Increment 3

Group 1: Car Master

Michael Rzepka (34) (SG-11)

Haritha Garikapati (14) (SG-11)

Wang Zhang (50) (SG-3)

Ran Chen (11) (SG-3)

Dr. Yugyung Lee

CS 551: Advance Software engineering

December 8, 2014

## Import Existing Services/API

NHTSA car recall information

<http://api.usa.gov/recalls/search.json?organization=nhtsa>

Edmunds developer network car information

<http://developer.edmunds.com/api-documentation/vehicle/spec_make/v2/>

Edmunds developer network car pricing data

<http://developer.edmunds.com/api-documentation/vehicle/price_tmv/v1/05_calculatetypicallyequippedusedtmv/api-description.html>

Edmunds developer network car style API

<http://developer.edmunds.com/api-documentation/vehicle/spec_style/v2/01_by_mmy/api-description.html>

Gas Station with prices API

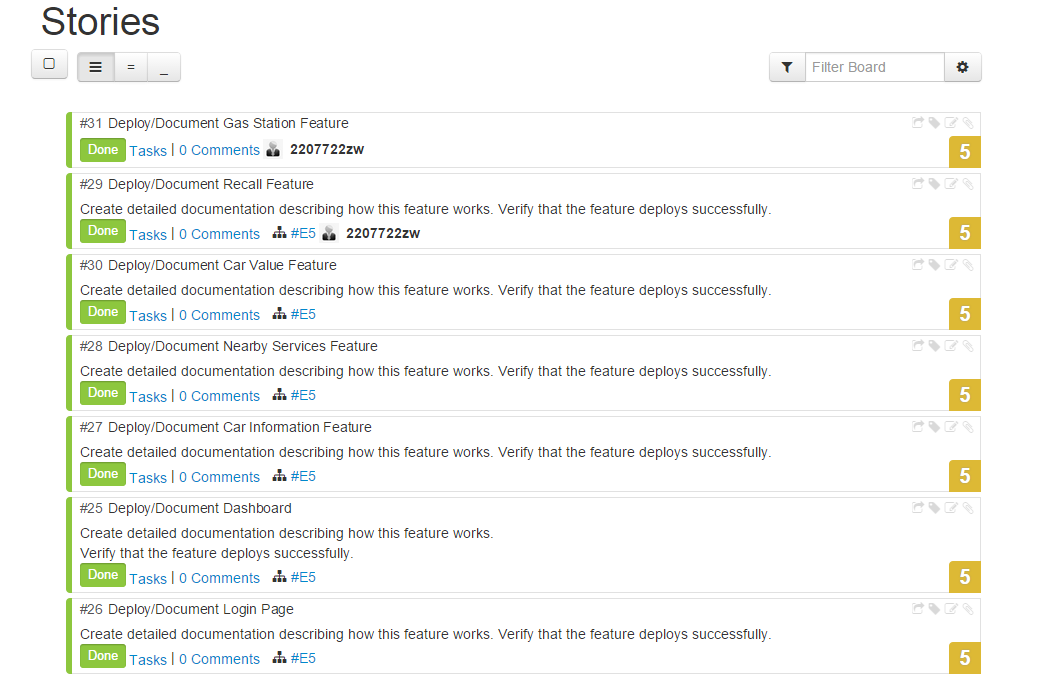
<http://www.mygasfeed.com/keys/api>

Google Map API v2

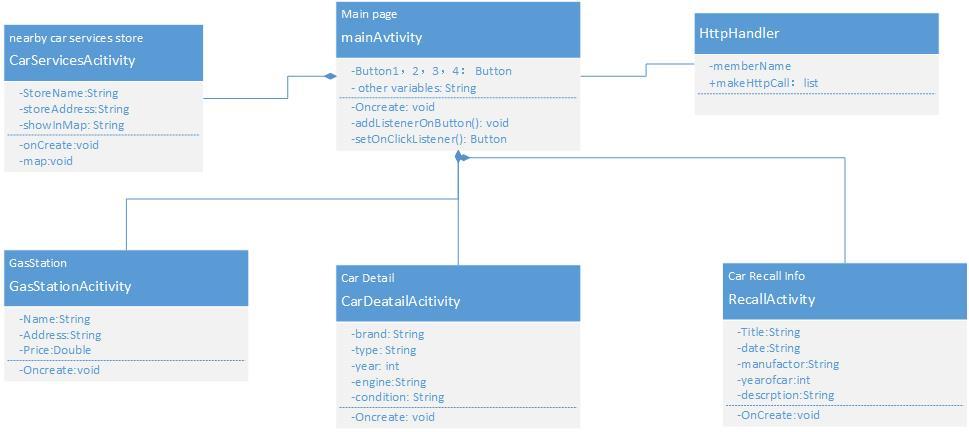
<https://maps.googleapis.com/maps/api>

**Detail Design of Service**

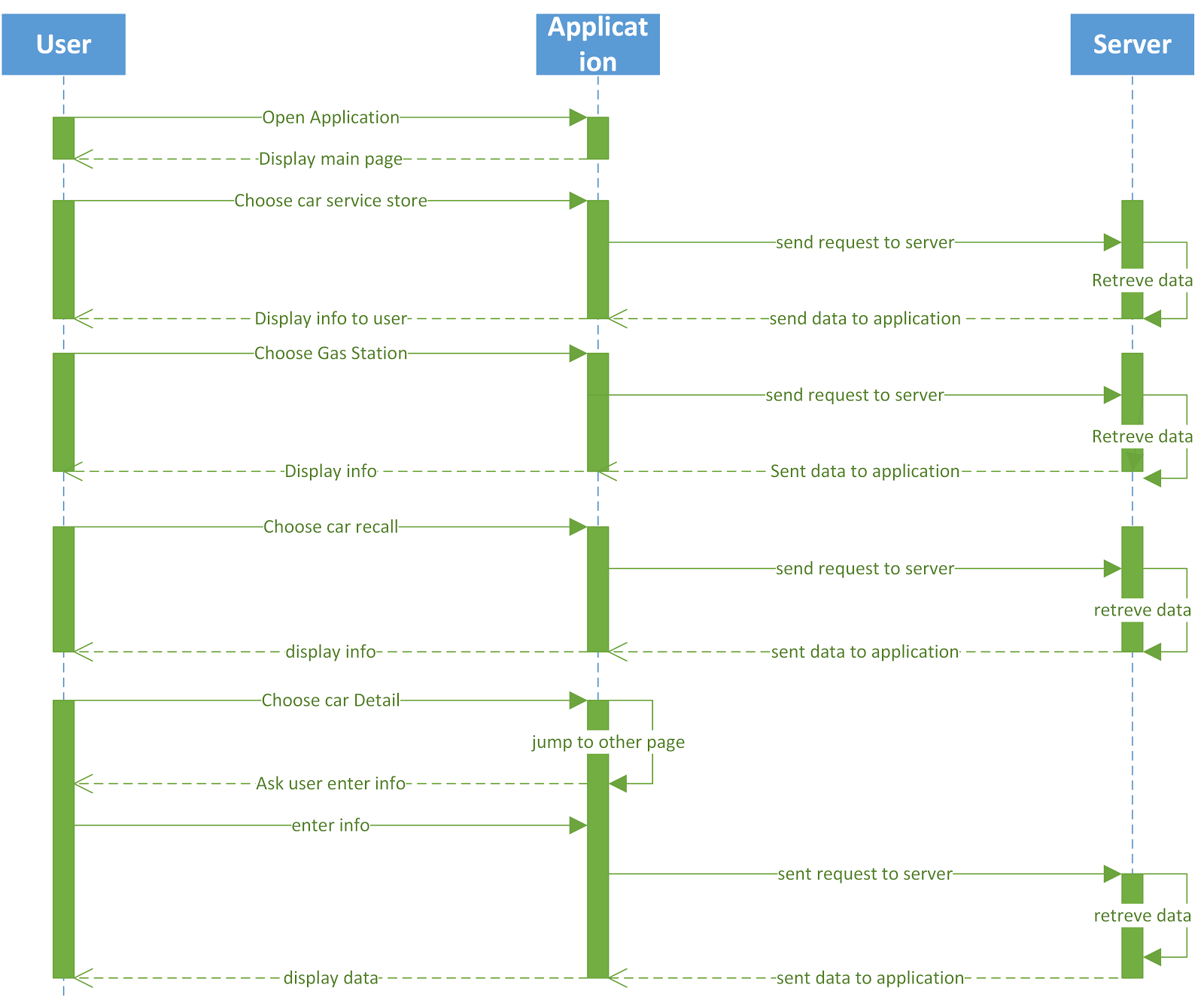
Write User Stories/Use Case (Using ScrumDo)



Class diagram (using Visio)



Sequence diagram (using Visio)



Design of Mobile Client Interface

Our project is an Android based application. For now, our project has five buttons on the main page with each button indicating a specific feature. The first button is the Car Detail feature with which allow the user will be able to see information like the engine, transmission, and others details for their car based on the users entry. The second button is the Car Recall feature with which the user will able to see recall information for their car based on what user entered. The third button is the Car Price feature with which the user will be able to see their car’s market value. The fourth button is the Nearby Services feature which with the user will able to check nearby service stores for their cars. The last one is the Nearby Gas Station feature which will allow the user to see nearby gas stations with the associated prices.

Design of Unit test cases (using NUnit tool)

No unit tests have been written yet as a part of increment 3, as this increment was primarily focused upon implementation work. The expectation is that the next increment would be more heavily focused on the testing aspect of the application. For the design of the test cases, since we are developing an Android application, we will not be testing the activities themselves that control the application flow through unit testing. This testing will be handled primarily through manual functional testing. The unit testing will cover aspects that the activity classes call, such as any service integration, JSON parsing, DB insertions, updates, deletions, and any other business logic that is implemented.

Design of User Registration and Login

The user when accessing the application has to sign up first by entering the username and password. The application will ask the user to confirm the password, when the passwords match, the user registration is successful. Also to access the application the user will have to enter the exact credentials to keep his data safe. This makes the same application accessible to different users by storing their individual or personal data. This is being implemented through SQlite Database. All required Login and Registration user-data will be save in application’s own Android SQLite database. The database will be stored in the application’s context, so that no other android applications will access the data.

Design of Car Pricing

The web service driving the car value feature will primarily be the Edmunds API. To use the used car value API, the Edmunds Style ID is required. To obtain this ID, we'll first need to make another call based on the cars make, model, year, and state (condition). The call to get the Style ID will likely return an array of multiple results. We'll need to determine how we will filter these result down when checking the value. Some options of this filtering are to prompt the user to pick a specific style, average the total of the multiple results, or simply pick the first result returned.

Averaging the results likely wouldn't work out very well, as that would require several API calls to get the final data (which would likely result in a slow response). The simplest way to move forward for now would be to pick the first Style, and then possibly expand the functionality later to prompt the user to choose a type based on the returned results.

When retrieving the car value specifically, the zip code is also required. This can be returned by using the Android's GPS functionality (or alternatively, prompting the user if GPS is unavailable).

All of the fields mentioned above should be saved to the Android in a SQLite database.

The table for this database could potentially be the car\_value table. The primary key of this table would likely be the Style ID. This could be expanded to include the User ID, if user login functionality is available. It is also reasonable to assume that the zip code might be available from another table (used by other areas of the application), where the data could be stored and simply retrieved for populating these calls and the resulting data. The fields on the car\_value table would be the make, model, year, and condition. It's possible that all of these values are stored in relation to the car information feature, but if not, they will be stored here at a minimum.

As far as the display, after the initial value is entered for a given make, model, and year, that information would be automatically retrieved when the car value page is entered. If available, the latest information will be pulled down and saved to the Android device (and also display it). Having the data in the database displayed by default will allow a value to always be available after the first time it is retrieved, if access to the web service becomes unavailable.

We could also possibly include the Instagram API functionality into this view, to show the car that the price is being retrieved for based on tagging from the make, model, year, and condition. This image could be shown in the middle of the screen above the price, and could possibly be saved by the user to the Android database if they find an image that they like.

Design documentation: <https://www.scrumdo.com/projects/project/2606/iteration/111857#story_814280>

Design Car Information

In the Car Information part. We used Edmunds API which is the same as the car price. In the front page. I design the six options for user : make, model, year, state, category and view. User needs to input the car’s basic information on it. Then the system according to the six information to find out the suitable car. In the next page, the user can get some basic information about the car. The information include engine size, engine cylinder, engine displacement, drive wheel and so on. According to these information, user can get recognize about car. All the car’s information were stored in the API. If user submit his requirement, the system can search it in the API, to finding out the suitable car. The output the basic information of the car.

In this part, user can get some knowledge about his car. It is convenience for user to change some basic equipment of the car.

Design Car Recall Information

For the car recall information feature. We used NHTSA API to get car recall records. But first thing is that user need to enter the car make, model and year in order to get recall information. More general, user would just choose car make or car make and model. it all depends on user. after user submit the car information, our application will show a list view of car recall information which only contain recall title and date. in order to get more information, user would click one of recall from list view. Our application will show more detail about that recall, such as recall description, defect summary, consequence summary, corrective summary, manufacturer of the car and notes from NHTSA. Also we provide a recall link from NHTSA.

Design Gas Station With Prices

In the Gas station part. We used MyGasFeed API to get gas station with prices. It is one click action. Then user will be able to see the list of station with address, distance from their current location, price of gas and updated time. If user click one of gas station from list, it will show a map with route which is from user current location to that gas station.

Design of Nearby Services

In this the user is able to find the near by services by accessing the GPS functionality through google maps API. With the help of Google Maps API Android V2, this feature has been created to find the nearest Service Stations required for maintenance. It takes the users current location latitude and longitude and gives out the service stations. Where the user can select the one nearest to him. The service stations comprise from nearest restaurants, service centres, maintenance centres, resting areas, hospitals etc.

# Implementation

Implementation of Car Pricing

The current state of the implementation exhibits the core functionality of the feature. It makes two distinct REST calls to the Edmunds API, one to look up the Style ID, and the other to find the car’s respective retail value. It currently considers the make, model, year, and condition of the entry to return the pricing information.

Some enhancements are still needed to this functionality. First, database integration should be entered to save this information in the Android devices directly. Secondly, error handling is needed if not all of the information is entered about the car. We should also take the extra effort to make sure that the price is correctly formatted. Lastly, including the Instagram API with this feature might add a nice touch of functionality, as it should make the user feel more attached to the workflow.

Implementation of Car Recall

In the feature, the user will need to choose the car make, model and year. Then, the user will able to see the recall information. If there is no recall information for their car, there will be a pop-up message window to notify the user. Also if the user click one of the recalls in the list, it will jump to another page to show more details about that recall. Used the usa.gov recall data for our car recall feature.

### Implementation of Gas Station

In this feature, First we implemented a GPS tracker which is used to obtain latitude and longitude for user current location. Then based on the user current location, we use MyGasFeed API to find all gas station nearby the user and sorted based on the distance which is the closest to user first, then display it as listview. Furthermore, user will able to click a station from list, our application will parse the station latitude and longitude from API and use GPS tracker get user current location, then application use google map API to show a route and display it.

Implementation of Car Information

The car information feature is done. It contains two pages. In the first page, user need input the car’s make, model, year, state, category and view. The state, category and view are used drop down list. User only need to choose the option in it. The six basic information is important to search the car in the API. So, it needs user to fill in them. If the six information were correct, the system will search the suitable car in the API. Since the API used JSON format to storage these information. First of all, we need to submit the six information to the system. In the coding part, the program will read the labels which obtain the six information. Then searched them in the API. If there is a car was suitable to these information. It will show all the information in the second page.

In the second page. Program needs to find out the suitable information about the car. It can searched in the API. In this part, API contains some loops in it. To read it, we used several loop in the program. Find out the object, then make a loop in it, read the detail information about the object. Then show all the information on the second page.

Implementation of Car Services

With the help of Google Maps API Android V2, this feature has been created to find the nearest Service Stations required for maintenance. It takes the users current location latitude and longitude and gives out the service stations. Where the user can select the ones nearest to him. It uses the base Android framework for it’s interface, certain changes are required to fix the crashing of application as, despite the code workout carefully and properly, we seem to experience some technical difficulties of application crashing.

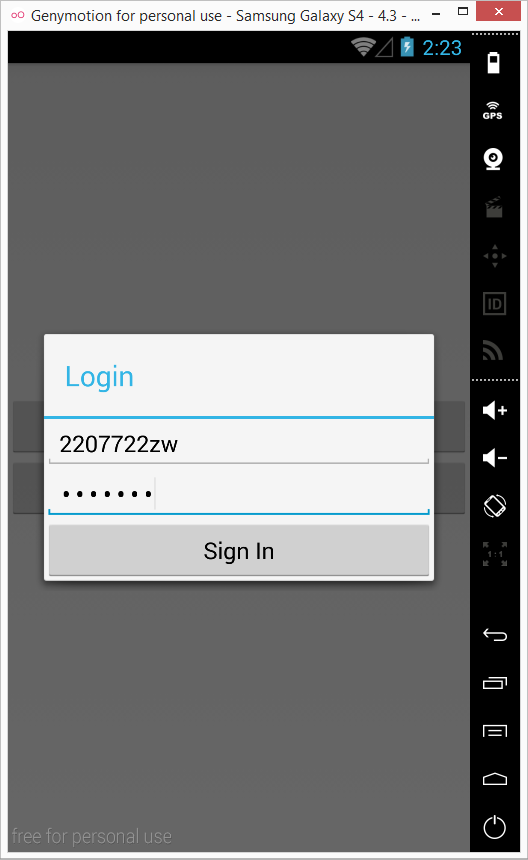
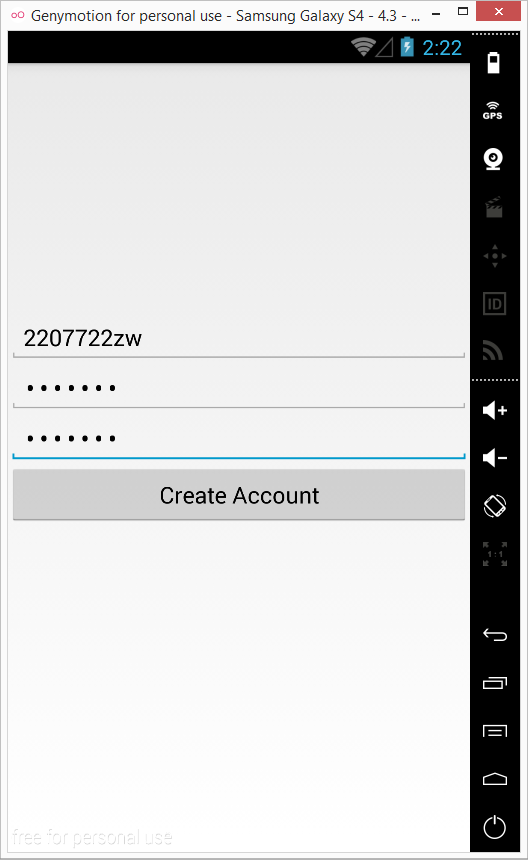
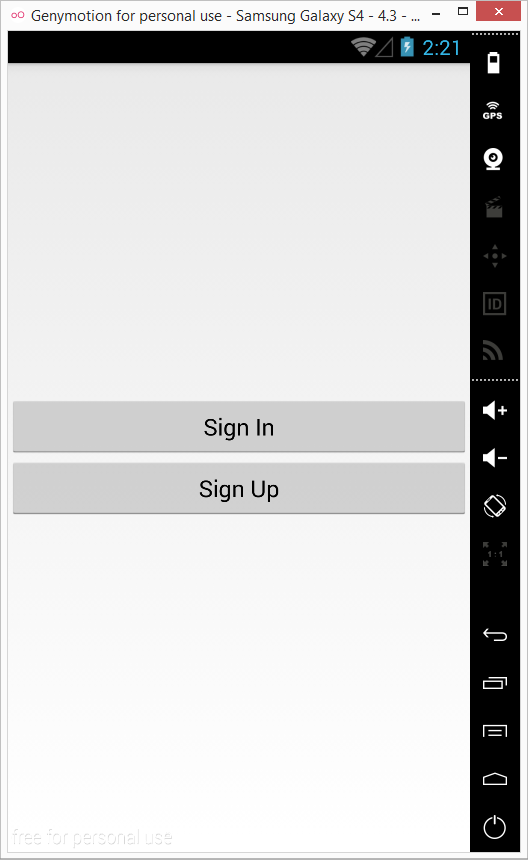
Implementation of User Interface (Mobile Apps)

The application currently uses the base Android framework for it’s interface.First of all, user can used username and password to log in the dashboard. In the dashboard, it concludes the logo and four features of the app. We used four image buttons to make user easy to recognize the features through pictures also. From there, each feature uses it’s own display, but is consistently styled as the same framework is used. In every page of the features, it shows the information of car. Every page is clean and easy for user to read.

# Deployment

Github URL: <https://github.com/mtrzepka/car_master/tree/master/Third%20Increment>

# Screenshots

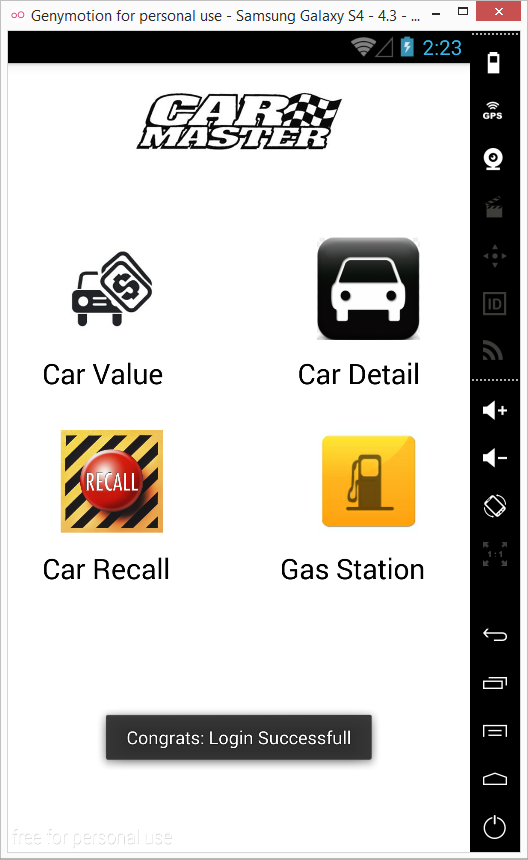


The user is presented with two options of Direct sign in or sign up. If the User has an already existing account, he will be successfully able to log in to the Car master android application. Else The application throws a message saying invalid Id or password and to sign up. There by after registering the user is able to log in. This has been developed through SQLite Database. The above are the screenshots for the user login and registration.

The Logo designed is as follows:

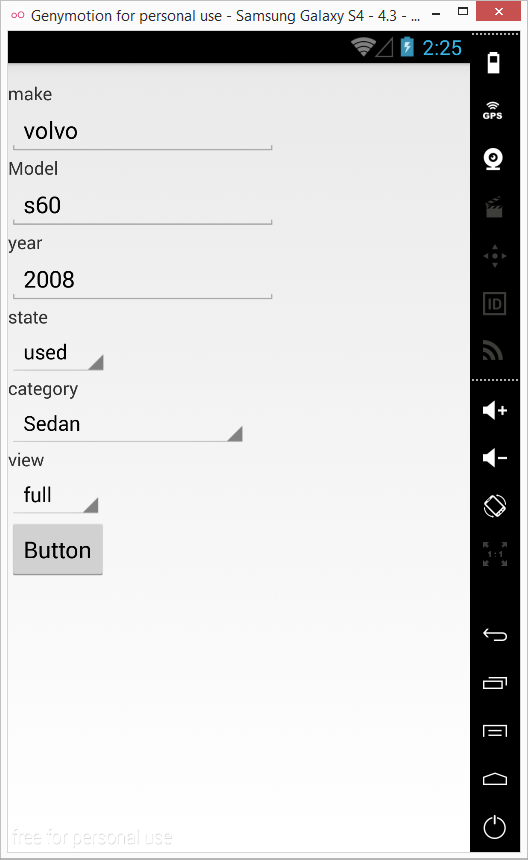


Main page of our Car Master Application, it contains four features Car value, car detail, car recall and nearby gas station.

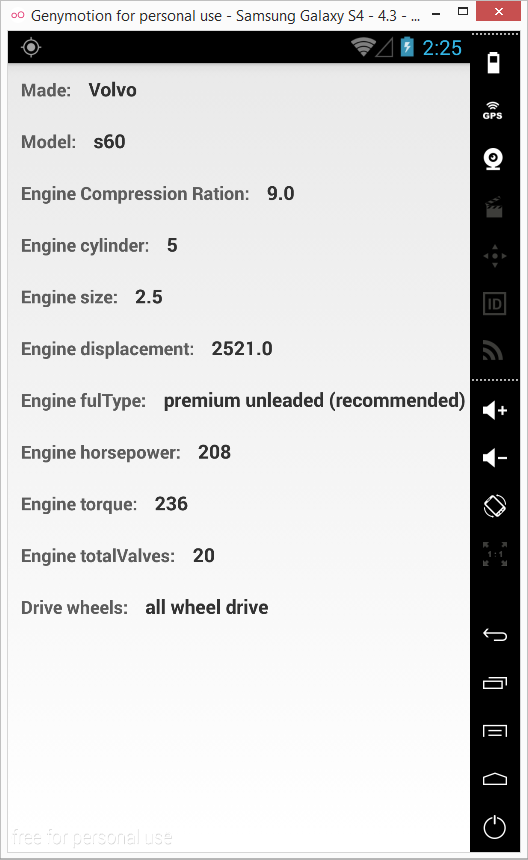


Car information front page.

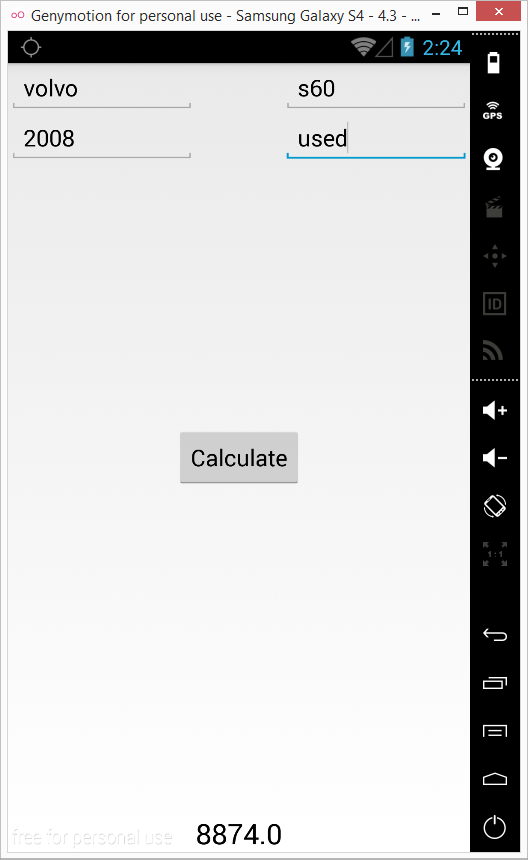
This is the first page of car information. In this page, it needs user input some basic information which contains make, model, year, state, category, and view. After that, user press the button to submit these information.



This is the second page of the car information. In this page, it shows all the information of the car, user can get car’s detail information in this page. Such as made, model, engine information and wheel information.

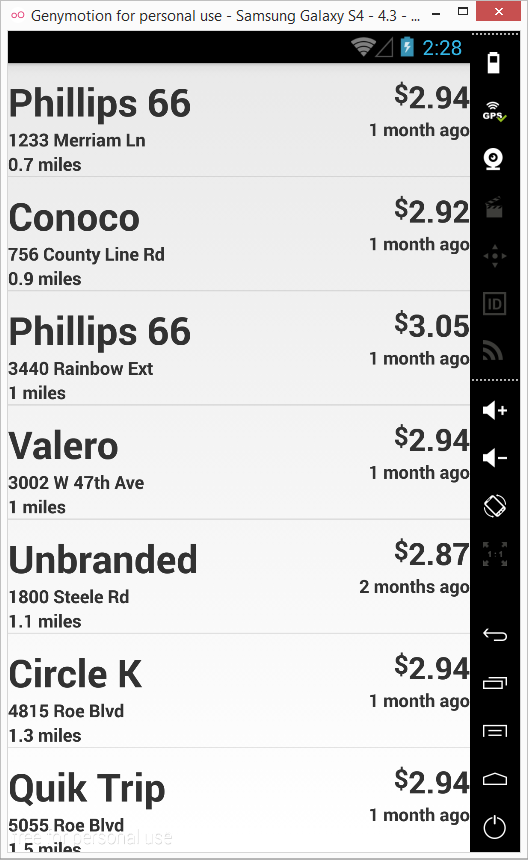


Car Pricing feature - sample call



Nearby gas station feature.

Based on user current location, it will display the list view of the station name, address, distance, price and updated date

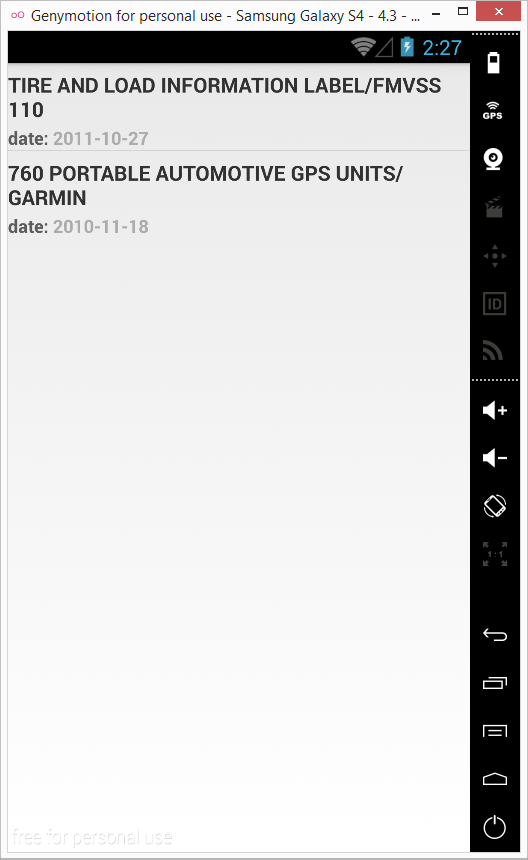
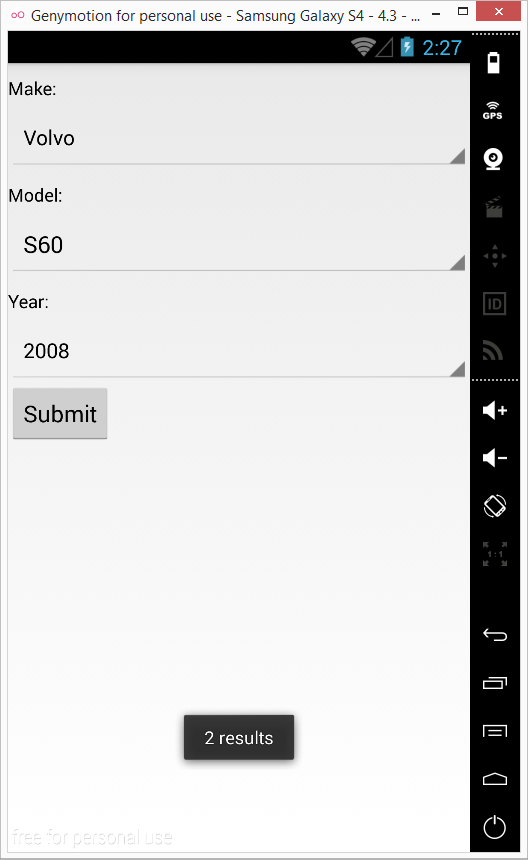


Click one of gas station from list, it will display a map to show the user route from use current location to gas station has been chosen.

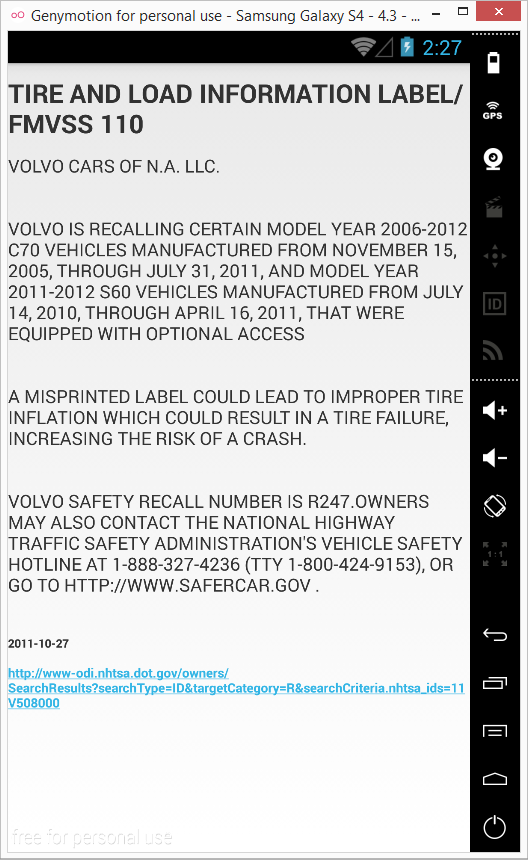


Car recall Feature.

After user enter car make, model and year, it will display the list of recall information for that car.



Click each recall information from list will display more detail about that recall.



# Project Management

ScrumDo URL: <https://www.scrumdo.com/projects/project/2606/summary>

Implementation status report

Work completed:

Ran: The car information part is done.

Mike: The car price part is done.

Haritha: The User Login and Registration Part is done

Wang:The gas station, recall information and main page UI design parts are done.

Issues/Concerns

There are a couple issues and concerns that will need to be overcome as a part of the work for the Car Master application. These center primarily around communication and integration of the various components. Since our group all has busy schedules outside of this class, we will need to put extra focus on being able to incorporate our work seamlessly together. We’ll also have to take extra effort not to repeat common work across features, as they will all likely use the same information as far as the car’s make, model, and year at minimum.