SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

# CZ2006 SOFTWARE ENGINEERING

## Project Name - DengueSafe

SSP4/Group 5

Team: Ctrl_AH_Defeat

**Team Members:**
Pal, Aratrika (Leader)(U1922069F)
Tan Mei Xuan (U1921903J)
Tan Min, Tricia (U1922619J)
Bachhas Nikita (U1921630F)
Loh Xin Yi (U1922030E)
Ong Li Wen (U1920679B)

# Table of Contents

# 1. Product Description

## 1.1 Purpose

Our DengueSafe app aims to help students of Nanyang Technological University by providing information about on campus reported dengue cases. These on campus dengue cases can only be reported to the app by Admin users belonging to NTU Staff, Fullerton or Jurong Hospital  in order to maintain a level of security and accountability.  In addition, what makes our app unique is being able to display the number of cases in each Hall of Residence using a hotspot map, since visual information can be perceived better. DengueSafe also analyses cases occurring in the rest of Singapore and sends alerts to users if the trend of high number of cases is shifting closer to the university. Effectively, DengueSafe serves two main purposes. They are as follows:

1. The first purpose is to show the active cases and the cases reported today in the university, using a Hotspot Map of the campus implemented using a Google Map API. The app is able to send an alert if any Hall of Residence has more than or equal to 16 active cases. Such areas are referred to as “Red Alert Areas”.
2. The second purpose is to retrieve real time data of dengue cases in Singapore and represent it using maps and line charts to be informative and useful for the students. The app is able to send an alert when the number of cases in Singapore are shifting towards the university, and it detects that there might soon be an inflation of dengue cases on campus.

## 1.2 Scope

Our DengueSafe app is an application that is targeted for students at Nanyang Technological University. It helps student users stay up to date with reported dengue cases on campus. Student users will be able to view all the dengue cases that are in NTU such as the active cases and cases reported today. Student users will also be able to view the dengue statistics of Singapore and Frequently Asked Questions on the application. Students will also be alerted when there are equal to or more than 16 cases in any Hall of

Residence. Admin users can login using their shared admin email and password to report new dengue cases on the application. This will help send updates to students about the newly reported cases. We believe that this app will help the students “Stay Safe, Stay Healthy.”

## 1.3 Users and stakeholders

The stakeholders of this application are NTU students (undergraduate and graduate), NTU’s Fullerton Clinic Staff, NTU Staff, Jurong Hospital Staff, and Android Users. The application is consistently updated by the admin users from NTU Fullerton Clinic staff, NTU Staff or Jurong Hospital Staff whenever they add a case to the application’s database. The cases are showcased as hotspots on NTU’s map, that is displayed using Google Maps API. Real-time statistics of Singapore are displayed using Government API sourced from NEA’s (National Environmental Agency) website. Android mobile users, who are staying in any of the halls in NTU Campus can utilise the application to find where dengue cases are clustered and are alerted whenever the cases reach the red level, which is more than 15 active cases. The application can be further developed to include other people (i.e. admin, other staff members and teaching staff) living on NTU campus when the locations are further developed to include staff residencies. This application is designed, developed and programmed by group Ctrl_AH_Defeat (Group 5), of the module CZ2006 Software engineering.

## 1.4 Assumptions

Users should have internet access on their cellphone in order to use this application.

Admin Users must belong to NTU Staff, Fullerton Hospital or Jurong Hospital and must have a shared admin email and password. Using these, they can log in to our app for the reporting of dengue cases.

## 1.5 Constraints

Our Dengue Safe App supports Android OS with Android Version Lollipop (Version 5.1) and above.

Our app is currently only available in English.

## 1.6 UI Mockups

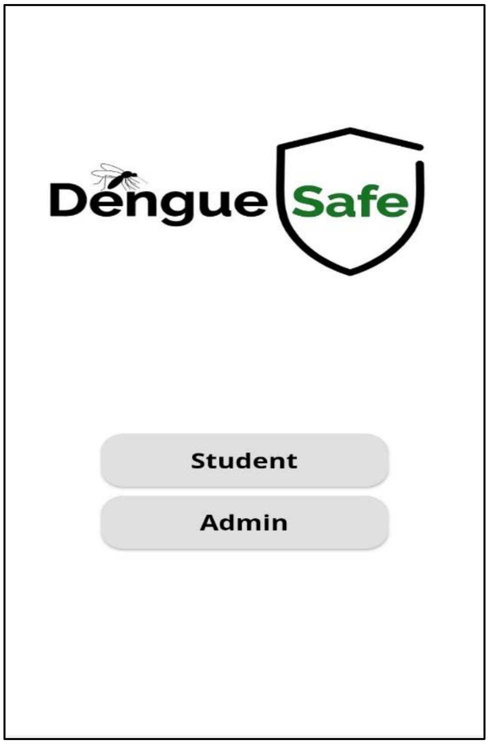


*Figure 1: Admin or Student View*

Figure 1 shows the screens used by the users of the app. The user will be directed to 2 different menus upon pressing on the 'Admin' or 'Student' view.

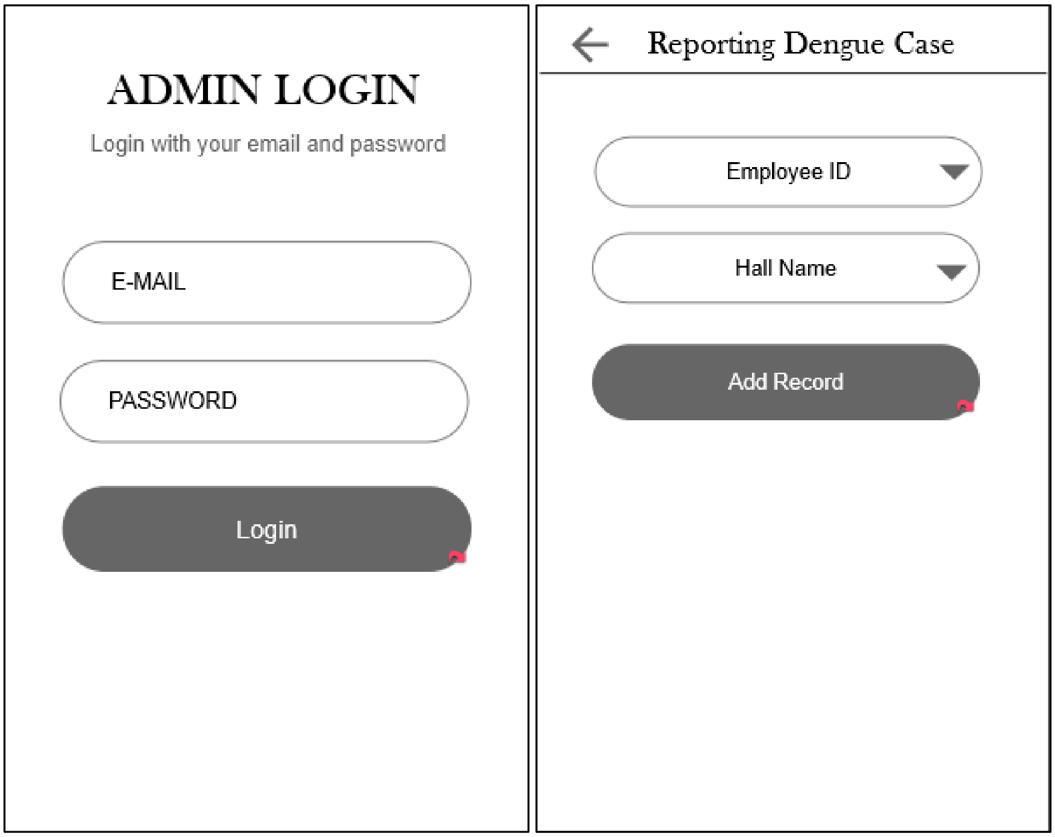
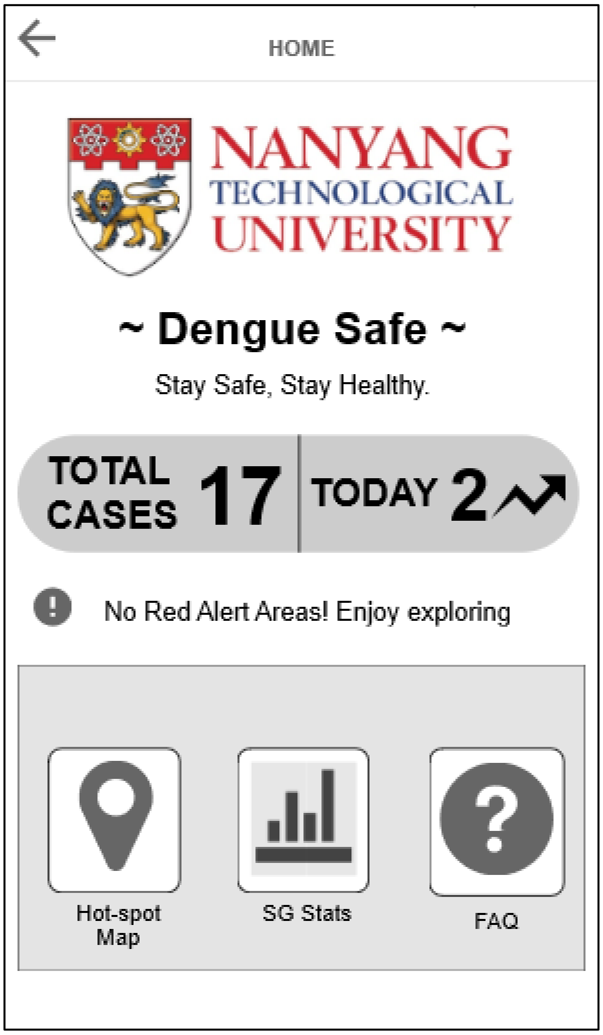


*Figure 2: Admin Login Screen and Report Case Screen*

Figure 2 shows the screens used by the Admin users of the app. They are the Admin Login Screen and the Report Case Screen. The user interface is kept simple and minimalistic. The login screen is similar to login screens on other apps to allow the user to input email and password and login with ease. The Report Case screen is made in an intuitive form fill-in style having 2 required fields Employee ID and Hall Name.



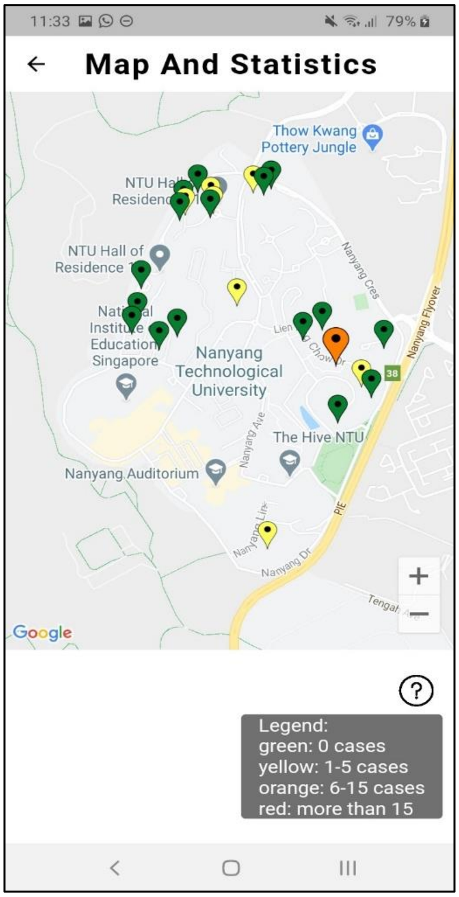
~ Dengue Safe ~

Stay Safe, Stay Healthy.

No Red Alert Areas! Enjoy exploring

*Figure 3: Home Screen for Student Users*

Figure 3 shows the interface for the Home Page. It is designed to provide important information such as the number of active cases, cases today, and Red Alert Areas, to the user at a glance. It also has 3 distinct buttons to allow users to navigate through the available functionality easily.



Legend:
green: 0 cases
yellow: 1-5 cases
orange: 6-15 cases
red: more than 15

*Figure 4: Hotspot Map Screen*

Figure 4 shows the interface for the Hotspot Map Screen. Here a Google Map of NTU will be shown. It will focus on the Halls of Residence and will show markers in 4 distinct colors- green, yellow, orange or red. Halls with 0 cases have green markers, halls with 1 to 5 cases have yellow markers, halls with 6 to 15 cases have orange markers and halls with 16 cases or above have red markers. The screen should be primarily occupied by the map to allow the user to interact with it and know about the number of cases in each Hall of Residence. Informative feedback about what the colors of the markers represent will be provided in a legend.

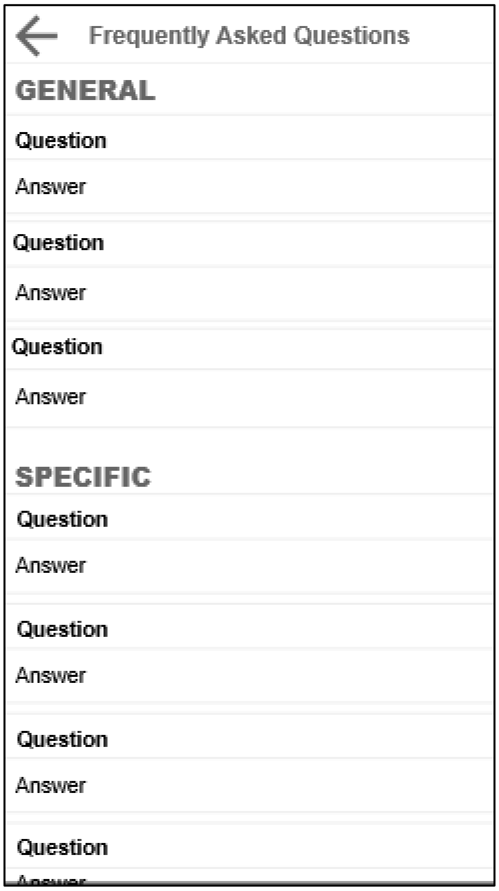


*Figure 5: Frequently Asked Questions Screen*

Figure 5 shows the Frequently Asked Questions Page. All the questions and their answers must be displayed in a simple and consistent card form. The questions must be divided based on their category, for example a general category may include general questions about the disease and prevention and a specific category may include specific questions relating to the app and its functionality.

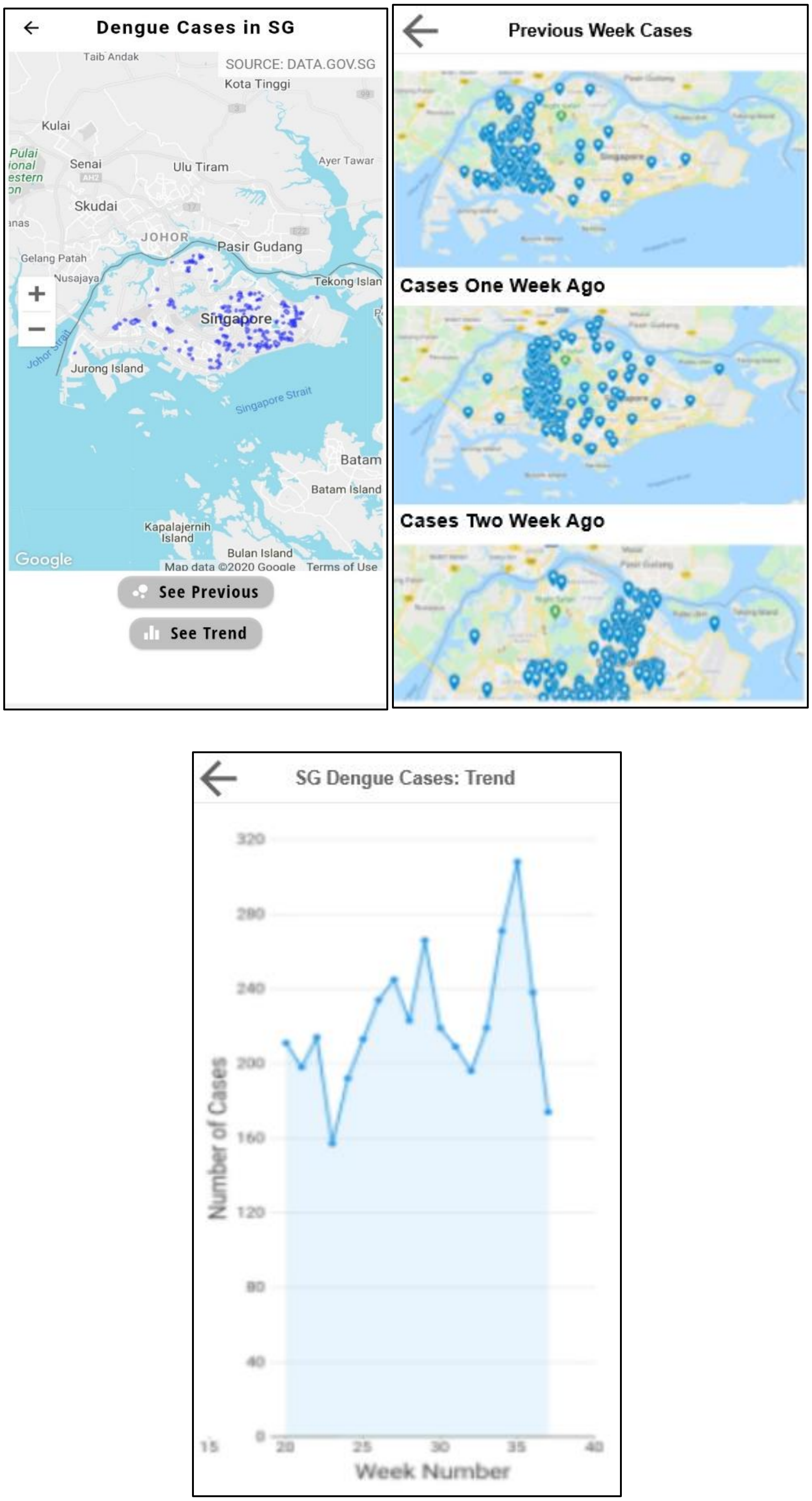


*Figure 6: Statistics Screens*

Figure 6 shows the screens that display statistics based on Government data of dengue cases in Singapore. They show the map of Singapore with the dengue clusters, based on the information retrieved from data.gov.sg. Upon pressing the ‘see previous’, the user will be directed to a page that will show the map of previous weeks. Next, upon pressing the ‘see trend’, the user will see a line chart of week number vs number of dengue cases since a visual graph represents information with more clarity. The interfaces will be kept clean and minimalistic, so that the user can focus on the information and not the design. In this way, maximum information can be conveyed to the user.

# 2. Functional Requirements

1. The System must start and must allow the beginning of user interaction.

1.1. The system must allow the Admin user to login to their registered account.

1.1.1. The system must require the Shared Admin Email for login.

1.1.1.1. The system must display an error message if the Shared Admin Email field is empty.

1.1.2. The system must require the password for login.

1.1.2.1. The system must display an error message if the password field is empty.

1.1.3. The system must be able to query the database to authenticate and process the user login request.

1.1.4. If the information of these 2 fields are incorrect, the system must display an error message.

1.2. The system must allow Admin users to report dengue cases that happened in a Hall of Residence.

1.2.1. The system must require the Employee ID for reporting a dengue case.

1.2.1.1. The system must display an error message if the Employee ID field is empty.

1.2.2. The system must require the Hall of Residence where the dengue case has occurred for reporting a dengue case.

1.2.2.1. The system must display an error message if the Hall of Residence field is empty.

1.2.3. The system must store the dengue case information and the timestamp when the case was reported into the database.

1.3. The system must display the Home Page for student users.

1.3.1 The Home Page must display the following information

1.3.1.1. The Home Page must display the app name and the app tagline.

1.3.1.2. The Home Page must display the total number of active dengue cases.

1.3.1.3. The Home Page must display the total number of dengues cases reported on that day.

1.3.2. The Home Page must allow the user to navigate to the Frequently Asked Questions Page.

1.3.2.1. The Frequently Asked Questions Page must show General Questions and their answers.

1.3.2.2. The Frequently Asked Questions Page must show Specific Questions and their answers.

1.3.3. The Home Page must allow the user to navigate to the HotSpot Map Page.

1.3.4. The Home Page must allow the user to navigate to the SG Stats Page.

1.4. The system must display a HotSpot Map of NTU.

1.4.1. The system must use Google Maps API to render a map of the campus of NTU.

1.4.2. The system must retrieve dengue cases information from the database.

1.4.3. The system must show markers on the map for every Hall of Residence based on the number of active cases in that Hall retrieved from the dengue cases information.

1.4.3.1. The system must display a green marker on a Hall of Residence if it has 0 active cases.

1.4.3.2. The system must display a yellow marker on a Hall of Residence if it has 1 to 5 active cases.

1.4.3.3. The system must display an orange marker on a Hall of Residence if it has 6 to 15 active cases.

1.4.3.4. The system must display a red marker on a Hall of Residence if it has 16 or more active cases.

1.4.4. The system must inform the users the meaning of the 4 colours of the markers.

1.4. The system must be able to show statistics and maps of dengue cases in Singapore.

1.4.1. The system must show the current dengue cluster information on a map of Singapore retrieved from the Government Data website.

1.4.2. The system must show the dengue cluster information for upto 4 previous weeks on the map of Singapore.

1.4.3.  The system must access Government Data API to retrieve information about the number of cases per week in Singapore.

1.4.4. The system must create a line chart of the number of cases per week in Singapore.

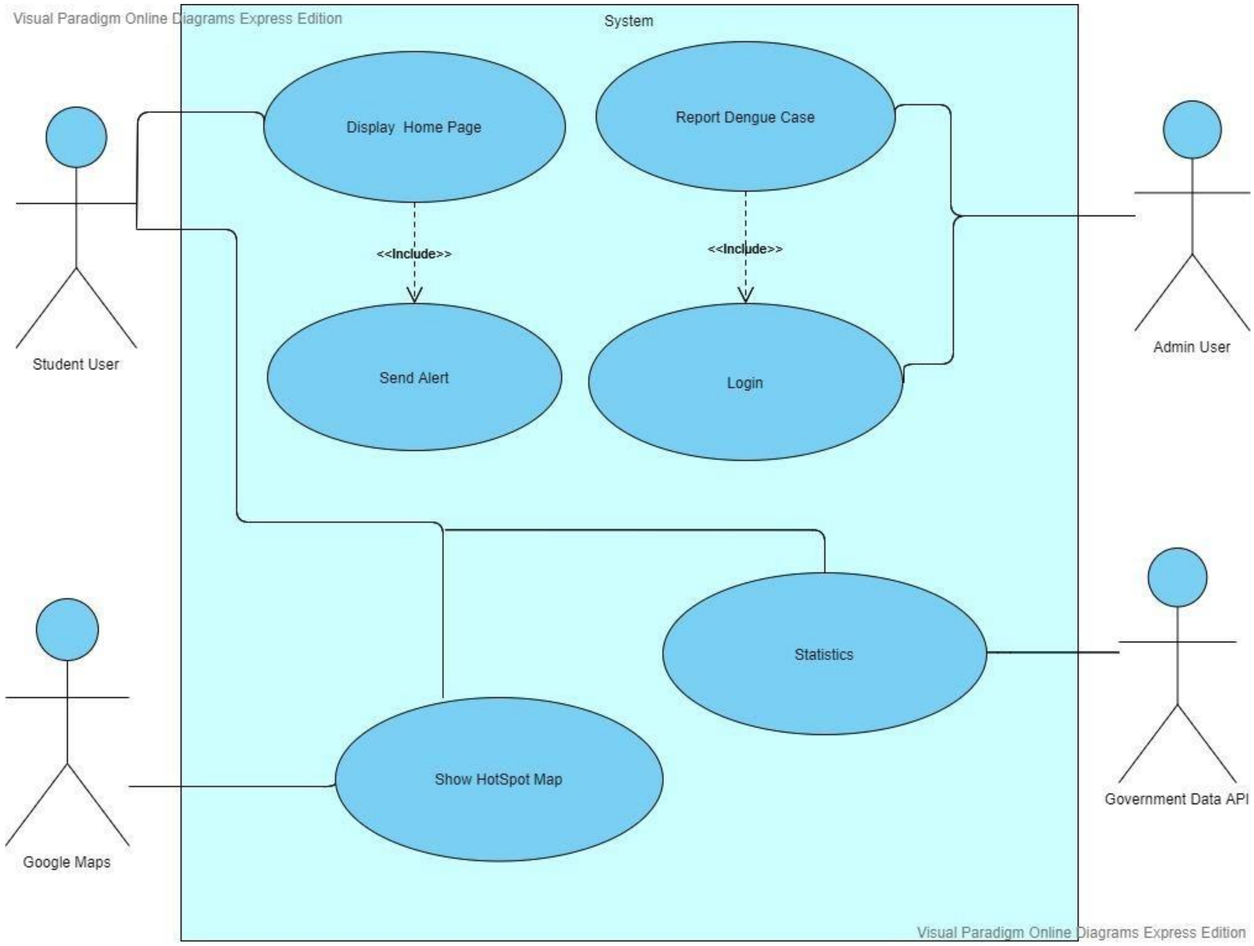
1.5 The system must be able to send alerts to student users.

1.5.1. The system must scan through the total number of active cases in each Hall of Residence.

1.5.2. The system must flag Halls of Residence as “Red Alert areas” if they have 16 or more active cases.

1.5.3. The system must display an alert message mentioning the Red Alert areas to warn users.

## 2.1 Use Case Diagram



## 2.2 Use Case Descriptions

| | | | |
|---|---|---|---|
| Use Case ID: | 1 | | |
| Use Case Name: | Admin Login | | |
| Created By: | Aratrika | Last Updated By: | Aratrika |
| Date Created: | 30/8/2020 | Date Last Updated: | 19/10/2020 |

| | |
|---|---|
| Actor: | Admin User (Initiating Actor) |
| Description: | This Use case allows an Admin User to login to the admin account using a shared email and a password. |
| Preconditions: | 1. The shared admin email address and password for the admin accounts must be stored in the database.<br>2. The Admin user must remember the shared admin email and the password required for login. |
| Postconditions: | 1. Admin User must be logged into the application.<br>2. The Admin User must be able to use the Report Dengue Case use case. |

| | |
|---|---|
| Priority: | After registering an account, the Admin User must login to enable the Report Dengue Case use case. |
| Frequency of Use: | Depends on the number of on-campus dengue cases and the Admin users. |
| Flow of Events: | 1. The System shows the login form for Admin User.<br>2. The Admin User must enter the shared admin email.<br>3. The Admin User must enter the password.<br>4. The Admin User clicks on the Login button.<br>5. If both the email and password have been entered, the system must query the database to authenticate the shared admin email and the password entered by the user.<br>6. If authentication succeeds, the System must log the user in. |
| Alternative Flows: | 1.AC.5.1. If the user has not entered the shared admin email, System displays an error message “Enter the email” and returns to step 1.<br>1.AC.5.1. If the user has not entered the password, System displays an error message “Enter the password” and returns to step 1.<br>1. AC.6.1. If the shared admin email does not exist in the database: then the system must display an error message “Please enter valid email” and return to step 1.<br>1.AC.6.2. If the password does not match for the shared admin email: the System must show an error message |

| | |
|---|---|
| | “Please enter valid email. Password doesn’t match” and return to step 1. |
| Exceptions: | |
| Includes: | |
| Special Requirements: | 1. Usability: Error messages must be written in English, in a clear, concise manner.<br>2. Performance: Once the user has entered the credentials, the system must give feedback by logging in or showing an error message within 10 seconds. |
| Assumptions: | |
| Notes and Issues: | |

| | | | |
|---|---|---|---|
| Use Case ID: | 1.1 | | |
| Use Case Name: | Report Dengue Case | | |
| Created By: | Nikita | Last Updated By: | Aratrika |
| Date Created: | 31/8/2020 | Date Last Updated: | 19/10/2020 |

| | |
|---|---|
| Actor: | Admin User (Initiating Actor) |
| Description: | Admin users can use this use case to report on-campus dengue cases that occurred in a Hall of Residence. |
| Preconditions: | 1. The Admin User must be logged into the System. |
| Postconditions: | 1. The Dengue case must be reported in the app.<br>2. The Dengue case must be stored in the database. |
| Priority: | This use case has among the highest priority in our app. It helps on-campus Dengue cases to be reported so that other students can be notified. |
| Frequency of Use: | depends on the number of dengue cases happening to on-campus Students. |

| | |
|---|---|
| Flow of Events: | 1. System shows the form for reporting a dengue case.<br>2. Admin User enters the Employee ID.<br>3. Admin User chooses the Hall of Residence from a drop down menu.<br>4. Admin User clicks "Add Record" to submit the form.<br>5. If the user has entered the Employee ID and the Hall of Residence, System updates the database with the timestamp and Hall of Residence of the newly reported Dengue Case.<br>6. System shows a dialog box asking the User if he wants to log out or report another case.<br>7. If the User clicks on logout, then he is logged out of the admin account. |
| Alternative Flows: | 1.1.AC.5.1 If the user has not filled in Employee ID, the app will display a message "Employee ID is required" and return to step 1.<br>1.1.AC.5.2 If the user has not chosen a Hall of Residence from the dropdown menu,  the app will display a message "Hall of Residence is required" and return to step 1.<br>1.1.AC.7.1. If the user clicks on "Continue Adding Record", go back to step 1. |
| Exceptions: | |
| Includes: | Includes the Login use case. |

| | |
|---|---|
| Special Requirements: | 1. Usability: Error message must be shown in English, in a clear and concise manner.<br>2. Usability: The user receives useful feedback from the System such as “Case Verified” and “Case Not Verified”. |
| Assumptions: | 1. Admin Users are Fullerton Hospital Staff or Nanyang Technological University Staff hence they will be responsible and will not report false cases. |
| Notes and Issues: | |

| | | | |
|---|---|---|---|
| Use Case ID: | 2 | | |
| Use Case Name: | Display Homepage | | |
| Created By: | Nikita | Last Updated By: | Nikita |
| Date Created: | 31/8/2020 | Date Last Updated: | 7/11/2020 |

| | |
|---|---|
| Actor: | Student User (Initiating Actor) |
| Description: | Users see the main page for the app upon opening it. The total number of dengue cases and the cases reported on that day are shown. They will be able to direct themselves to different pages such as the Hotspots Map page, Statistics page or Frequently Asked Questions Page. |
| Preconditions: | 1. Users must choose the Student option upon opening the app. |
| Postconditions: | 1. User should leave the homepage upon exiting the app, return to login page upon clicking on the back button or successfully select and go to another page among:<br>1. NTU Hotspot Map<br>2. See SG Dengue Stats |

| | |
|---|---|
| | 3. Frequently Asked Questions |
| Priority: | |
| Frequency of Use: | Every time a Student User uses the app. |
| Flow of Events: | 1. System must load the Homepage with the 3 buttons for navigating to the following pages:<br>a. NTU Hotspot Map<br>b. See SG Dengue Stats<br>c. Frequently Asked Questions.<br>2. User presses the floating action button to refresh the page.<br>3. System must retrieve dengue case data from the database.<br>4. System must display active dengue cases on Campus.<br>5. System must display the total number of new dengue cases on that day.<br>6. System must execute the Send Alert use case. |
| Alternative Flows: | 2.AC.1. System will return to the login page if the user presses the back button. |
| Exceptions: | |

| | |
|---|---|
| Includes: | Includes the execution of the Send Alert use case. |
| Special Requirements: | |
| Assumptions: | 1. To get new data, the user will refresh the page by pressing the floating action button. |
| Notes and Issues: | |

| | | | |
|---|---|---|---|
| Use Case ID: | 2.1 | | |
| Use Case Name: | Send Alert | | |
| Created By: | Nikita | Last Updated By: | Aratrika |
| Date Created: | 31/8/2020 | Date Last Updated: | 20/10/2020 |

| | |
|---|---|
| Actor: | |
| Description: | If any Hall of Residence has 16 or more dengue cases, it will be called a Red Alert Area and an alert message will be shown on the homepage. Otherwise, a normal message will be shown on the homepage. |
| Preconditions: | |
| Postconditions: | 1. An alert message or a normal message must be displayed on the homepage. |
| Priority: | This use case has high priority because it informs students of Halls of Residence which are dangerous because they have 16 or more dengue cases. |

| | |
|---|---|
| Frequency of Use: | Depends on the number of dengue cases in each Hall of Residence. |
| Flow of Events: | 1. System retrieves data about on-campus dengue cases from the database.<br>2. System calculates the total number of active dengue cases in each Hall of Residence which were reported in the last 8 days.<br>3. System checks if any Hall of Residence has equal to or more than 16 dengue cases and adds it to a list that stores all Red Alert Areas.<br>4. If there is only one Red Alert Area, System shows an alert message such as “Hall 16 is a Red Alert Area! Be Aware.” |
| Alternative Flows: | 2.1.AC.3.1. If no Hall of Residence has equal to or more than 16 dengue cases, System shows a normal message “No Red Alert Areas! Enjoy exploring.”<br>2.1.AC.4.1. If there are multiple Red Alert Areas, System shows an alert message such as “Hall 16, Hall 3 are Red Alert Areas! Be Aware.” |
| Exceptions: | |
| Includes: | |
| Special Requirements: | 1. Usability: Alert message is written in English, is friendly and concise to capture attention. |

| | |
|---|---|
| Assumptions: | |
| Notes and Issues: | |

| | | | |
|---|---|---|---|
| Use Case ID: | 3 | | |
| Use Case Name: | Show Hotspot Map | | |
| Created By: | Nikita | Last Updated By: | Nikita |
| Date Created: | 31/8/2020 | Date Last Updated: | 7/11/2020 |

| | |
|---|---|
| Actor: | Google Maps (Regular Actor) |
| Description: | This use case shows the map of Nanyang Technological University with green, yellow, red and orange markers placed on every Hall of Residence location to visually denote the number of dengue cases. It also shows a legend where the colours of the markers and the number of dengue cases mapping is shown. |
| Preconditions: | 1. At least one recorded case is present in the database. |
| Postconditions: | 1. System updates the colour of the markers on the hotspot map every time a case is added to the database. |
| Priority: | This use case has the second highest priority in the application. |

| | |
|---|---|
| Frequency of Use: | Depends on Student User, but usually each time the app is opened. |
| Flow of Events: | 1. System retrieves data about on-campus dengue cases from the database.<br>2. System calculates the number of dengue cases that were reported in the last 8 days in each Hall of Residence.<br>3. System sets marker colours according to the following rules:<br>a. Total number of cases: 0 -  Green color<br>b. Total number of cases: 1 to 5 - Yellow color<br>c. Total number of cases: 6 to 15 - Orange color<br>d. Total number of cases: 16 and more - Red color<br>4. System sets up a connection with Google Maps through the Google Maps API.<br>5. System displays the Map of Nanyang Technological University with markers for each Hall of Residence.<br>6. System creates a legend explaining the mapping from the number of dengue cases in a Hall of Residence to the color of the marker for that Hall of Residence. |
| Alternative Flows: | |
| Exceptions: | |

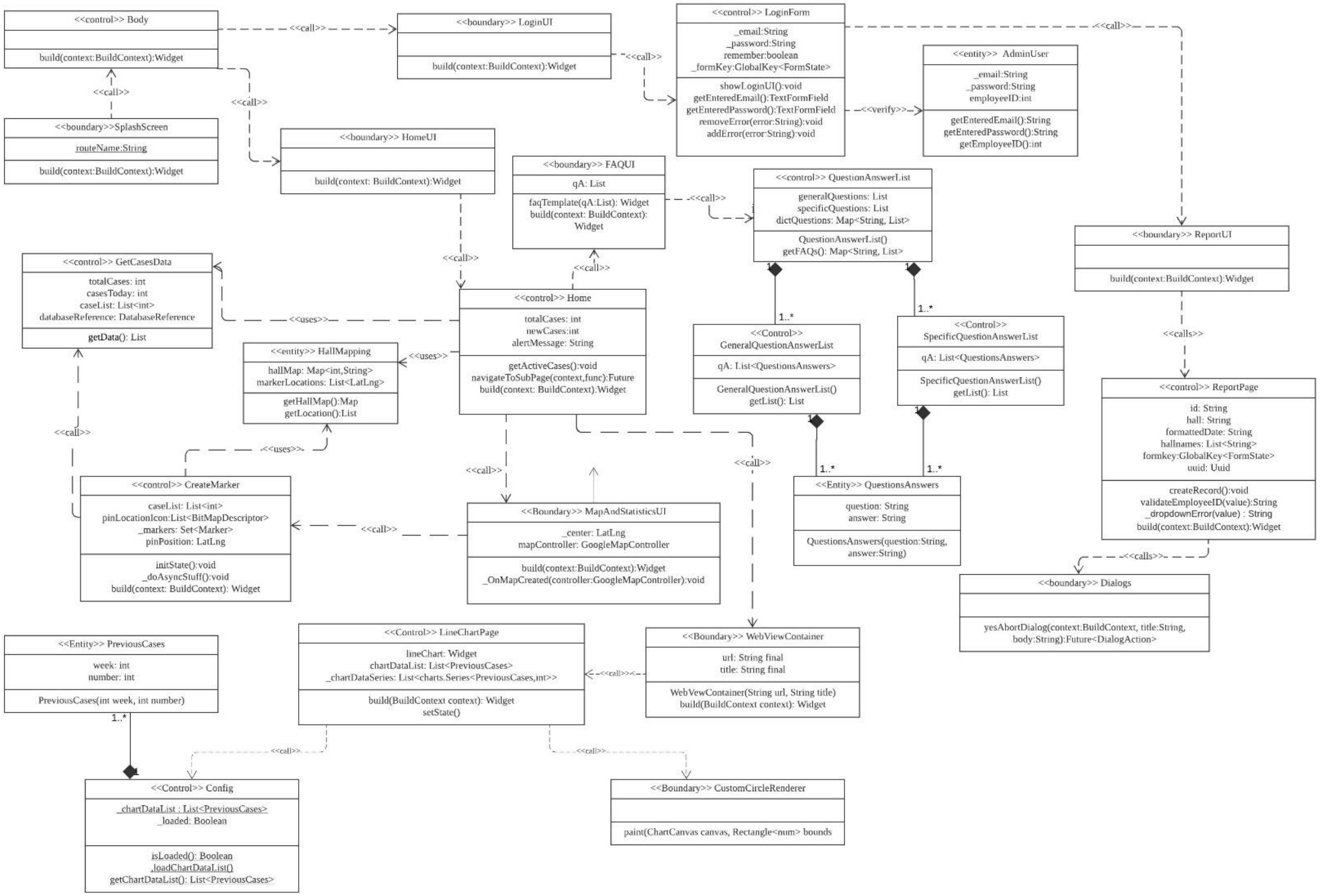
| | |
|---|---|
| Includes: | |
| Special Requirements: | 1. Usability: Hotspots Maps uses 4 different colored markers to help users differentiate between which areas on Campus are safer from the risk of Dengue.<br>2. Usability: Concise language used in the legend for user clarity.<br>3. Performance: The google map of Nanyang Technological University loads within 10 seconds.<br>4. System must have an API key to access Google Maps API. |
| Assumptions: | |
| Notes and Issues: | |

| | | | |
|---|---|---|---|
| Use Case ID: | 4 | | |
| Use Case Name: | Statistics | | |
| Created By: | Nikita | Last Updated By: | Nikita |
| Date Created: | 31/8/2020 | Date Last Updated: | 7/11/2020 |

| | |
|---|---|
| Actor: | Government Data API (Regular Actor) |
| Description: | Data about dengue cases in Singapore is retrieved from the Government Data API and displayed in maps and line charts. |
| Preconditions: | There must be data on dengue cases available on the Government Data website and Government Data API. |
| Postconditions: | 1. System automatically updates the locations of the clusters and the number of clusters based on the addition of data on the Government Data website and Government Data API. |
| Priority: | This use case is of the third highest priority in the application as it informs users of the trends and locations of clusters of dengue cases in Singapore. |
| Frequency of Use: | Depends on the Student user, but usually each time the app is opened |

| | |
|---|---|
| Flow of Events: | 1. The user must click on “SG Stats” on the Home Page.<br>2. The system must retrieve and display dengue cluster maps of Singapore from an http link on the Government data website.<br>3. The user must click on the “See Previous” button.<br>4. The system must show dengue cluster maps of Singapore upto 4 weeks ago.<br>5. The user must click on the “See Trends” button.<br>6. The system must access the Government Data API to retrieve the weekly number of dengue cases data.<br>7. The system must represent the data in a line chart for visualisation. |
| Alternative Flows: | |
| Exceptions: | |
| Includes: | |
| Special Requirements: | 1. System needs access to the URL holding the Government Data API. |
| Assumptions: | |
| Notes and Issues: | |

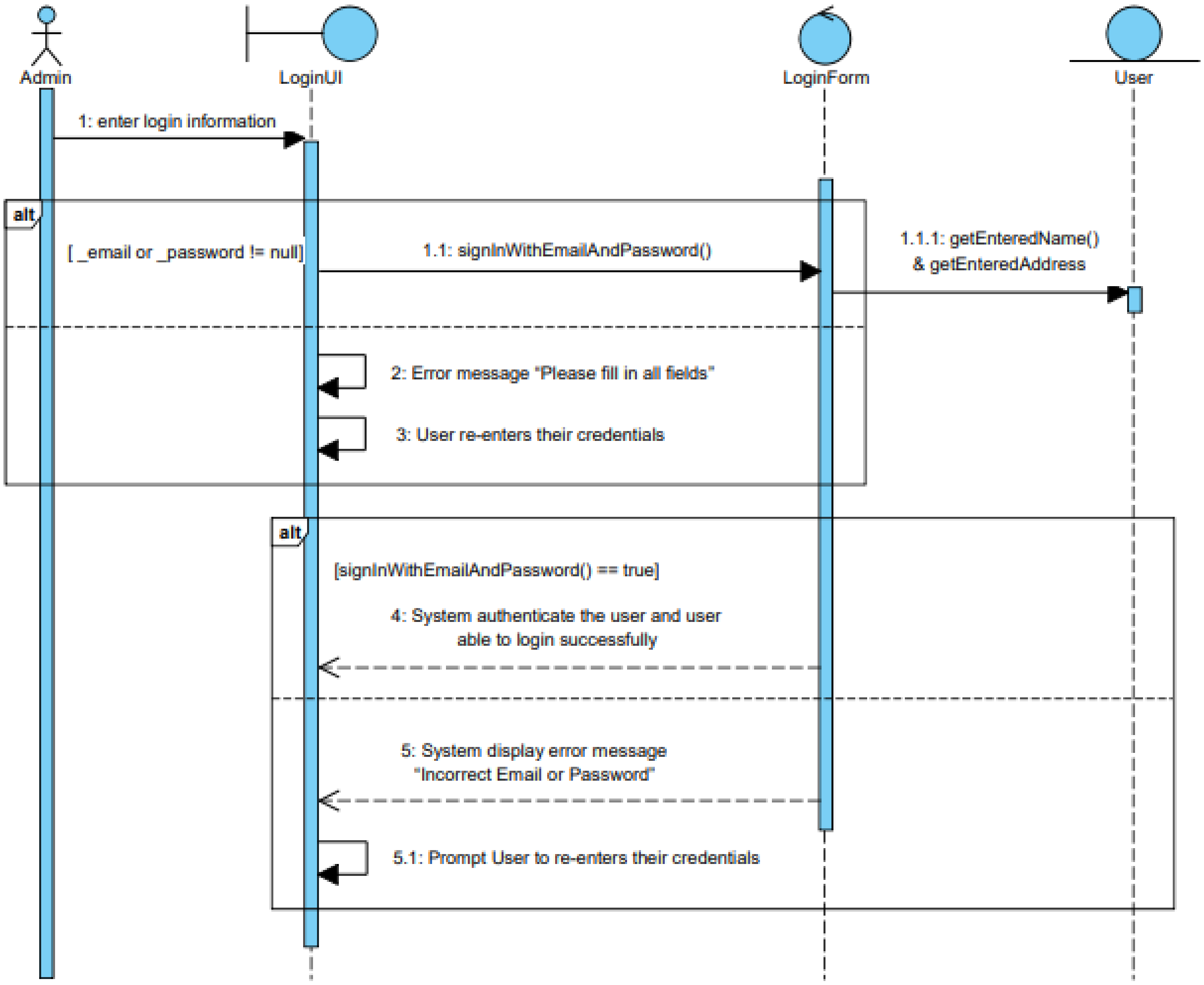
## 2.3 Class Diagram



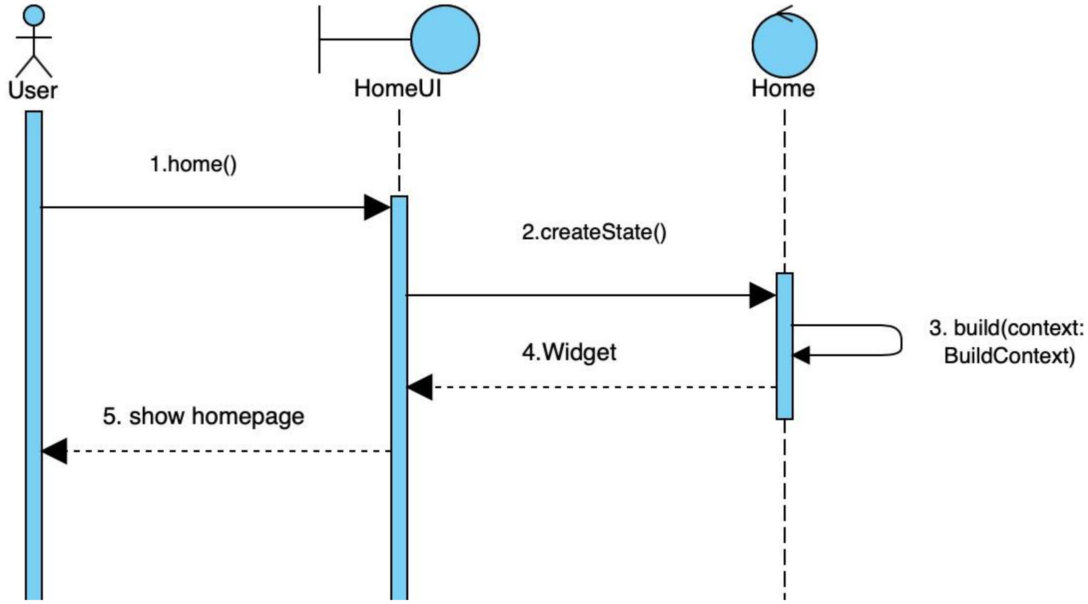
<<control>> Body
build(context:BuildContext):Widget
<<call>>
<<boundary>> LoginUI
build(context:BuildContext):Widget
<<call>>
<<control>> LoginForm
_email:String
_password:String
remember:boolean
_formKey:GlobalKey<FormState>
showLoginUI():void
getEnteredEmail():TextFormField
getEnteredPassword():TextFormField
removeError(error:String):void
addError(error:String):void
<<verify>>
<<entity>> AdminUser
_email:String
_password:String
employeeID:int
getEnteredEmail():String
getEnteredPassword():String
getEmployeeID():int
<<call>>
<<call>>
<<call>>
<<boundary>>SplashScreen
routeName:String
build(context:BuildContext):Widget
<<boundary>> HomeUI
build(context: BuildContext):Widget
<<boundary>> FAQUI
qA: List
faqTemplate(qA:List): Widget
build(context: BuildContext):
Widget
<<call>>
<<control>> QuestionAnswerList
generalQuestions: List
specificQuestions: List
dictQuestions: Map<String, List>
QuestionAnswerList()
getFAQs(): Map<String, List>
<<boundary>> ReportUI
build(context:BuildContext):Widget
<<call>>
<<call>>
<<control>> GetCasesData
totalCases: int
casesToday: int
caseList: List<int>
databaseReference: DatabaseReference
getData(): List
<<uses>>
<<control>> Home
totalCases: int
newCases:int
alertMessage: String
getActiveCases():void
navigateToSubPage(context,func):Future
build(context: BuildContext):Widget
1
1..*
1
1..*
<<Control>>
GeneralQuestionAnswerList
qA: List<QuestionsAnswers>
GeneralQuestionAnswerList()
getList(): List
<<Control>>
SpecificQuestionAnswerList
qA: List<QuestionsAnswers>
SpecificQuestionAnswerList()
getList(): List
<<calls>>
<<entity>> HallMapping
hallMap: Map<int,String>
markerLocations: List<LatLng>
getHallMap():Map
getLocation():List
<<uses>>
<<control>> ReportPage
id: String
hall: String
formattedDate: String
hallnames: List<String>
formkey:GlobalKey<FormState>
uuid: Uuid
createRecord():void
validateEmployeeID(value):String
_dropdownError(value) : String
build(context:BuildContext):Widget
<<call>>
<<uses>>
<<call>>
<<call>>
1
1..*
1
1..*
<<control>> CreateMarker
caseList: List<int>
pinLocationIcon:List<BitMapDescriptor>
_markers: Set<Marker>
pinPosition: LatLng
initState():void
_doAsyncStuff():void
build(context: BuildContext): Widget
<<call>>
<<Boundary>> MapAndStatisticsUI
_center: LatLng
mapController: GoogleMapController
build(context:BuildContext):Widget
_OnMapCreated(controller:GoogleMapController):void
<<Entity>> QuestionsAnswers
question: String
answer: String
QuestionsAnswers(question:String,
answer:String)
<<calls>>
<<boundary>> Dialogs
yesAbortDialog(context:BuildContext, title:String,
body:String):Future<DialogAction>
<<Entity>> PreviousCases
week: int
number: int
PreviousCases(int week, int number)
1..*
<<Control>> LineChartPage
lineChart: Widget
chartDataList: List<PreviousCases>
_chartDataSeries: List<charts.Series<PreviousCases,int>>
build(BuildContext context): Widget
setState()
<<call>>
<<Boundary>> WebViewContainer
url: String final
title: String final
WebVewContainer(String url, String title)
build(BuildContext context): Widget
<<call>>
<<call>>
1
<<Control>> Config
_chartDataList : List<PreviousCases>
_loaded: Boolean
isLoaded(): Boolean
,loadChartDataList()
getChartDataList(): List<PreviousCases>
<<Boundary>> CustomCircleRenderer
paint(ChartCanvas canvas, Rectangle<num> bounds

## 2.4 Sequence Diagrams

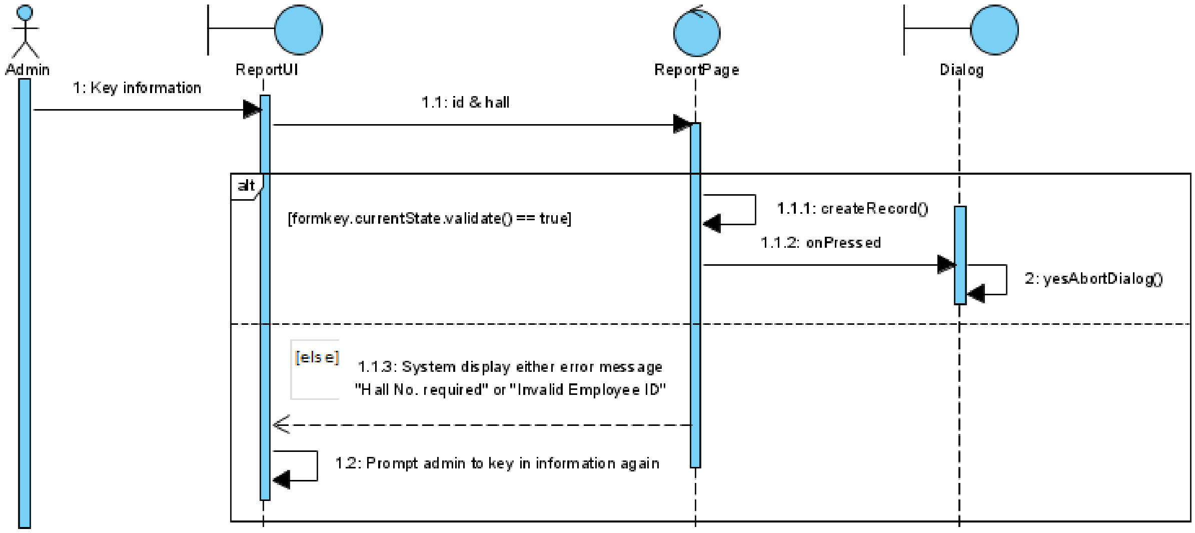
### 2.4.1 Login


Admin
LoginUI
LoginForm
User
1: enter login information
alt
[ _email or _password != null]
1.1: signInWithEmailAndPassword()
1.1.1: getEnteredName()
& getEnteredAddress
2: Error message "Please fill in all fields"
3: User re-enters their credentials
alt
[signInWithEmailAndPassword() == true]
4: System authenticate the user and user
able to login successfully
5: System display error message
"Incorrect Email or Password"
5.1: Prompt User to re-enters their credentials

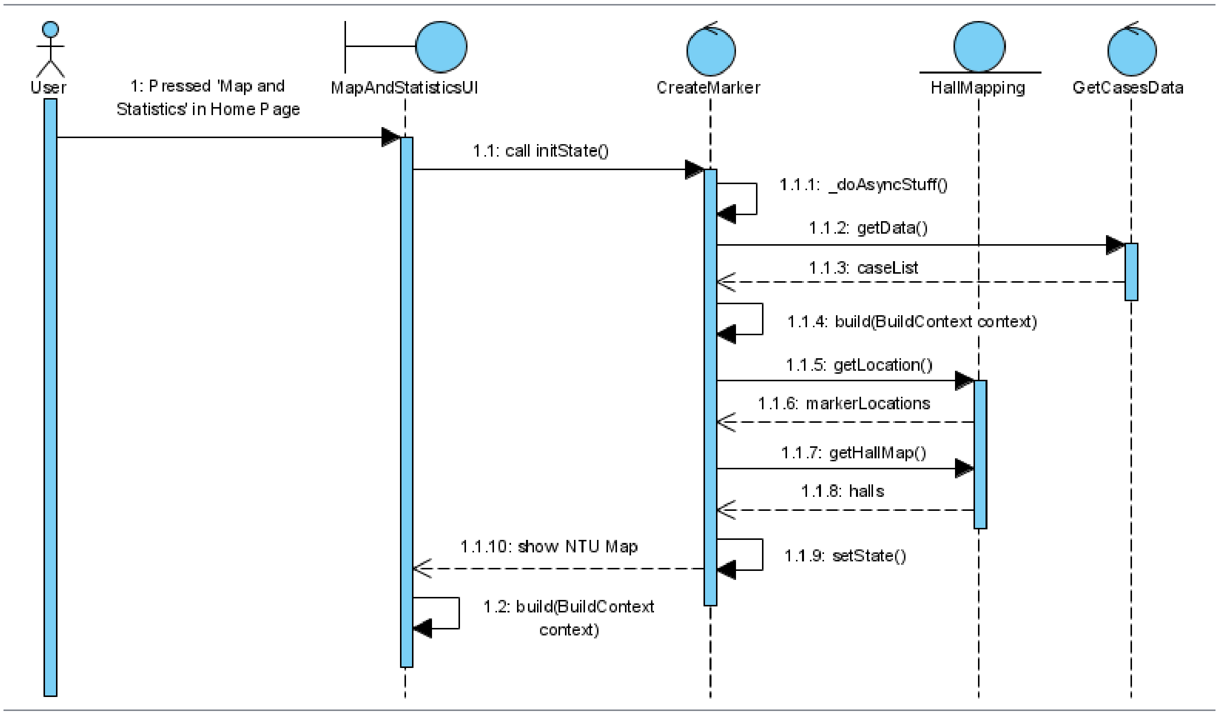
### 2.4.2 Display Homepage



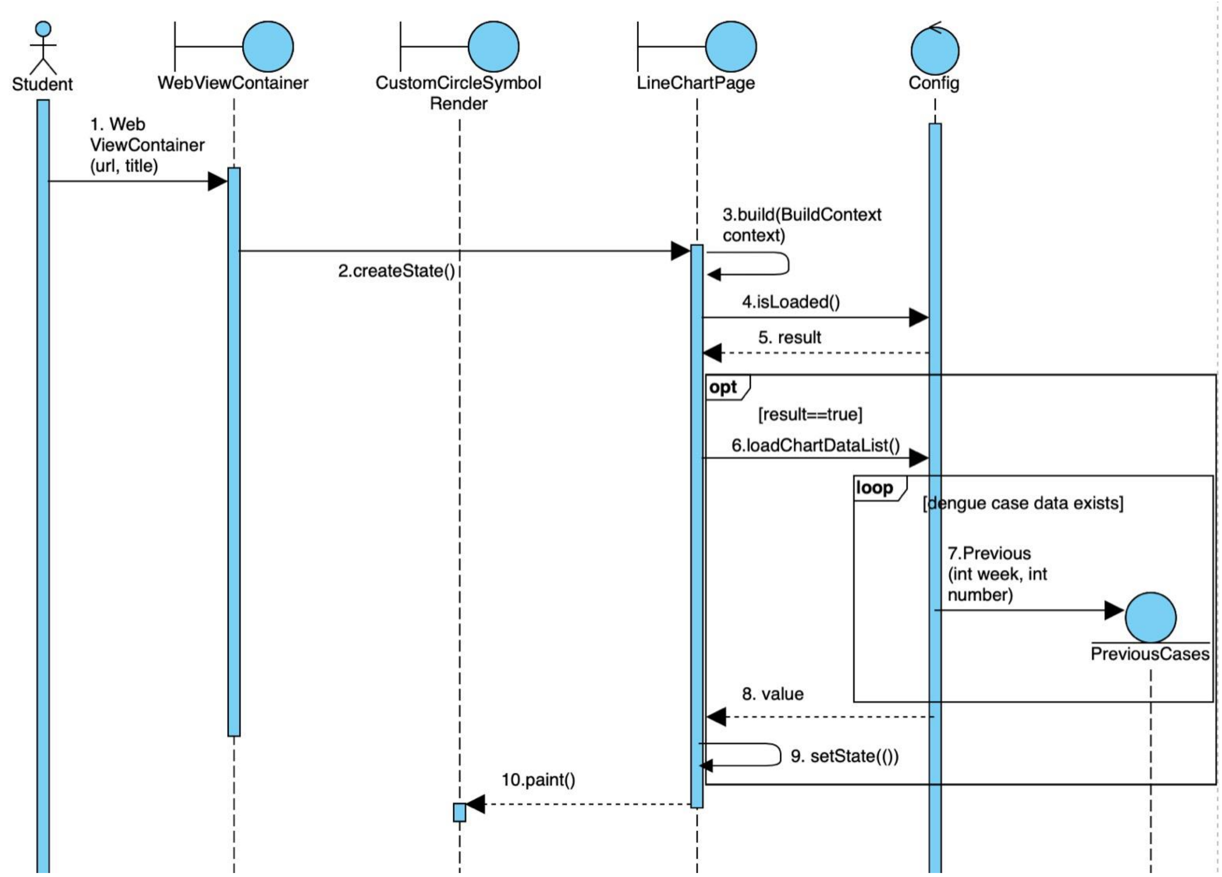
### 2.4.3 Reporting Cases


Admin
ReportUI
ReportPage
Dialog
1: Key information
1.1: id & hall
alt
[formkey.currentState.validate() == true]
1.1.1: createRecord()
1.1.2: onPressed
2: yesAbortDialog()
[else]
1.1.3: System display either error message "Hall No. required" or "Invalid Employee ID"
1.2: Prompt admin to key in information again

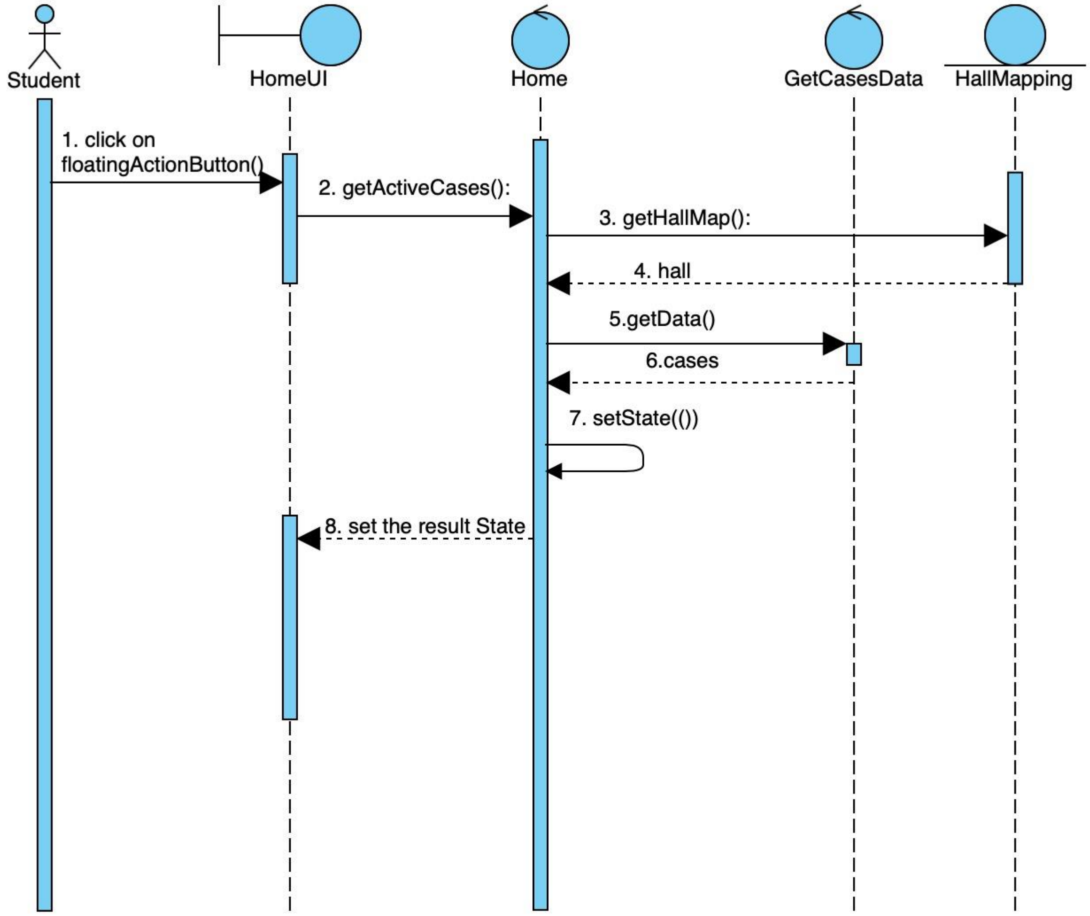
### 2.4.4 Hotspot Maps



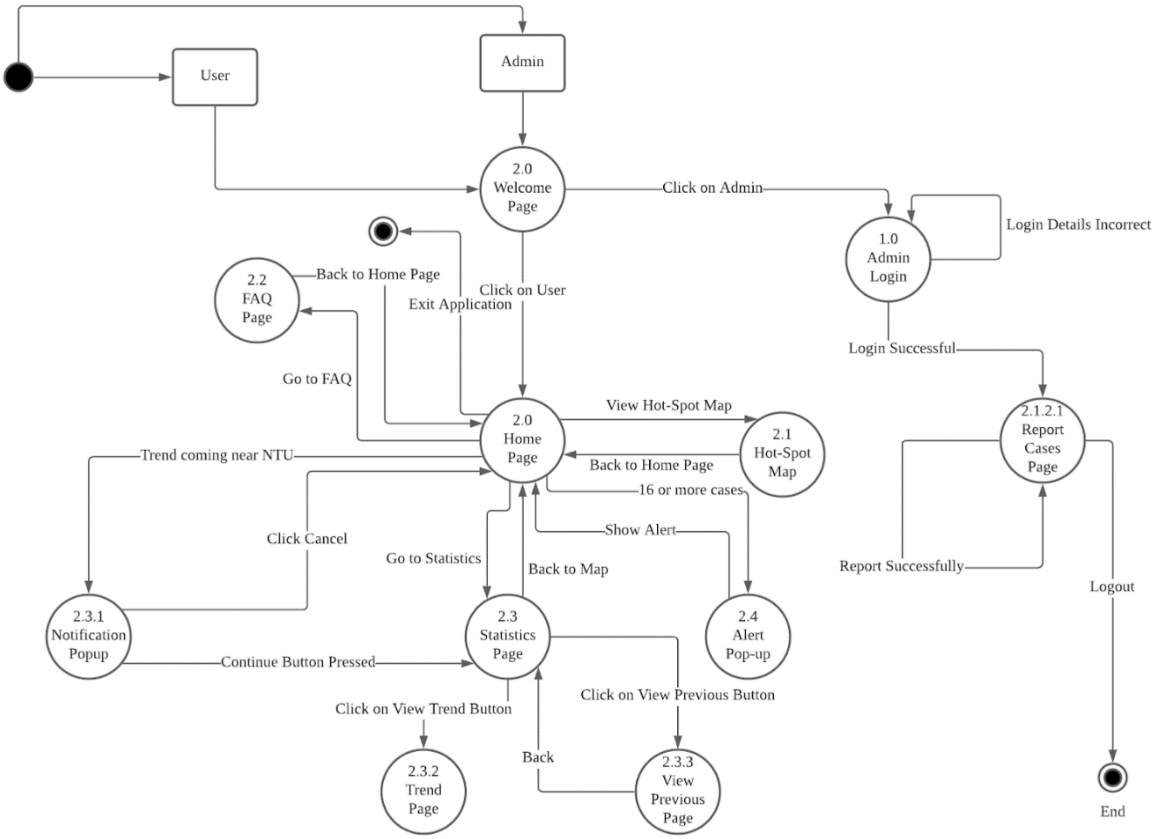
### 2.4.5 Statistics



### 2.4.6 Alert


Student
HomeUI
Home
GetCasesData
HallMapping
1. click on floatingActionButton()
2. getActiveCases():
3. getHallMap():
4. hall
5.getData()
6.cases
7. setState(())
8. set the result State

## 2.5 Dialog Map



# 3. Non-Functional Requirements

1. *Usability Requirements:*

1.1. The system must strive for consistency.

1.1.1. Consistent visuals, and colour scheme must be used throughout the application.

1.1.2. Open Sans, from Google fonts must be used throughout the application.

1.1.3. Texts must be displayed in English throughout the application.

1.2. The system must offer informative feedback.

1.2.1. The system must inform the users what the meanings of the markers are on the HotSpot Map.

1.2.2. The system must show an informative error message when a process fails.

1.3. The system must reduce short term memory load.

1.3.1. Each page must have maximum 5 elements to reduce the short term memory load on the user.

2. *Performance Requirements:*

2.1. After application restarts, the full system functionally must be restored within 3 minutes.

2.2. Application must display accurate values with minimal error.

2.3. The system must not crash during user interaction.

2.4. The application must be able to load the different pages in the app within 10 seconds.

3. *Security Requirements:*

3.1. The system allows only Admin Users from NTU Staff, Fullerton NTU or Jurong Hospital to login and report dengue cases.

3.2.The system will mask the password field in order to prevent any potential shoulder surfing.

*4. Extendibility Requirements:*

4.1.The system must be designed with layered architecture and design patterns to support any future enhancements and facilitate extendibility.

4.2.To facilitate easy data access using other platforms, the data collected will be separated from the system and stored in an online database

# 4. Interface Requirements

## 4.1 User

DengueSafe caters to all students of NTU-graduate and undergraduate, primarily those with on campus residence, who would like to be informed and aware about the dengue cases in the Halls of Residence at NTU and in Singapore.

## 4.2 Hardware

DengueSafe requires the usage of the internet hence it requires hardware devices with a supported internet connection.

## 4.3 Software

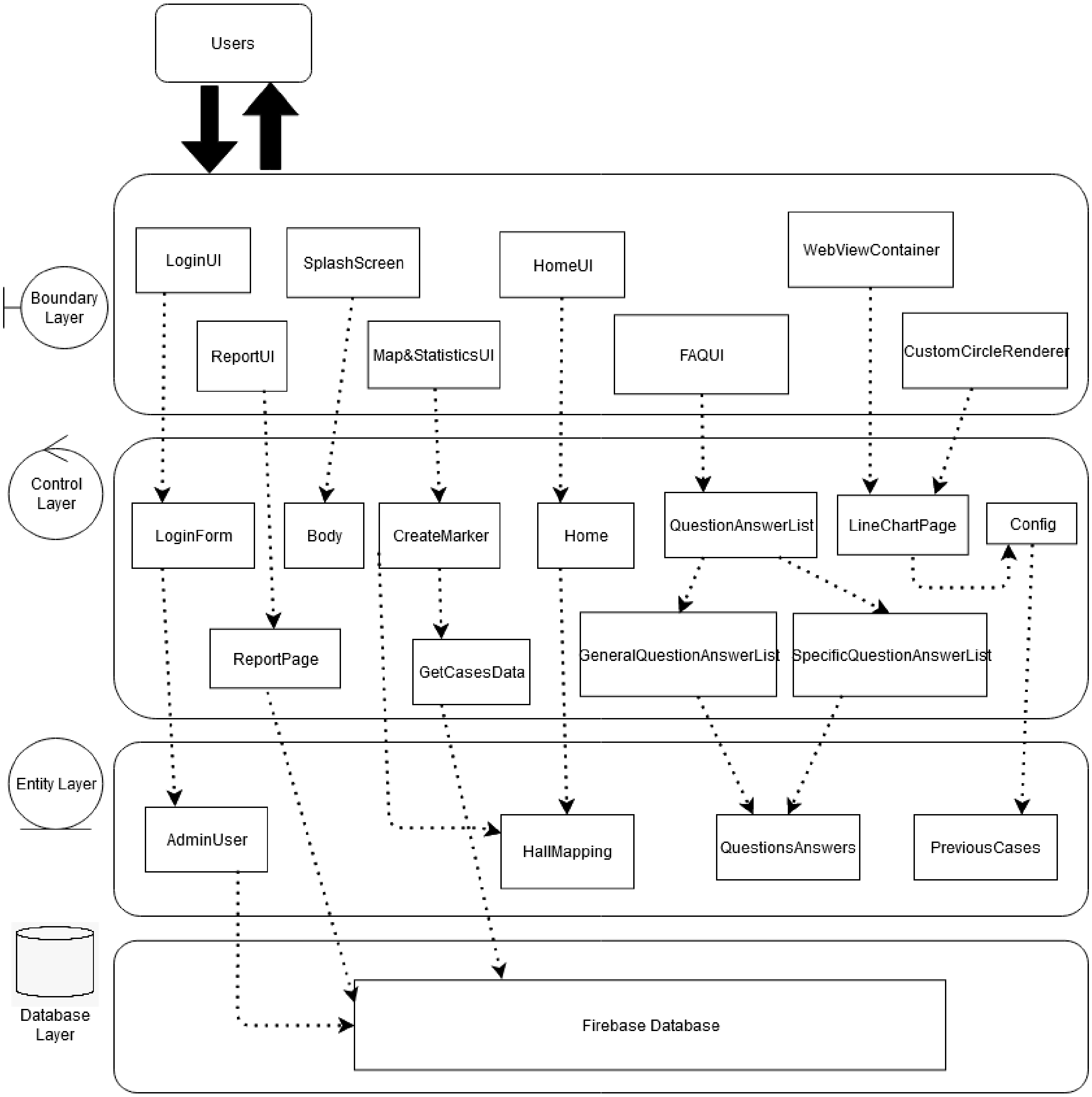
DengueSafe is being designed to work on Android Devices. Since Flutter supports cross platform development, it will later work for Android and iOS devices both.

## 4.4 Communication

DengueSafe will be accessed over the Internet. All features will be accessible through the application.

# 5. Architecture Design

## 5.1 System Architecture Diagram



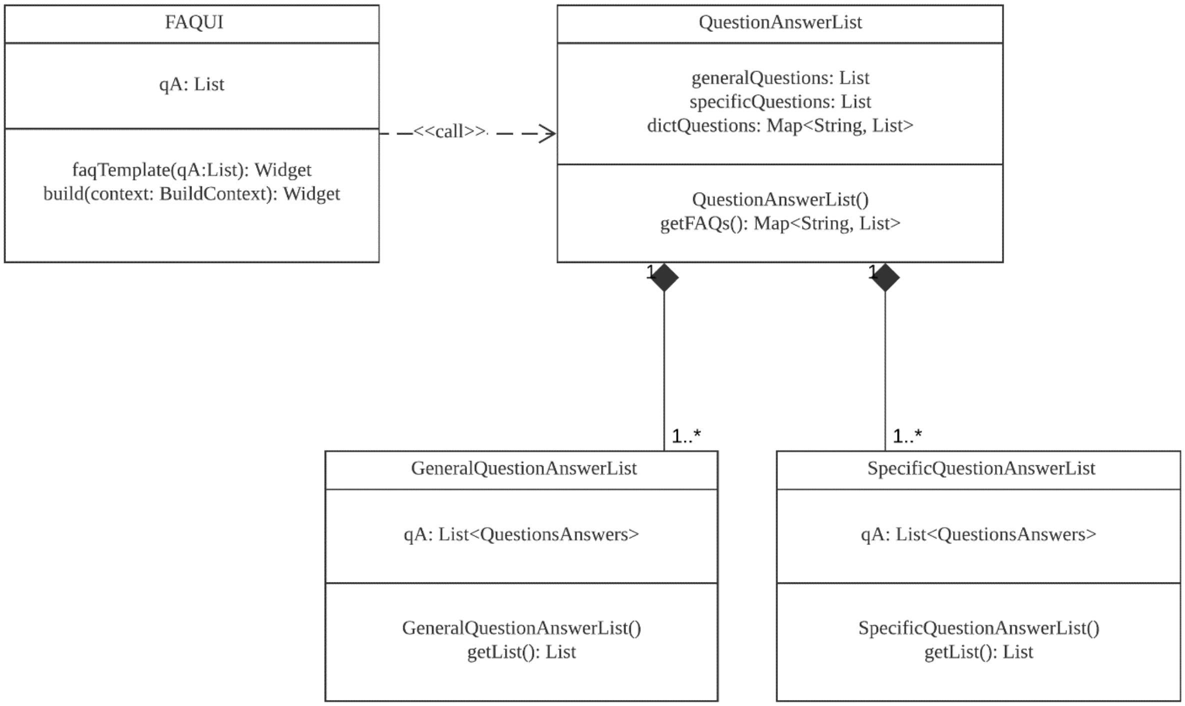
## 5.2 Design Pattern

*Problem*

In the Frequently Asked Questions, there are already 2 categories of questions: General Questions and Specific Questions. In the future, we envision more categories being added and making the client directly access each of them to display them in a separate list causes high coupling and it is not good design because it is not extensible.

*Solution*

**Facade Pattern**


FAQUI
qA: List
faqTemplate(qA:List): Widget
build(context: BuildContext): Widget
<<call>>
QuestionAnswerList
generalQuestions: List
specificQuestions: List
dictQuestions: Map<String, List>
QuestionAnswerList()
getFAQs(): Map<String, List>
1
1
1..*
1..*
GeneralQuestionAnswerList
qA: List<QuestionsAnswers>
GeneralQuestionAnswerList()
getList(): List
SpecificQuestionAnswerList
qA: List<QuestionsAnswers>
SpecificQuestionAnswerList()
getList(): List

We can apply the facade pattern here to create a higher-level interface to make the subsystem easier to use. Here the client will only be coupled with the QuestionAnswerList class which will be a composition of the different categories of questions such as General Questions and Specific Questions. This also makes the code easier to understand and decouples the client from a potentially complex system.

# 6. Data Dictionary

| Term | Definition |
|---|---|
| NTU | Nanyang Technological University |
| Student | Main users of the application who are students of Nanyang Technological University and stay on-campus residential hall. The users report on-campus dengue cases and receive alerts from the system. |
| Admin | Admin users are control users belonging to hospitals or NTU Staff who report on campus Dengue Cases. |
| GPS | Global Positioning System (GPS) is a satellite-based system that provides the location of the users. |
| Report | Reporting of the Dengue cases for each hall. |
| Google Maps | Google Maps is a web mapping service developed by Google. It provides street maps of all regions of the world. |
| API | Application Programming Interface (API) is a computing interface that defines the interaction between multiple applications. |
| Dengue | A debilitating viral disease of the tropics, transmitted by mosquitoes, and causing sudden fever and acute pains in the joints. |

| | |
|---|---|
| FAQ | Frequently Asked Questions. These are answers to questions that are often asked about the application or about dengue. |
| General Questions | Frequently asked questions generally related to dengue and its prevention. |
| Specific Questions | Frequently asked questions specific to app functionality. |
| Hall/ Hall of Residence | Buildings to provide on-campus residence to students of Nanyang Technological University. |
| Shared Admin Email | Shared Email Address of the Admin users of the application. |
| Active case | A reported dengue case is considered active if it has been reported within the last 8 days. |
| Red Alert Area | Any Hall of Residence with more than or equal to 16 active dengue cases. |
| Shoulder Surfing | Shoulder Surfing is a type of social engineering technique where another person spies on the user when they are typing their credentials to obtain their password. |

# 7. Testing

## 7.1 Black Box Testing

1. Login

    1. Generic cases

| Test Id | Scenario | Expected Result | Actual Result |
|---|---|---|---|
| 1 | Login with valid email and password | The system displays the report cases page for the admin to report new cases. | The system displays the report cases page for the admin to report new cases. |
| 2 | Login without filling up the required fields | The system prompts the user to fill up the required fields for logging in. | The system prompts the user to fill up the required fields for logging in. |
| 3 | Login with invalid email and password | The system prompts users to enter email and password again. | The system prompts users to enter email and password again. |

    b. Specific Cases (Combination)

| Email | Password | Expected Result | Actual Result |
|---|---|---|---|
| admin@ntu.edu.sg | ntuadmin123 | Successful Login | Successful Login |
| fullerton@ntu.edu.sg | fullerton123 | Successful Login | Successful Login |
| juronghospital@ntu.edu.sg | juronghosp123 | Successful Login | Successful Login |
| wrong@ntu.edu.sg | ntuadmin123 | Invalid email/password | Invalid email/password |

| | | | |
|---|---|---|---|
| Empty | ntuadmin123 | Please enter valid email | Please enter valid email |
| admin@ntu.edu.sg | 123456 | Invalid email/password | Invalid email/password |
| admin@ntu.edu.sg | Empty | Please enter password | Please enter password |
| Empty | Empty | Please enter valid email, Please enter password | Please enter valid email, Please enter password |

## 2. Equivalence class and boundary value testing for Initialisation of MapMarker

- Number of cases per Hall valid equivalence class (x=0)
  - Lower Boundary 0: -1, **0**, 1
  - Upper Boundary 0: -1, 0, 1
- Number of cases per Hall valid equivalence class (1<=x<=5)
  - Lower Boundary 1: 0, **1**, 2
  - Upper Boundary 5: 4, **5**, 6
- Number of cases per Hall valid equivalence class (6<=x<=15)
  - Lower Boundary 6: 5, **6**, 7
  - Upper Boundary 15: 14, **15**, 16
- Number of cases per Hall valid equivalence class (16<=x)
  - Lower Boundary 16: 15, **16**, 17

Hence:

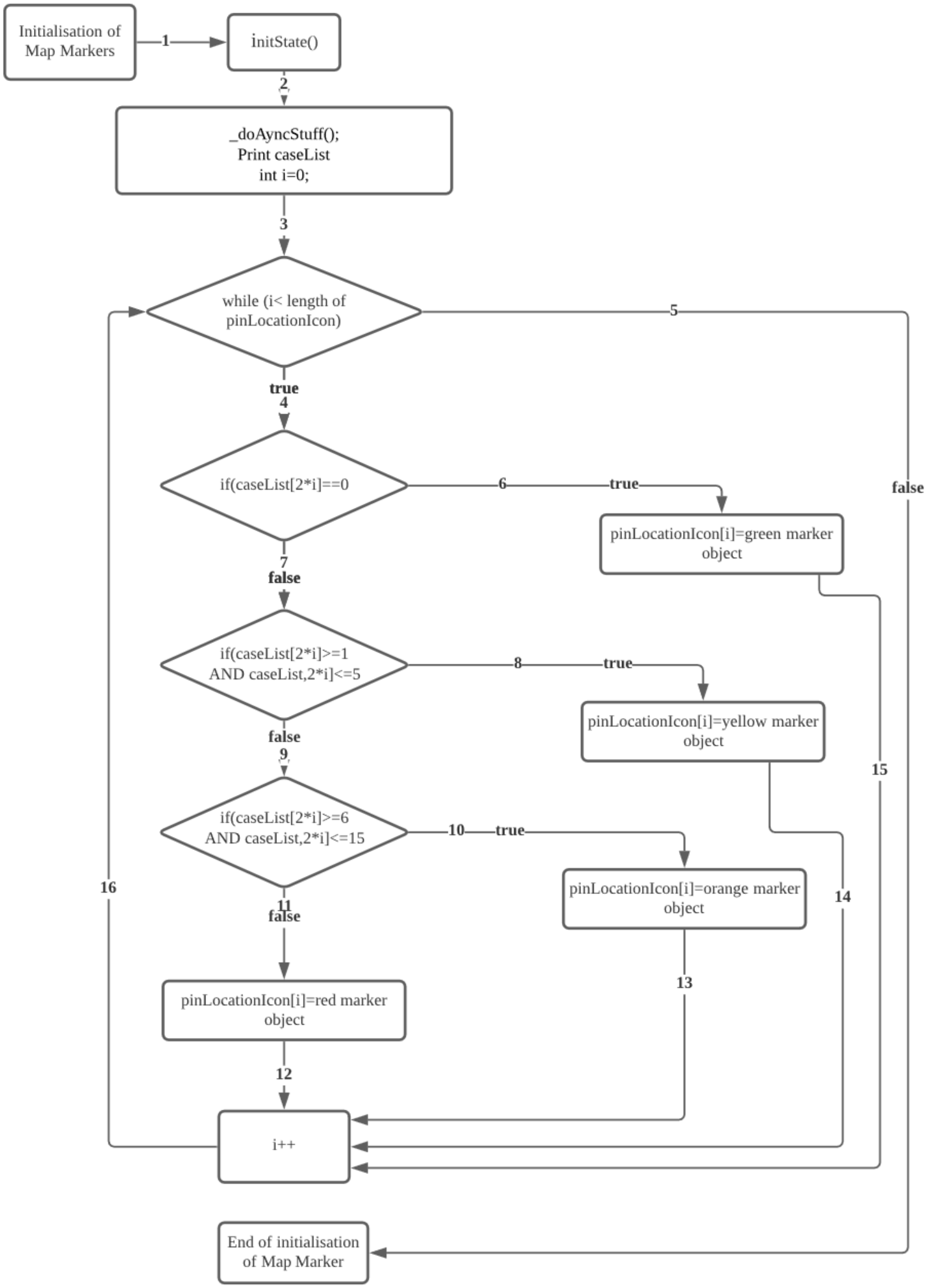
- Valid Boundary Values: {0, 1, 5, 6, 15, 16}
- Invalid Boundary Values: {-1}

| TestID | Number of Cases | Expected Result | Actual Result |
|---|---|---|---|
| 1. | 0 | Green marker object created | Green marker object created |
| 2. | 1 | Yellow marker object created | Yellow marker object created |
| 3. | 5 | Yellow marker object created | Yellow marker object created |
| 4. | 6 | Orange marker object created | Orange marker object created |
| 5. | 15 | Orange marker object created | Orange marker object created |
| 6. | 16 | Red marker object created | Red marker object created |
| 7. | -1 | Test case will never occur | Test case will never occur |

**Note:** The test case where the number of cases in a Hall of Residence is equal to -1 will never occur. This is because our Firebase database stores the timestamps for the reported dengue cases and it can store either 0 or more timestamps. The code retrieves the number of timestamps for each Hall and performs a count to find the number of cases per Hall. This count of the number of timestamps can never be negative hence the number of cases cannot be -1.

## 7.2 White Box Testing

1. Initialisation of Map Markers


Initialisation of Map Markers
1
initState()
2
_doAyncStuff();
Print caseList
int i=0;
3
while (i< length of pinLocationIcon)
5
false
true
4
if(caseList[2*i]==0
6
true
pinLocationIcon[i]=green marker object
7
false
if(caseList[2*i]>=1 AND caseList,2*i]<=5
8
true
pinLocationIcon[i]=yellow marker object
15
false
9
if(caseList[2*i]>=6 AND caseList,2*i]<=15
10
true
pinLocationIcon[i]=orange marker object
14
16
11
false
13
pinLocationIcon[i]=red marker object
12
i++
End of initialisation of Map Marker

Now we analyse the Control Flow Graph to find the Cyclomatic Complexity.

Cyclomatic Complexity= | edges | - | nodes | +2 = 16 - 13 +2 = 5

Alternatively, Cyclomatic Complexity= | decision points | +1 = 4 + 1 = 5

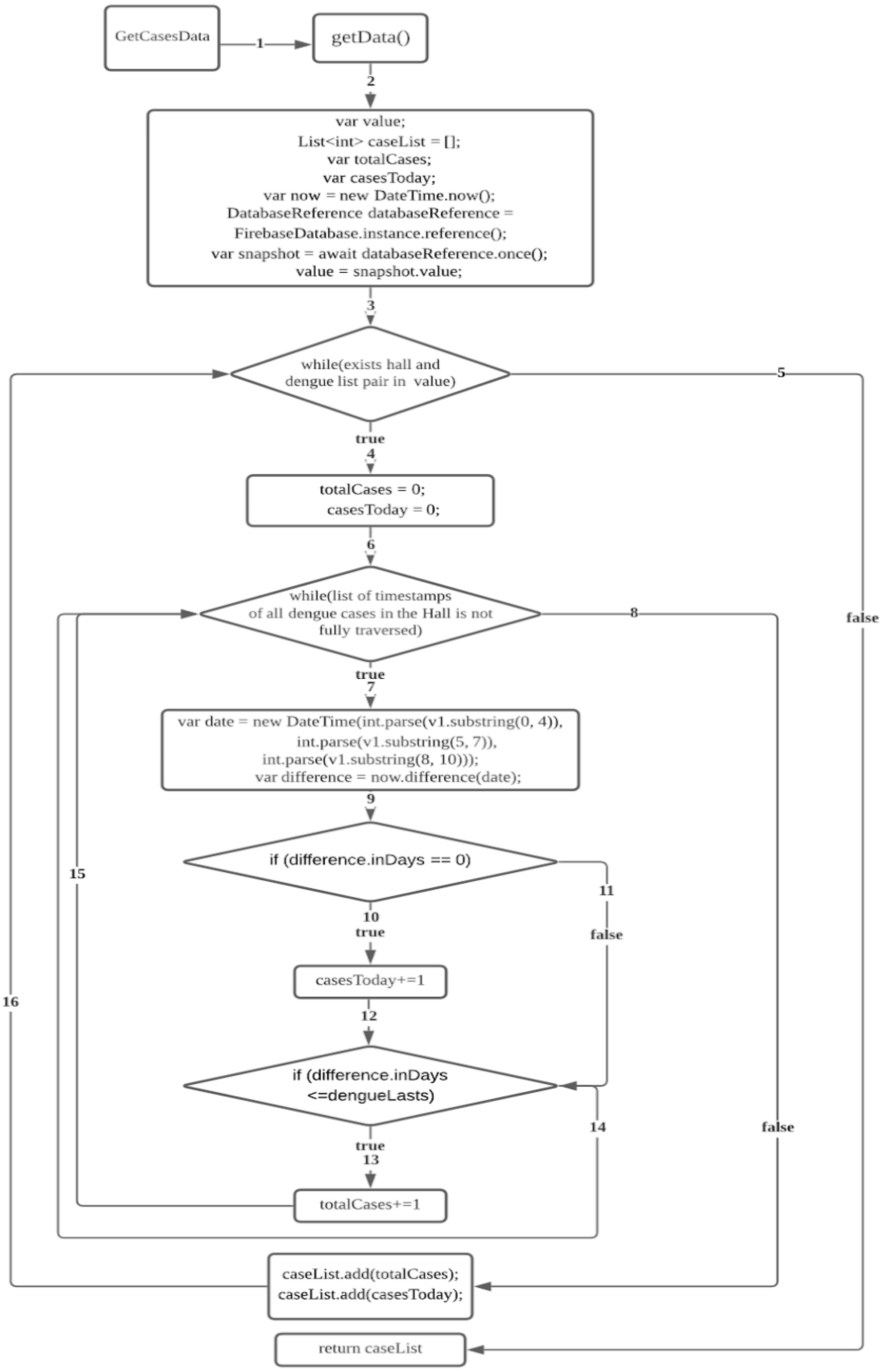
Since the CC is 5, we need to find 5 basis paths for testing.

Basis paths and the inputs satisfying the paths:

- 1, 2, 3, 5 for input i = 27
- 1, 2, 3, 4, 6, 15, 16, 5 for input i = 0, caseList[2*i] = 0
- 1, 2, 3, 4, 7, 8, 14, 16, 5 for input i = 0, caseList[2*i] = 1
- 1, 2, 3, 4, 7, 9, 10, 13, 16, 5 for input i = 0, caseList[2*i] = 6
- 1, 2, 3, 4, 7, 9, 11, 12, 16, 5 for input i = 0, caseList[2*i] = 16

| Serial Number | Test Case | Expected Output | Actual Output |
|---|---|---|---|
| 1. | i = 27 | Loop ends, end of initialisation of Map Marker | Loop ends, end of initialisation of Map Marker |
| 2. | i = 0, caseList[2*i] = 0 | pinLocationIcon[i]=green marker object | pinLocationIcon[i]=green marker object |
| 3. | i = 0, caseList[2*i] = 1 | pinLocationIcon[i]=yellow marker object | pinLocationIcon[i]=yellow marker object |
| 4. | i = 0, caseList[2*i] = 6 | pinLocationIcon[i]=orange marker object | pinLocationIcon[i]=orange marker object |
| 5. | i = 0, caseList[2*i] = 16 | pinLocationIcon[i]=red marker object | pinLocationIcon[i]=red marker object |

2. Get Cases Data



```
var value;
List<int> caseList = [];
var totalCases;
var casesToday;
var now = new DateTime.now();
DatabaseReference databaseReference =
FirebaseDatabase.instance.reference();
var snapshot = await databaseReference.once();
value = snapshot.value;
```

```
totalCases = 0;
casesToday = 0;
```

```
var date = new DateTime(int.parse(v1.substring(0, 4)),
        int.parse(v1.substring(5, 7)),
    int.parse(v1.substring(8, 10)));
  var difference = now.difference(date);
```

```
casesToday+=1
```

```
totalCases+=1
```

```
caseList.add(totalCases);
caseList.add(casesToday);
```

```
return caseList
```

GetCasesData
1
getData()
2
3
while(exists hall and dengue list pair in value)
true
4
5
false
6
while(list of timestamps of all dengue cases in the Hall is not fully traversed)
true
7
8
9
if (difference.inDays == 0)
10
true
11
false
12
if (difference.inDays <=dengueLasts)
true
13
14
15
16
false

*Note that the value of dengueLasts variable is 8 because we assume that dengue lasts for 8 days.*

Now we analyse the Control Flow Graph to find the Cyclomatic Complexity.

Cyclomatic Complexity= | edges | - | nodes | +2 = 16 - 13 +2 = 5

Alternatively, Cyclomatic Complexity= | decision points | +1 = 4 + 1 = 5

Since the CC is 5, we need to find 5 basis paths for testing.

Basis paths and the inputs satisfying the paths:

- 1, 2, 3, 5 for input (hall and dengue list pair does not exist in value)
- 1, 2, 3, 4, 6, 8, 16, 5 for input (hall and dengue list pair exists in value, list of timestamps of all dengue cases in the hall is fully traversed)
- 1, 2, 3, 4, 6, 7, 9, 10, 12, 13, 15, 8, 6, 5 for input (hall and dengue list pair exists in value, list of timestamps of all dengue cases in the hall is not fully traversed,

  difference.inDays = 0)
- 1, 2, 3, 4, 6, 7, 9, 11, 13, 15, 8, 16, 5 for input (hall and dengue list pair exists in value, list of timestamps of all dengue cases in the hall is not fully traversed,

  difference.inDays = 1 )
- 1, 2, 3, 4, 6, 7, 9, 11, 14, 8, 6, 5 for input (hall and dengue list pair exists in value, list of timestamps of all dengue cases in the hall is not fully traversed,
- difference.inDays = 9 )

| Serial Number | Test Case | Expected Output | Actual Output |
|---|---|---|---|
| 1. | hall and dengue list pair does not exist in value | Outer loop ends, return caseList | Outer loop ends, return caseList |
| 2. | hall and dengue list pair exists in value, list of timestamps of all dengue cases in the hall is fully traversed | caseList.add(totalCases);<br>caseList.add(casesToday);<br>Go back to outer loop to iterate through dengue case list of the next hall; | caseList.add(totalCases);<br>caseList.add(casesToday);<br>Go back to outer loop to iterate through dengue case list of the next hall; |
| 3. | hall and dengue list pair exists in value, list of timestamps of all dengue cases in the hall is not fully traversed,<br>difference.inDays = 0 | casesToday += 1;<br>totalCases += 1;<br>caseList.add(totalCases);<br>caseList.add(casesToday);<br>Go back to outer loop to iterate through dengue case list of the next hall; | casesToday += 1;<br>totalCases += 1;<br>caseList.add(totalCases);<br>caseList.add(casesToday);<br>Go back to outer loop to iterate through dengue case list of the next hall; |
| 4. | hall and dengue list pair exists in value, list of timestamps of all dengue cases in the hall is not fully traversed,<br>difference.inDays = 1 | totalCases += 1;<br>caseList.add(totalCases);<br>caseList.add(casesToday);<br>Go back to outer loop to iterate through dengue case list of the next hall; | totalCases += 1;<br>caseList.add(totalCases);<br>caseList.add(casesToday);<br>Go back to outer loop to iterate through dengue case list of the next hall; |
| 5. | hall and dengue list pair exists in value, list of timestamps of all dengue cases in the hall is not fully traversed,<br>difference.inDays = 9 | caseList.add(totalCases);<br>caseList.add(casesToday);<br>Go back to outer loop to iterate through dengue case list of the next hall; | caseList.add(totalCases);<br>caseList.add(casesToday);<br>Go back to outer loop to iterate through dengue case list of the next hall; |

# 8. Appendix

Please refer to the Read-Me.pdf for information on our other folders.

## 8.1 Demo Video

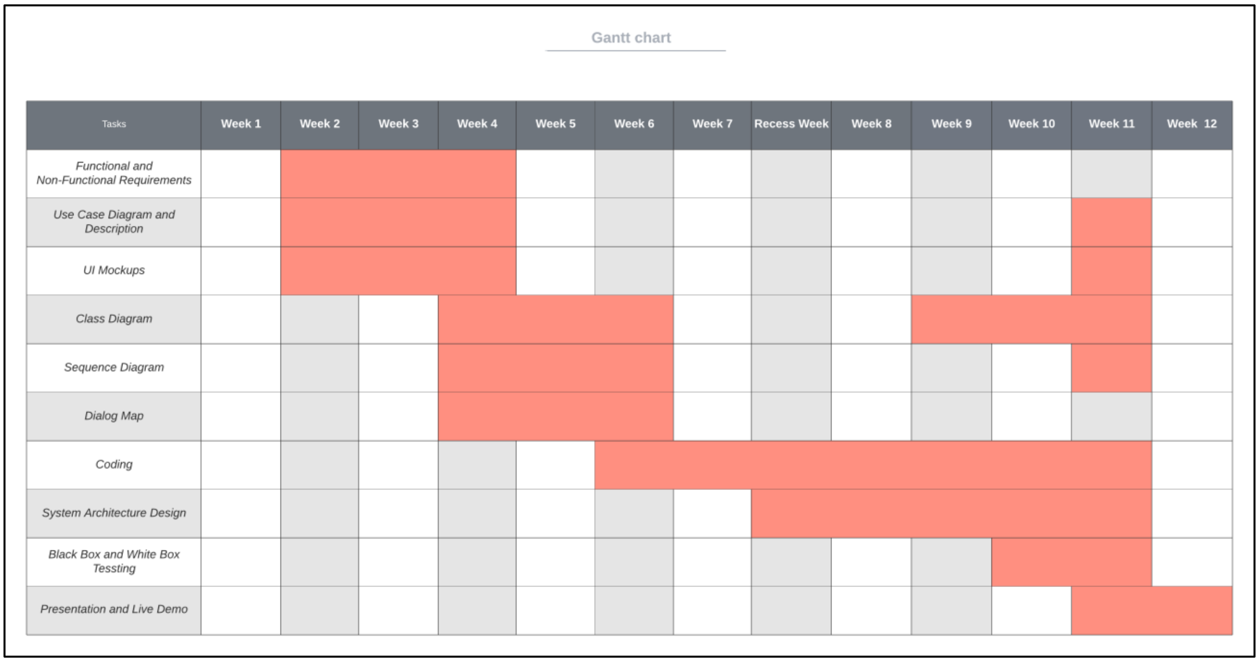
For more information and a detailed demo of our DengueSafe app, please refer to the YouTube link below for our video demo:- https://youtu.be/l07JdE6q9GU

## 8.2 Work Division Table

| Team Member Name | Work Contribution |
|---|---|
| Pal Aratrika | Use Case Diagram, Use Case Description, Class Diagram, Coding for Student Homepage, FAQ, Hotspot Map and SG Stats, ie, all the pages and functionalities in Student Mode, System Architecture Diagram, Design Pattern, Demo-Script (Live App Demo), Demo Video, Black Box Testing and White Box Testing. |
| Tan Mei Xuan | UI Mockups, Sequence Diagram (Login, Reporting and Hotspot), Coding for Report Cases, System Architecture Diagram, Demo-Script (Tech Stack and Sequence Diagram), and Black Box Testing. |
| Tan Min, Tricia | Functional and Non-Functional Requirement , Sequence Diagram (HomePage, Statistics and Alert), Coding for Login Page,  System Architecture Diagram, Demo-Script (SDLC Model and Live App Demo), Demo Video and Black Box Testing. |
| Bachhas Nikita | Use Case Diagram, Use Case Description, Class Diagram, Coding for HotSpot Map page and SG Stats Page for “See Previous” clusters, System Architecture Diagram, Demo-Script (Use Case Diagram and Functionality) and Black Box Testing. |
| Loh Xin Yi | UI Mockups, Dialog Map, Coding for student homepage, FAQ, overall UI, System Architecture Diagram, Demo-Script (System |

| | |
|---|---|
| | Architecture Diagram and Future Upgrades), Demo Video and Black Box Testing. |
| Ong Li Wen | Data Dictionary, Dialog Map, Coding for student homepage. SG Stats Page, overall UI, System Architecture Diagram, Demo-Script (Product Statement and Product Description) and Black Box Testing. |

## 8.3 Workflow Timeline



Gantt chart

| Tasks | Week 1 | Week 2 | Week 3 | Week 4 | Week 5 | Week 6 | Week 7 | Recess Week | Week 8 | Week 9 | Week 10 | Week 11 | Week 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| *Functional and Non-Functional Requirements* | | 🟥 | 🟥 | 🟥 | | | | | | | | | |
| *Use Case Diagram and Description* | | 🟥 | 🟥 | 🟥 | | | | | | | | 🟥 | |
| *UI Mockups* | | 🟥 | 🟥 | 🟥 | | | | | | | | 🟥 | |
| *Class Diagram* | | | | 🟥 | 🟥 | 🟥 | | | | 🟥 | 🟥 | 🟥 | |
| *Sequence Diagram* | | | | 🟥 | 🟥 | 🟥 | | | | | | 🟥 | |
| *Dialog Map* | | | | 🟥 | 🟥 | 🟥 | | | | | | | |
| *Coding* | | | | | | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | |
| *System Architecture Design* | | | | | | | | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | |
| *Black Box and White Box Tessting* | | | | | | | | | | | 🟥 | 🟥 | |
| *Presentation and Live Demo* | | | | | | | | | | | | 🟥 | 🟥 |