



**CMPE-277**

Individual Android App

**Smart Bus Tracker**

**Submitted To**

Prof. Chandrasekar Vuppalapati

**Date of Submission**

26th April 2017

**Submitted By**

Sih-Han Chen - 011498552

|  |           |
|--|-----------|
| <b>Learning Objective:</b>                             | <b>3</b>  |
| <b>GitHub:</b>   | <b>3</b>  |
| <b>1. The Main Function of this App</b>                | <b>4</b>  |
| <b>2. Architecture Design</b>                          | <b>5</b>  |
| <b>3. Bus Client Android App Design</b>                | <b>6</b>  |
| 3.1 Activity - Bus Line & Stop Selection               | 6         |
| 3.2 Activity - Bus Tracker Map                         | 7         |
| 3.3 Use Case - Passenger Waiting, Get On, Get Off bus. | 10        |
| <b>4. Bus Simulator Server Design</b>                  | <b>13</b> |
| 4.1 Backend Dashboard                                  | 14        |
| 4.2 Real Time Bus Tracking Map                         | 15        |
| <b>5. Reference</b>                                    | <b>16</b> |

# Learning Objective:

Individual Android App

Learning objective:

Apply class learnings - storage, services, intents, sensors, maps, and others to create one utility App

Think creatively and develop the App

Must upload

Source code

Read me document

Architecture and report document

---

## GitHub:

[https://github.com/stephen-sh-chen/CMPE277\\_Smartphone\\_App/tree/master/FinalAndroidApp](https://github.com/stephen-sh-chen/CMPE277_Smartphone_App/tree/master/FinalAndroidApp)

# 1. The Main Function of this App

This Bus Tracker App can help user to know the real time position of a moving bus and can notify users when the bus which they are going to take is coming. Also it can inform user to get off the bus when the destination stop is arriving. Therefore, passengers can fully control the bus route even they are in a new city or country. As the Figure 1 shown, it is the two main Activity in this App. We will present the detailed design and implementation in the rest of this document.

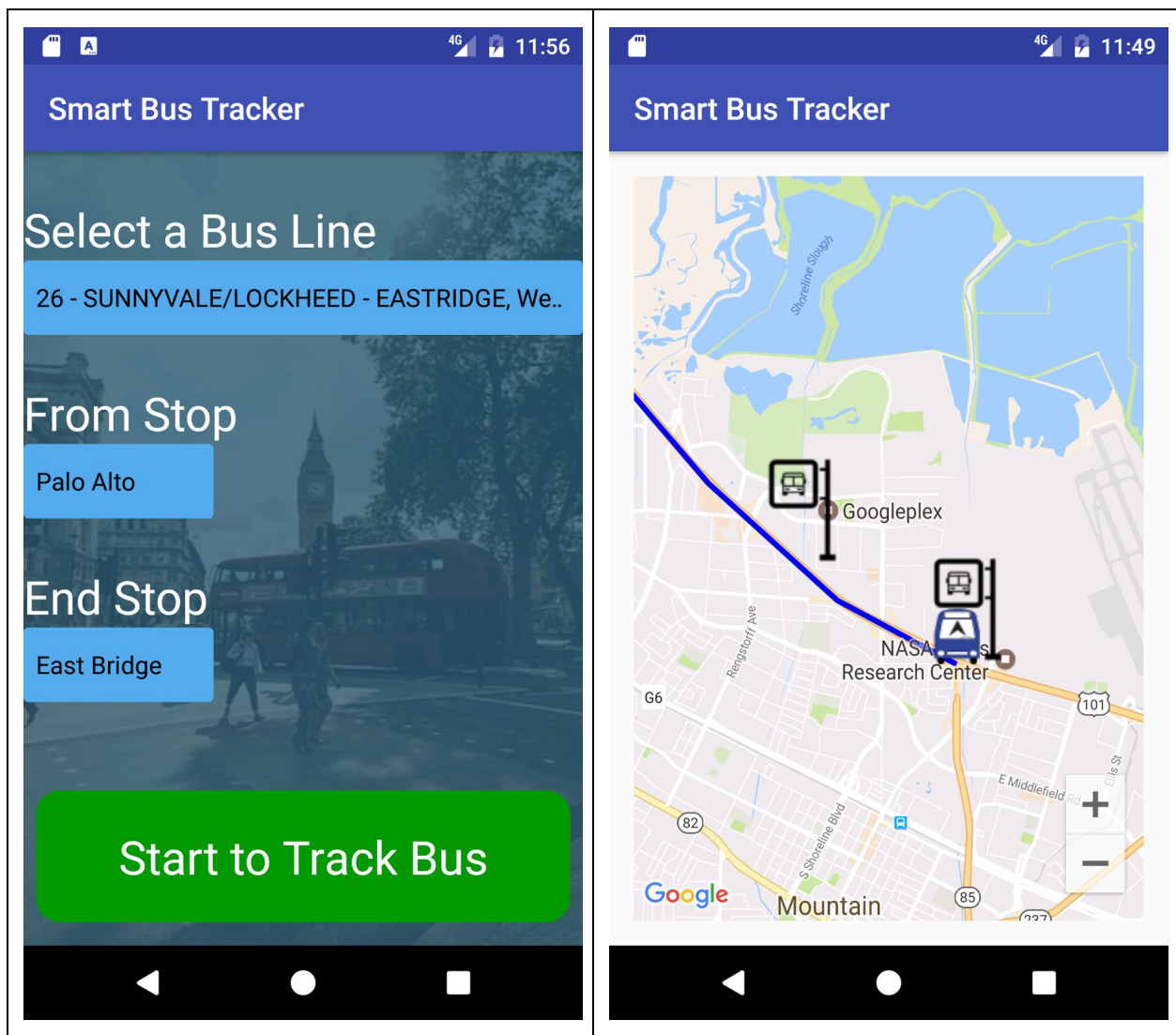
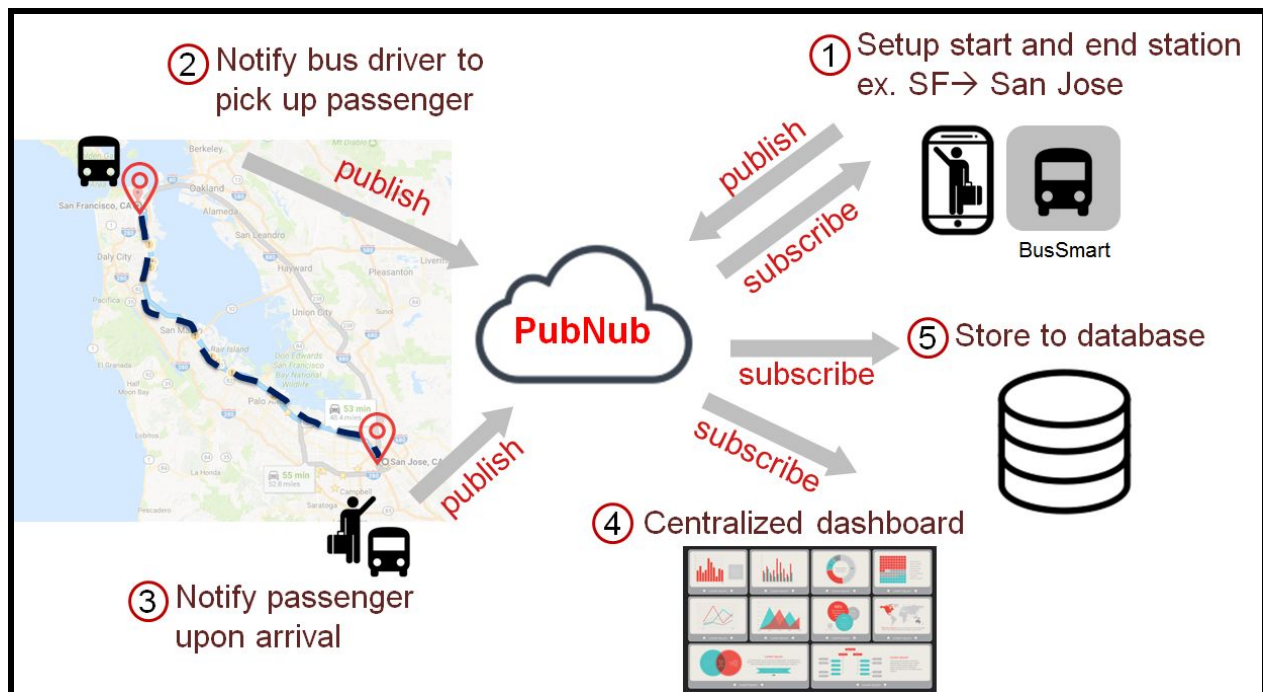


Figure 1. The overview of this App GUI

## 2. Architecture Design

The following figure is our system architecture. We use the PubNub API to be our message queue in our system. The main responsibility of PubNub is to receive and forward the real time geolocation information of the moving bus. The scenario is that passengers use this App to choose the bus line, and the starting/ending bus stop they want. And users can quickly get information of the bus real position. User will get notification when the bus is coming. Another feature is that once users enter the bus, they will receive another notification when the destination is coming. The most important part is that user don't need to depend on the GPS sensors to get the bus position. We use the PubNub to publish the bus location to the user App. Therefore, we can get a maximum powers saving by this architecture.



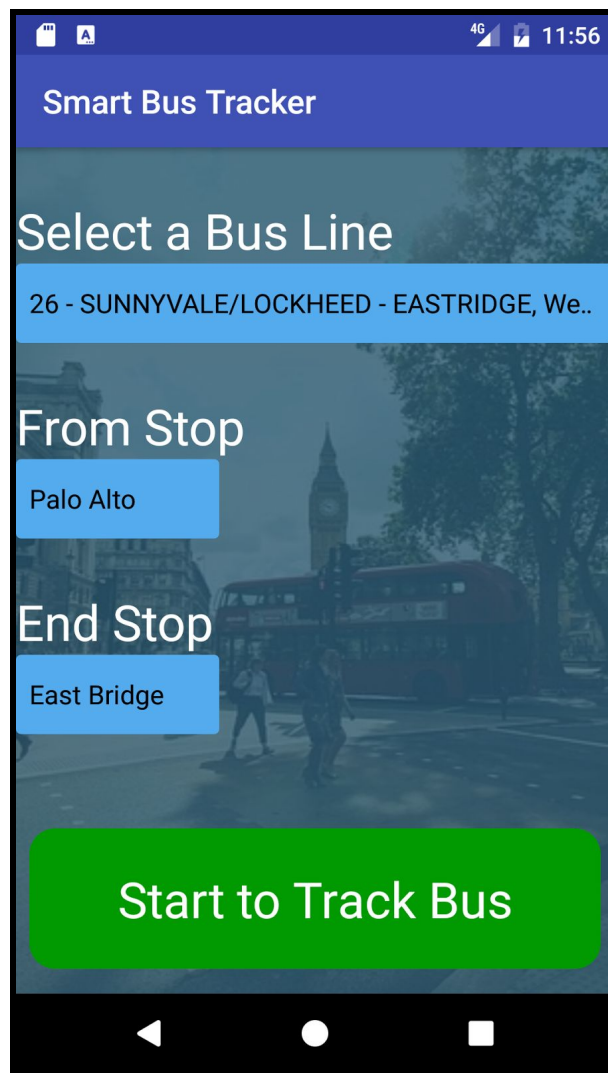
**Figure 2.** System Architecture Diagram

## 3. Bus Client Android App Design

In this section, We will give an introduction of each Activity in Section 3.1 and Section 3.2. Then a normal use case will be presented in Section 3.3 about how to use this App.

### 3.1 Activity - Bus Line & Stop Selection

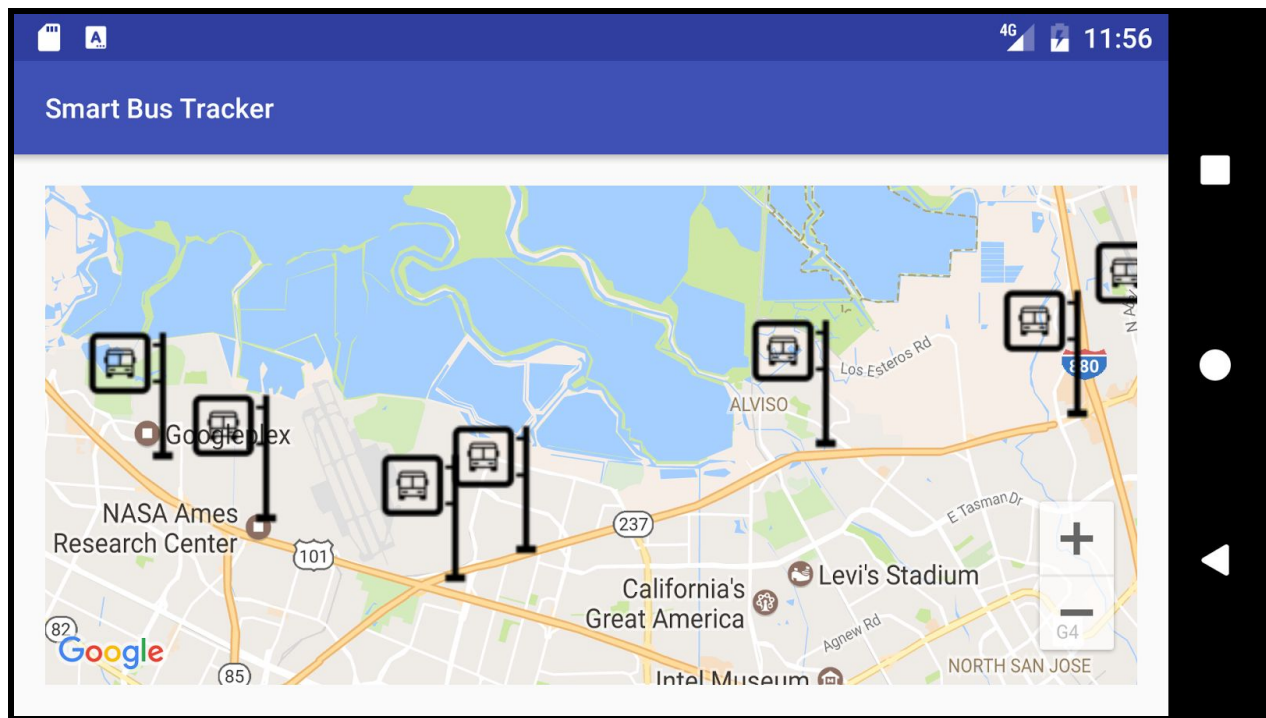
Figure 3 is our first Activity. In this Activity, user can select the bus line they are going to take. Then user can pick up one of the stop of this bus line they want to get on. The third step is to choose the destination bus stop. After the determination of the bus information, user just click the button “Start to Track Bus” to get how far the bus from them.



**Figure 3.** Activity - Bus Line & Stop Selection

## 3.2 Activity - Bus Tracker Map

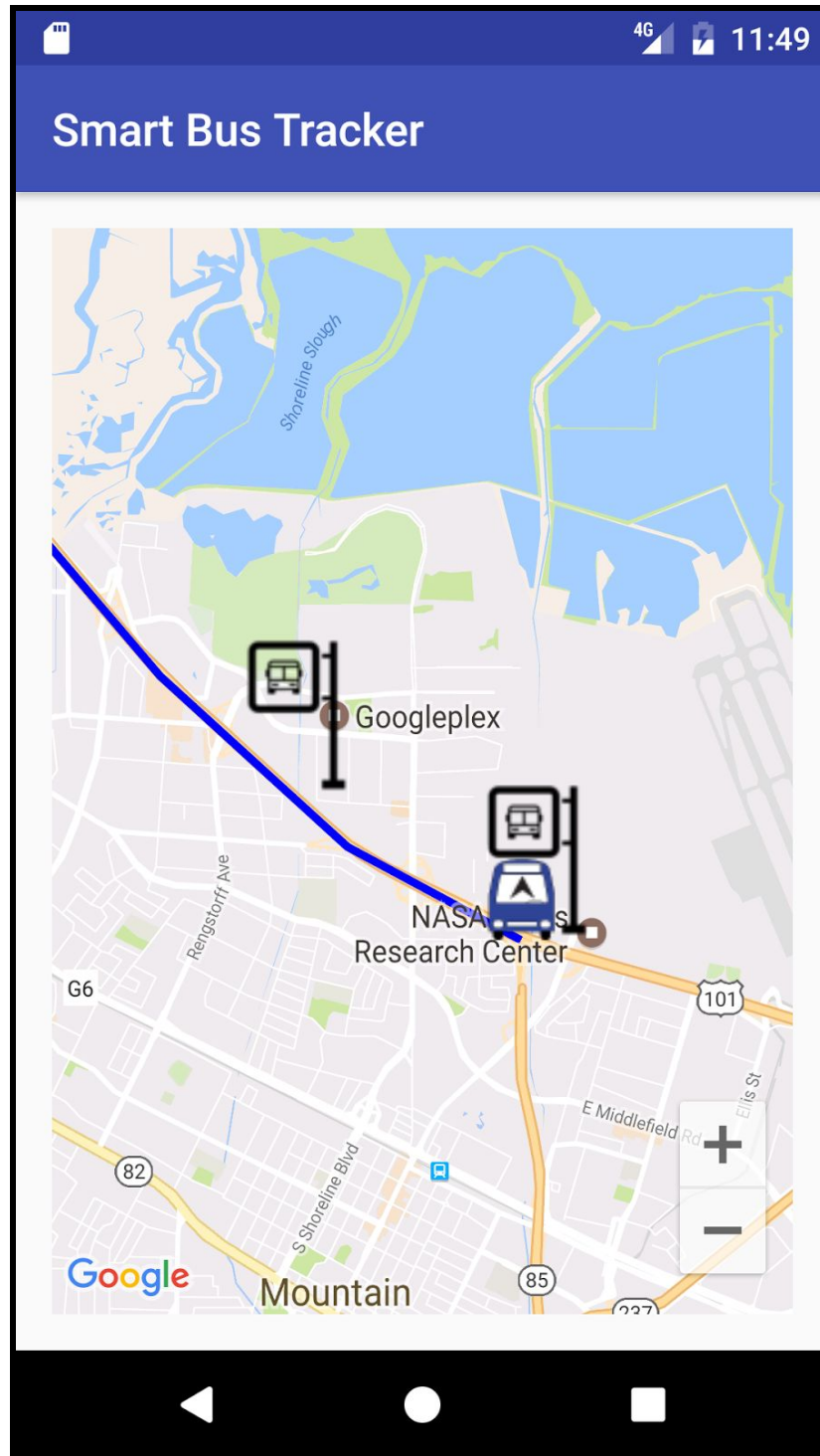
In this section, we will see the exactly GUI and function of the map activity. Figure 4 is the landscape orientation view for this map activity. Users can clearly see all the bus stops for the bus they choose.



**Figure 4.** Show All bus stops on the map for a specific bus line



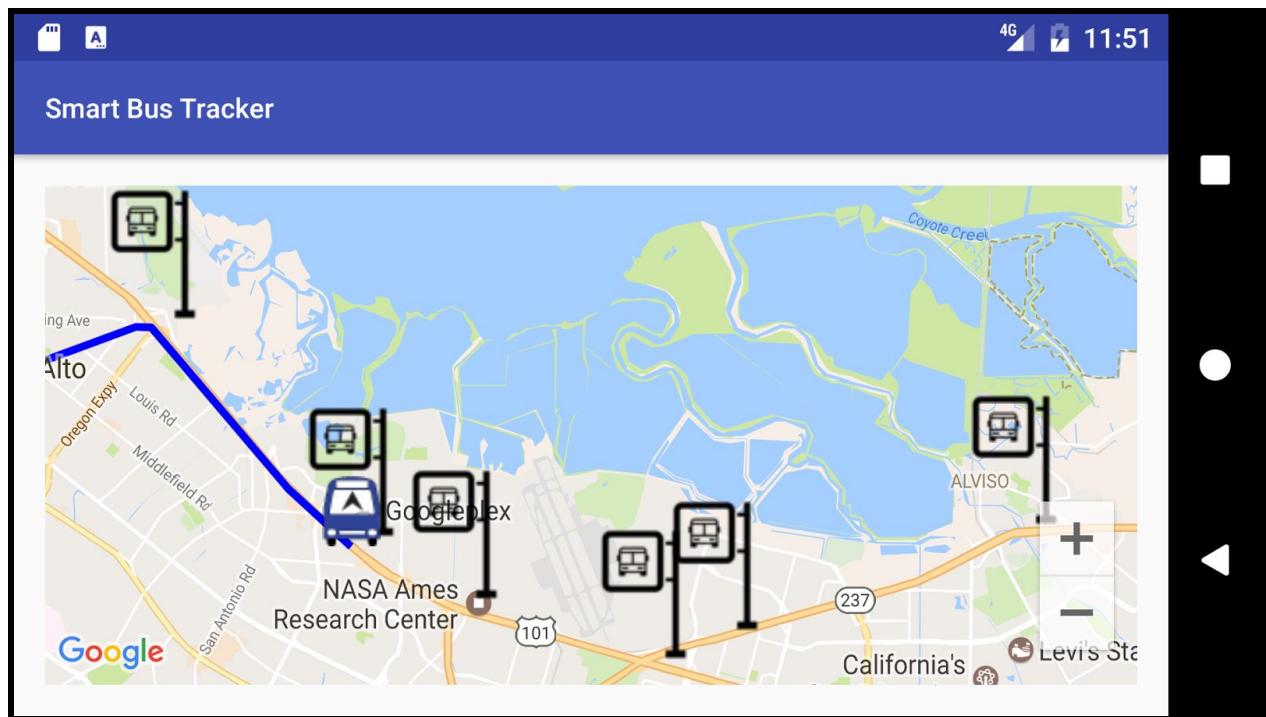
Figure 5 is the portrait orientation view of this map activity. Also user can easily to get the bus stop location and the real time bus position in this map activity.



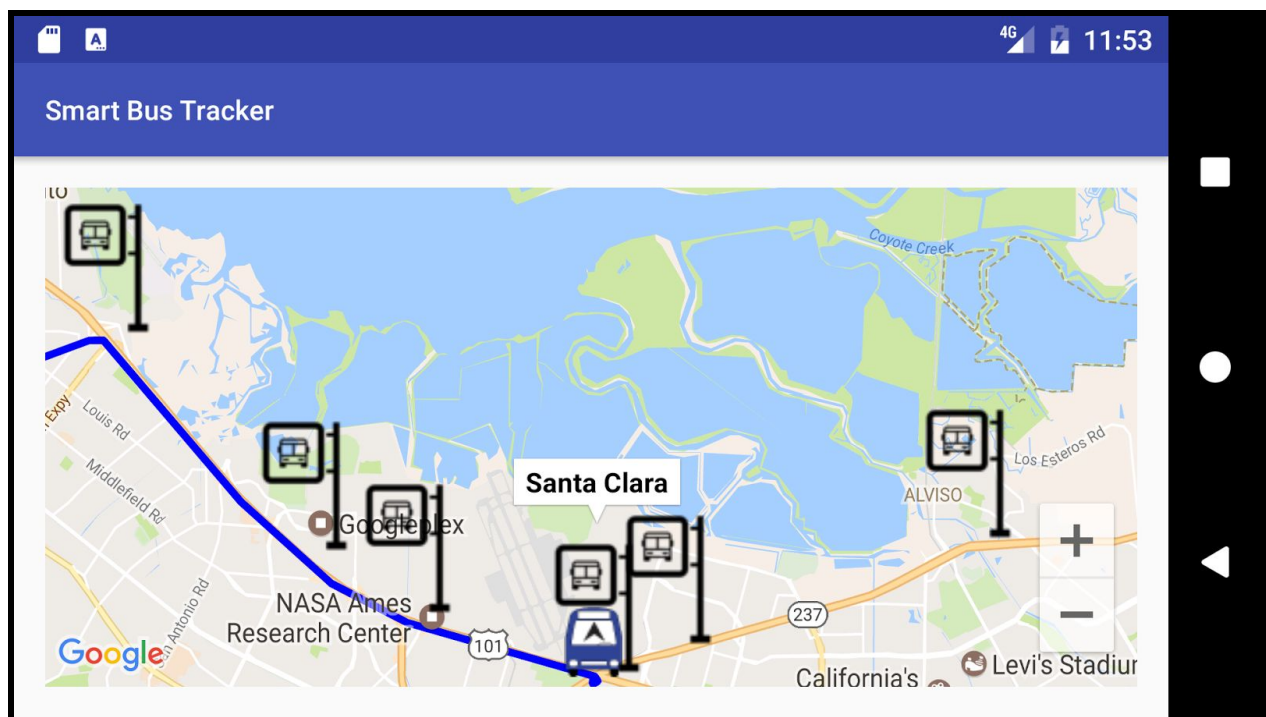
**Figure 5.** Real Time Position of the moving bus



In Figure 6 and Figure 7, users can know the real time bus position and each stop's information.



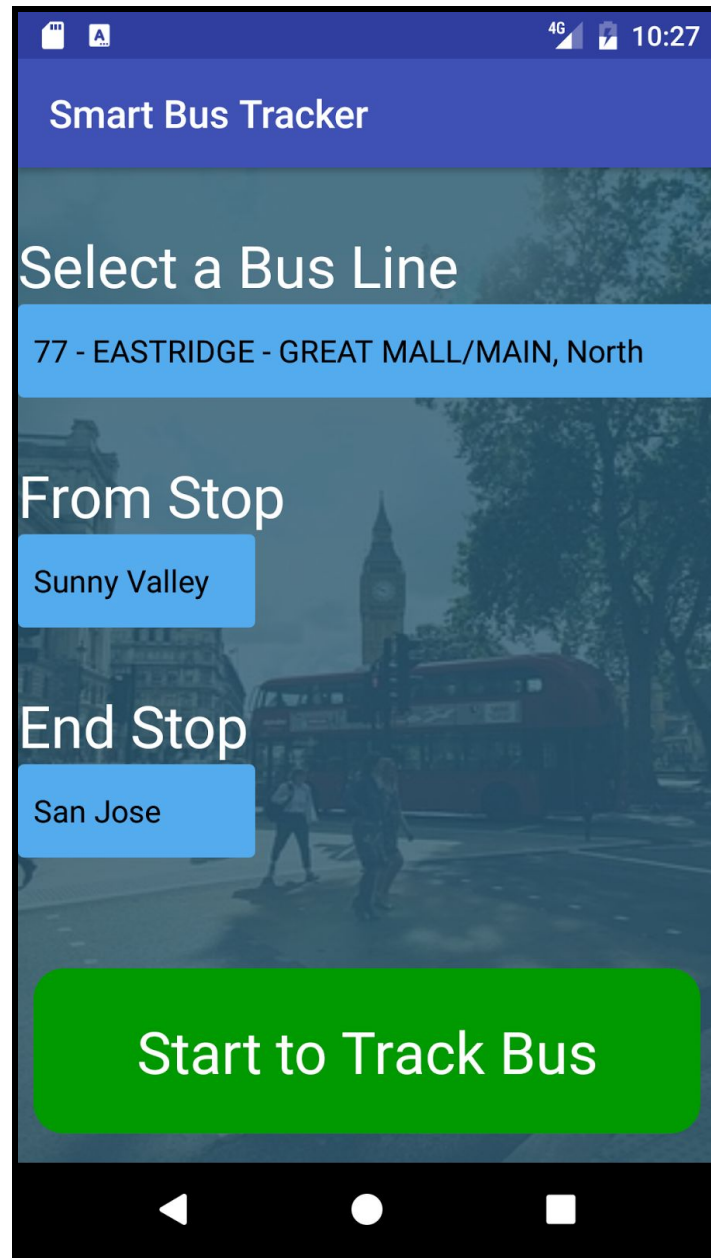
**Figure 6.** Real Time Position of the moving bus



**Figure 7.** Real Time Position of the moving bus

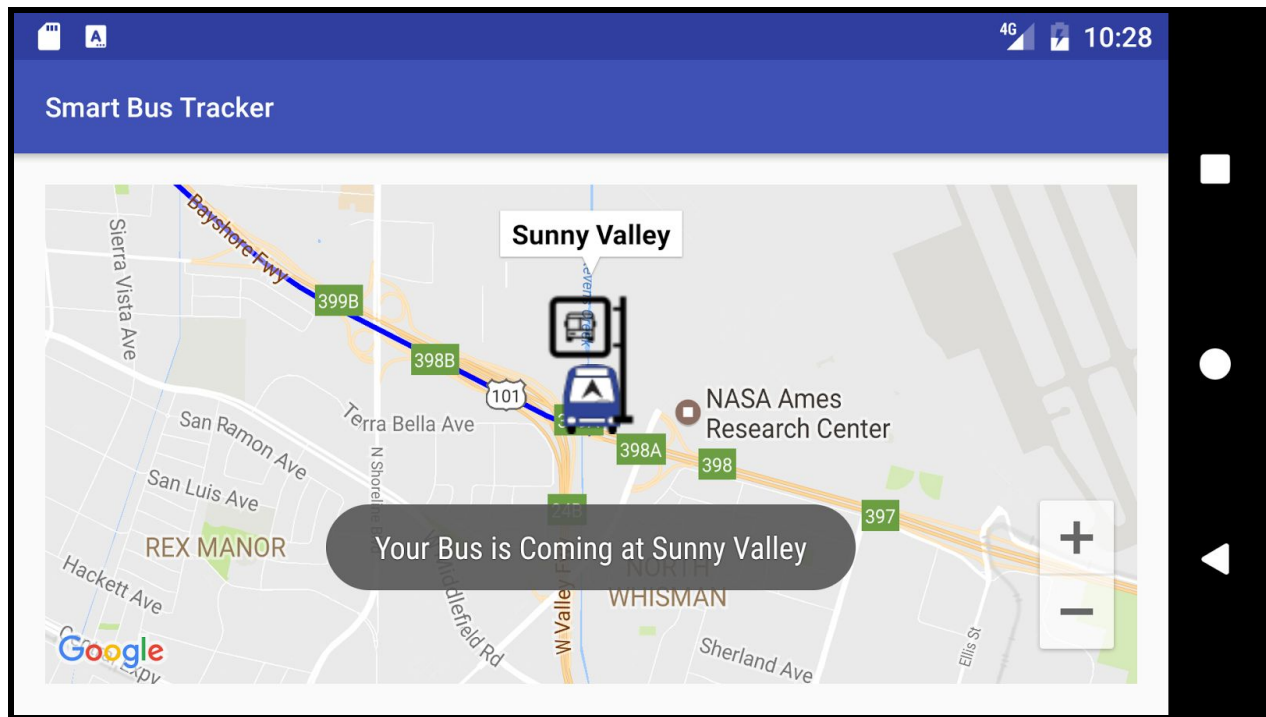
### 3.3 Use Case - Passenger Waiting, Get On, Get Off bus.

Imagine that you are going to take a bus from Sunny Valley to San Jose by bus line 77. The first step is to select the bus line as 77 in this App. Second, choose the starting bus stop you want to get on the bus. Third, choose the end stop you want to get off the bus. The last step is to click the “Start to Track Bus” button.



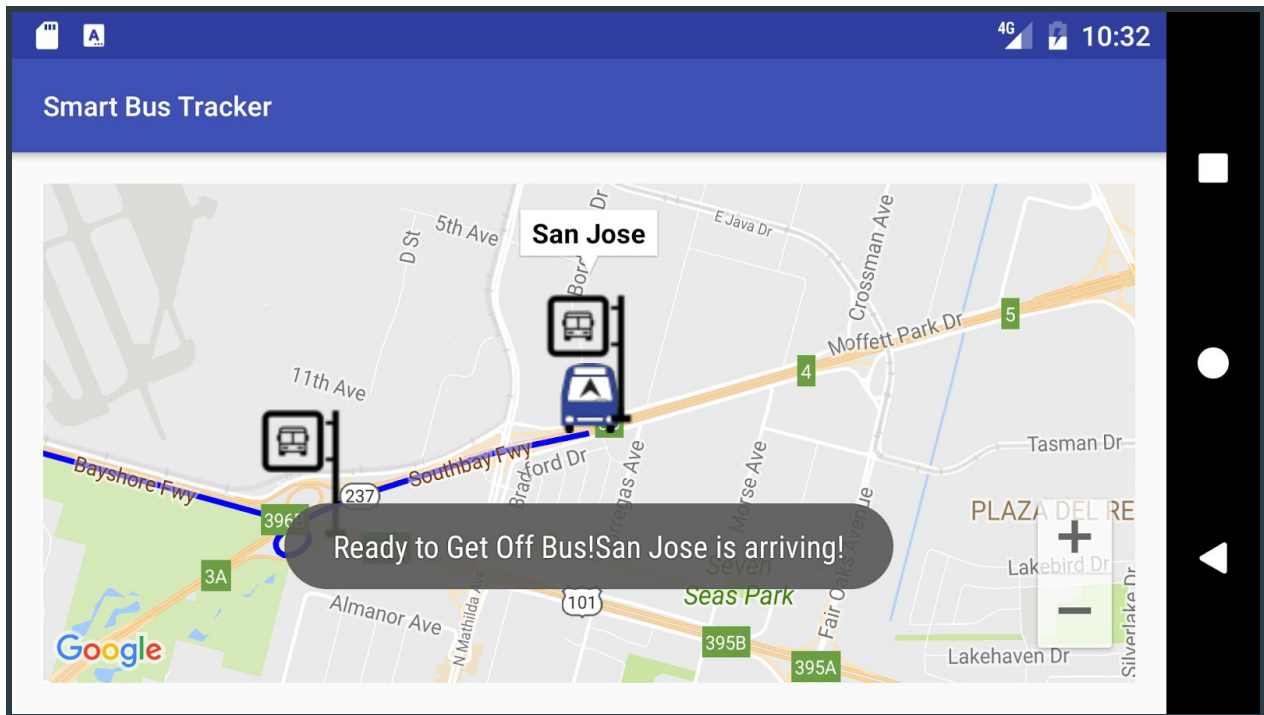
**Figure 8.** Enter the Bus information in the App when waiting the bus

Once click the button on the previous step, the screen will switch to the map activity. You can easily get the real position of the bus and all the bus stop information. When the bus is coming to the bus stop that you are waiting, the app will give you a notification.



**Figure 9.** Notify user when bus is coming

As the Figure 10 shown, when the destination bus stop is arriving, this app will notify you to prevent missing your destination.

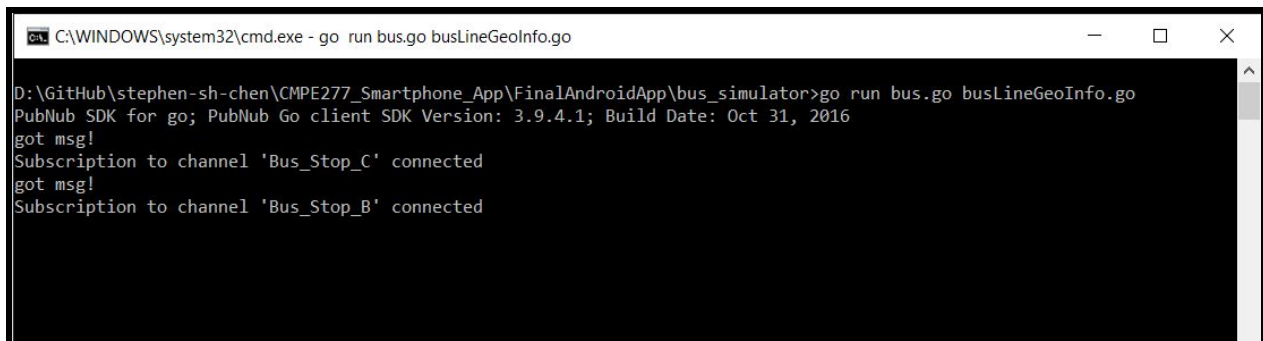


**Figure 10.** Notify user to get off the bus when the destination is arriving

## 4. Bus Simulator Server Design

In the section, we will introduction our backend bus simulator server. Since it is hard to get the GPS information from the bus vendor and in order to verify our client Android App, I made this bus simulator system to simulate the behavior of a real bus.

Please note! This simulator is programmed by Golang, that means you need to create a Golang compiler environment before you are going to run it. After your Golang programming environment is ready, please enter the “bus\_simulator” folder and type the command “go run bus.go busLineGeoInfo.go” to launch this service in the command window.



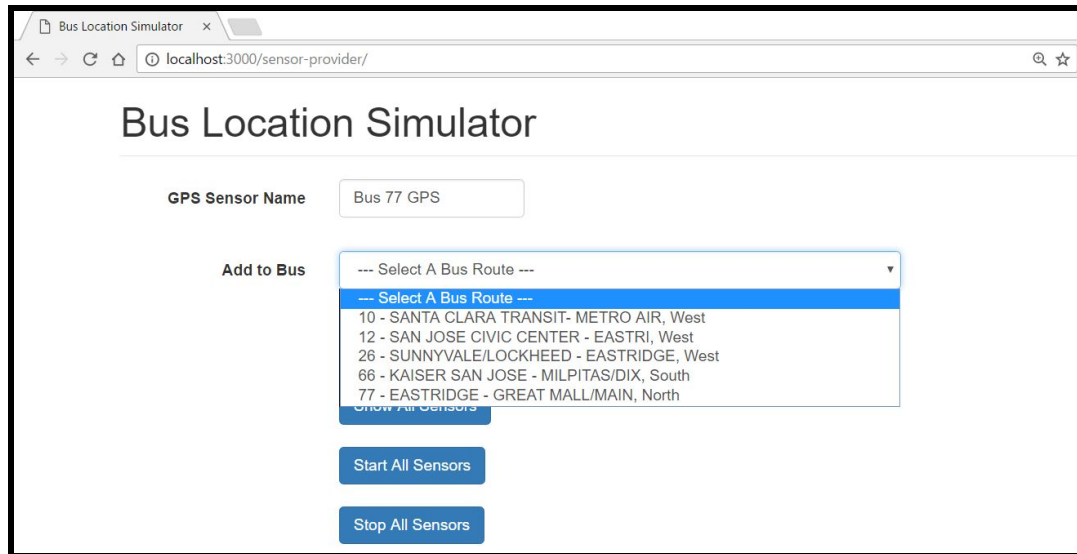
```
C:\WINDOWS\system32\cmd.exe - go run bus.go busLineGeoInfo.go

D:\Github\stephen-sh-chen\CMPE277_Smartphone_App\FinalAndroidApp\bus_simulator>go run bus.go busLineGeoInfo.go
PubNub SDK for go; PubNub Go client SDK Version: 3.9.4.1; Build Date: Oct 31, 2016
got msg!
Subscription to channel 'Bus_Stop_C' connected
got msg!
Subscription to channel 'Bus_Stop_B' connected
```

Figure 11.

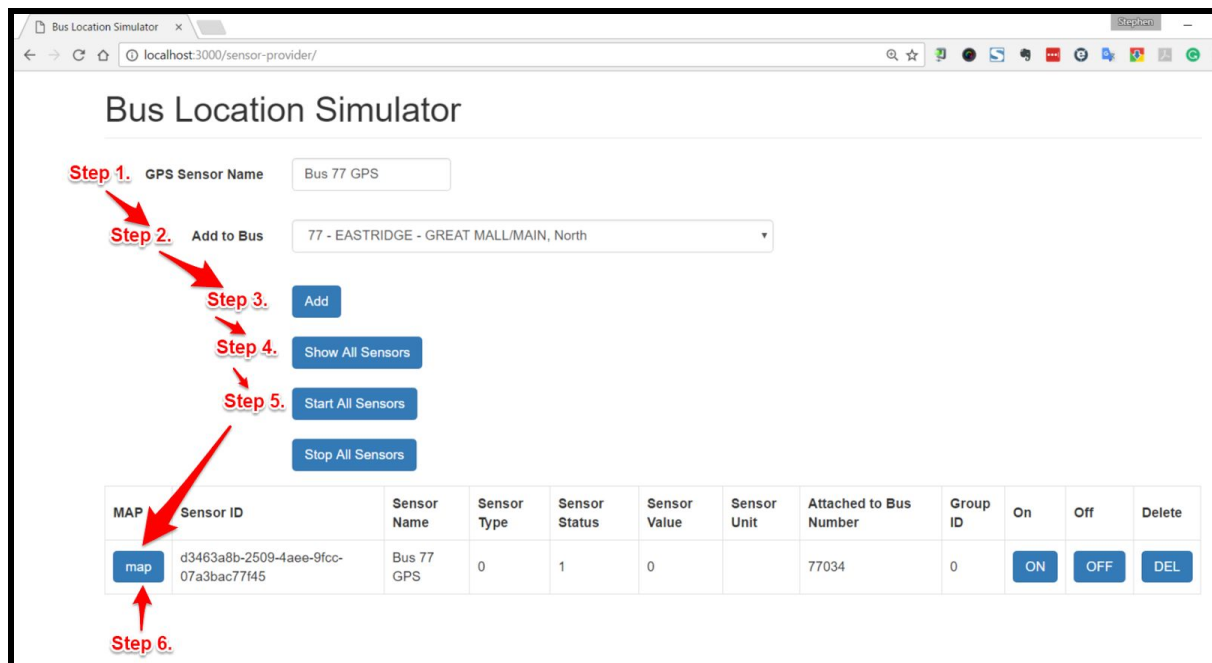
## 4.1 Backend Dashboard

After the bus simulator is launched, please open your browser and go to this URL “<http://localhost:3000/sensor-provider/>” to open the dashboard.



**Figure 12.** Bus Location Simulator Dashboard

Please follow the steps in the Figure 13 to add a new bus in the simulator.

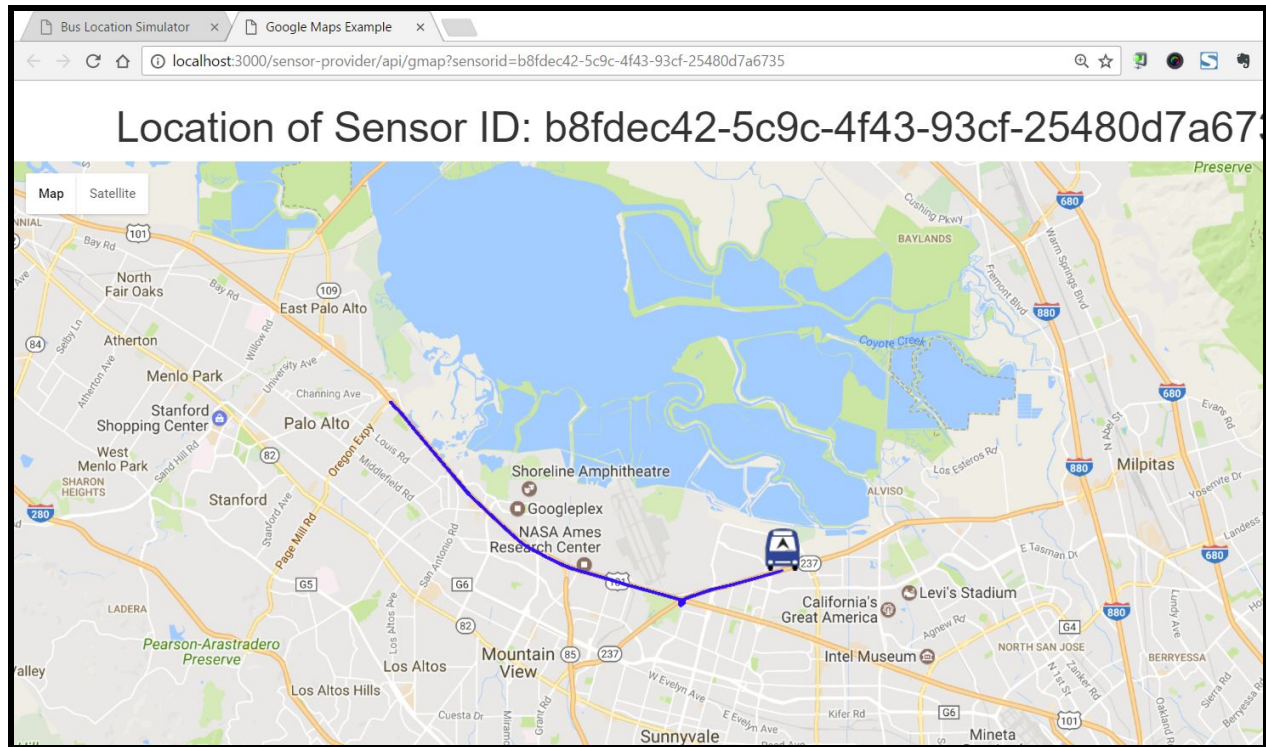


**Figure 13.** Bus Location Simulator Dashboard II



## 4.2 Real Time Bus Tracking Map

After create a new bus in the simulator, you can click the map button which show in the step 6 of Figure 13. It will open a new web to present the position of the moving bus in google map.



**Figure 14.** The real time bus location from the Bus Simulator



## 5. Reference

- [1] <https://developers.google.com/maps/documentation/android-api/start>
- [2] <https://developer.android.com/reference/org/w3c/dom/Document.html>
- [3] <https://www.pubnub.com/docs/android-java/pubnub-java-sdk>