Shuttle Track

Project - Shuttle Track App

(University shuttle tracking through crowd sourcing)

Status : Complete

Sonali Sahu

wb8997

CS4521 - Mobile and Topics in Web

ssahu2@horizon.csueastbay.edu

Project - Shuttle Track App

Name - Sonali Sahu

Email - ssahu2@horizon.csueastbay.edu

Section 0

Introduction and Proposal:

This is developed app from the needs of personal experience. Many a times I have been waiting for the shuttle in the stop and at times wait for hours before the shuttle turn up. Even though we have the schedule for the shuttle, the students at many times due to delay of turn up of the shuttle often waste a lot of time waiting for the bus to come or in anticipating the arrival of bus.

This app will give the student the idea of where their intended shuttle is and how much time will it take to arrive at the nearest stop to the student. The students will have an option to report whether they have boarded the shuttle or that the shuttle is delayed.

Through crowd sourcing the location of the shuttle is depicted.

Audience:

This app is aimed at the college students or staff or basically everyone who uses the CSU East Bay college shuttle.

Application Cost and Projected Success: Since this app is for students, I would charge a minimal fee of 0.99 cents. As this app will try to help students to know exactly how much time they must wait it will save a lot of their time and energy. If they find out that a scheduled shuttle is taking more than expected time then they will freely look for other options in case they don’t want to wait.

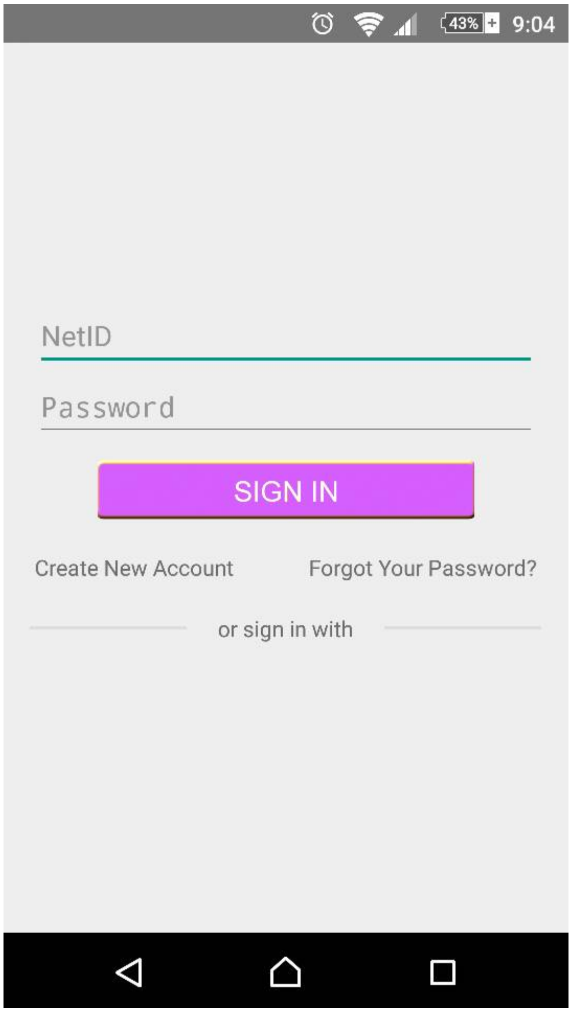
Monetization:

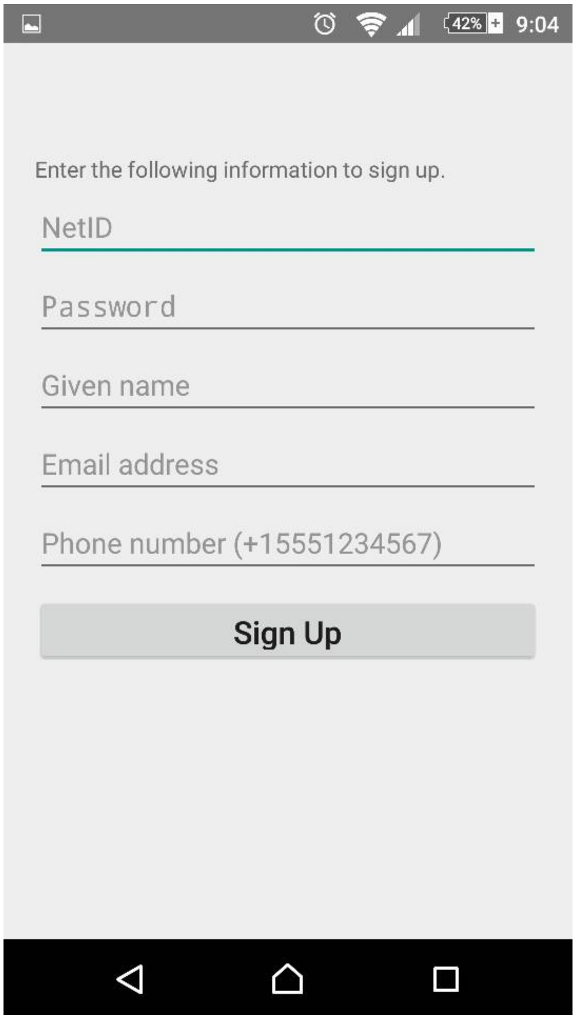
To earn revenue, I have thought of 2 ways.

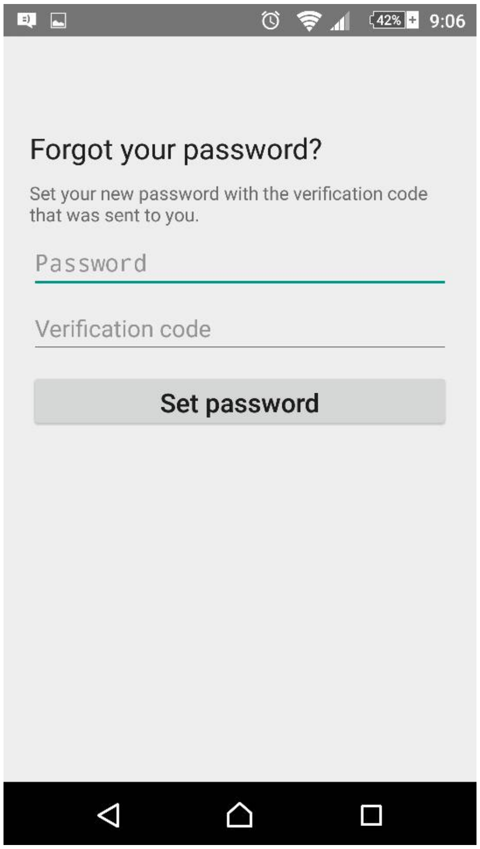
1. Advertising: I have placed in-app advertisements through which I can collect money from advertisers.

2. College Funding: since I would be developing an app for the college students I would try to convince the college body or ASI to annually provide funds to my applications so that I could maintain and improve its features which will help to serve the students better.

Interface Mockups:




Enter the following information to sign up.

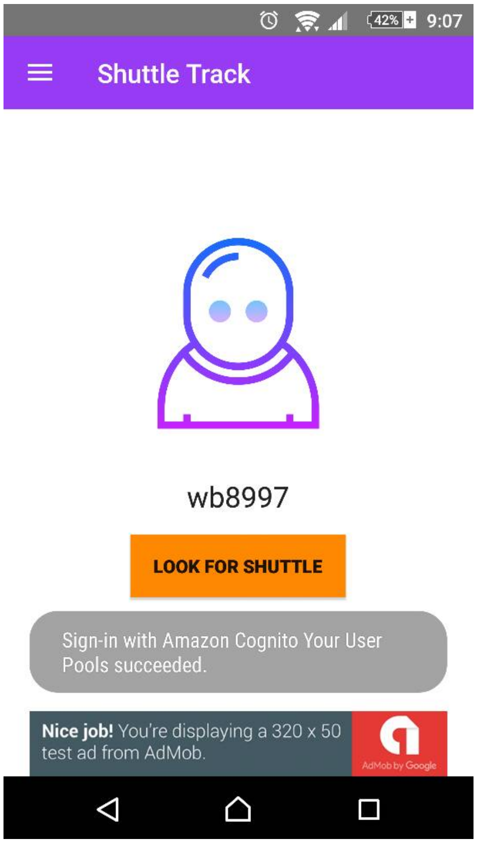


Forgot your password?

Set your new password with the verification code that was sent to you.

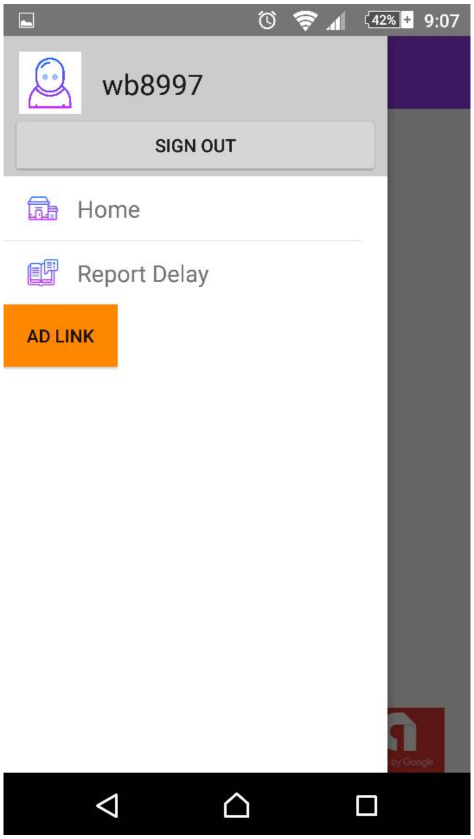
Password

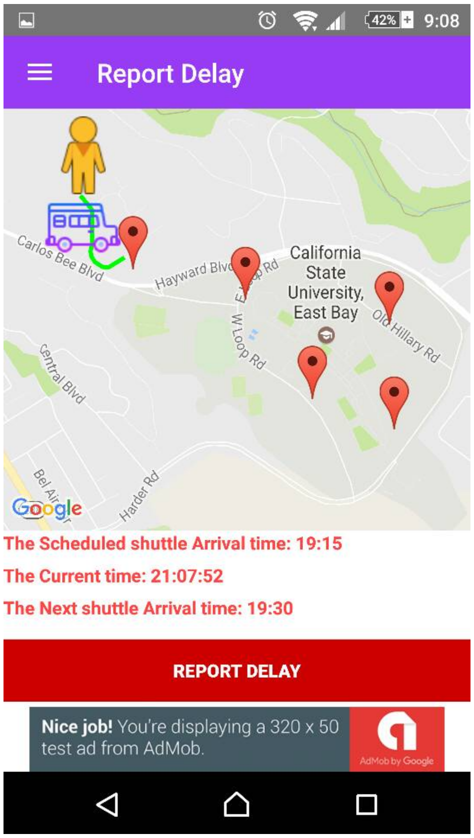
Verification code



wb8997

Sign-in with Amazon Cognito Your User Pools succeeded.

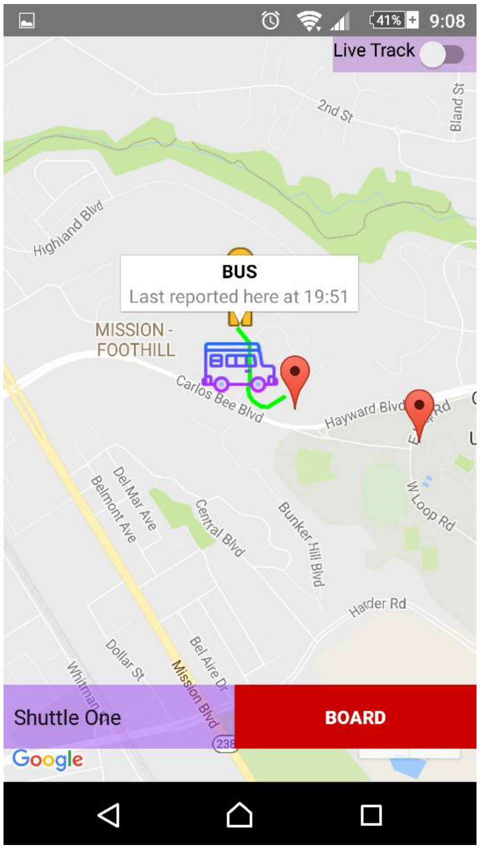


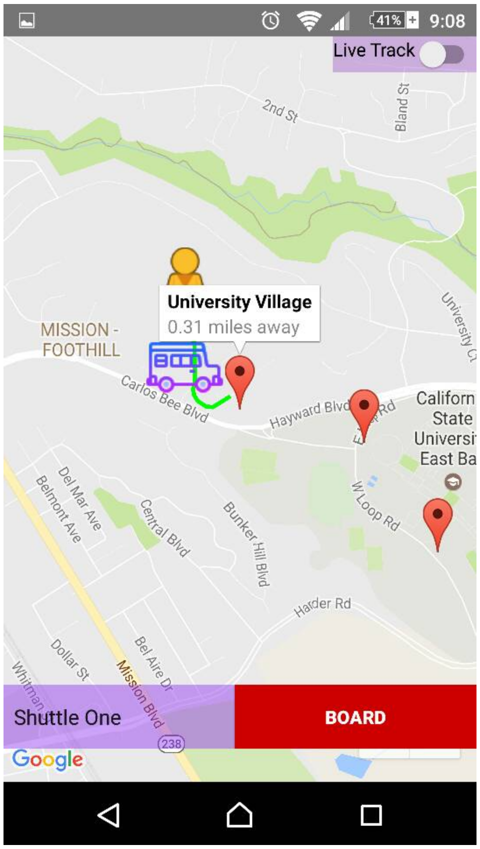


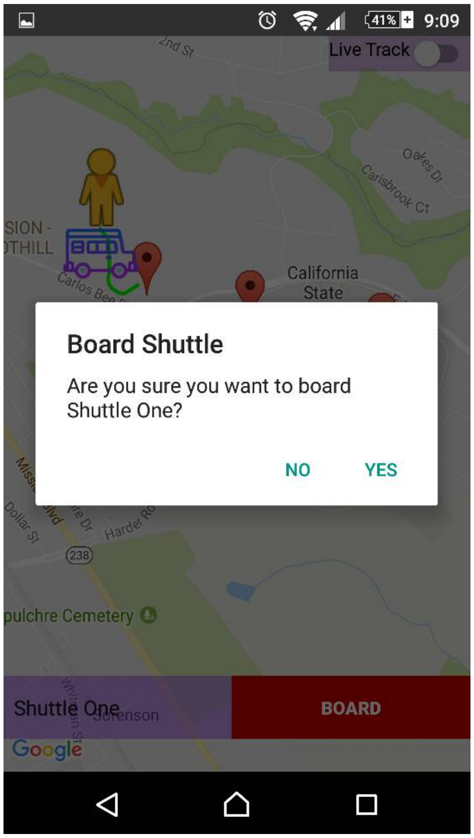
The Scheduled shuttle Arrival time: 19:15

The Current time: 21:07:52

The Next shuttle Arrival time: 19:30



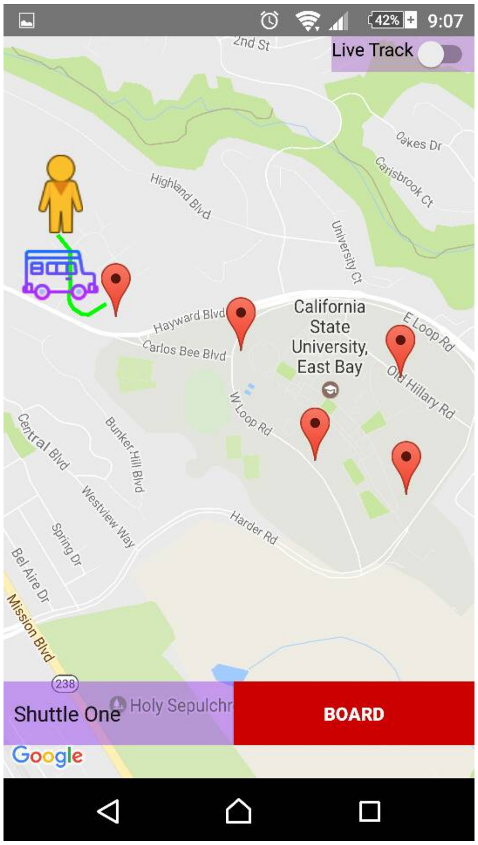




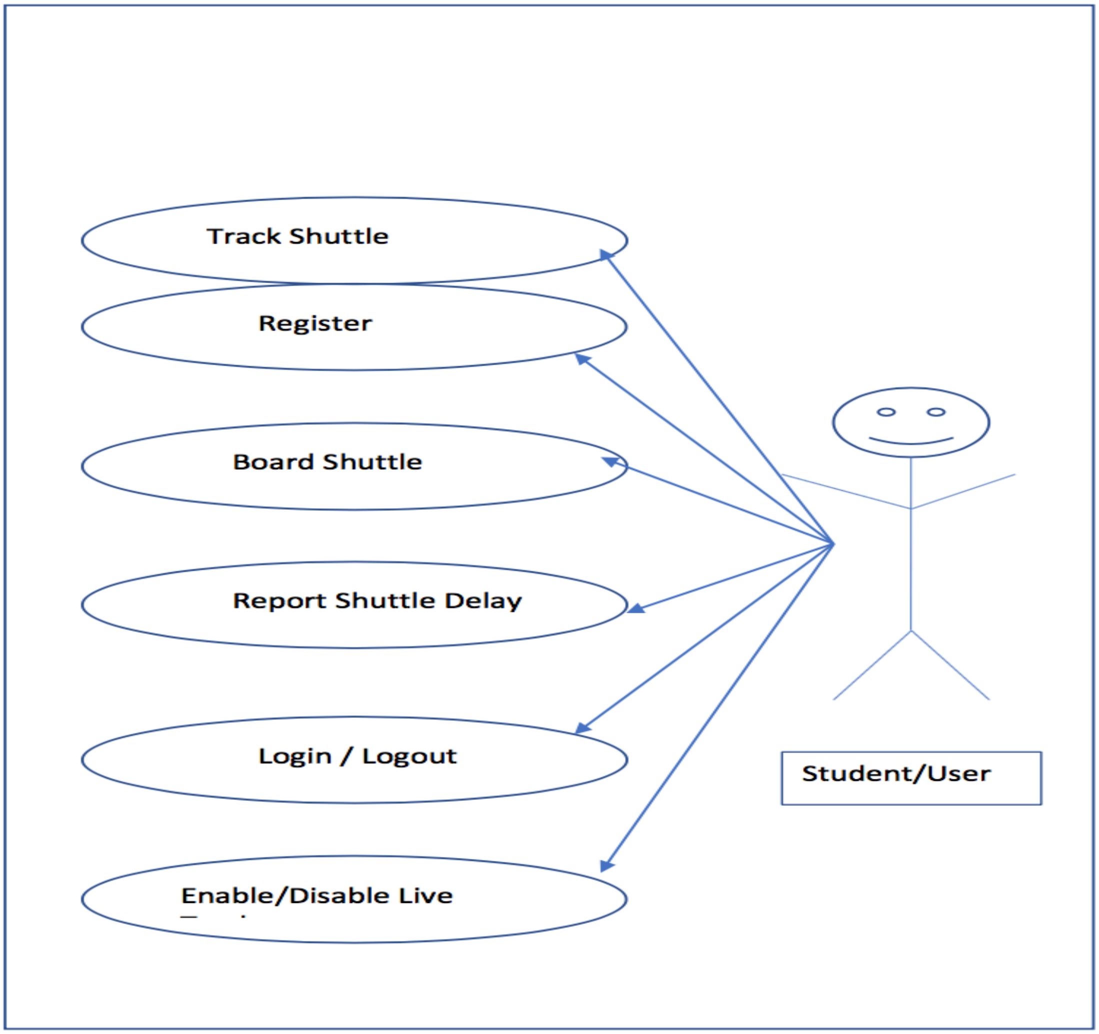
Board Shuttle

Are you sure you want to board Shuttle One?

NO YES



Use Case:



Description of steps involved:

- Sign in or Sign up to the app.
- Register with verification code sent to the given mobile number.
- Upon successful sign in reach the main home page.
- Click on “Look For Shuttle” to check existing location of shuttle and the path to the nearest bus stop.
- Select a shuttle from the dropdown and then click on board to board the shuttle.
- To check the distance and the stops of the shuttle click on the location markers.

- Click on live track to send the live updates of your location to the Database to aid in tracking of shuttle.
- To report delay of a shuttle, click on Navigation Draw and click on report delay. An entry will be logged in Database with the time stamp.
- Click on the bus marker to know about the last seen time.
- Click sign-out in Navigation draw to close the current login session

References:

- http://www.codingsection.com/real-time-car-booking-app-using-google-maps-javascript-api- and-html5.html
- https://medium.freecodecamp.com/how-to-build-your-own-uber-for-x-app-33237955e253

Device Sensor Processing Routines:

Used the following device sensors and techniques:

1. Device GPS.
2. Google Map API
3. Web-server to handle incoming requests from the client and to route information.
4. AWS Database to store and query information and locations.
5. Device mobile data to connect to webservers.

AVD specification:

Device name: Pixel XL API 23 x86
Android Version: 6.0
Marshmallow API Version: 23
Screen specifics: 5.5” 1440x2560 HD

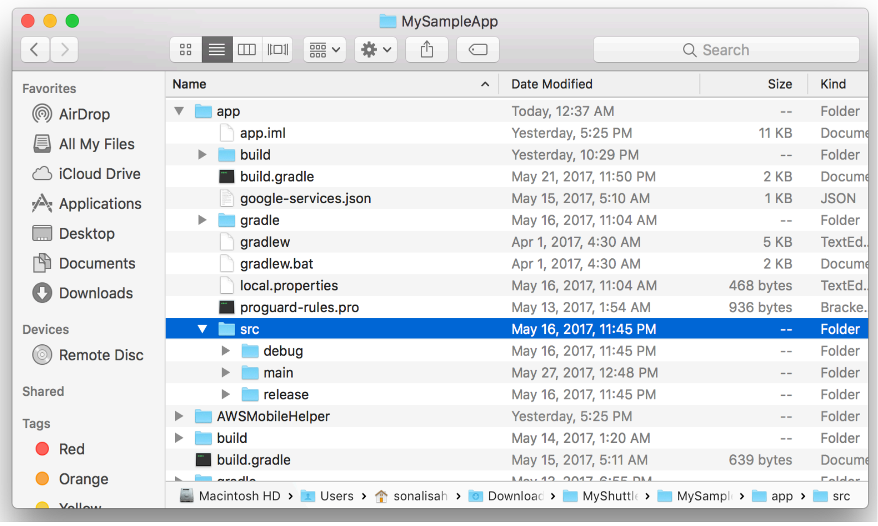
Location of Code:

MyShuttleApp.zip folder contains all required source code and document.

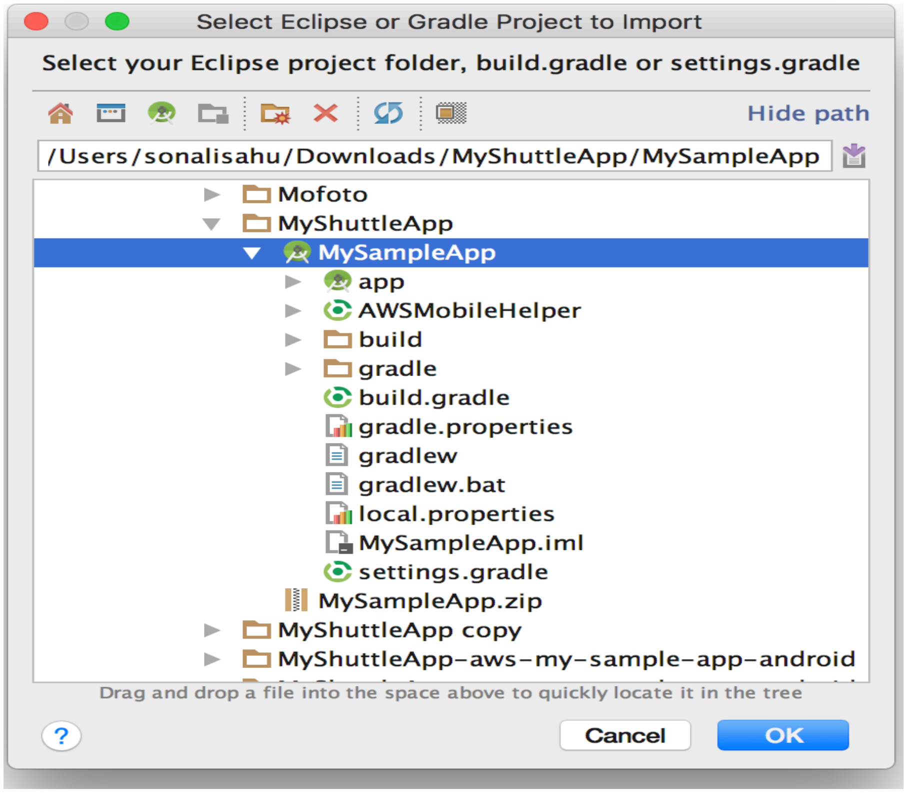
Section 1

Execution Instructions:

1. Download the zip folder MyShuttleApp.zip from Blackboard.
2. Go to the download location and unzip the folder.
3. Open Android Studio. Go to File -> New -> Import projects… and import the downloaded project ->MyShuttleApp ->MySampleApp



Name	Date Modified	Size	Kind
▼ app	Today, 12:37 AM	--	Folder
app.iml	Yesterday, 5:25 PM	11 KB	Docume
▶ build	Yesterday, 10:29 PM	--	Folder
build.gradle	May 21, 2017, 11:50 PM	2 KB	Docume
google-services.json	May 15, 2017, 5:10 AM	1 KB	JSON
▶ gradle	May 16, 2017, 11:04 AM	--	Folder
gradlew	Apr 1, 2017, 4:30 AM	5 KB	TextEd.
gradlew.bat	Apr 1, 2017, 4:30 AM	2 KB	Docume
local.properties	May 16, 2017, 11:04 AM	468 bytes	TextEd.
proguard-rules.pro	May 13, 2017, 1:54 AM	936 bytes	Bracke.
▼ src	May 16, 2017, 11:45 PM	--	Folder
▶ debug	May 16, 2017, 11:45 PM	--	Folder
▶ main	May 27, 2017, 12:48 PM	--	Folder
▶ release	May 16, 2017, 11:45 PM	--	Folder
▶ AWSMobileHelper	Yesterday, 5:25 PM	--	Folder
▶ build	May 14, 2017, 1:20 AM	--	Folder
build.gradle	May 15, 2017, 5:11 AM	639 bytes	Docume



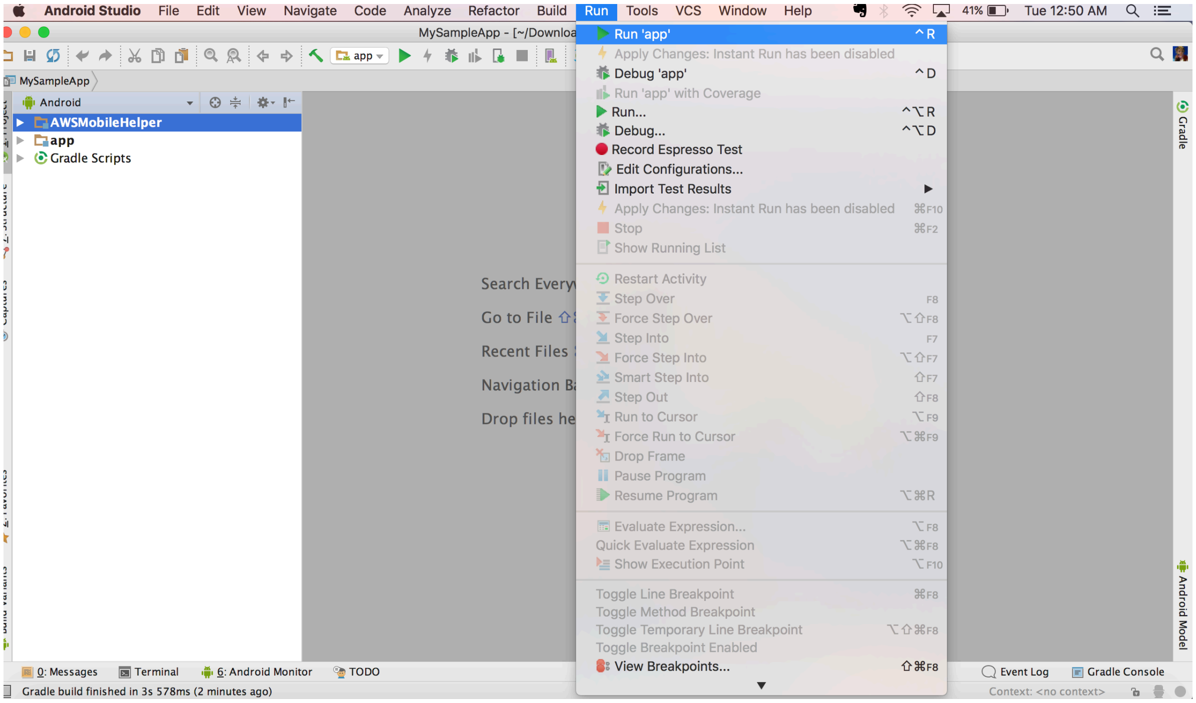
Select your Eclipse project folder, build.gradle or settings.gradle

/Users/sonalisahu/Downloads/MyShuttleApp/MySampleApp

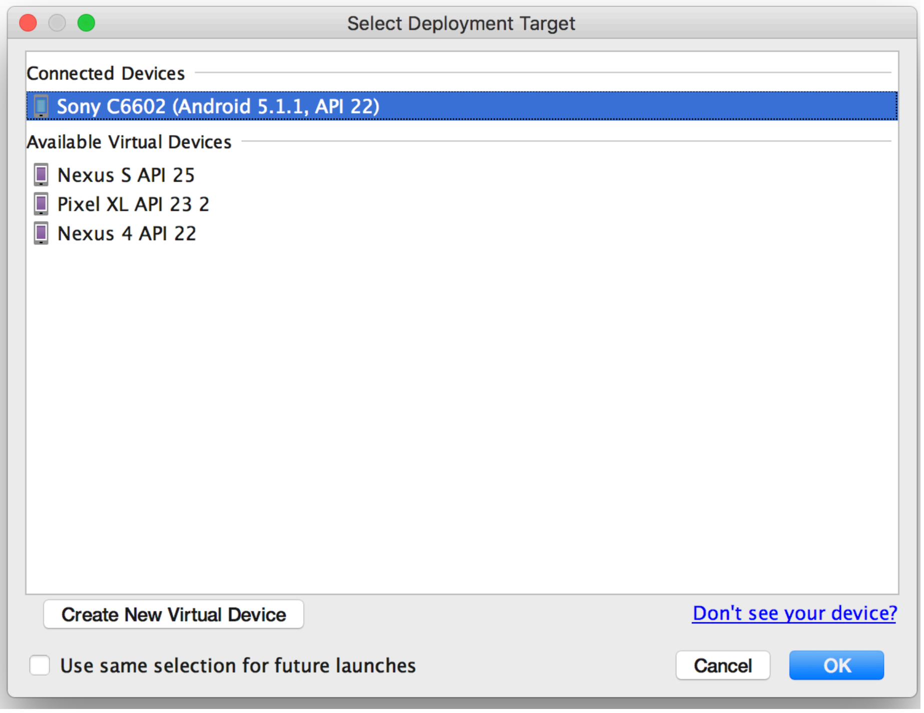
```
▶ Mofoto
▼ MyShuttleApp
  ▼ MySampleApp
    ▶ app
    ▶ AWSMobileHelper
    ▶ build
    ▶ gradle
      build.gradle
      gradle.properties
      gradlew
      gradlew.bat
      local.properties
      MySampleApp.iml
      settings.gradle
    MySampleApp.zip
▶ MyShuttleApp copy
▶ MyShuttleApp-aws-my-sample-app-android
```

Drag and drop a file into the space above to quickly locate it in the tree

4. Connect an android device, go to Run -> Run “app” or click the run icon in toolbar



5. Choose the target android device from the list and click “OK” to run the App.


Select Deployment Target
Connected Devices
Sony C6602 (Android 5.1.1, API 22)
Available Virtual Devices
Nexus S API 25
Pixel XL API 23 2
Nexus 4 API 22
Create New Virtual Device
Don't see your device?
□ Use same selection for future launches
Cancel
OK

Section 2

Code Description:

Java Files:

Application.java: responsible for initializing singletons and other common components.

HomeMapFragment.java: Is the fragments that launches report delay screen giving user to check for the last reported shuttle, scheduled arrival time of current shuttle and next shuttle.

MainActivity.java: Is the Main frame of the application consisting links to tack shuttle activity, report delay and navigation draw and sign-out fragments.

MapsActivity.java: Activity that launches the Track Shuttle screen, giving the user options to check the direction for nearest bus stop, check distance to all bus stops, log the boarding of the shuttle and enable or disable live track.

OnMapAndViewReadyListener.java: Helper class that will delay triggering the OnMapReady callback until both the GoogleMap and the View having completed initialization. This is only necessary if a developer wishes to immediately invoke any method on the GoogleMap that also requires the View to have finished layout ie. anything that needs to know the View's true size like snapshotting).

SignInHandler.java: Handles Re-directing to the main activity upon sign-in.
DemoConfiguration.java: Base class for each small Demo Fragment.

IdentityDemoFragment.java: Launches the default user identity fragment with user Id and session management functions

HomeDemoFragment.java: Launches the default home page containing the User Identity fragment and lick to start shuttle track activity.

DemoFragmentBase.java: Base framework class for all fragment in App.

SplashActivity.java: plash Activity is the start-up activity that appears until a delay is expired or the user taps the screen. When the splash activity starts, various app initialization operations are performed.

ReportDO.java, RoutesDO.java, ShuttleDO.java, ShuttleTimeTableDO.java, StopDO.java, UserInfoDO.java: Database Objects class for AWS DB calls.

AWSConfiguration.java: This class defines constants for the AWS resource identifiers and API keys

AWSMobileClient.java: The AWS Mobile Client bootstraps the application to make calls to AWS services. It creates clients which can be used to call services backing the features you selected in your project.

Layout Files:

activity_main.xml: Main activity layout.

activity_splash.xml: Layout of the Welcome Screen.

fragment_demo_home.xml: Layout for Home page fragment.

fragment_demo_identity.xml: Layout for User Identity fragment.

nav_drawer_item.xml: Layout for Navigation Draw.

report_delay.xml: Layout for report Delay fragment screen.

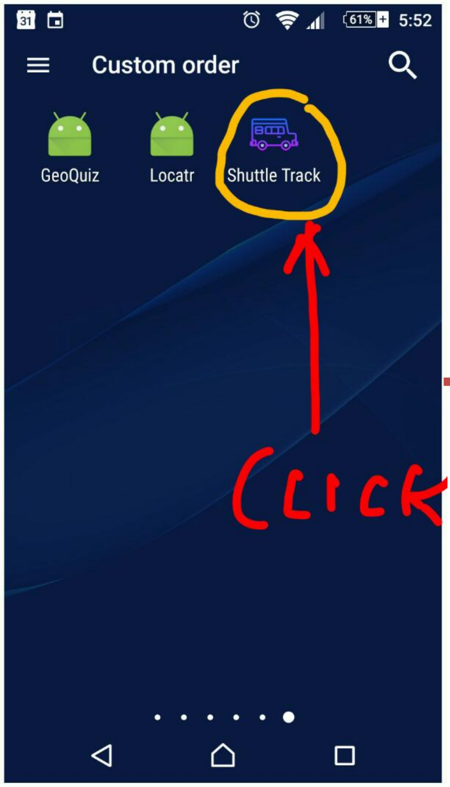
shuttle_track.xml: Layout of shuttle tack Activity Screen.

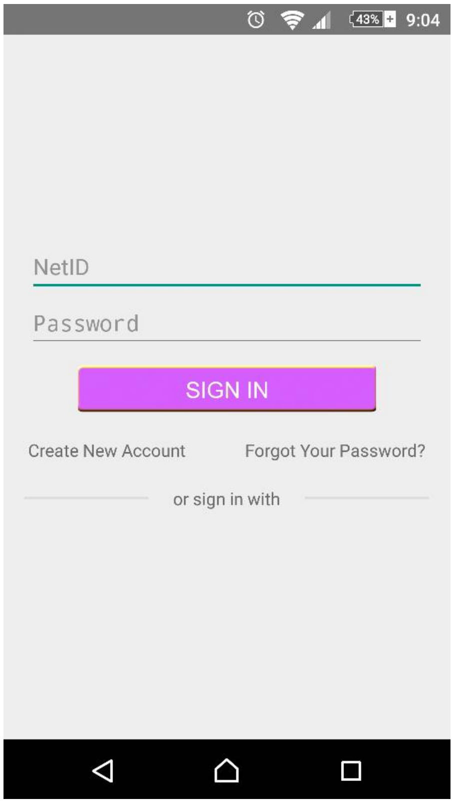
Section 3

Testing:

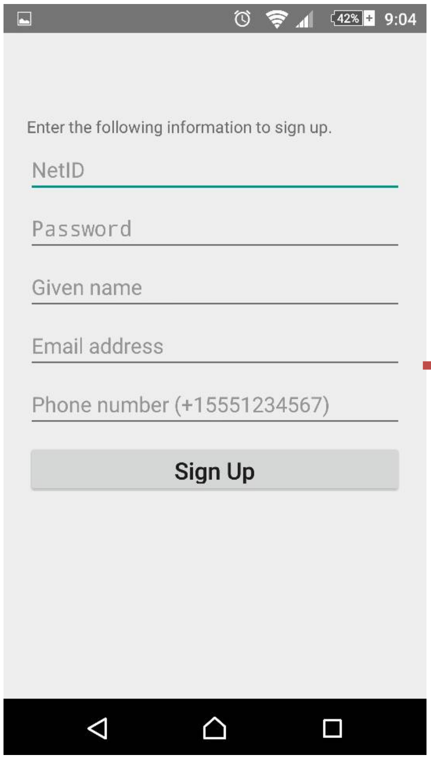
Section 3.1: Starting application.

After installing the App on device click on Shuttle Track icon to launch the App.

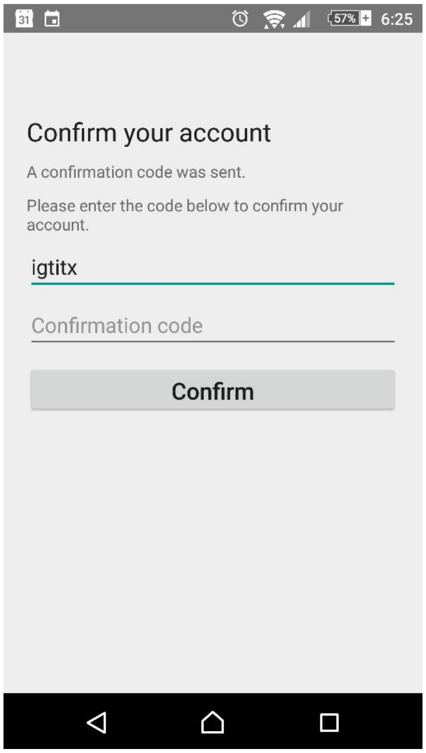




- Click on Create New Account for New user or Sign In for existing user.
- Fill up the Sign-up Form and then Verify account with received code.



Enter the following information to sign up.



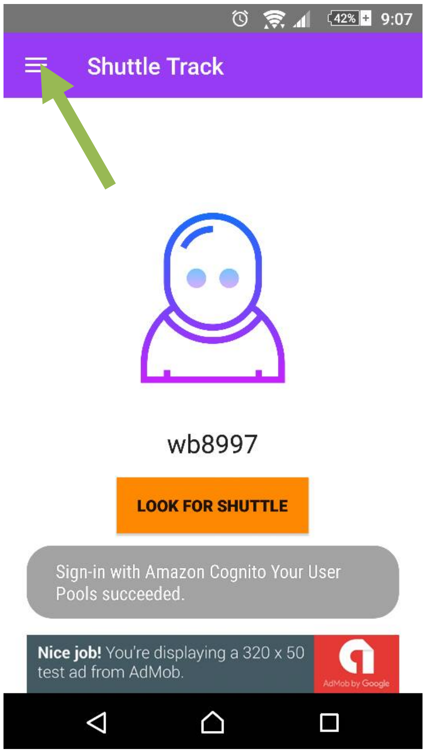
Confirm your account

A confirmation code was sent.

Please enter the code below to confirm your account.

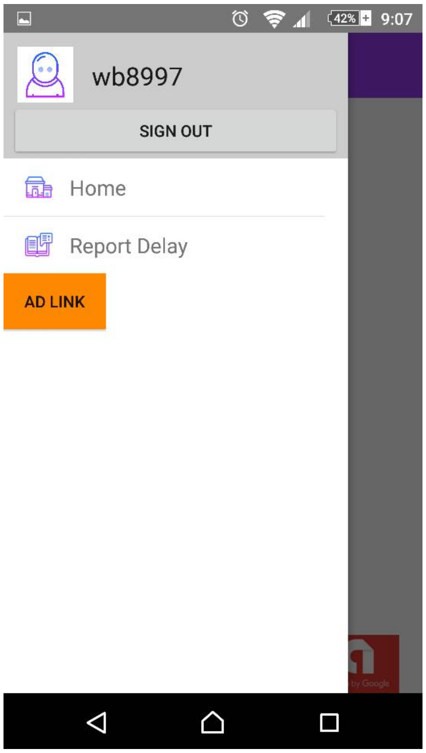
igtitx

- Next Home Page will be displayed. Click on top List icon to show Navigation items.

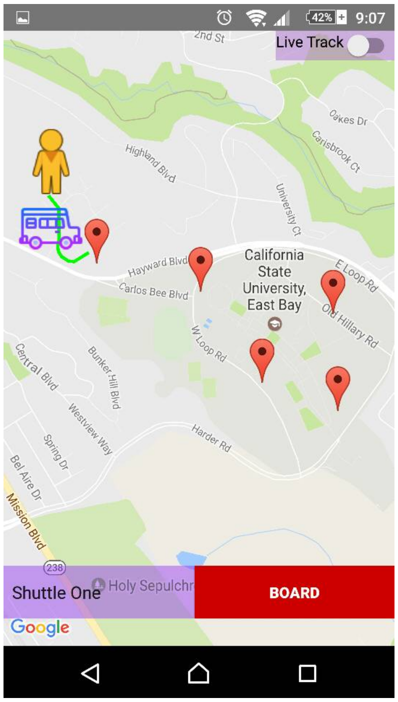


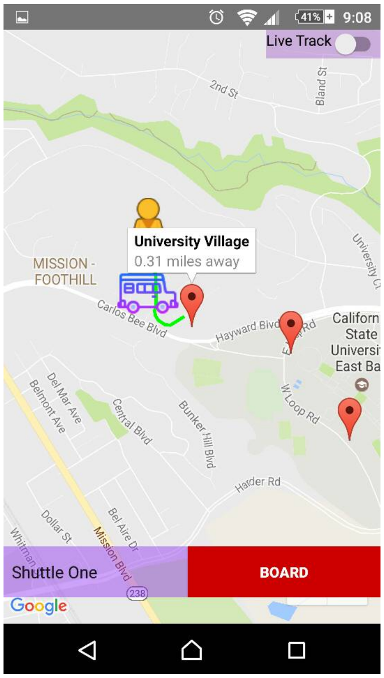
Sign-in with Amazon Cognito Your User Pools succeeded.

Nice job! You're displaying a 320 x 50 test ad from AdMob.

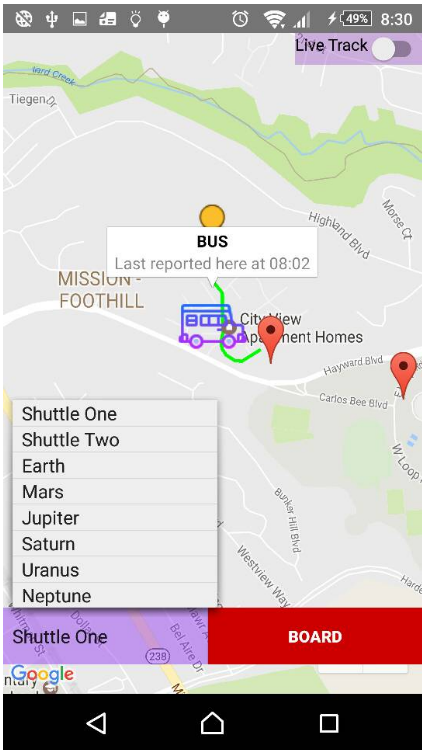


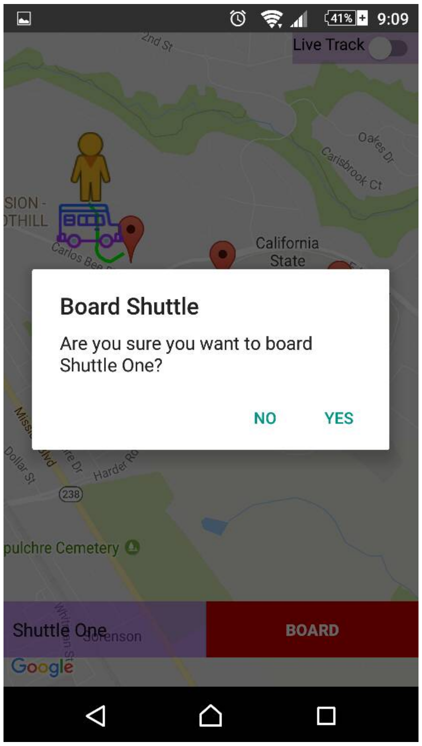
- Click on Look for Shuttle button to check for the location of the shuttle. *There would be a prompt to allow the App to use **GPS** Service of the device. Click on allow to enable functioning of the app.*





- Click on Markers to know about the shuttle stop and distance from your location. Also, Direction from your current position to the nearest shuttle stop is marked in green path lines.

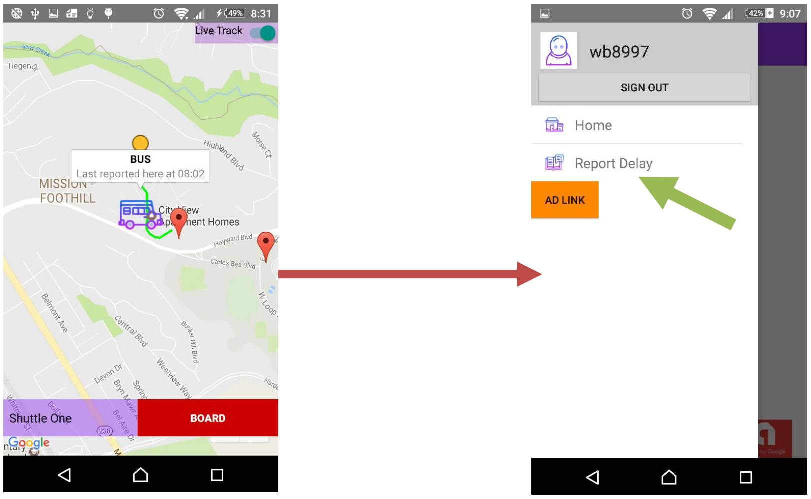




Board Shuttle

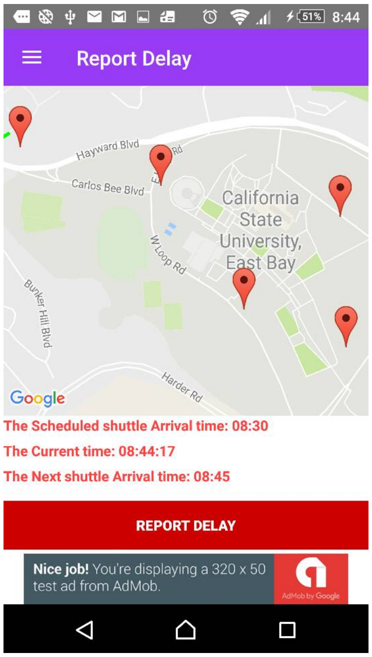
Are you sure you want to board Shuttle One?

- Select the intended shuttle from the purple dropdown menu and click on board shuttle. To confirm click on yes.



- To unable live track option which will facilitate better locating of shuttle for the users click on the green switch on top right corner.

- Go back to the main menu by clicking on back and select Report Delay log delay of the shuttle.

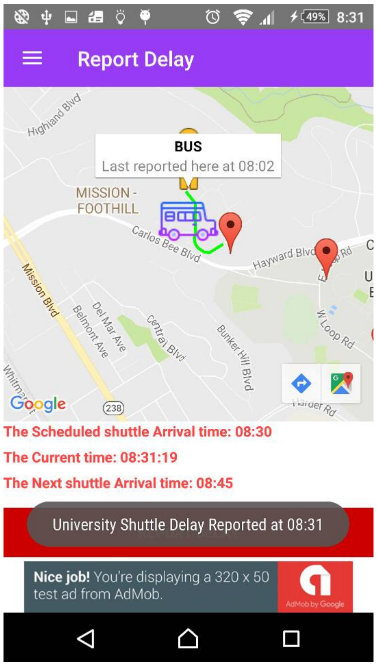


The Scheduled shuttle Arrival time: 08:30

The Current time: 08:44:17

The Next shuttle Arrival time: 08:45

Nice job! You're displaying a 320 x 50 test ad from AdMob.



BUS
Last reported here at 08:02

The Scheduled shuttle Arrival time: 08:30

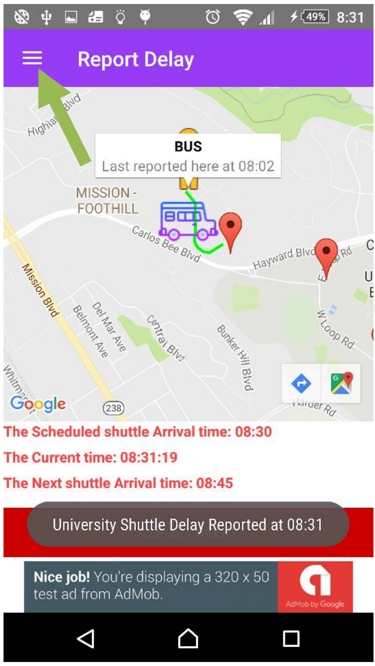
The Current time: 08:31:19

The Next shuttle Arrival time: 08:45

University Shuttle Delay Reported at 08:31

Nice job! You're displaying a 320 x 50 test ad from AdMob.

- To sign out click on sign out in navigation draw.



BUS
Last reported here at 08:02

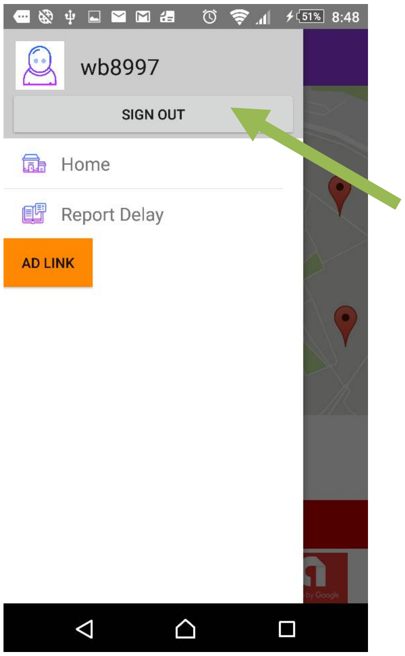
The Scheduled shuttle Arrival time: 08:30

The Current time: 08:31:19

The Next shuttle Arrival time: 08:45

University Shuttle Delay Reported at 08:31

Nice job! You're displaying a 320 x 50 test ad from AdMob.



Section 4

Comments:

My Shuttle Track Application since heavily dependents on device GPS does not serve its intended purpose when run in Android emulator as the default location for the emulator is set to Mountain View CA. For testing purpose, I have added the routes and schedule for just one shuttle i.e. Campus Loop Shuttle.

Certain redundant buttons from the project proposals has been consolidated into minimal

Section 5

YouTube URL:

Video 1: **https://youtu.be/Bywvxttvr8k**

Video 2: **https://youtu.be/sQ09miB1EUw**