Diploma in Electronic and Computer Engineering (DECE)

EGE312: Mobile Computing Project

Project

NYP NANYANG THE INNOVATIVE POLYTECHNIC

Effective Date : 15 Apr 2024

Project

- In this project, you will build a Singapore Transport app:
  - To use API to fetch Bus Arrival from LTA DataMall
  - To use API to fetch Platform Crowd Density from LTA DataMall
  - To use API to fetch all Taxi Stands from LTA DataMall
  - To log taxi trips and store in Firestore Database

Project Architecture

Flutter app

Land Transport Authority
We Keep Your World Moving

DATAMALL
an
LTA Open Data Initiative

- 2.1 Bus Arrival
- 2.4 Bus Stops
- 2.10 Taxi Stands
- 2.25 Platform Crowd Density

Download API User Guide & Documentation
https://datamall.lta.gov.sg/content/dam/datamall/datasets/LTA_DataMall_API_User_Guide.pdf

Firebase Authentication
- For user to login / register
- Updates display name

Firestore Database
- fares collection

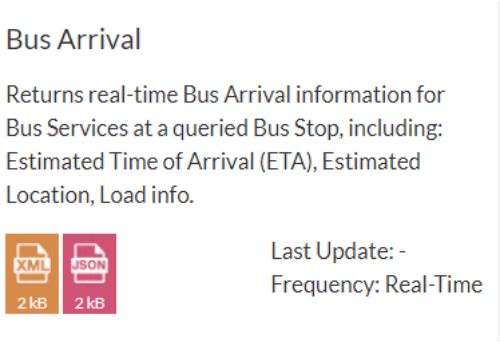
Collection

fares
origin
dest
fare
date
userid

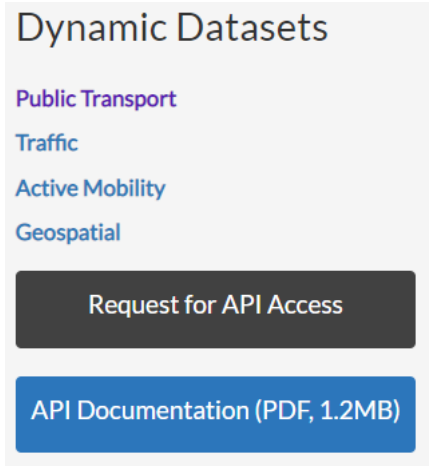
LTA DataMall

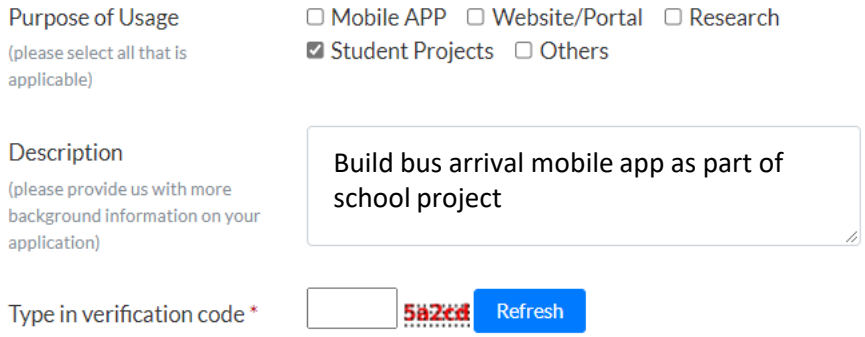
LTA DataMall

- Go to https://datamall.lta.gov.sg/content/datamall/en/dynamic-data.html



- Request for API Access





LTA DataMall

- Download the API User Guide & Documentation
- https://datamall.lta.gov.sg/content/dam/datamall/datasets/LTA_DataMall_API_User_Guide.pdf

- Read up on **Section 1 Making API Calls** (Page 6-7)
- Read up on **Section 2.1 Bus Arrival** (Page 12)

2.1 BUS ARRIVAL

URL	http://datamall2.mytransport.sg/ltaodataservice/BusArrivalv2		
Description	Returns real-time Bus Arrival information of Bus Services at a queried Bus Stop, including Est. Arrival Time, Est. Current Location, Est. Current Load.		
Update Freq	1 minute		
Request			
Parameters	**Description**	**Mandatory**	**Example**
BusStopCode	Bus stop reference code	Yes	*83139*
ServiceNo	Bus service number	No	*15*

LTA DataMall

- Read up on **Section 2.4 Bus Stops** (Page 20)

2.4 BUS STOPS

URL	http://datamall2.mytransport.sg/ltaodataservice/BusStops	
Description	Returns detailed information for all bus stops currently being serviced by buses, including: Bus Stop Code, location coordinates.	
Update Freq	Ad hoc	
Response		
Attributes	**Description**	**Sample**
BusStopCode	The unique 5-digit identifier for this physical bus stop	*01012*
RoadName	The road on which this bus stop is located	*Victoria St*
Description	Landmarks next to the bus stop (if any) to aid in identifying this bus stop	*Hotel Grand Pacific*
Latitude	Location coordinates for this bus stop	*1.29685*
Longitude		*103.853*

LTA DataMall

- Read up on **Section 2.10 Taxi Stands** (Page 26)

2.10 TAXI STANDS

URL	http://datamall2.mytransport.sg/ltaodataservice/TaxiStands	
Description	Returns detailed information of Taxi stands, such as location and whether is it barrier free.	
Update Freq	Monthly	
Response		
Attributes	**Description**	**Sample**
TaxiCode	Code representation of Taxi facility.	*A01*
Latitude	Latitude map coordinates for the start point of this road incident.	*1.303980684*
Longitude	Longitude map coordinates for the start point of this incident.	*103.9191828*
Name	Name of Taxi facility.	*Orchard Rd along driveway of Lucky Plaza*

LTA DataMall

- Read up on **Section 2.25 Platform Crowd Density Real Time** (Page 43)

2.25 PLATFORM CROWD DENSITY REAL TIME

URL	http://datamall2.mytransport.sg/ltaodataservice/PCDRealTime		
Description	Returns real-time platform crowdedness level for the MRT/LRT stations of a particular train network line		
Update Freq	10 minutes		
Request			
Parameters	**Description**	**Mandatory**	**Example**
TrainLine	Code of train network line. Train lines supported: • CCL (for Circle Line) • CEL (for Circle Line Extension – BayFront, Marina Bay) • CGL (for Changi Extension – Expo, Changi Airport) • DTL (for Downtown Line) • EWL (for East West Line) • NEL (for North East Line) • NSL (for North South Line) • BPL (for Bukit Panjang LRT) • SLRT (for Sengkang LRT) • PLRT (for Punggol LRT)	Yes	*EWL*

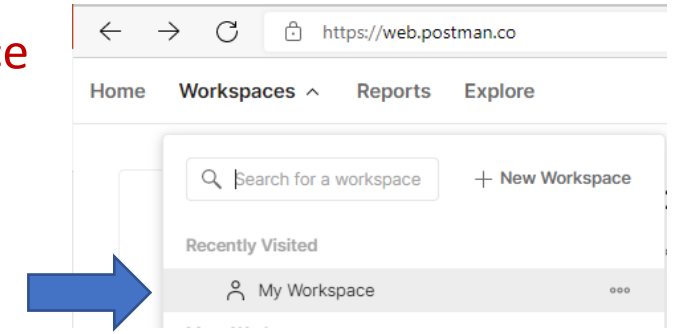
LTA DataMall – Bus Arrival


```
{
  "odata.metadata": "http://datamall2.mytransport.sg/ltaodataservice/$metadata#BusArrivalv2/@Element",
  "BusStopCode": "83139",
  "Services": [
    {
      "ServiceNo": "15",
      "Operator": "GAS",
      "NextBus": {
        "OriginCode": "77009",
        "DestinationCode": "77009",
        "EstimatedArrival": "2024-04-12T15:44:55+08:00",
        "Latitude": "1.3288096666666667",
        "Longitude": "103.90548016666666",
        "VisitNumber": "1",
        "Load": "SEA",
        "Feature": "WAB",
        "Type": "SD"
      },
```

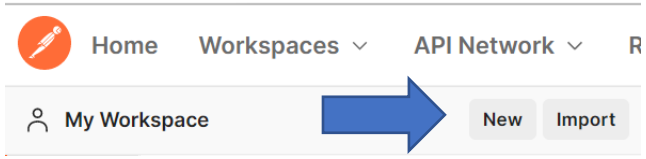

Response		
Attributes	**Description**	**Example**
ServiceNo	Bus service number	*15*
Operator	Public Transport Operator Codes: • SBST (for SBS Transit) • SMRT (for SMRT Corporation) • TTS (for Tower Transit Singapore) • GAS (for Go Ahead Singapore)	*GAS*
NextBus NextBus2 NextBus3	Structural tags for all bus level attributes^ of the next 3 oncoming buses. Note that if there is only one last bus left on the roads (e.g. at night), attributes values in NextBus2 and NextBus3 will be empty / blank.	
^ OriginCode	Reference code of the first bus stop where this bus started its service	*77009*
^ DestinationCode	Reference code of the last bus stop where this bus will terminate its service	*77131*
^ EstimatedArrival	Date-time of this bus' estimated time of arrival, expressed in the UTC standard, GMT+8 for Singapore Standard Time (SST)	*2017-04-29T07:20:24+08:00*
^ Latitude	Current estimated location coordinates of this bus at point of published data	*1.42117943692586*
^ Longitude		*103.831477233098*
^ VisitNumber	Ordinal value of the n^{th} visit of this vehicle at this bus stop; 1=1^{st} visit, 2=2^{nd} visit	*1*
^ Load	Current bus occupancy / crowding level: ▪ SEA (for Seats Available) ▪ SDA (for Standing Available) ▪ LSD (for Limited Standing)	*SEA*
^ Feature	Indicates if bus is wheel-chair accessible: • WAB • (empty / blank)	*WAB*
^ Type	Vehicle type: • SD (for Single Deck) • DD (for Double Deck) • BD (for Bendy)	*SD*

Postman

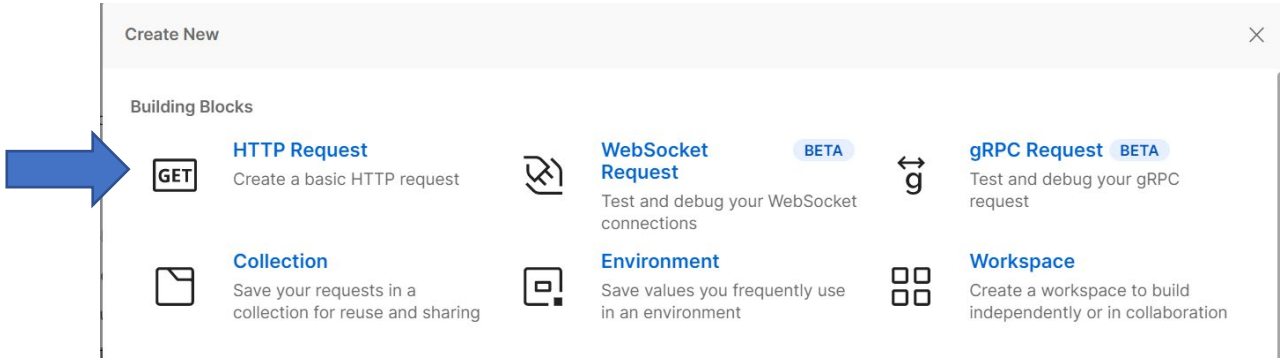
- We can use Postman to make a API call and view the response.
- Go to https://web.postman.co/ and login using your Google account.
- Go to Workspaces > My Workspace



- Click New.

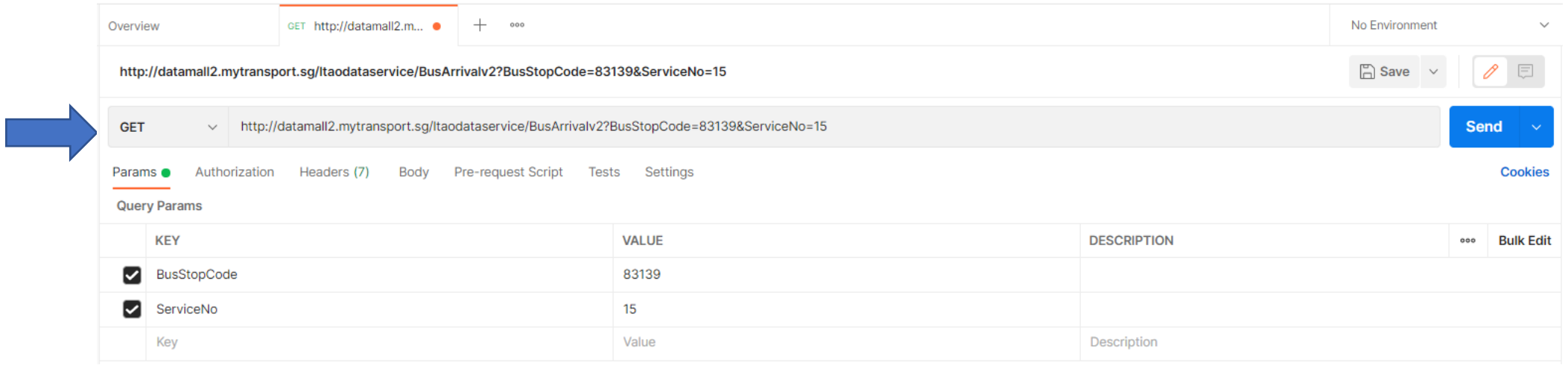


- Click HTTP Request.



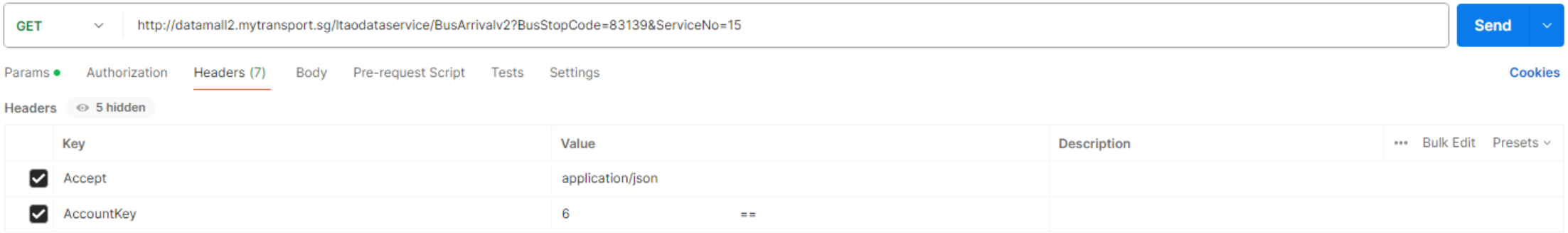
Postman

- Make sure that http method is set to GET.
- Enter the URL: http://datamall2.mytransport.sg/ltaodataservice/BusArrivalv2
- Under Params, add 'BusStopCode' as KEY and '83139' as VALUE.
- Under Params, add 'ServiceNo' as KEY and '15' as VALUE.



Postman

- Under Headers, add ‘AccountKey’ in KEY and ‘<YOUR-KEY>’ in VALUE.
- Under Headers, add ‘Accept’ in KEY and ‘application/json’ in VALUE.



- Click Send.

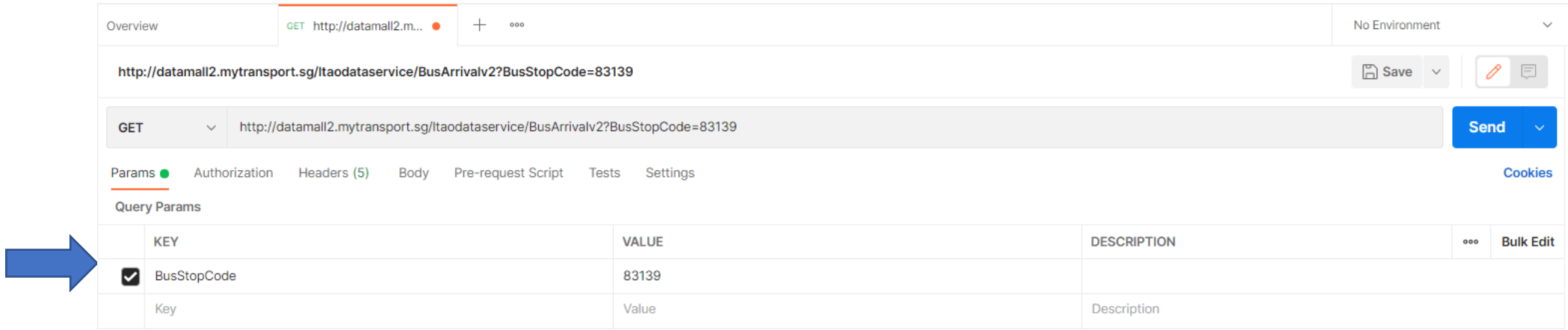
Postman

- This is the response.


```
{
   "odata.metadata":
"http://datamall2.mytransport.sg/ltaodataservice/$metadata#BusArrivalv2/@Element",
   "BusStopCode": "83139",
   "Services": [
      {
         "ServiceNo": "15",
         "Operator": "GAS",
         "NextBus": {
            "OriginCode": "77009",
            "DestinationCode": "77009",
            "EstimatedArrival": "2024-04-12T15:44:55+08:00",
            "Latitude": "1.3288096666666667",
            "Longitude": "103.90548016666666",
            "VisitNumber": "1",
            "Load": "SEA",
            "Feature": "WAB",
            "Type": "SD"
         },
         "NextBus2": {
            "OriginCode": "77009",
            "DestinationCode": "77009",
            "EstimatedArrival": "2024-04-12T15:46:45+08:00",
            "Latitude": "1.333443",
            "Longitude": "103.9039145",
            "VisitNumber": "1",
            "Load": "SEA",
            "Feature": "WAB",
            "Type": "SD"
         },
         "NextBus3": {
            "OriginCode": "77009",
            "DestinationCode": "77009",
            "EstimatedArrival": "2024-04-12T15:57:20+08:00",
            "Latitude": "1.3488268333333333",
            "Longitude": "103.9286775",
            "VisitNumber": "1",
            "Load": "SEA",
            "Feature": "WAB",
            "Type": "SD"
         }
      }
   ]
}
```

Retrieve all Bus Services at a Bus Stop

- If we want to retrieve all bus services at a bus stop, we can enter only the BusStopCode.
- Under Params, add ‘BusStopCode’ as KEY and ‘83139’ as VALUE.



Retrieve all Bus Services at a Bus Stop

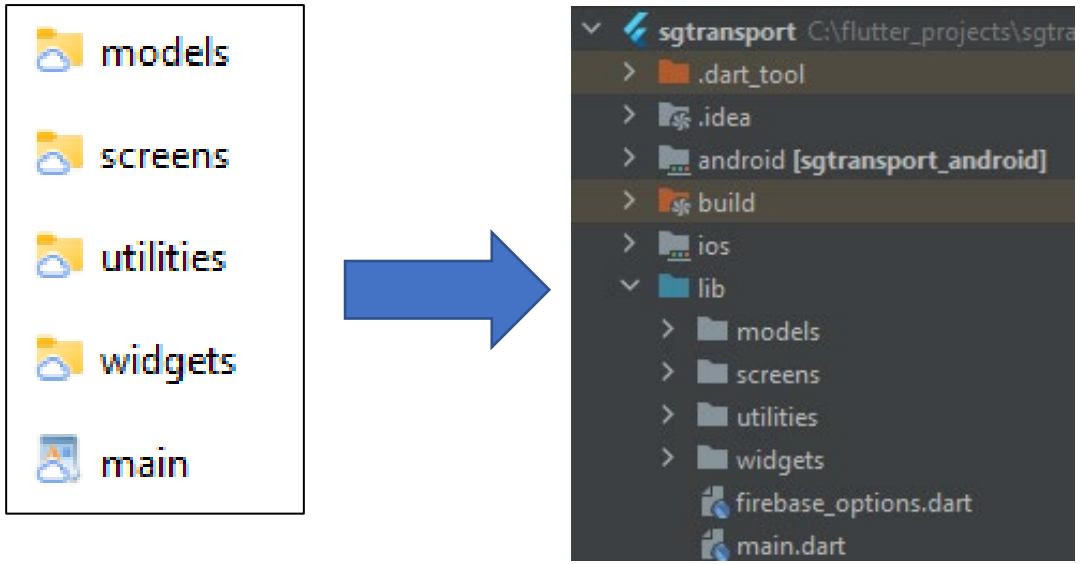
- Note that Services is an array of 3 objects with ServiceNo 15, 150 and 155.
- Some fields have been removed for clarity.


```
{
   "odata.metadata":
"http://datamall2.mytransport.sg/ltaodataservice/$metadata#BusArrivalv2/@El
ement",
   "BusStopCode": "83139",
   "Services": [
      {
         "ServiceNo": "15",
         "Operator": "GAS",
         "NextBus": {
                  "EstimatedArrival": "2024-04-12T09:42:49+08:00",
         },
         "NextBus2": {
                  "EstimatedArrival": "2024-04-12T09:47:16+08:00",
         },
         "NextBus3": {
                  "EstimatedArrival": "2024-04-12T09:53:11+08:00",
         }
      },
      {
         "ServiceNo": "150",
         "Operator": "SBST",
         "NextBus": {
                  "EstimatedArrival": "2024-04-12T09:53:24+08:00",
         },
         "NextBus2": {
                  "EstimatedArrival": "2024-04-12T10:08:24+08:00",
         },
         "NextBus3": {
                  "EstimatedArrival": "2024-04-12T10:23:24+08:00",
         }
      },
      {
         "ServiceNo": "155",
         "Operator": "SBST",
         "NextBus": {
                  "EstimatedArrival": "2024-04-12T09:44:04+08:00",
         },
         "NextBus2": {
                  "EstimatedArrival": "2024-04-12T09:56:37+08:00",
         },
         "NextBus3": {
                  "EstimatedArrival": "2024-04-12T10:11:31+08:00",
         }
      }
   ]
}
```

App Development

Create New Project

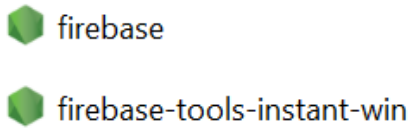
- Start the Android Studio IDE and create a new Flutter project: sgtransport.
- Ensure that the Project location is set to "C:\flutter_projects\sgtransport" and click Finish.
- Download and extract the sgtransport_start.zip.
- Copy the folders (models, screens, utilities and widgets) to your flutter project sgtransport/lib folder.



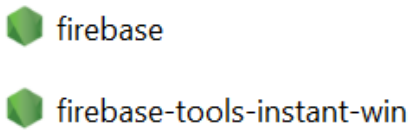
- Replace main.dart with the one in the extracted folder.

Add Firebase Configuration to Flutter app

- In Windows Explorer, move these 2 files from C:\flutter_projects\firebase to C:\flutter_projects\sgtransport.

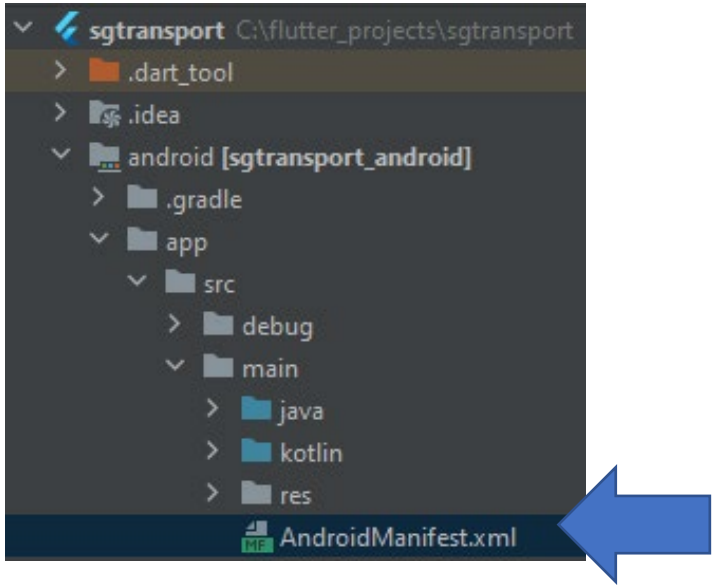


- Execute firebase-tools-instant-win.exe from Windows Explorer.
- You should still be logged in. Otherwise, issue this command: firebase login.
- **Extend** the width of the console (you need to type in package name later).
- Issue this command to install the FlutterFire CLI: flutterfire configure
- Choose the same Firebase-Flutter project and press Enter.
- Ensure that both android and ios are checked and press Enter.
- Type com.example.sgtransport for the package name.
- A firebase_options.dart file is automatically created in lib folder.
- In the Firebase console, you will see that another 2 apps have been added to the project.
- **Move these 2 files to C:\flutter_projects folder.**



Running the App

- Go to: sgtransport\android\app\src\main\AndroidManifest.xml and add the highlighted codes.



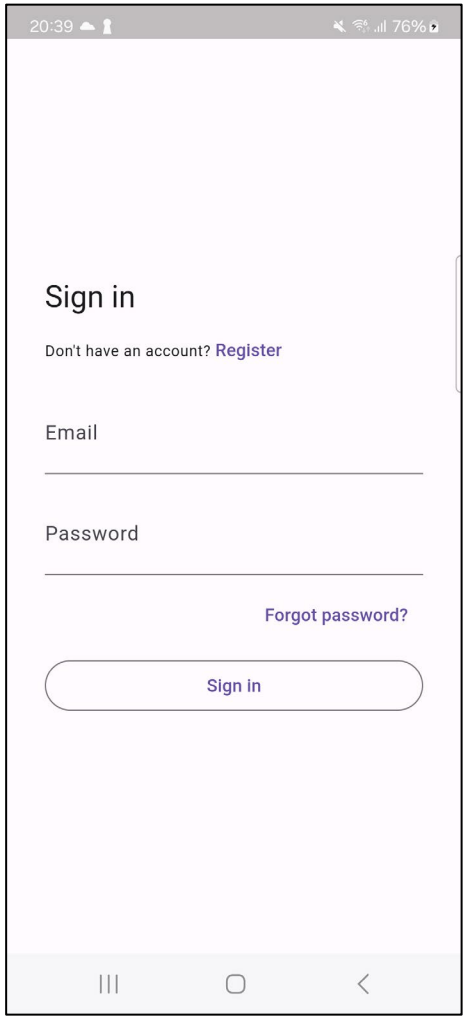

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
  package="com.example.nyp_courses">

  <queries>
    <!-- If your app opens https URLs -->
    <intent>
      <action android:name="android.intent.action.VIEW" />
      <data android:scheme="https" />
    </intent>
  </queries>

  <application
    android:label="nyp_courses"
    android:icon="@mipmap/ic_launcher">
```

Running the App

- Add the necessary plugins in pubspec.yaml

http: ^1.2.1
firebase_core: ^2.24.0
firebase_auth: ^4.15.0
firebase_ui_auth: ^1.14.0
cloud_firestore: ^4.13.3
url_launcher: ^6.1.11

- Press Pub get.
- You should be able to run the app with no errors at this point.
- You may sign in using the same account for Lab 10 or register a new account.



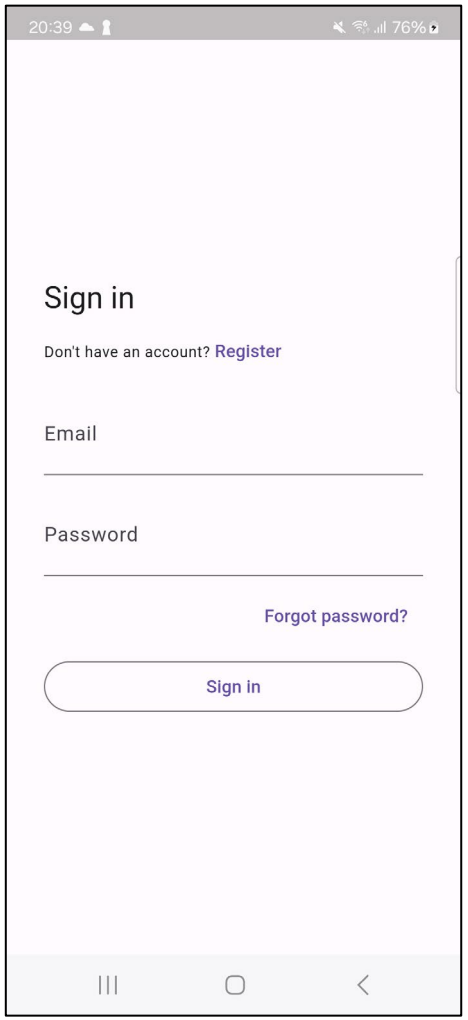
Screens

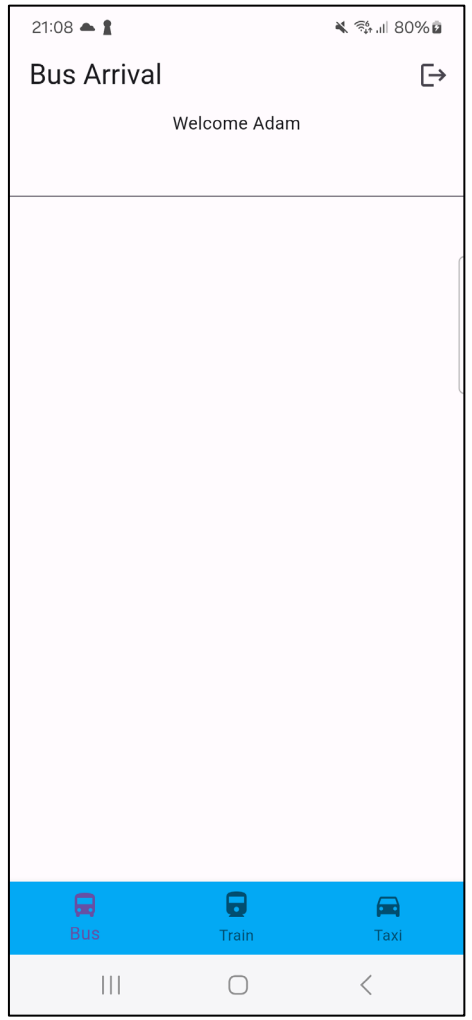
LoginScreen

- For Firebase user authentication:

1. Show **SignInScreen()**
2. If displayname == null > **ProfileScreen()**
3. Otherwise, show **BusScreen()**

ALL DONE!



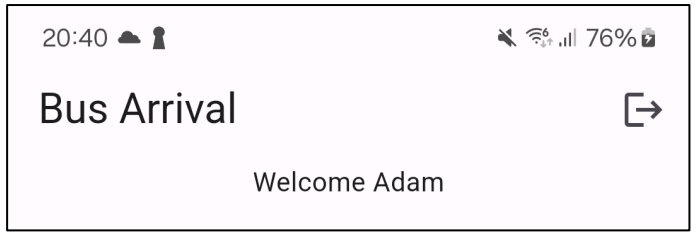


BusScreen

- In this screen, user can perform a search using bus stop description or road name to retrieve all the bus arrival timings of the buses at that bus stop.

RoadName	The road on which this bus stop is located	*Victoria St*
Description	Landmarks next to the bus stop (if any) to aid in identifying this bus stop	*Hotel Grand Pacific*

- **DONE:** Shows app title, user display name and logout button.



BusScreen

- **TASK 1:** Executes API call to get all bus stops (with bus stop code, road name, description, latitude and longitude) (see **2.4 Bus Stops**) and store in _allBusStops.

URL	http://datamall2.mytransport.sg/ltaodataservice/BusStops	
Description	Returns detailed information for all bus stops currently being serviced by buses, including: Bus Stop Code, location coordinates.	
Update Freq	Ad hoc	
Response		
Attributes	**Description**	**Sample**
BusStopCode	The unique 5-digit identifier for this physical bus stop	*01012*
RoadName	The road on which this bus stop is located	*Victoria St*
Description	Landmarks next to the bus stop (if any) to aid in identifying this bus stop	*Hotel Grand Pacific*
Latitude	Location coordinates for this bus stop	*1.29685*
Longitude		*103.853*

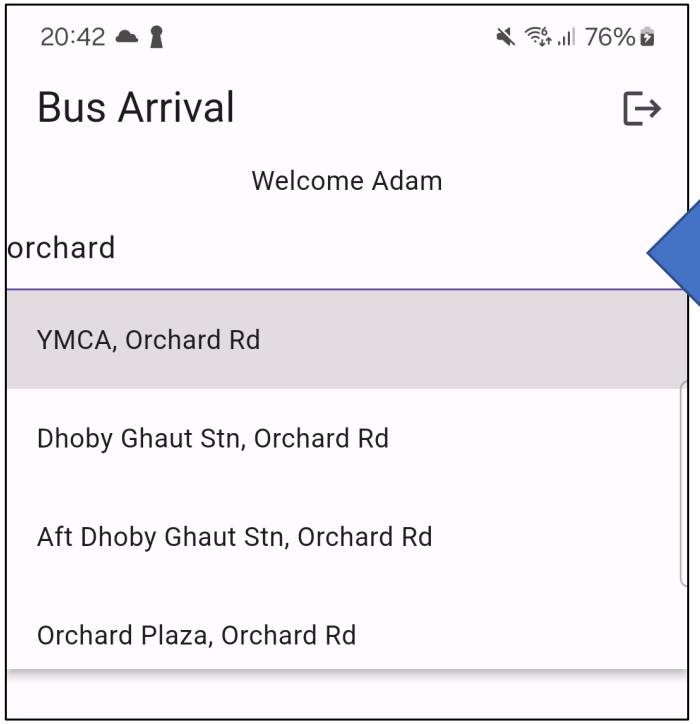

```
"odata.metadata": "http://datamall2.mytransport.sg/ltaodataservice/$metadata#BusStops",
"value": [
    {
        "BusStopCode": "01012",
        "RoadName": "Victoria St",
        "Description": "Hotel Grand Pacific",
        "Latitude": 1.29684825487647,
        "Longitude": 103.85253591654006
    },
    {
        "BusStopCode": "01013",
        "RoadName": "Victoria St",
        "Description": "St. Joseph's Ch",
        "Latitude": 1.29770970610083,
        "Longitude": 103.8532247463225
    },
```


Response

BusScreen

TASK 2:

- Use AutoComplete widget to provide a list of options that user can select from as they type.
- For example, when user types ‘orchard’, all the bus stop options in _allBusStops that contain ‘orchard’ in road name or description will appear as a list of options.
- When user selects one of the options, it is saved in _selectedBusStop.



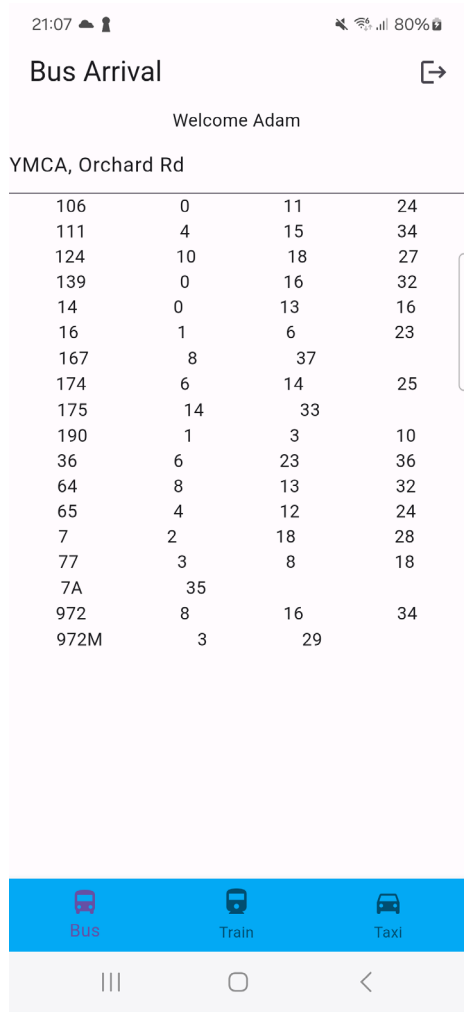
AutoComplete

BusScreen

TASK 3:

- Make an API call to retrieve the next bus arrival information of all the buses at the _selectedBusStop (see **2.1 Bus Arrival**).
- Useful information such as load, feature and type should be included. You may use different colors, icons or images to show these information.

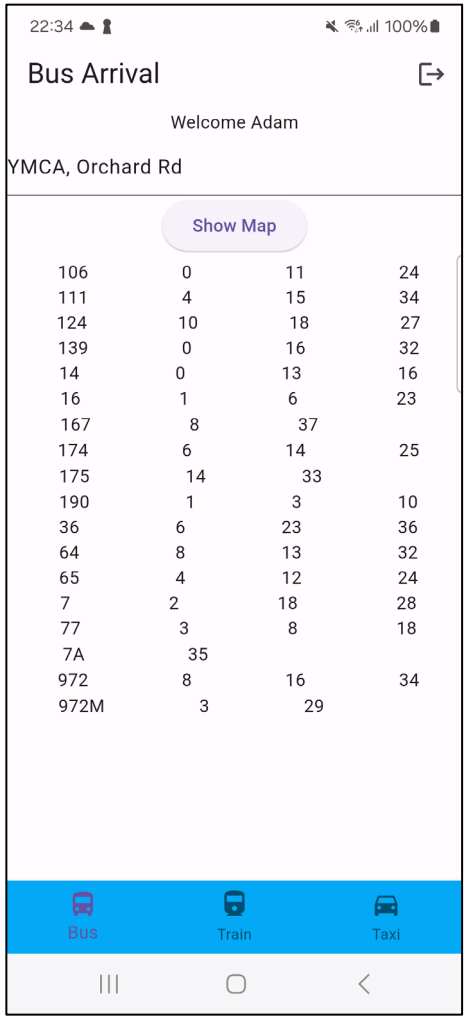
^ Load	Current bus occupancy / crowding level: ▪ SEA (for Seats Available) ▪ SDA (for Standing Available) ▪ LSD (for Limited Standing)	*SEA*
^ Feature	Indicates if bus is wheel-chair accessible: • WAB • (empty / blank)	*WAB*
^ Type	Vehicle type: • SD (for Single Deck) • DD (for Double Deck) • BD (for Bendy)	*SD*

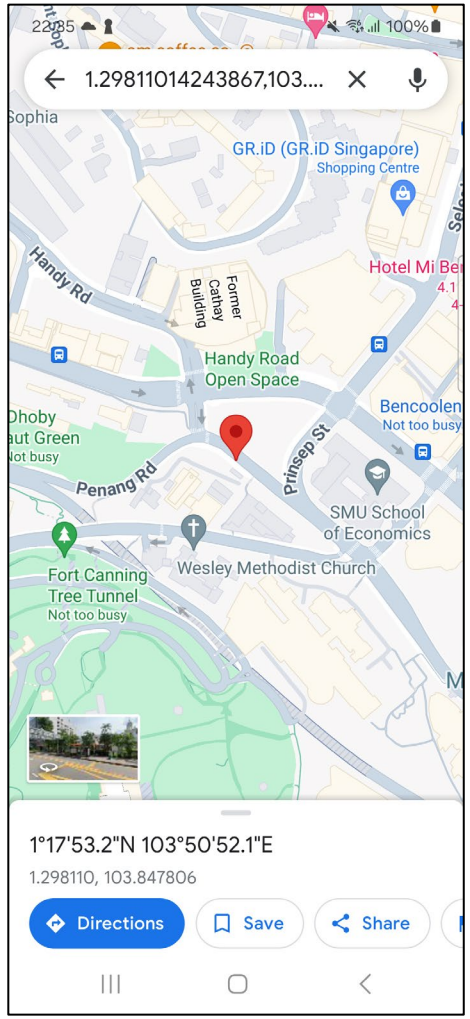


BusScreen

TASK 4:

- Add a button for url launcher to show user the location of the bus stop in a Google map.
- The URL is https://www.google.com/maps/search/?api=1&query=$lat,$lng
- Refer to utilities/my_url_launcher.dart





TrainScreen

- In this screen, user can enter a station name/code to enquire about the platform crowd density.
- A list of train stations with their station codes, station names, train lines and train line codes is given in TrainStationsRepository.

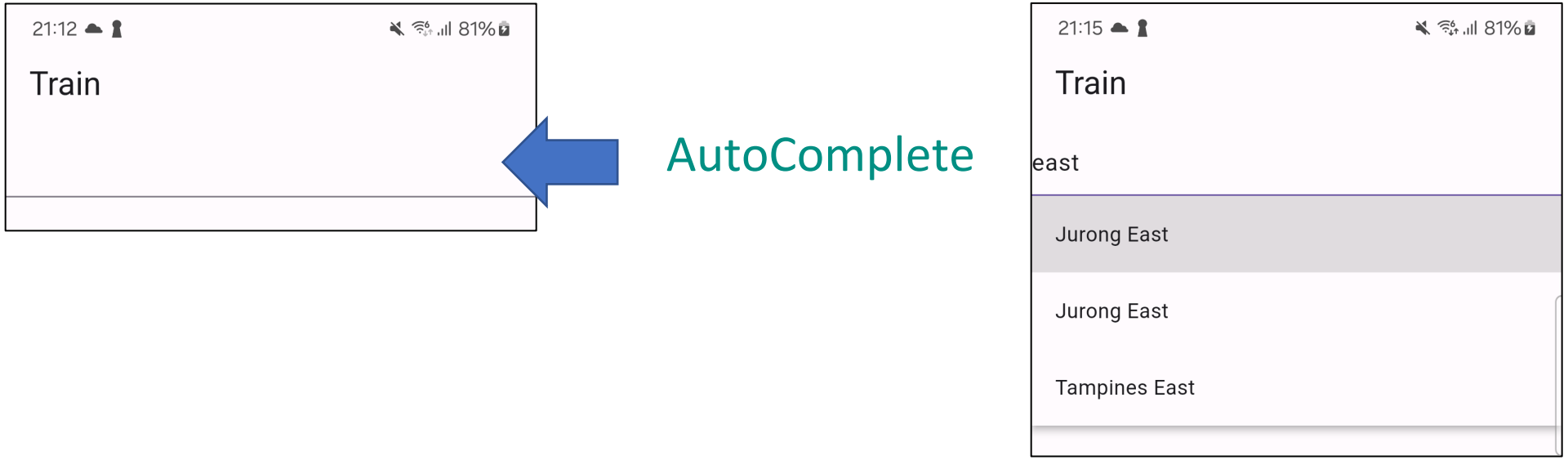
```
class TrainStationsRepository {
 UnmodifiableListView<TrainStation> get allTrainStations {
  return UnmodifiableListView(_allTrainStations);
 }

 int get allTrainStationsCount {
  return _allTrainStations.length;
 }
```

```
 final List<TrainStation> _allTrainStations = [
  TrainStation(
     stnCode: "NS1",
     stnName: "Jurong East",
     trainLine: "North-South Line ",
     trainLineCode: "NSL"),
  TrainStation(
     stnCode: "NS2",
     stnName: "Bukit Batok",
     trainLine: "North-South Line ",
     trainLineCode: "NSL"),
  ...
  ];
}
```

TrainScreen

TASK 5:

- Use AutoComplete widget to provide a list of options from allTrainStations that user can select from as they type.
- For example, when user types ‘east’, options such as ‘Jurong East’ and ‘Tampines East’ are provided.
- When user makes a selection, it is saved in _selectedTrainStation.



TrainScreen

TASK 6:

- Make an API call to retrieve the platform crowd density of the _selectedTrainStation (see **2.25 Platform Crowd Density**).

URL	http://datamall2.mytransport.sg/ltaodataservice/PCDRealTime		
Description	Returns real-time platform crowdedness level for the MRT/LRT stations of a particular train network line		
Update Freq	10 minutes		
Request			
Parameters	**Description**	**Mandatory**	**Example**
TrainLine	Code of train network line. Train lines supported: • CCL (for Circle Line) • CEL (for Circle Line Extension – BayFront, Marina Bay) • CGL (for Changi Extension – Expo, Changi Airport) • DTL (for Downtown Line) • EWL (for East West Line) • NEL (for North East Line) • NSL (for North South Line) • BPL (for Bukit Panjang LRT) • SLRT (for Sengkang LRT) • PLRT (for Punggol LRT)	Yes	*EWL*

Response		
Attributes	**Description**	**Example**
Station	Station code	EW13
StartTime	The start of the time interval	2021-09-15T09:40:00+08:00
EndTime	The end of the time interval	2021-09-15T09:50:00+08:00
CrowdLevel	The crowdedness level indicates: • l: low • h: high • m: moderate • NA	l

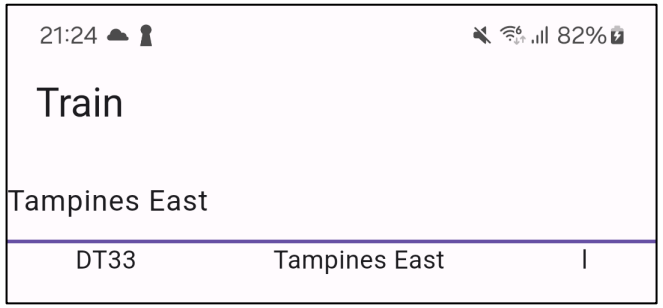
TrainScreen

TASK 6:

- Make an API call to retrieve the platform crowd density of the _selectedTrainStation (see **2.25 Platform Crowd Density**).


```
"odata.metadata": "http://datamall2.mytransport.sg/ltaodataservice/$metadata#PcdRealTime",
"value": [
    {
        "Station": "NS10",
        "StartTime": "2024-04-13T22:30:00+08:00",
        "EndTime": "2024-04-13T22:40:00+08:00",
        "CrowdLevel": "l"
    },
    {
        "Station": "NS11",
        "StartTime": "2024-04-13T22:30:00+08:00",
        "EndTime": "2024-04-13T22:40:00+08:00",
        "CrowdLevel": "l"
    },
```


Response



TaxiScreen

Part 1

- In this screen, user can perform a search for taxi stands using taxi stand names.

TASK 7:

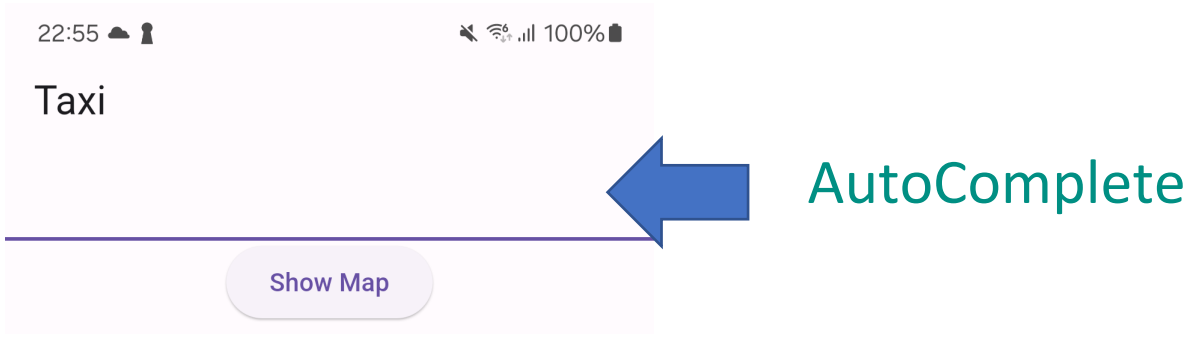
- Executes API call to get all taxi stands (with name, latitude and longitude) (see **2.10 Tax Stands**) and store in _allTaxiStands.

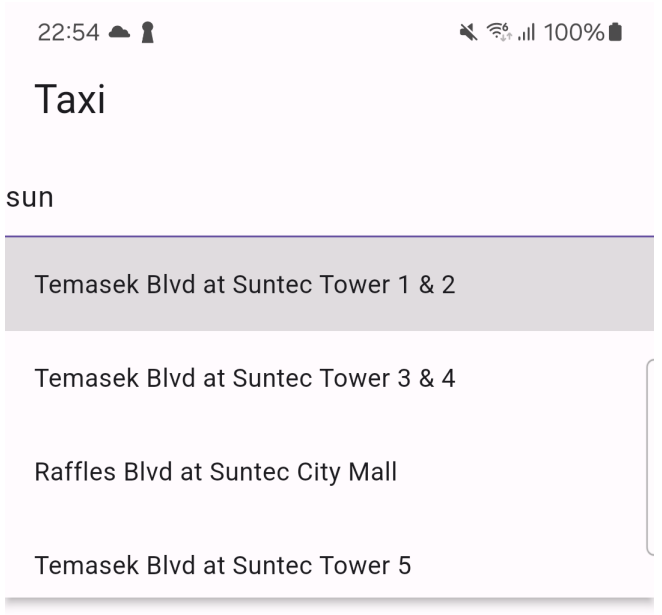
URL	http://datamall2.mytransport.sg/ltaodataservice/TaxiStands
Description	Returns detailed information of Taxi stands, such as location and whether is it barrier free.
Update Freq	Monthly

TaxiScreen

TASK 8:

- Use AutoComplete widget to provide a list of options that user can select from as they type.
- For example, when user types ‘sun’, all the taxi stand options in _allTaxiStands that contain ‘sun’ in name will appear as a list of options.
- When user selects one of the options, it is saved in _selectedTaxiStand.

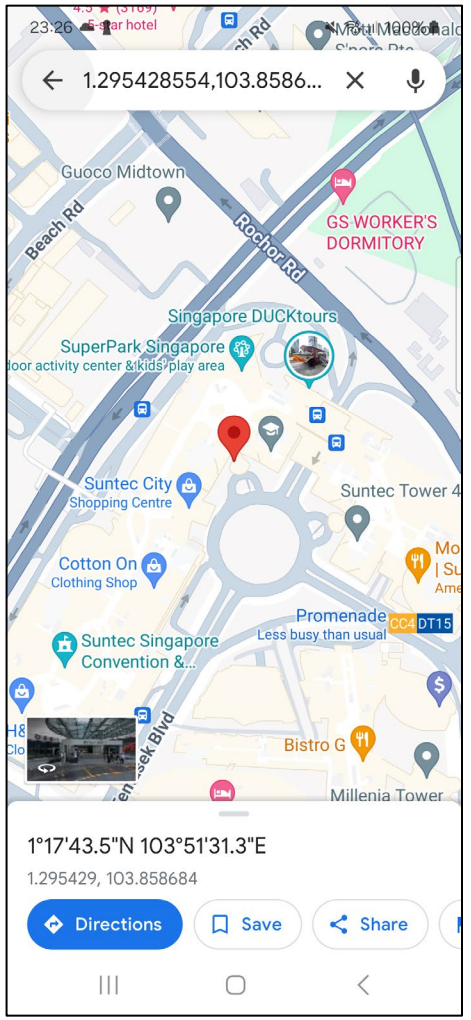




TaxiScreen

TASK 9:

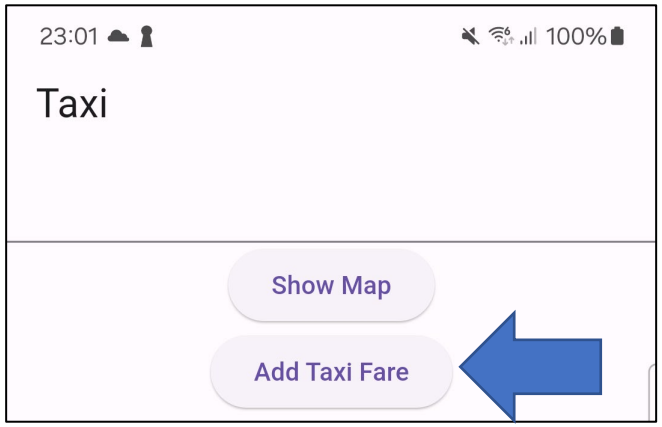
- Add a button for url launcher to show user the location of the taxi stand in a Google map.
- The URL is https://www.google.com/maps/search/?api=1&query=$lat,$lng
- Refer to utilities/my_url_launcher.dart



TaxiScreen

Part 2

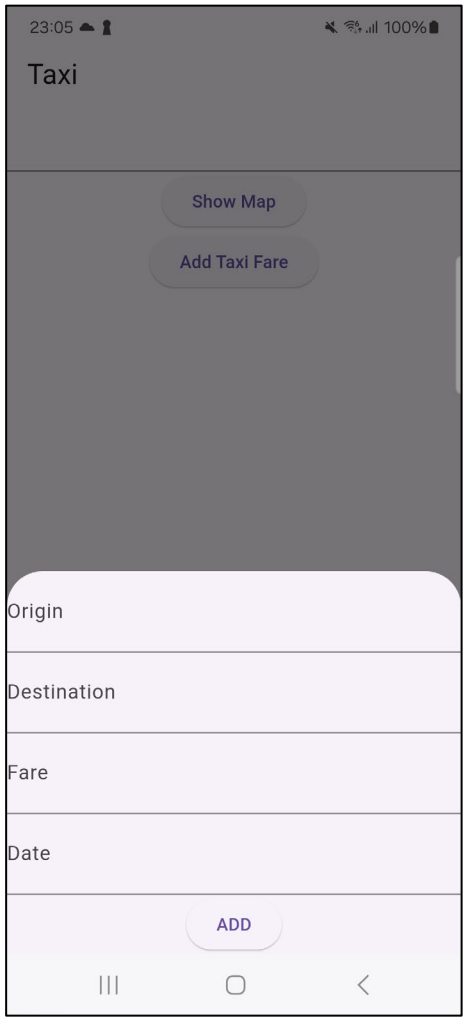
- In this screen, user can also log the taxi fares that he/she took and store in **fares** collection inside Firestore Database.
- **DONE:** Includes button for user to add new taxi fare > **AddTaxiScreen()**.



AddTaxiScreen

TASK 10:

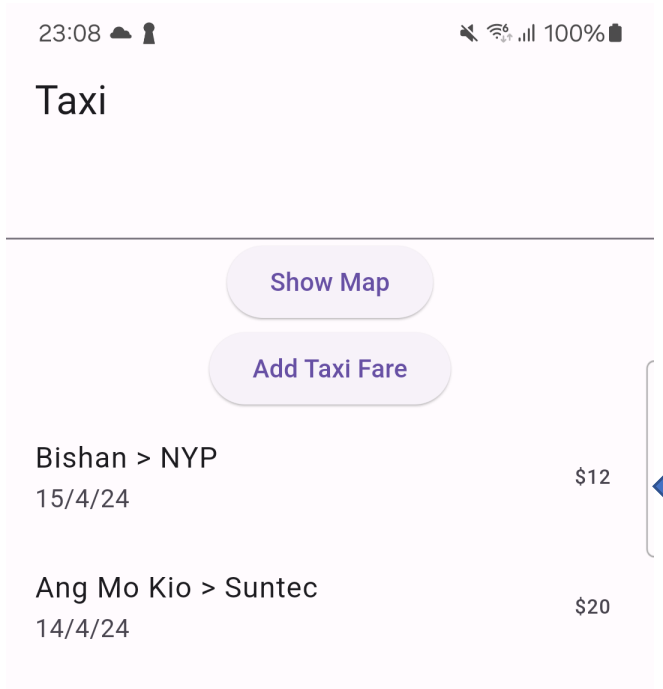
- Provides TextField widgets (or equivalent) for user to enter origin, destination, fare and date of trip.
- Provides ElevatedButton for user to add the new taxi trip to **fares** collection.

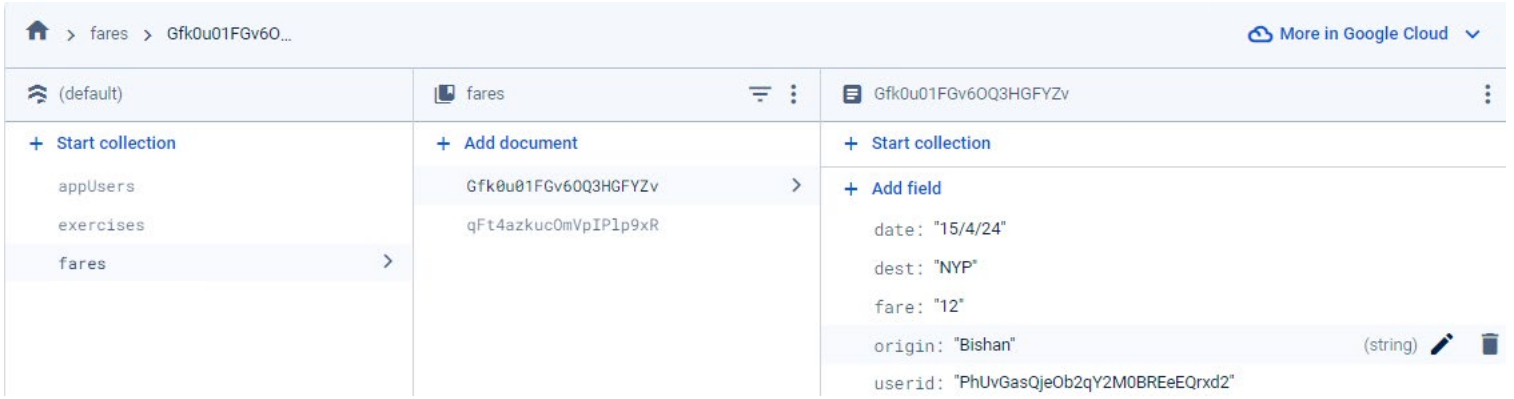


TaxiScreen

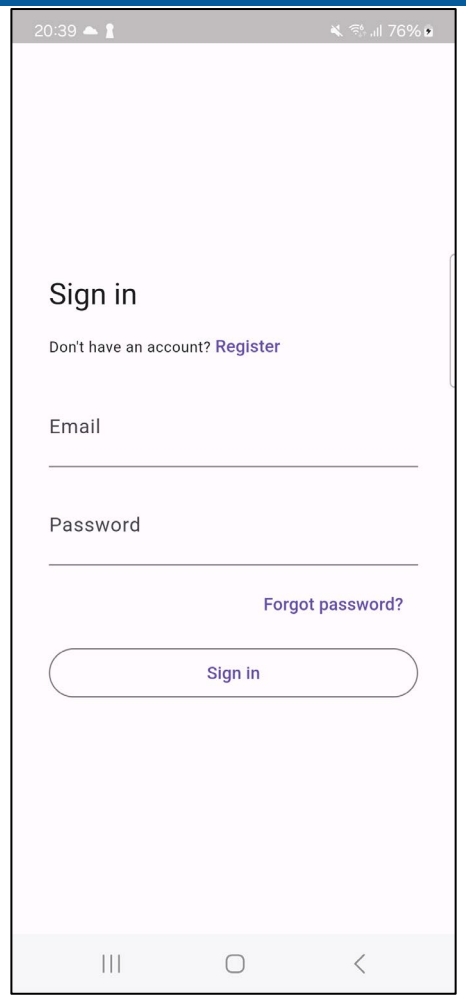
TASK 11:

- Reads from **fares** collection in Firestore Database and displays information in a StreamBuilder in UI.

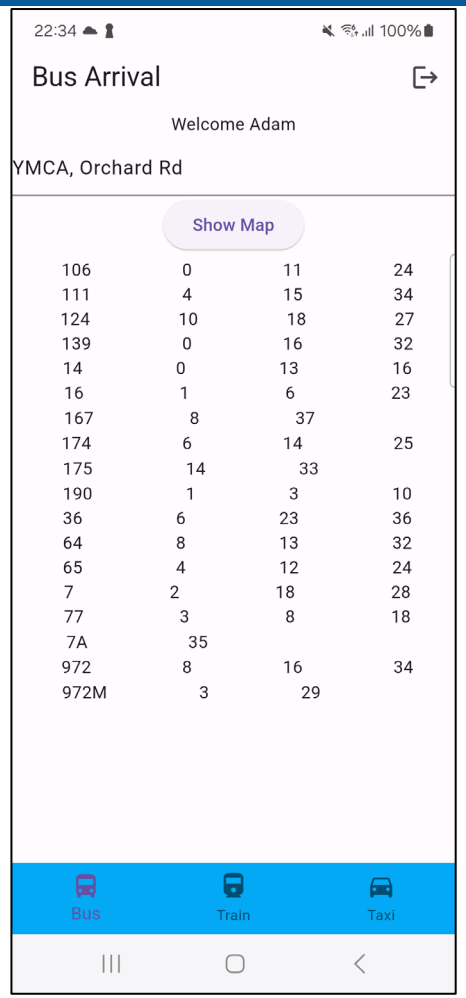




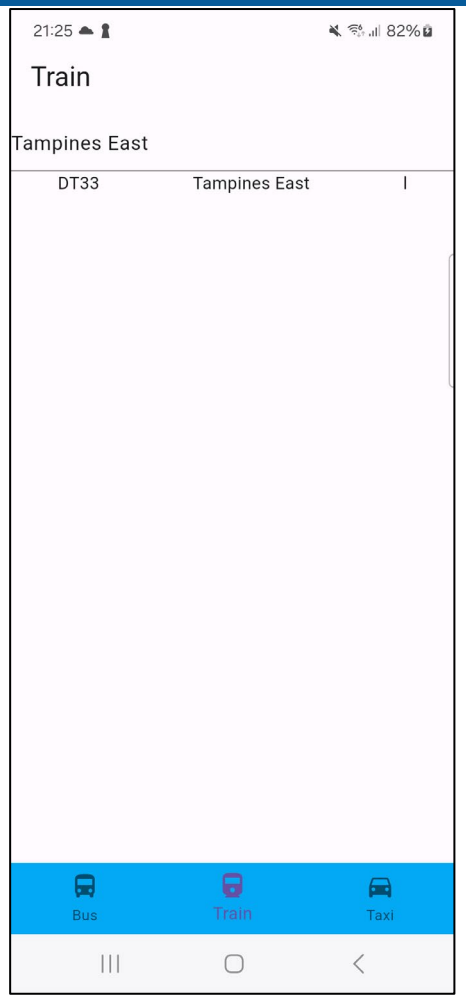
Minimum Requirement



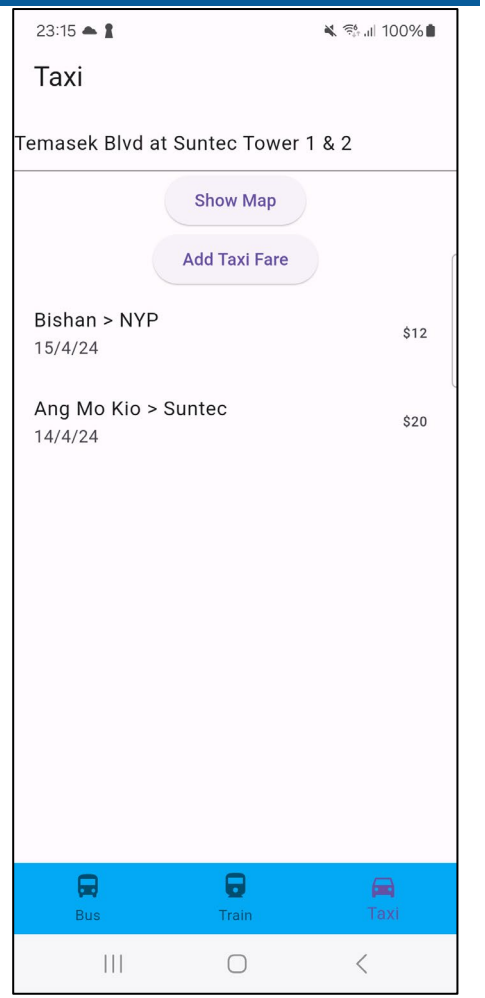
LoginScreen()



BusScreen()



TrainScreen()



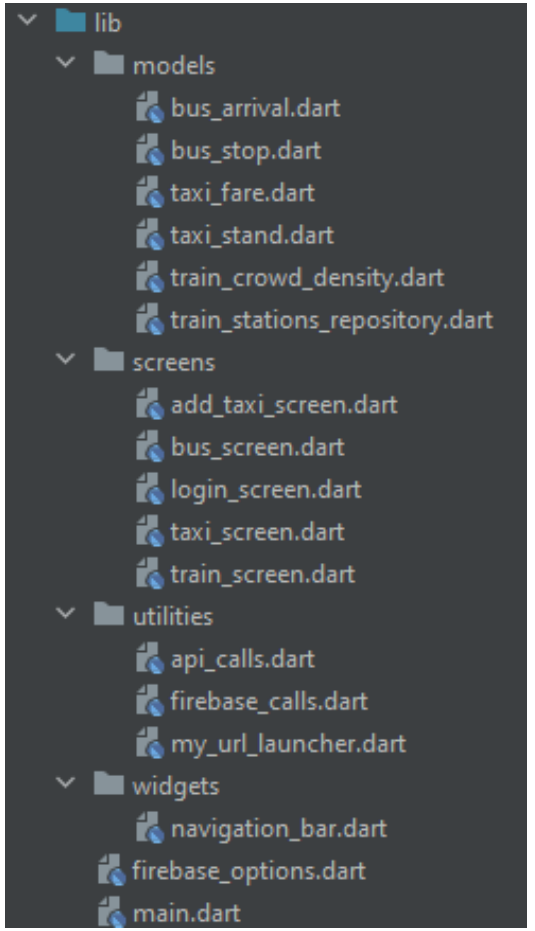
TaxiScreen()

Project Structure

Project Structure

- Basic Project Structure

models > bus_arrival.dart	**BusArrival** model class which contains information of next 3 bus arrivals of a bus service number from http response: serviceNo, nextBus. **NextBus** model class which contains information of the next bus: estimatedArrival (in minutes), load, feature and type.
models > bus_stop.dart	**BusStop** model class which contains information about a bus stop from http response: busStopCode, roadName, description, latitude & longitude.
models > taxi_fare.dart	**TaxiFare** model class which contains information of a taxi fare to be stored in **fares** collection in Firestore Database: origin, dest, fare, date, etc.
models > taxi_stand.dart	**TaxiStand** model class which contains information about a taxi stand from http response: name, latitude & longitude.
models > train_crowd_density.dart	**CrowdDensity** model class which contains information about platform crowd density from http response: station & crowdLevel.
models > train_stations_repository.dart	**TrainStation** model class which contains information about a train station: stnCode, stnName, trainline, trainLineCode. **TrainStationsRepository** which contains information about all train stations.

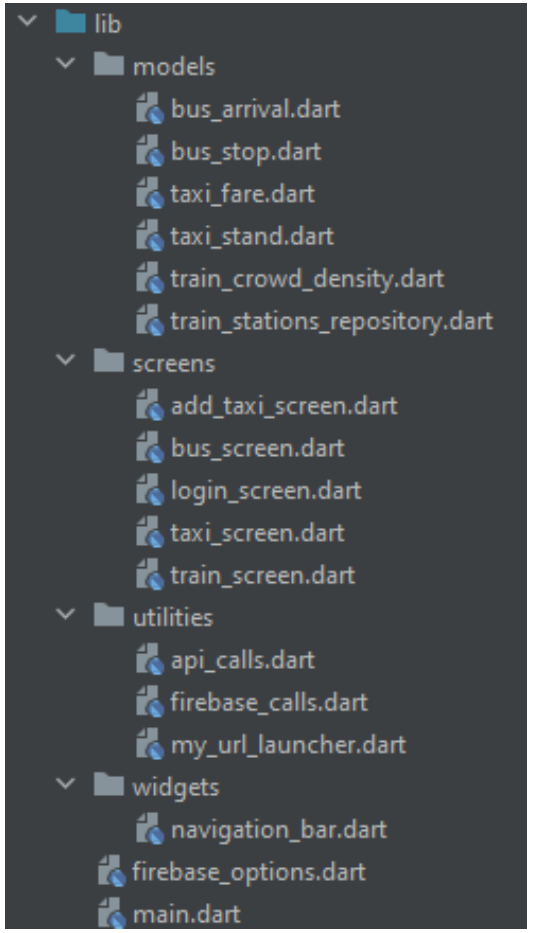


Project Structure

screens > login_screen.dart	For Firebase user authentication 1. **SignInScreen()** 2. If displayname == null > **ProfileScreen()** 3. Otherwise > **BusScreen()**
screens > bus_screen.dart	• Shows app title, user display name and logout button. • Makes API call to retrieve all bus stops. • Allows user to search for bus stop and makes API call to retrieve bus arrival information of all buses at that bus stop. • Includes button for user to view location of bus stop in Google map.
screens > train_screen.dart	• Reads from **TrainStationsRepository** to retrieve all train stations • Allows user to search for train station • Make API call to retrieve platform crowd density for that train station
screens > taxi_screen.dart	• Makes API call to retrieve all taxi stands. • Allows user to search for taxi stand and includes button for user to view location of taxi stand in Google map. • Includes button for user to add new taxi fare > **AddTaxiScreen()**. • Reads from **fares** collection in Firestore Database and displays in UI.
screens > add_taxi_screen.dart	Allows user to enter details of a taxi ride and store in **fares** collection in Firestore Database

Project Structure

utilities > api_calls.dart	Contains functions to make http api calls: fetchBusStops, fetchBusArrival, fetchCrowdDensity, fetchTaxiStands, etc.
utilities > firebase_calls.dart	Contains functions to add data to collections in Firestore Database and Firebase authentication.
utilities > my_url_launcher.dart	Contains functions to launch url to Google map.
widgets > navigation_bar.dart	Bottom navigation bar to navigate to **BusScreen()**, **TrainScreen()** and **TaxiScreen()**.
firebase_options.dart	Automatically generated file after setting up the Firebase connection
main.dart	



Additional Requirement(s)

Add AT LEAST one more API from LTA DataMall that is relevant to the SGTransport app

Rubrics

Project Rubrics (40%)

Criteria	Below Expectation	Approaching Expectation	Good	Excellent	Criterion Score
Code Functionality	3 points Some of the API calls and Firebase functions are implemented correctly, but did not address potential errors.	7 points Most of the API calls and Firebase functions are implemented correctly, but did not address potential errors.	11 points Most of the API calls and Firebase functions are implemented correctly, and made some efforts to address potential errors.	15 points All the API calls and Firebase functions are implemented correctly with care taken to address potential errors.	/ 15
Code Efficiency	3 points Poor code efficiency with long blocks of repeating codes, little use of functions and classes, inappropriate data types, etc.	7 points Attempt to optimize codes through minimal use of reusable components, functions, classes, appropriate data types, etc.	11 points Good code efficiency with reusable components, functions, classes, appropriate data types, etc.	15 points Excellent code efficiency with reusable components, functions, classes, appropriate data types, etc.	/ 15
User Interface	3 points UI was poorly designed with some missing widgets and/or wrong information from API calls and Firebase.	7 points Most of the widgets are arranged with min UI design and showed correct information from API calls and Firebase.	11 points Most of the widgets are arranged in good layout with customized UI design and showed correct information from API calls and Firebase. Incorporated the use of customizable UI widgets from Flutter library.	15 points Excellent use of customizable UI widgets from Flutter library to create stunning UI with correct information from API calls and Firebase.	/ 15
User Experience	3 points Poor user experience without visual feedback & did not use graphics, colors, styling, global themes, animations, reduce excessive widget rebuilds, etc.	7 points Satisfactory user experience with visual feedback & some use of graphics, colors, styling, global themes, animations, reduce excessive widget rebuilds, etc.	11 points Good user experience with visual feedback & appropriate use of graphics, colors, styling, global themes, animations, reduce excessive widget rebuilds, etc.	15 points Excellent user experience with visual feedback & good use of graphics, colors, styling, global themes, animations, reduce excessive widget rebuilds, etc.	/ 15
Additional Features	5 points Met app requirement with no enhancement features	10 points Met app requirement & incorporated minimal enhancement features	15 points Met app requirement & incorporated substantial enhancement features	20 points Exceeded app requirement & incorporated substantial enhancement features	/ 20
Q & A	5 points Able to answer some questions.	10 points Able to answer most questions with short answers.	15 points Able to answer most questions with elaboration.	20 points Able to answer all questions with elaboration.	/ 20
Total					/ 100

Project Presentation Rubrics (10%)

- Present key features of your mobile app in 10 mins
- Include screen captures / videos and code snippets in Powerpoint slides

Criteria	Below Expectation 10 points	Approaching Expectation 15 points	Good 20 points	Excellent 25 points	Criterion Score
Presentation Delivery	Audience has difficulty following presentation and flow of information can be improved.	Audience is able to follow presentation which is delivered well but too heavily scripted.	Audience is able to follow presentation which is delivered well and smoothly.	Presentation is interesting, eloquently delivered and with enthusiasm.	/ 25
Presentation Preparedness	The script is unprepared and not synchronized and coherent with the slides. There are many moments of silence and nervous mannerisms during the presentation.	The script is prepared but not synchronized and coherent with the slides. There are some moments of silence and nervous mannerisms during the presentation.	The script is prepared and fully synchronized and coherent with the slides. There are no moments of silence but some nervous mannerisms during the presentation.	The script is prepared and fully synchronized and coherent with the slides. There are no moments of silence and the presentation was delivered confidently.	/ 25
Content Organization	Lack of a logical sequence of information and the presentation is unclear and disorganized. No emphasis on key parts of the presentation.	Logical sequence of information and the presentation is unclear and disorganized. Some emphasis on key parts of the presentation.	Logical sequence of information and the presentation is clear and organized. Some emphasis on key parts of the presentation.	Logical sequence of information and the presentation is clear and organized. Strong emphasis on key parts of the presentation.	/ 25
Content Visual Aids	Little attempt to capture audience's attention. Visual aids are present but irrelevant to the presentation.	Good attempt to capture audience's attention. Visual aids are present and relevant to the presentation but not used effectively to drive the points across.	The team is able to capture audience's attention. Visual aids are present and relevant to the presentation and are used effectively to drive the points across.	The team successfully used unique strategies to capture audience's attention. Visual aids are very interesting and used effectively to excite and inform the audience about the project.	/ 25