

# Increment 1

## 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. Yogyung Lee

CS 551: Advance Software engineering

October 15, 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/](http://developer.edmunds.com/api-documentation/vehicle/spec_make/v2/)

Edmunds developer network car pricing data

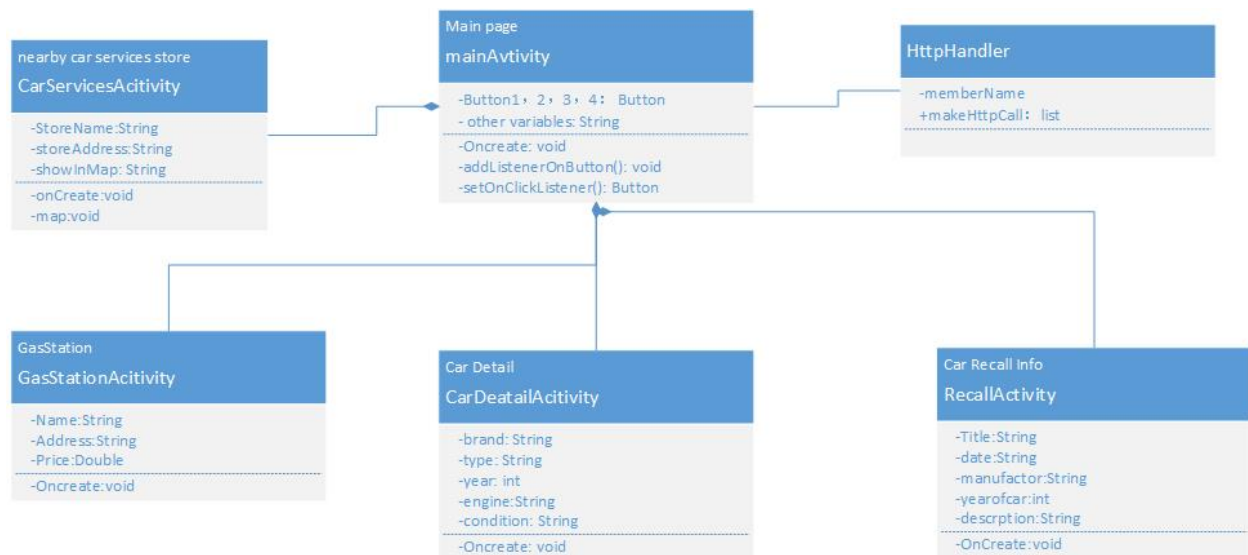
[http://developer.edmunds.com/api-documentation/vehicle/price\\_tmv/v1/05\\_calculatetypicallyequippedusedtmv/api-description.html](http://developer.edmunds.com/api-documentation/vehicle/price_tmv/v1/05_calculatetypicallyequippedusedtmv/api-description.html)

Edmunds developer network car style API

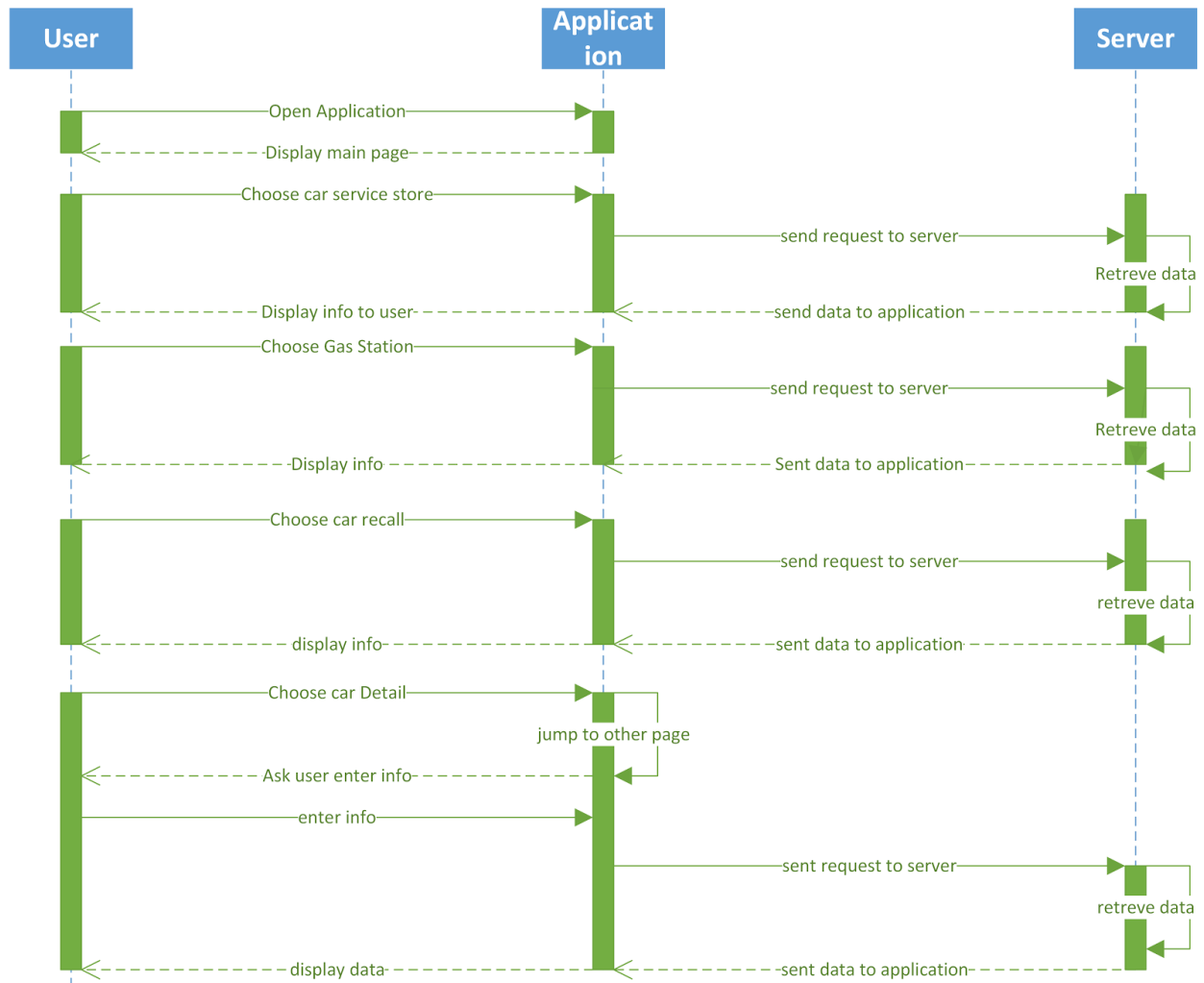
[http://developer.edmunds.com/api-documentation/vehicle/spec\\_style/v2/01\\_by\\_mmy/api-description.html](http://developer.edmunds.com/api-documentation/vehicle/spec_style/v2/01_by_mmy/api-description.html)

## Detail Design of Service

Class diagram (using Visio)



## Sequence diagram (using Visio)



## Design of Mobile Client Interface

Our project is Android based Application, For now, our project has five buttons on the main page which each button indicates a feature. The first button is the car detail, which is a feature that users will be able to see information like engine, transmission and others for their car based on user input. The second button is the car recall, which is a feature that users will be able to see recall information for their car based on what user entered. The third button is the car price, which is a feature for users to be able to see their car's market value. The fourth button is the Nearby services, which is a feature that users will be able to check nearby services stores for their cars. The last one is the nearby gas station, this feature will allow users to see nearby gas station locations and their prices.

## Design of Unit test cases (using NUnit tool)

No unit tests have been written yet as a part of increment 1, as this increment was primarily focused upon interaction design and some initial implementation work. The expectation is that

the later increments would be more heavily focused on the testing aspect of the application, specifically iteration 3 and 4. For the design of the test cases, since we're 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, and deletions, and any other business logic that is implemented.

## Design of Car Pricing

The web service driving the car value feature will primarily be the Edmunds API. To use the 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. There will be a "refresh" or "update" button available to pull down the latest information and save it 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.

Design documentation: [https://www.scrumdo.com/projects/project/2606/iteration/111857#story\\_814280](https://www.scrumdo.com/projects/project/2606/iteration/111857#story_814280)

# Implementation

## Implementation of user interface (Mobile Apps)

The car recall feature is most likely done. In the feature, user will need to choose the car make, model and year. then user will able to see recall information. If there is no recall information for their car, there will be a pop-up message window to notify user. Also if user click one of recall for list, it will jump another page to show more details about that recall.

## Implementation of test cases

Used the usa.gov recall datas for our car recall feature.

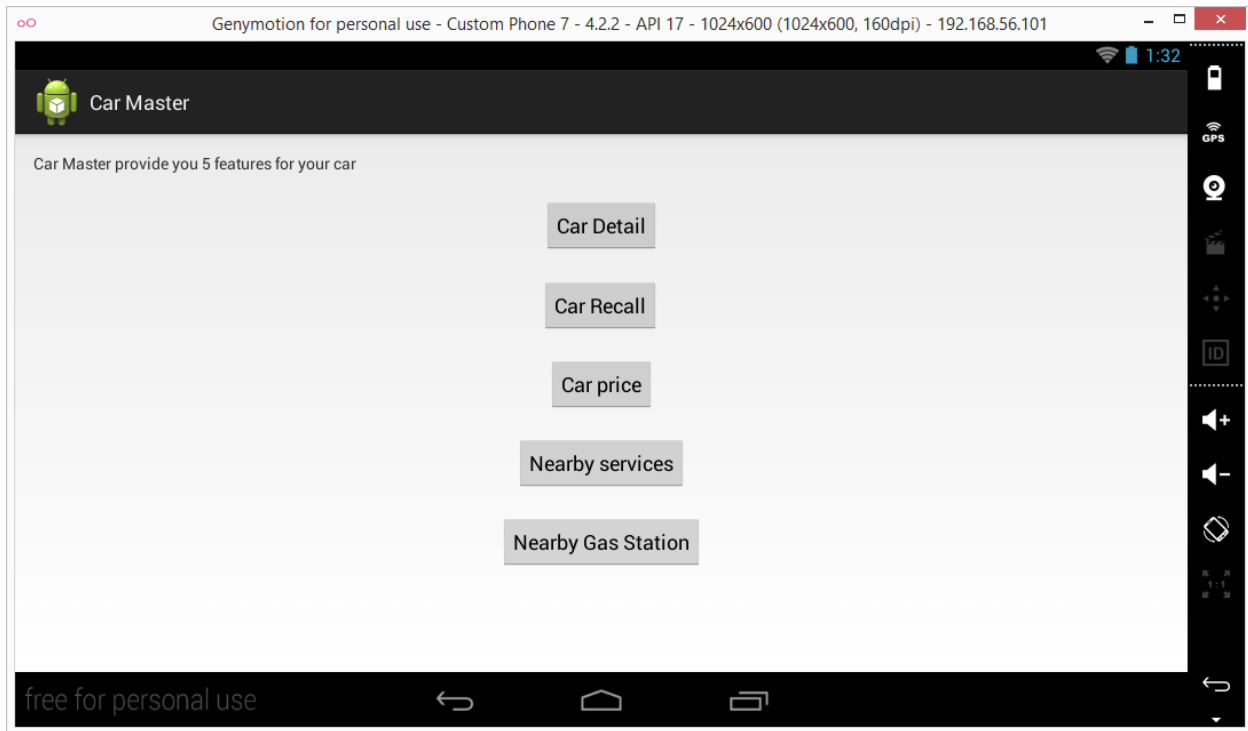
# Deployment

Post your report and source code to your GitHub site and include the URL to the report and the google spreadsheet.

[https://github.com/mtrzepka/car\\_master](https://github.com/mtrzepka/car_master)

## Screenshots and explanation on the design, implementation and testing

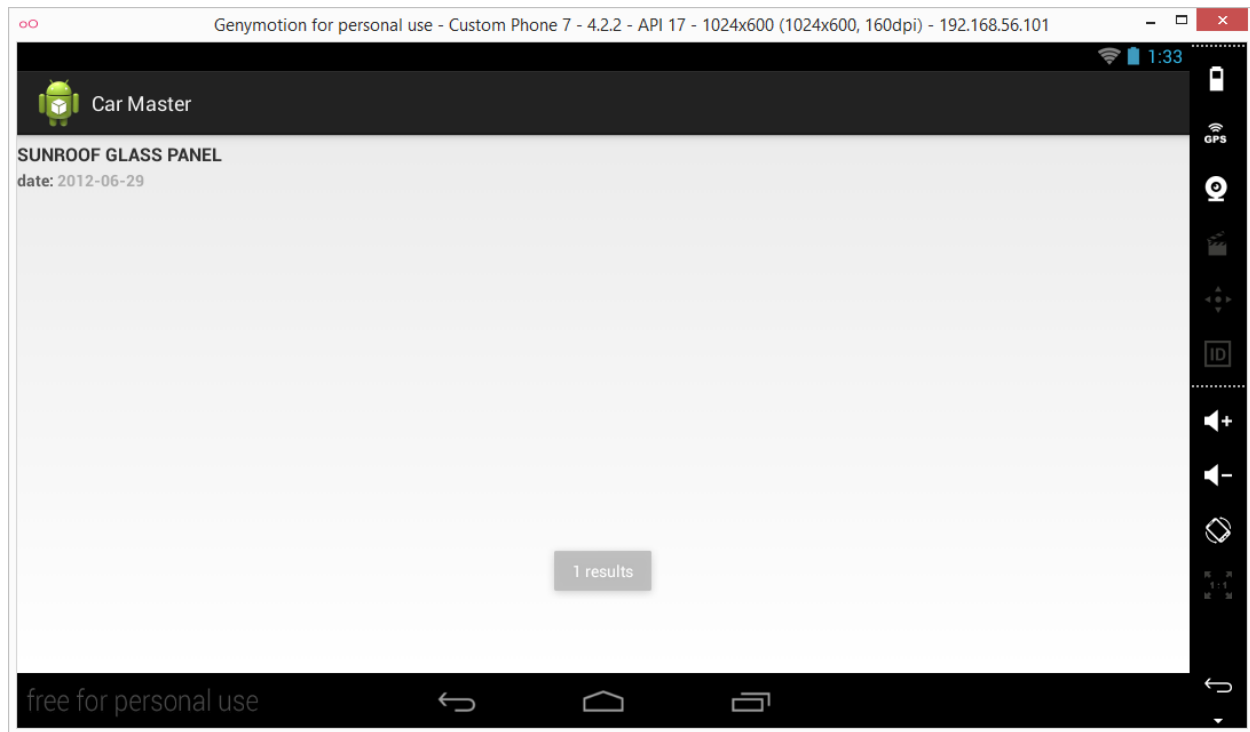
Main page of our Car Master app. it contains 5 buttons indicate for each features.



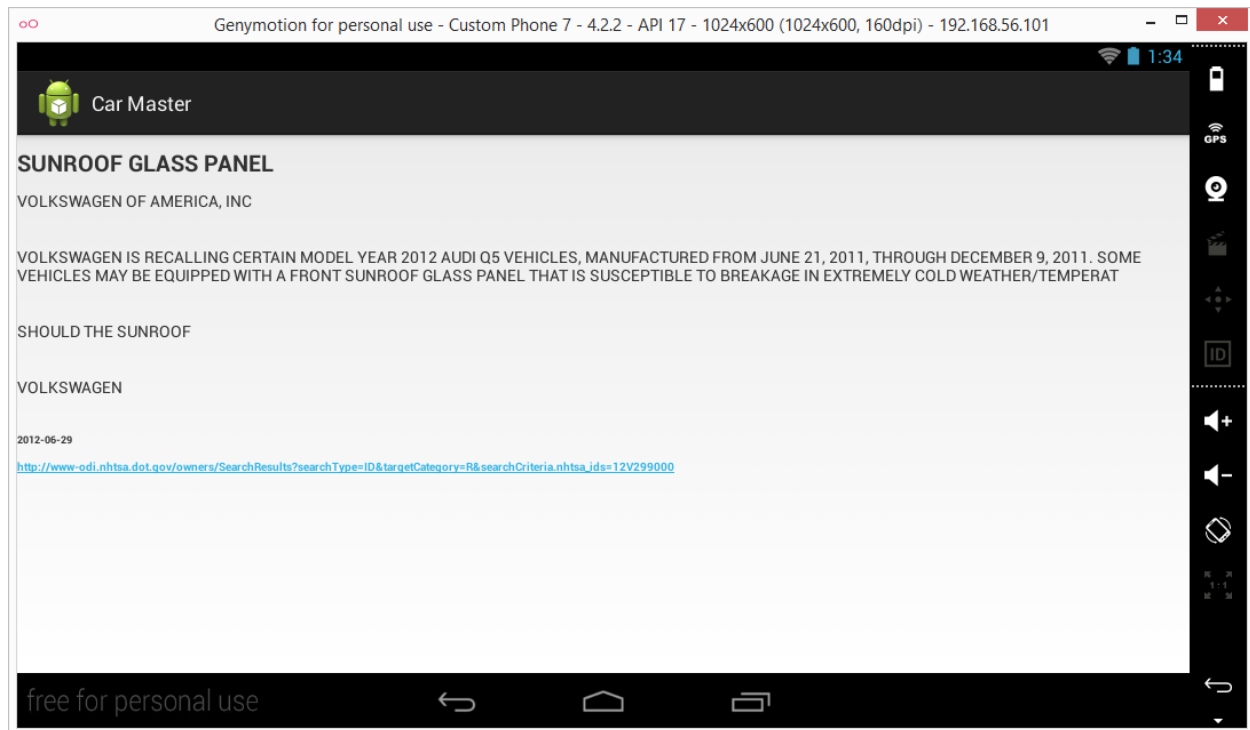
car recall page, in this page user need to choose make, model and year for their car.



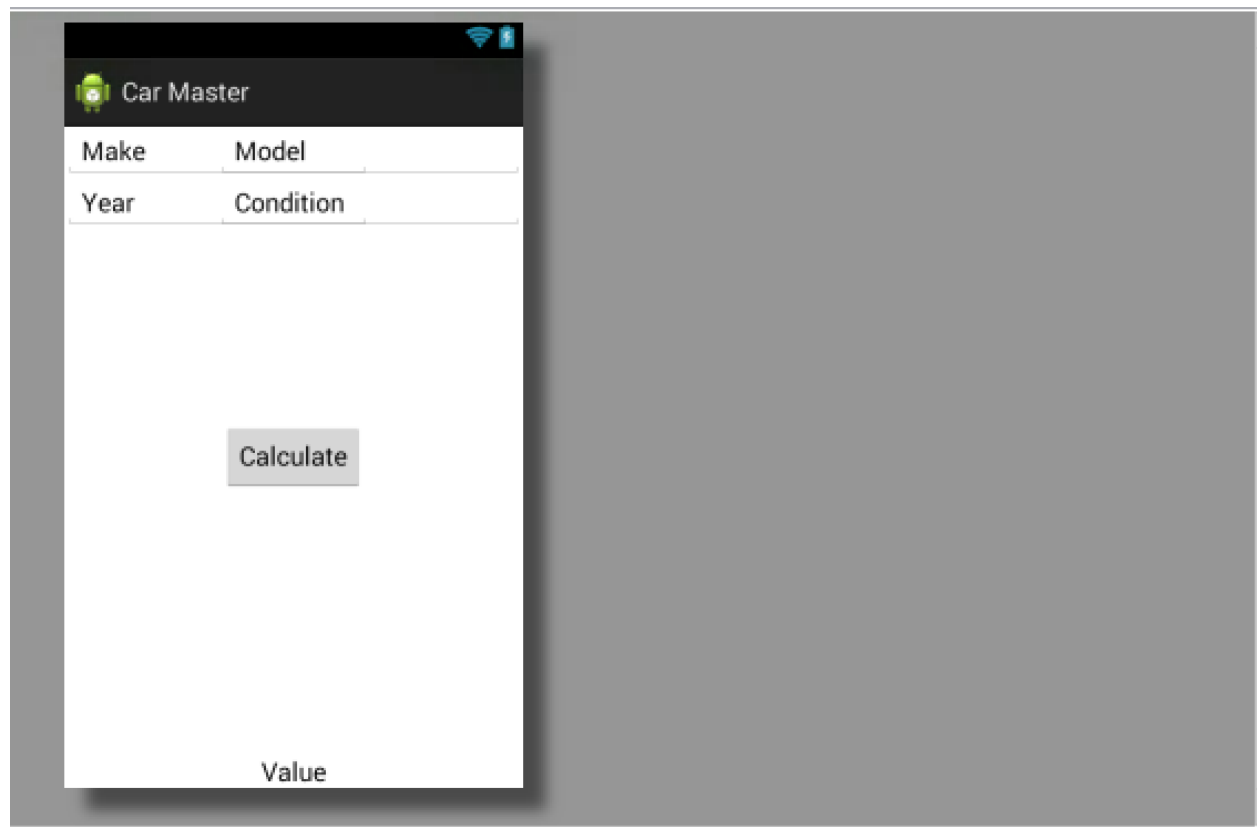
This is recall list page, in this case there is only one recall information.



Detail page, in this page user will able to see the detail recall information from list they choose



## Car Pricing - Initial Layout Design



The image shows a mobile application interface for "Car Master". At the top, there is a black header bar with a green Android robot icon and the text "Car Master". Below the header, there is a white form area. The form contains two rows of input fields: "Make" and "Model" in the first row, and "Year" and "Condition" in the second row. Below these fields is a large, light gray "Calculate" button. At the bottom of the form, the word "Value" is displayed. The entire form is set against a dark gray background.

Make	Model
Year	Condition

Calculate

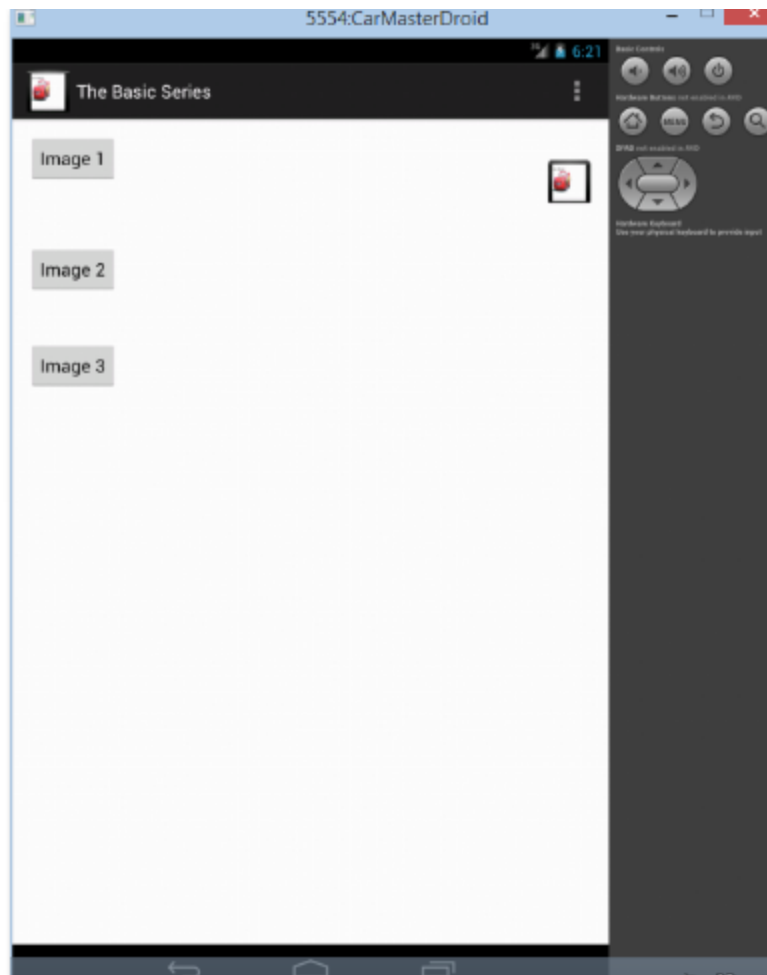
Value

The Logo designed is as follows:





The initial Layout of App, later the buttons named Images have been modified as to our features being implemented. It also shows the Logo attached to our application.



### *Project Management:*

update your project ScrumDo site with the progress and plan of your project and include the URL

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

## Implementation status report

Work completed:

Wang: Most likely finished the car recall part. Spend about 10 hours.

Ran: Get api information for the car information. Design what will show on the every page and the App style. What is the information need user input. And user will get what is kind of the informations. Write small part of code. Spend about 6 hours.

Mike: Focused on the design of the car pricing functionality. Wrote a detailed design specification, along with proposing an initial layout for that design. Spent roughly 6 hours.

Haritha: Created the initial Layout, with 3 buttons redirecting to specific functionality of the app. Also designed the Application Logo to appear in Mobiles at startup. Approximately 6 hours.

### Work to be completed

Wang: Change some format of recall display page and start to implement the nearby gas station feature. These may take more than 24 hours.

Ran: Start to writing code for car details. These may take more than 20 hours.

Mike: Implement the Car Pricing functionality into the application, including feature testing and attempting to integrate with the other areas of the application (to promote reuse). Expectation is close to 30 hours remaining.

Haritha: Will be Designing the User Login, Creating the test cases, and will be working on building rich GUI. 25 hours approximately.

## 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.