**Assignment 2**

Design Document

CIS 1250

Pocket Mechanic

Ajai Gill

Aeman abdulmuhssen

Ian Wadsley

Jamie Chee Man Shing

Raza Mahmood

Yajing Huang

# Content

1. **Introduction**
   1. Introduction of the Innovative Software Product
   2. Goals and Objectives
   3. Definitions
   4. About the software
2. **Requirements**
3. **Design Choices**
   1. Design choices
   2. Key design components
4. **Limitations and constraints**
5. **Software prototype**
6. **Design justifications**
7. **Summary**
8. **References**

# Section 1: Introduction

**1.1 Introduction of the Innovative Software Product**

Pocket Mechanic is an app on the phone directed towards people who would like to keep up with the condition of their car. Pocket Mechanic is always connected to the user's car to gather information on the car. From that, Pocket Mechanic will provide a variety of information based on the car's systems. Pocket Mechanic will be used, the majority of the time, whenever the car experiences technical difficulties. However, the user may use Pocket Mechanic to be informed about the mileage and other related measures.

If the car experiences problems Pocket Mechanic will warn the user and provide information on nearby places for service, or provide instructions to fix the car. This product is useful and is needed because it makes sure that people invest economically for their car and quickly get the best deals.

**1.2 Goals and Objectives/ Vision Statement**

**Goal:** The goal of the software is to help user to save time on finding mechanic shops or save money when they can easily fix the car problems by themselves.

**Objectives:** The design of the software is to help the user with their car problems and to support people who aren't familiar with cars.

**Vision Statement:** The purpose of the software is to help user to save time and money on car repairs.

**1.3 Definitions, Acronyms and Abbreviations**

* **Map:** This is a satellite view of the area around you (google maps).
* **Price(s):** Things like the price for a car service, car maintenance, or car repair.
* **Account ID:** Email and password of the user's account.
* **Finger ID:** The finger sensor data about your finger that is linked to your account so that you can login.
* **Face ID:** The picture data about what your face looks like so that you can login to your account
* **Car/Computer System(s):** Anything that monitors the cars equipment like tire pressure by using sensors mounted at different parts of the vehicle.
* **Common Fix:** Any fix that requires average knowledge on cars and requires basic tools.
* **Complex Fix:** Any fix that requires above average knowledge on cars and more complex tools
* **Account:** The is the account the user has made in our servers

**1.4 About the Software**

The software must be connected to the computer system in the car. From that, the software provides information on things like tire pressure and mileage as it monitors the car when it’s on. If the software recognizes a problem within the car's system, it will warn the user about the problem. The user then can view what's specifically wrong with the car by viewing the information the software collects.

There are two things the software can do if a problem exists. One, it can provide instructions on how to fix the problem. Two, the software triangulates your position and provides information on nearby mechanic shops. If the second scenario occurs the software displays the information on the mechanic shops. this information entails: prices, phone numbers, address, distance, etcetera. Of course, the user can view the detailed data on their own and choose their own course of action.

Considering the two ways the user can deal with the problem, they will make the most financially viable choice for them and the app will help them if the user chooses to use the app.

# Section 2: Requirements

**Accessibility Requirements**

1. Software must allow the user to log in with face ID
2. Software must allow the user to log in with finger ID
3. Software must allow the user to log in with username & password

**Compatibility Requirements**

1. Software must be compatible with IOS devices
2. Software must be compatible with Android devices
3. Software must be able to perform the following requirements with all vehicles from the larger brands (Toyota, Kia, Ford, BMW, etc...)
4. Software must display Car’s model and specifications (For example: 2011 Hyundai Elantra GLS 1.8L 4 cyl)

**Offline Feature Requirements**

1. Software must provide history of previous issues with the car including dates
2. Software must recognize the source of the issue in any of the car’s internal system
3. Software must display source of the issue
4. Software must display description of the issue
5. Software must display whether the issue has a common fix
6. Software must provide an option to push notifications for general car servicing.
7. Software must warn the user if an issue arises
8. The software must be able to gather data on the car used by the user through the car's internal computer system
9. Software must indicate user in case of emergency problems and instructions to follow.
10. Software must display a step by step instruction set should the issue have a common fix

**Online Feature Requirements**

1. Software must display all of the nearby mechanic places within the user’s maximum distance
2. Software must allow the user to choose a maximum search distance
3. Software must display the prices associated with the fix at each of the nearby places
4. Software must allow the user to arrange an appointment at selected nearby place
5. Software must allow the user to access reviews for the nearby mechanic places
6. Software must provide a map with GPS that helps the user to locate nearby places based on their current location
7. Software must provide option to notify user whenever special deals are offered from the mechanic shops
8. Software must allow user to call mechanics within search distance should they have roadside assistance options
9. Software must be able to monitor multiple cars
10. Software must be able to search for cars

# Section 3: Design Choices

**3.1 Complex Design Choices**

*3.1.1 Car mechanics must provide their information in a database that is connected to the software so the user can receive information on the services.*

* This provides a way to have up to date information on the local market for car servicing. Thus, making the software a much more reliable for users.
* The main drawback is that we need to create deals with mechanic shops so that they put their information on these databases.

*3.1.2 The software must be able to convert raw data on the car's systems to simpler language.*

* Some people may not understand the technical jargon, so requiring this is imperative for welcoming a wider variety of users regardless of their past experiences or knowledge.
* When gathering the data on the car's performance, we would need to learn exactly what each value means and be able to implement a system that is capable of translating this data into good or bad traits.

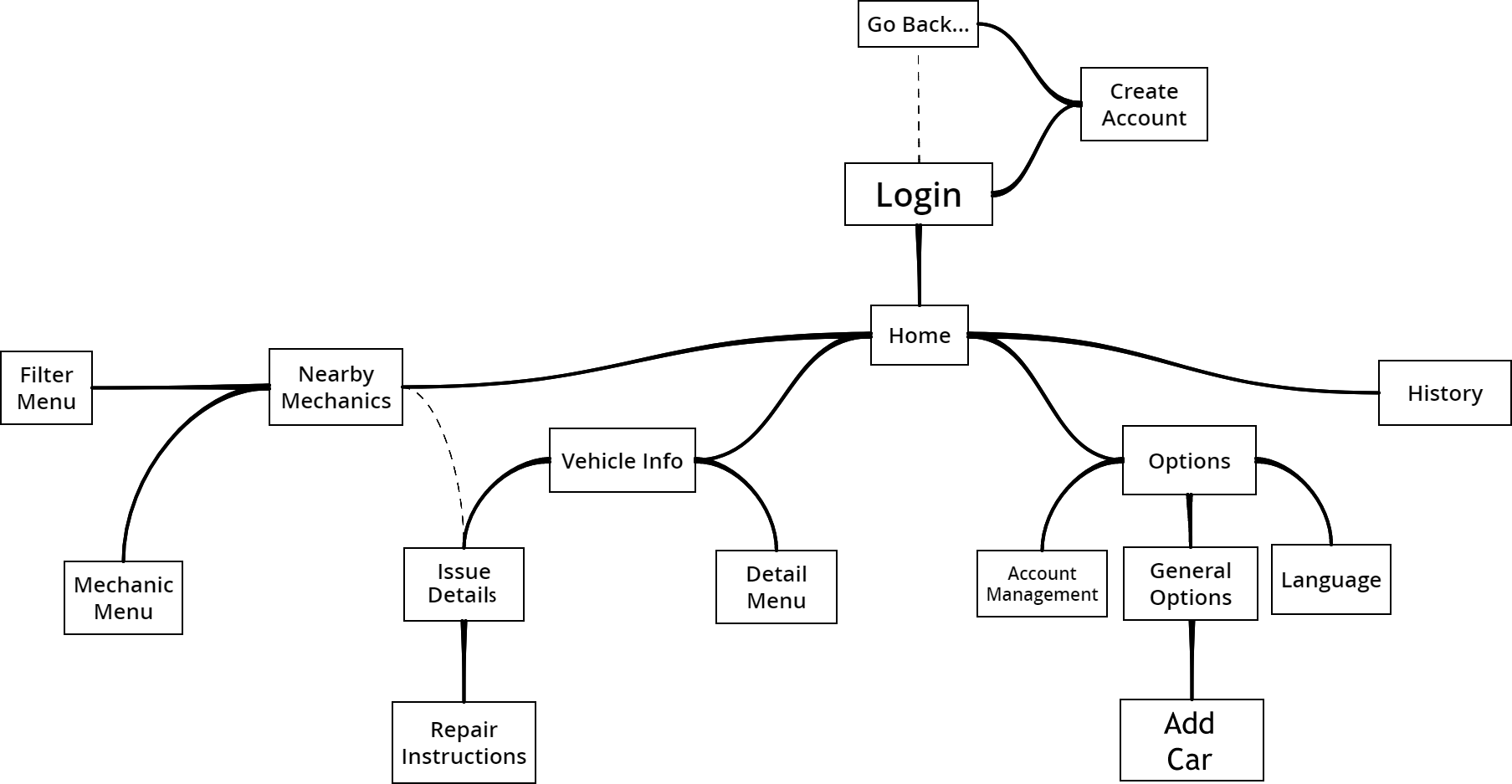
*3.1.3 The software must have basic fixes on the device and more complex ones on a server to be downloaded.*

* Doing this reduces the space the software takes up in the device. Users who have no interest in doing repairs on their own don't have to worry about the space taken from the app. Users who want to repair their car on their own are allowed to download more complex fixes and use the more basic fixes on their device already.

*3.1.4 The software must be able to show technical data on the car's systems for the user if they want to know it*

* This allows people more informed on how cars work to view their car's diagnostics in their own way instead of using the user-friendly language to explain what is going on with the car. Ultimately, this allows a different level of control for people who are far more experienced in cars.

**3.2 Key Design Components**



**Login:** This page allows the user to login to their account that they have created. They have the ability to go to the create an account page, login to their account (face ID, finger ID, or account ID), or reset password. We implemented a feature of accessibility, forgiveness, and constraint in this page because we allow the user to login in many ways, allow the user to reset their password if it is forgotten, and prevent the user from using a login technique that hasn't been established.

**Create Account:** In this screen, the user may set up their account that will be connected to our servers. They must provide their email, a password, and set up finger or face ID(optional). Constraint is used to prevent the user from setting up a login technique that their device is incapable of doing.

**Home:** This is where the user can navigate to all the other sections of the app freely. This page shows whether or not the app is connected to a car's systems. The user may navigate with the buttons provided to vehicle info, options, history, nearby mechanics, or logout. The principle of entry points is used here giving the user a point of prospect, which gives them a small insight of what to expect before the user actually uses the app.

**History:** This page is designed to show the user past problems, warnings and maintenance the car has encountered with timestamps. The principle of the cost-benefit tradeoff was used because a search bar is on the page used for searching for specific parts to reduce the work for finding the history of specific problems.

**Options:** This page is where the user may adjust a few things about how the app works. Here the user may choose to go to account management, general options, or language options.

**Account Management:** This page allows the user to change their password or email, authorize other devices to share the account, edit or create security questions/answers, and logout.

**General Options:** This page allows the user to allow metric conversion, turn on/off notifications, currency used, download extra fixes, allows the user to add a car, and delete a car.

**Add Car:** This screen allows the user to search for an existing car. The user may select the car and add it to the roaster of cars connected. The principle of confirmation will be used to prevent the user from accidentally adding the wrong car.

**Language Settings:** This page allows the user to change the language used in the app.

**Vehicle Info:** On this page, the user can view the connected car's model, year and specifications, and the list of the monitored parts with their condition. From here the user may also navigate to the detail menu or the issue details. The user can navigate to issue detail only if there is a problem detected and if they tap on the part that has the problem.

**Detail Menu:** All types of reading data is viewed in this screen. We use a component of the principle of comparison, benchmarking, and use benchmark variables to show the average expected output of each variable so that it is easy for the user to compare the values and make conclusions of their own.

**Issue Detail:** This section displays what the issue is and why it is a problem. We use an expository advanced organizer to supplement this information to help users understand. The user then can choose to use the app to find a nearby mechanic shop or fix it themselves with the two buttons provided. Selecting to use a mechanics help sends the user to the nearby mechanics page and selecting to fix it themselves sends them to the repair instructions page

**Repair Instructions:** This screen provides the user with the tools required to succeed in the task, step by step instructions with pictures, and a video available for quick and easy understanding. In this section, we use the principle of consistency, meaning we keep this page similar for each repair instruction so that users don't have to relearn the layout of the instructions every time.

**Nearby Mechanics:** On this screen, the user views a map centered on their GPS location. Here the user may view the nearby mechanics on the map and select which one they want to go to. On this screen, the user can also navigate to the mechanic menu and the filter menu.

**Filter Menu:** In this menu, the user may change their search criteria for the map. They can change the search distance on the map, filter by price of average checkup, by reviews, and by features.

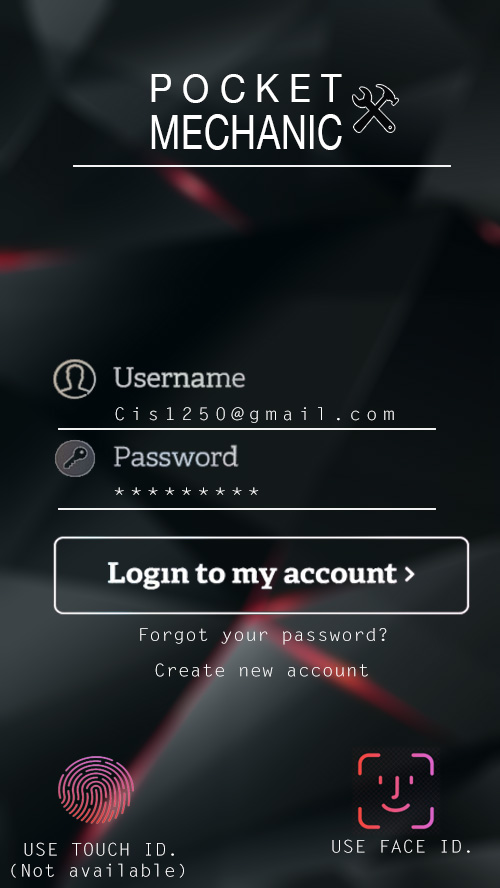
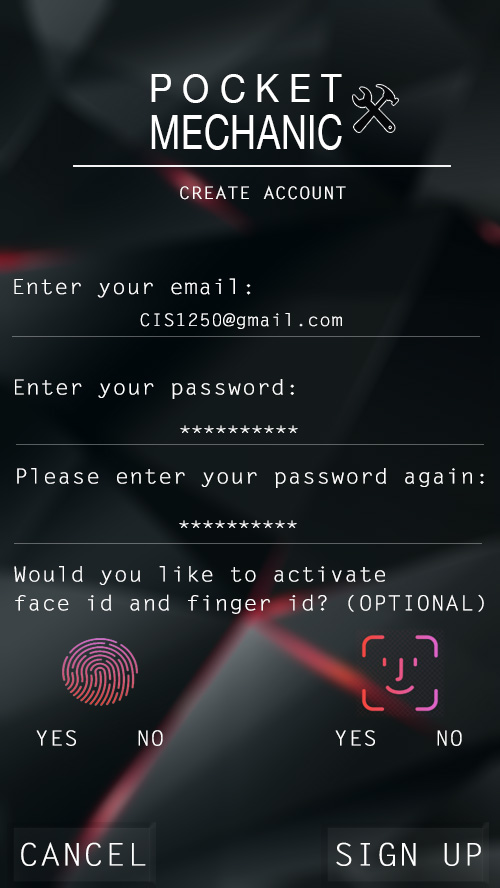
**Mechanic Menu:** This menu displays the information on a selected mechanic shop. It shows the telephone number, address, name, review score, and average price. The user can use the button "call" to call the shop to book an appointment.

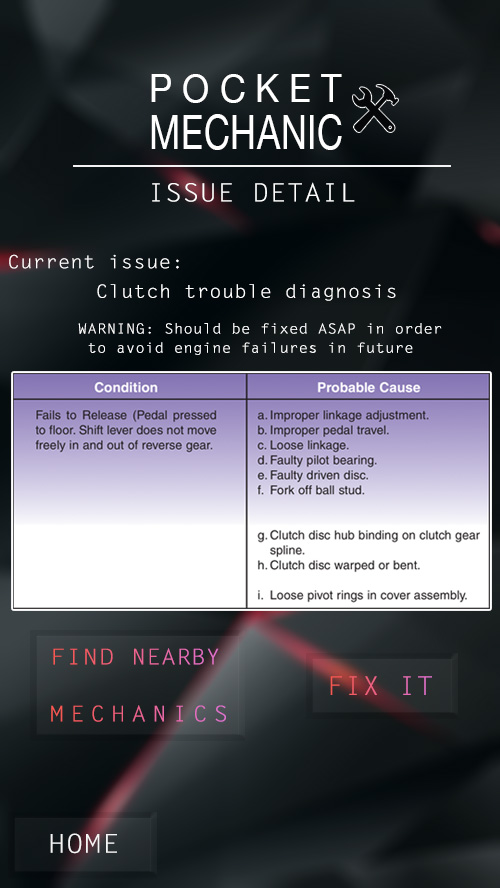
**Note:** At every page, the user may navigate back to home or back to the previous page.

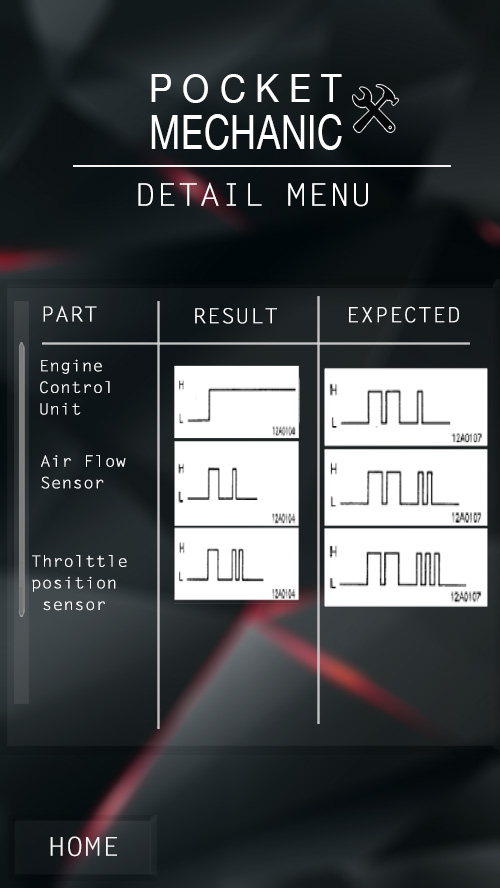
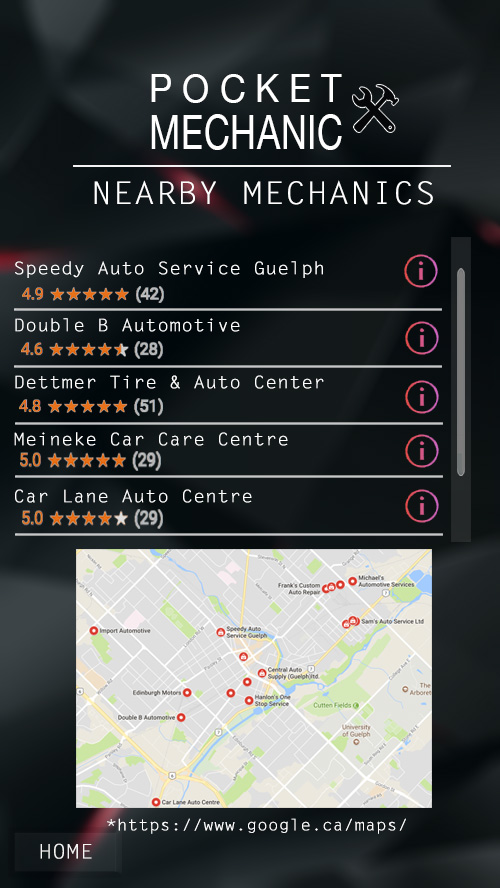
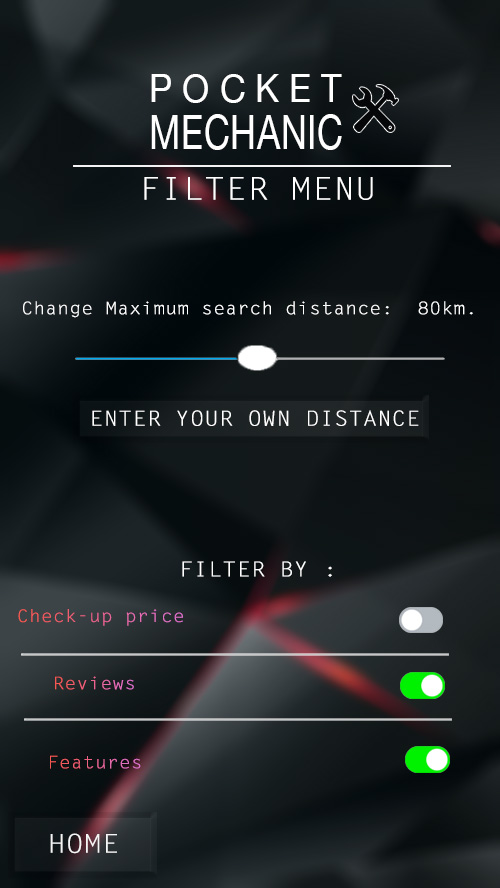
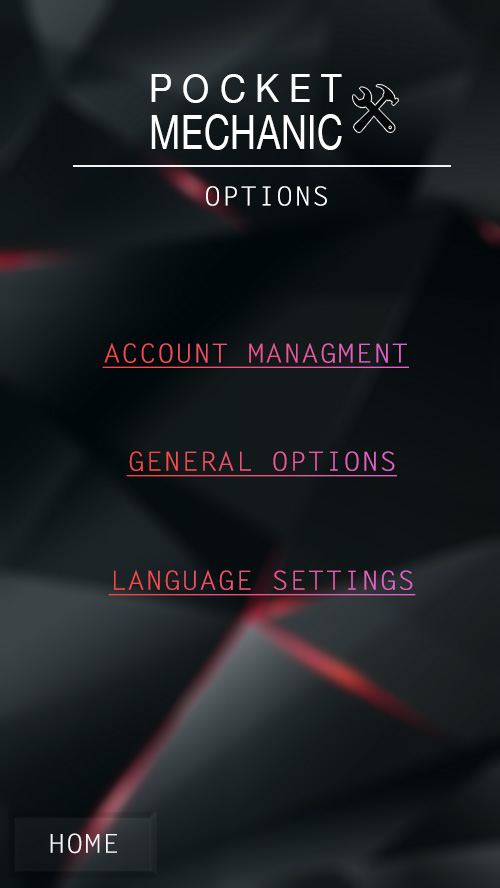
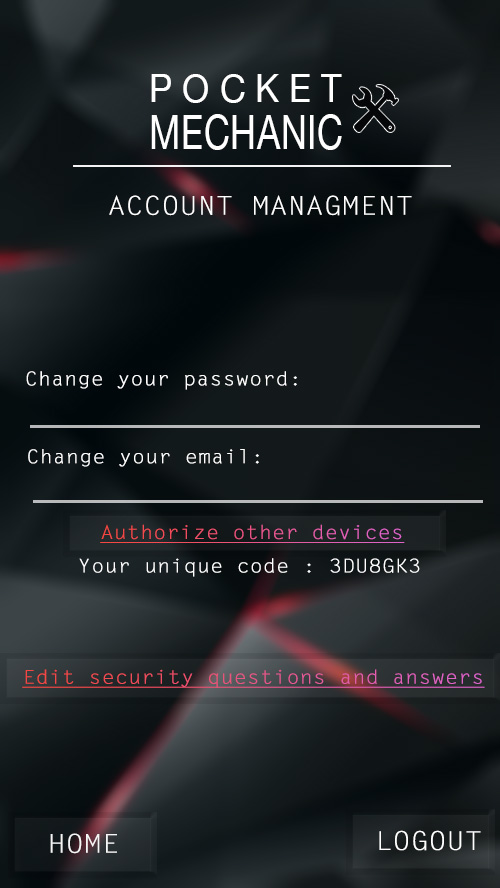
# Section 4: Limitations and constraints

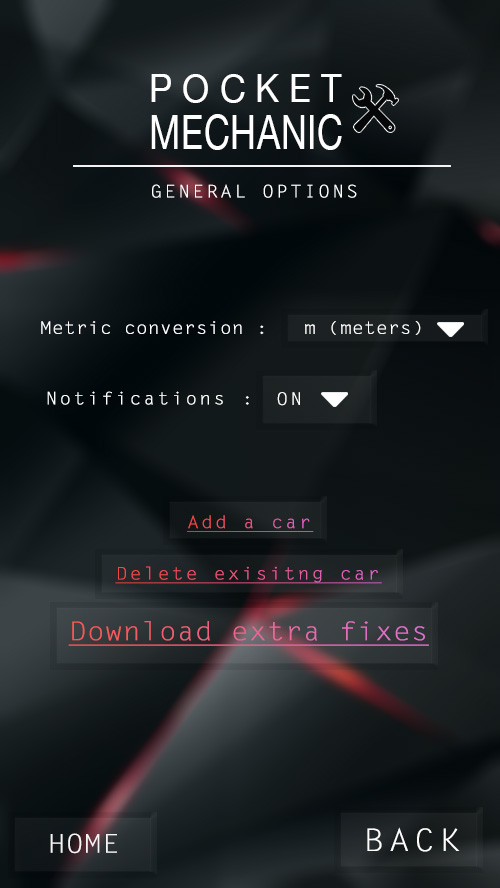
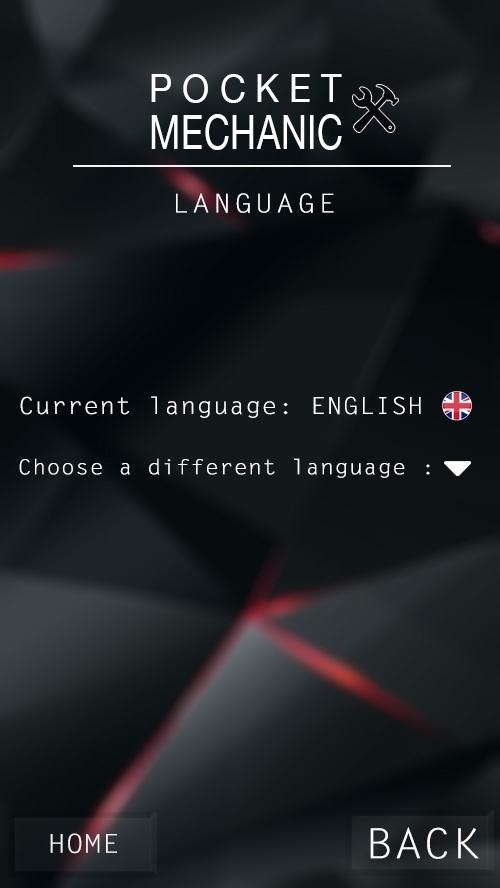
* Software needs to connect to a computer built inside the vehicle in order to receive information regarding the parts' condition from the user's vehicle. Problem is, some of the car models are not equipped with such a computer that provides the required information in order make this software possible.
* The software requires a database containing all mechanics as well as their prices for services which would limit the software's use to specific regions that the developers have looked though. This would also cause the database to be updated frequently in order to have the software not send people to closed mechanics as well as be able to send them to newly opened locations.
* This software requires the car manufacturers to set up integration with the vehicles on board computing systems to send messages to the user’s phone. This would most likely involve a large sum of money in order to convince them to put in this integration functionality.
* The software is limited only when the vehicle is turned on. Otherwise, no data will be received.
* The software requires constant connection to the internet in order to receive updated data from any of the mechanic shops, current location or get more complex fixes off the internet.
* The software requires a database and be constantly updating for vehicle fixes from the larger brands since many vehicles/brands have repairs that would only really work for those specific brands/models.
* The finger ID and face ID options would only be possible on certain models of phones that have those capabilities.

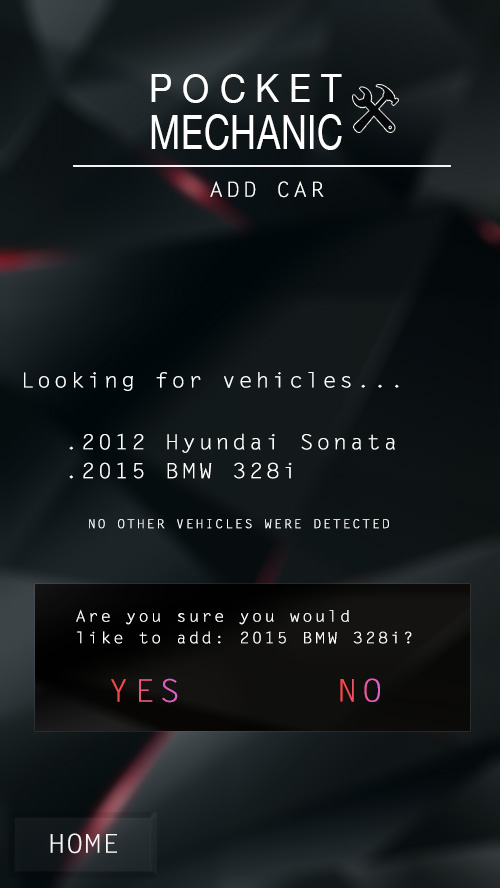
# Section 5: Software Prototype

# Section 6: Design Justification

* We made problems viewed on the vehicle info and history page stand out with colours
  + The principle of colours states that certain colours can bring the attention of the user so bright colours are used
* Many of the buttons on the screens are made to look similar to real buttons
  + Based on the principle of affordance the user will end up having an easier time using the app because of how the buttons will afford pressing and no one will have to explain to them that they have to do that
* Many elements on the screen shown above all center around the center of the screen
  + Based on the alignment principle this will effectively increase aesthetic appeal and boost the aesthetic-usability effect
* We chose that writing on each screen is minimal
  + Putting too much writing on one screen would cause certain users to dislike the app because of the amount of reading.
  + The 1-6-6/1-7-7 rule supports this and we try to use this when explaining the issues in the issue detail and repair instructions
* We chose to use icons to represent the features
  + In the home page, we used the principle of iconic representation because the images help the user to recognize it easier. Thus, user do not need to read the words to know what each one is, they can simply know its control by looking at the icons.
  + The principle of archetypes clearly explains that this choice will work to a high degree of effectiveness
* Scroll bars are used to indicate that the screen has more to show and we used sliders and drop-down bars to indicate that there are multiple selections for one option
  + This makes it easy to communicate to the user the amount of things on the screen
* Confirmation
  + We applied the principle of confirmation in the add car page because it gives the chance to the user to verify their action before it performed.

# Section 8: Summary

In summary, Pocket Mechanic is an application you can download on the phone and connect with your car to keep track on the many components of the car. First of all, you need to login with your account, if you don’t have one, you can always register for one. When you connect your car with the application, it tells you if there’s any problems with the car. The software also allows those with more knowledge of vehicles to look at more technical information about the components and their performance. You can either choose to fix it by yourself with the tutorial videos or the step by step instructions with images, or else find a nearby mechanic shop with the ability to call and look at reviews and prices.

# Section 9: References

William Lidwell, Kritina Holden & Jill Butler. (2012). *Universal Principles of Design, Revised and Updated: 125 Ways to Enhance Usability, Influence Perception, Increase Appeal, Make Better Design Decisions, and Teach through Design*.

Chris Bateman. (2007) *Design Document: Play With Fire* <https://www.gamasutra.com/view/feature/130127/design_document_play_with_fire.php>

Google Maps (2005). Google

<https://www.google.ca/maps>