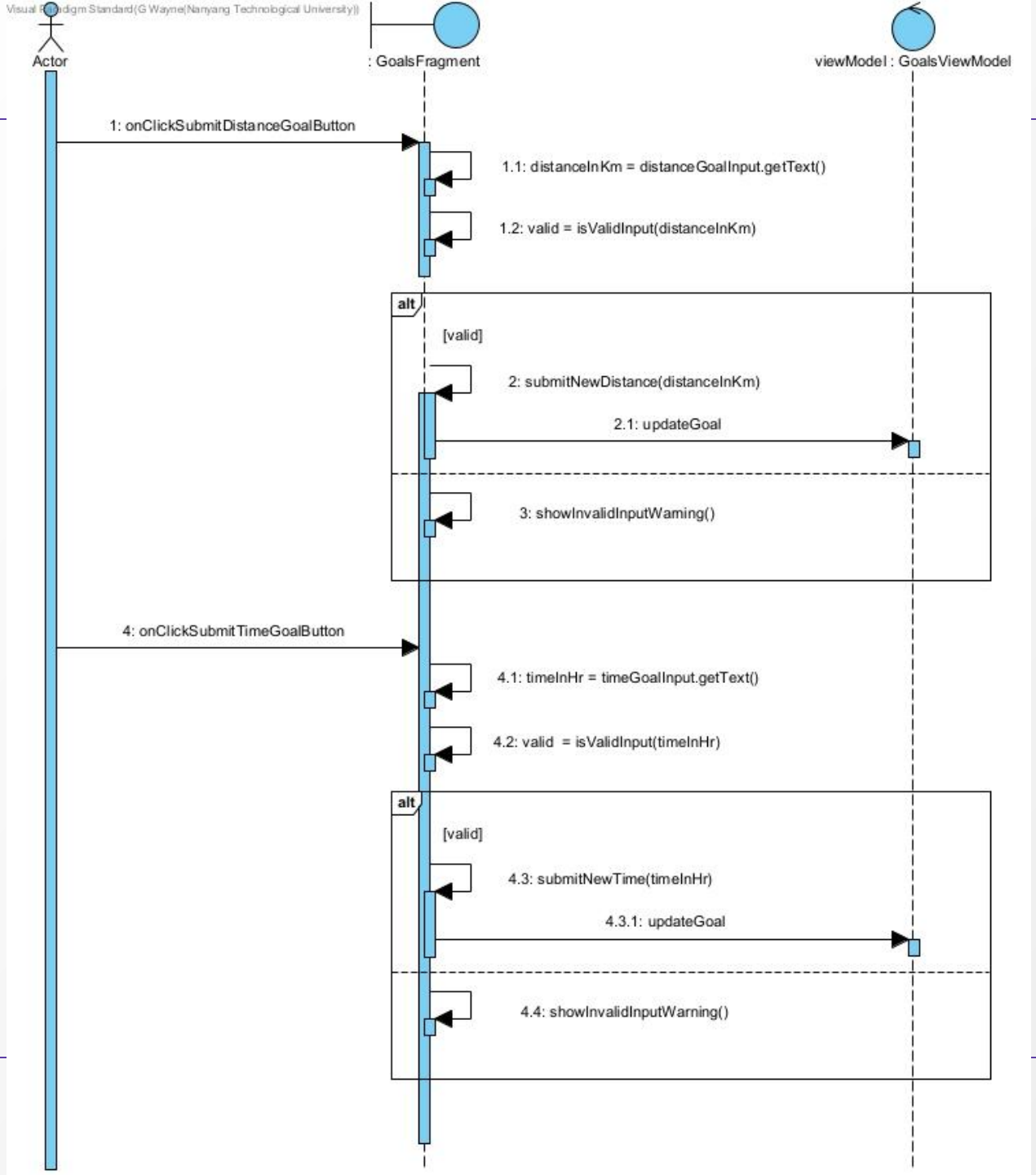


BIKER-X USE CASES - View & Edit Goals

Design Principles

- Separate aspects that vary from what stays the same
- Open for extension, closed for modification
- Loose Coupling




Visual Paradigm Standard(G Wayne(Nanyang Technological University))
Actor
: GoalsFragment
viewModel : GoalsViewModel
1: onClickSubmitDistanceGoalButton
1.1: distanceInKm = distanceGoalInput.getText()
1.2: valid = isValidInput(distanceInKm)
alt
[valid]
2: submitNewDistance(distanceInKm)
2.1: updateGoal
3: showInvalidInputWarning()
4: onClickSubmitTimeGoalButton
4.1: timeInHr = timeGoalInput.getText()
4.2: valid = isValidInput(timeInHr)
alt
[valid]
4.3: submitNewTime(timeInHr)
4.3.1: updateGoal
4.4: showInvalidInputWarning()

DEMO


THE NEXT DAY

Black Box Testing (Set and View Distance Goals)

Equivalence Class for distance goals

Valid EC	0 to 9999
Invalid EC 1	>= 10,000
Invalid EC 2	Non-numeric input

Boundary Values for distance goals

Valid EC (0 to 9999)	Two Boundary Values	0 (lower boundary)
		9999 (upper boundary)
	Lower Boundary (0)	0,1
	Upper Boundary(9999)	9998,9999,10000
Invalid EC (10,000 to infinity)	Two Boundary Values	10000 (lower boundary)
		infinity (upper boundary)
	Lower Boundary (10000)	9999,10000,10001
	Upper Boundary(infinity)	A number much larger than 10000

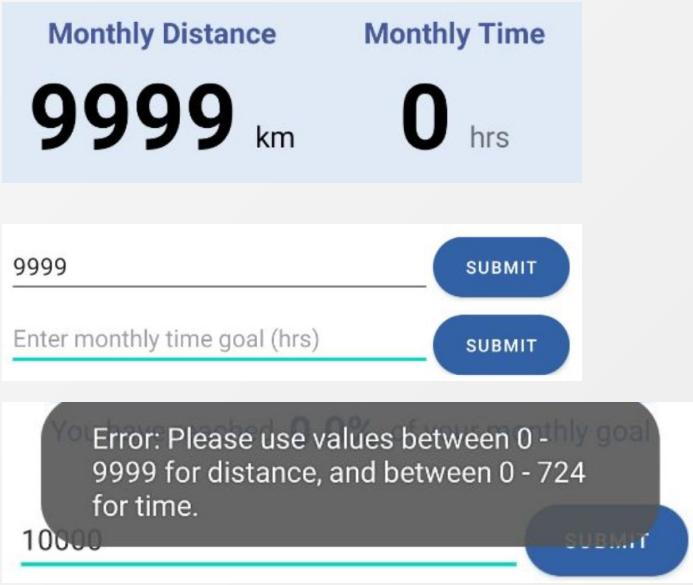
Black Box Testing (Test Cases)

Test Case 1: Distance Goal

* - Error Message (“Error: Please use values between 0-9999 for distance, and between 0-744 for time.)

^ - live updated value to be seen directly on the screen

Input	Expected Result	Actual Result
0	^0	^0
9999	^9999	^9999
99.9	^100	^100
10 000	*	*
999 999	*	*
.	*	*



Black Box Testing (Set and View Time Goals)

Equivalence Class for time goals

Valid EC	0 to 744
Invalid EC 1	>= 745
Invalid EC 2	Non numeric input

Boundary Values for time goals

Valid EC (0 to 744)	Two Boundary Values	0 (lower boundary)
		744 (upper boundary)
	Lower Boundary (0)	0,1 (negative inputs not allowed)
	Upper Boundary(744)	743,744,745
Invalid EC (745 to infinity)	Two Boundary Values	745 (lower boundary)
		infinity (upper boundary)
	Lower Boundary (745)	744,745,746
	Upper Boundary(infinity)	A number much larger than 745

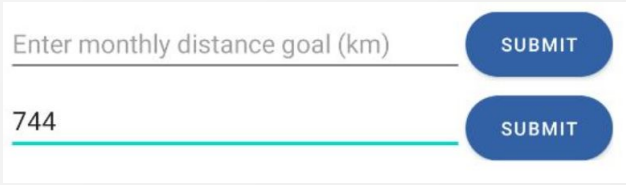
Black Box Testing (Test Cases)

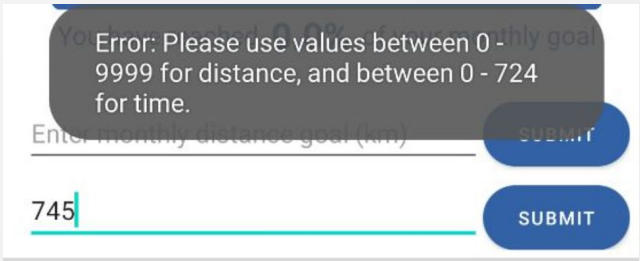
Test Case 2: Time Goal

* - Error Message (“Error: Please use values between 0-9999 for distance, and between 0-744 for time.)

^ - live updated value to be seen directly on the screen

Input	Expected Result	Actual Result
0	^0	^0
744	^744	^744
99.9	^100	^100
745	*	*
999 999	*	*
.	*	*

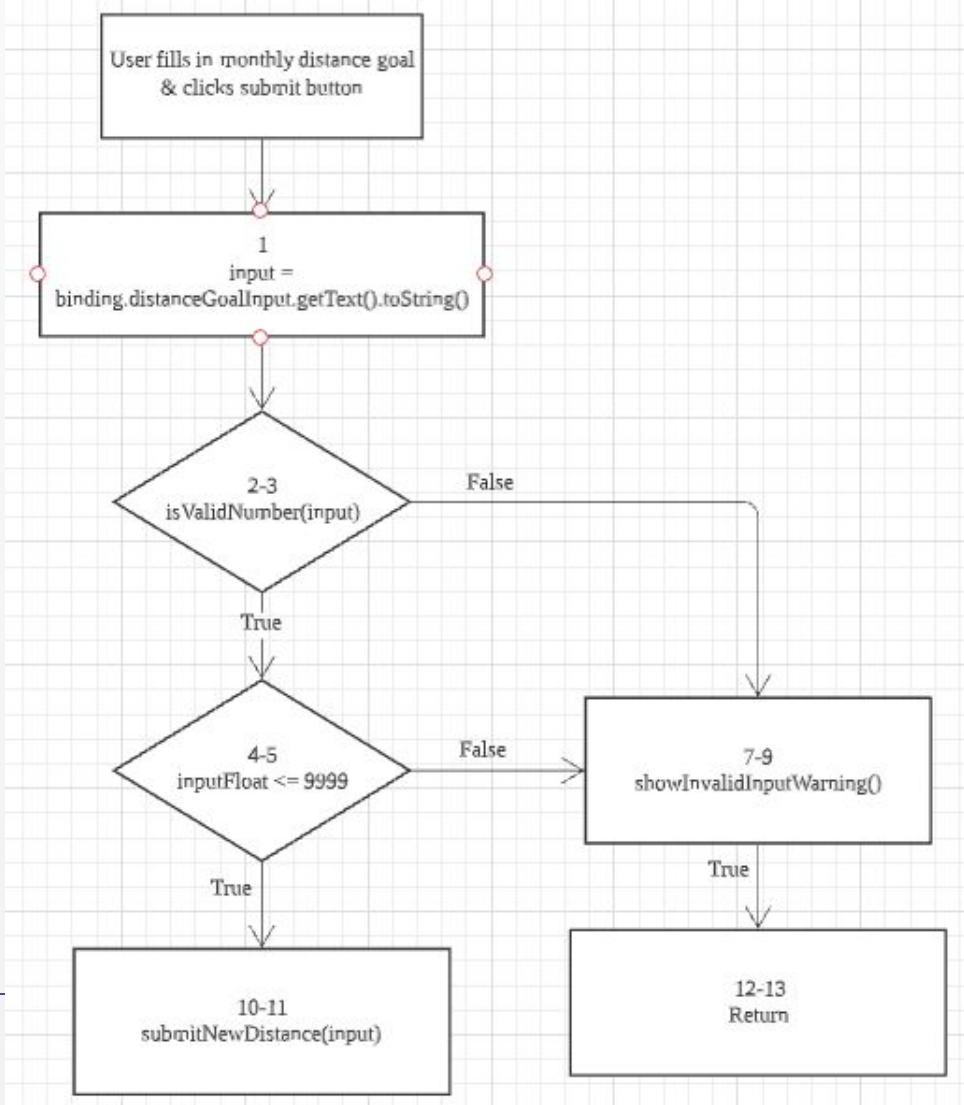




White Box Testing(distance goals)

```
    public void onClick(View view) {
1                   String input = binding.distanceGoalInput.getText().toString();
2                   boolean valid = isValidDistance(input);
8                   if (!valid) {
9                   showInvalidInputWarning();
10                  } else {
11                  submitNewDistance(input);
12          }
13          return
            }
    }

    private boolean isValidDistance(String input) {
3           if (isValidNumber(input)) {
4             float inputFloat = Float.parseFloat(input);
5             if (inputFloat <= 9999) return true;
6           }
7           return false;
    }
```



White Box Testing(distance goals)

Cyclomatic Complexity

Cyclomatic complexity = |decision points| + 1 = 2 + 1 = 3

Test Cases

I. User fills in valid number and distance
II. User fills in valid number and invalid distance
III. User fills in invalid number and distance

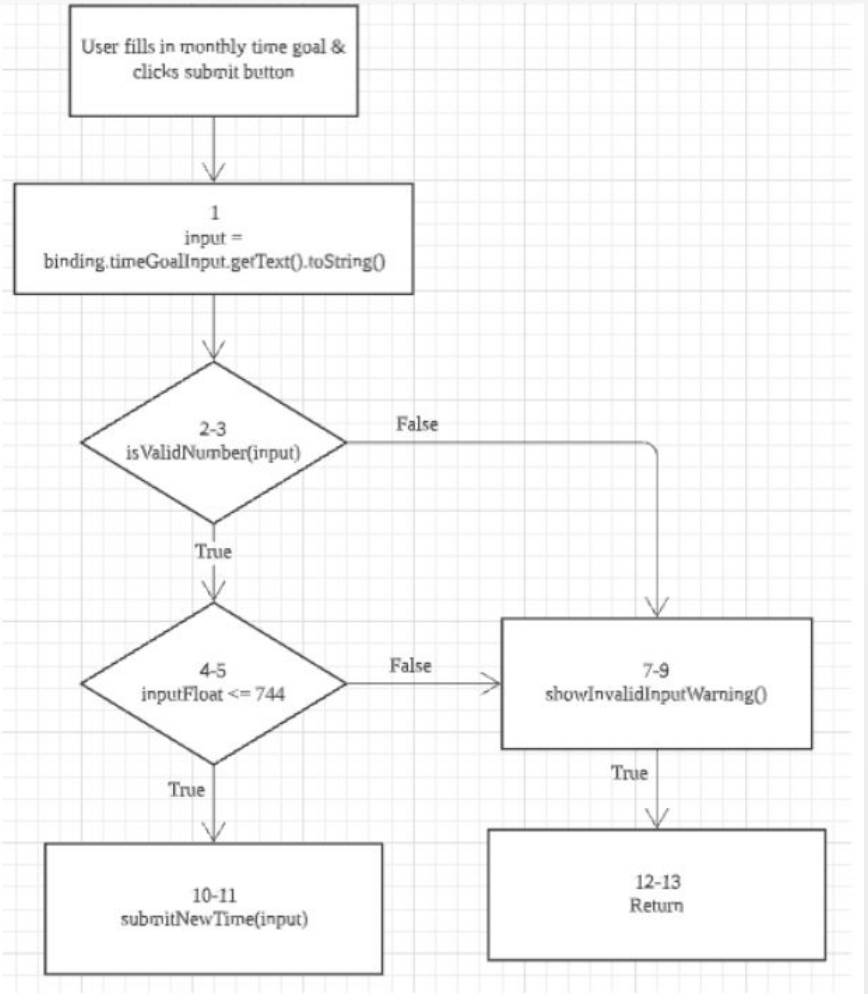
Execution Paths

I. Path 1 (Baseline): 1, 2-3, 4-5, 10-11
II. Path 2: 1, 2-3, 4-5, 7-9, 12-13 (User fills in valid number and invalid distance)
III. Path 3: 1, 2-3, 7-9, 12-13 (User fills in invalid number and distance)

White Box Testing(time goals)

```
    public void onClick(View view) {
1                   String input = binding.timeGoalInput.getText().toString();
2                   boolean valid = isValidDistance(input);
8                   if (!valid) {
9                   showInvalidInputWarning();
10                  } else {
11                  submitNewTime(input);
12              }
13              return
    }

    private boolean isValidTime(String input) {
3           if (isValidNumber(input)) {
4               float inputFloat = Float.parseFloat(input);
5               if (inputFloat <= 744) return true;
6           }
7           return false;
    }
```



White Box Testing(time goals)

Cyclomatic Complexity

Taking Cyclomatic complexity: |decision points| + 1 = 2 + 1 = **3**

Test Cases

I. User fills in valid number and time
II. User fills in valid number and invalid time
III. User fills in invalid number and time

Execution Paths

I. Path 1 (Baseline): 1, 2-3, 4-5, 10-11
II. Path 2: 1, 2-3, 4-5, 7-9, 12-13 (User fills in valid number and invalid time)
III. Path 3: 1, 2-3, 7-9, 12-13 (User fills in invalid number and time)

Project Timeline

Sprint 1: Ideation

- Requirements Elicitation
- UI Design and Mockup

Sprint 2: Refine Use Case & Model

- Use Case Descriptions
- Use Case Diagrams

Sprint 3: Visualise Dynamic Model

- Sequence Diagram
- Dialog Map
- Class Diagrams
- System Architecture

Sprint 4 and 5: Implement Design

- Database implementation
- Front & Back-end implementation
- Code documentation

Sprint 6:Testing

- Generate Test cases
- Code refinements
- Fixing bugs

SOFTWARE ENGINEERING PRACTICES

AGILE

Incremental Delivery

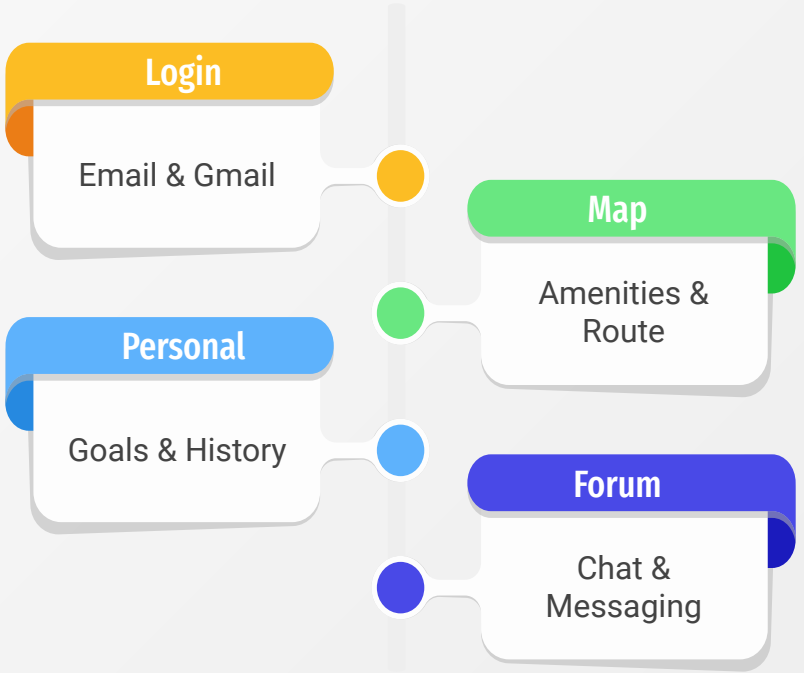
Embrace Change

Maintain Simplicity

SOFTWARE ENGINEERING PRACTICES

Incremental Delivery

The software is developed in increments with the customer specifying the requirements to be included in each increment



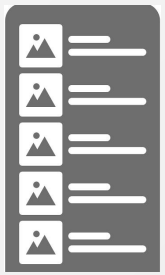
SOFTWARE ENGINEERING PRACTICES

Embrace Change

Expect system requirements to change, thereby design system to accommodate these changes

Open for Extension, Closed for Modification

Dimensions and values stored in resource file



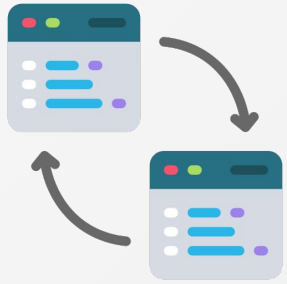
Recycler Views

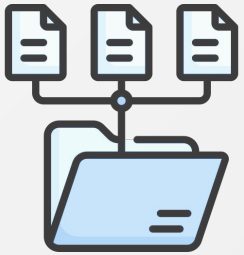
SOFTWARE ENGINEERING PRACTICES

Maintain Simplicity

Simplicity in both the software being developed and development process & actively eliminate system complexity

Separate Aspects That Vary From What Stays The Same





Refactor code to ensure readability, eliminate redundancy & illustrate clear sequential logical process

THANK YOU!