Software Requirements Specification for Greenway: Save money on every trip

Team #11, Roadkill
Priyansh Shah, shahp36
Utsharga Rozario, rozariou
Jash Mehta, mehtaj8
Bilal Shaikh, shaikb2
Pranay Kotian, kotianp
Sharjil Mohsin, mohsis2

October 9, 2022

Contents

1 Introduction **1**
- 1.1 Purpose of Document . . . 1
- 1.2 Characteristics of Intended Reader . . . 1
- 1.3 Organization of Document . . . 1

2 Project Drivers **2**
- 2.1 The Purpose of the Project . . . 2
- 2.2 Scope of Requirements . . . 2
- 2.3 Stakeholders . . . 2
  - 2.3.1 Product Users . . . 2
  - 2.3.2 Development Team . . . 2
  - 2.3.3 Teaching Staff/Instructors . . . 2

3 General System Description **3**
- 3.1 System Context . . . 3
- 3.2 User Characteristics . . . 4

4 Behavior **5**

5 Requirements **6**
- 5.1 Functional Requirements . . . 6
- 5.2 Non-functional Requirements . . . 9

6 Likely Changes **15**

7 Unlikely Changes **15**

8 Traceability Matrices and Graphs **16**

Revision History

Date	Version	Notes
05-10-2022	1.0	Software Requirements Specification Version 1
08-10-2022	1.1	Software Requirements Specification Updated

1 Introduction

Navigation applications are commonly used applications while driving to get directions from point A to B, but these applications never tell you how much it costs you to get there or how much gas was used on the trip. When carpooling with friends, the driver of the vehicle always asks everyone in the group for gas money and often times these calculations are mere estimates that are not always very accurate. People often wonder how much it costs to get to a destination before starting your journey, having an application perform these cost and fuel calculations based on real time gas pricing information ensures you save the most money on your journey while minimizing gas usage to encourage a more sustainable lifestyle. The introduction section focuses on the" Purpose of the Document", the "Scope of Requirements", the "Characteristics of the Intended Reader", and the "Organization of Document".

1.1 Purpose of Document

The purpose of this document is to outline the overall description and target functionality of the application to stakeholders. It will communicate, without any ambiguity, the system requirements for Greenway and the different use cases and contexts of use. These design aspects will be communicated using a combination of diagrams and written descriptions. In addition, it will also serve as a reference for verifying correctness and validation of the product during the testing phases.

1.2 Characteristics of Intended Reader

The intended readers characteristics include having knowledge in the following domains, the understanding of mapping software, the knowledge of how data is stored and will be used in the context of the google maps API, which algorithms need to be used to calculate the most fuel efficient route, the understanding of how terrain information will be used to alter car mileage calculations, and the understanding of how information will be retrieved for all types of data being used. The intended reader needs this knowledge as it is required to understand the rest of the SRS document; this ensures the reader understands what the information is that is being conveyed and how the technology is going to be used to put out the final product.

1.3 Organization of Document

The document is organized into sections with smaller subsection that will aid in conveying information. The intention is that these sections will work together to provide a more informed understanding of the product. The document is seperated into sections that include, "General System Description", "Specific System Description", "Requirements", "Likely Changes", "Unlikely Changes", "Traceability Matrices and Graphs", "Development Plan", and the "Values of Auxiliary Constants".

2 Project Drivers

2.1 The Purpose of the Project

The goal of Greenway is to create an application that allows users to provide car information and a destination, in return the application supplies the amount of money it will take for the user to reach that destination using the most fuel-efficient route calculated by the app. The application uses real-time gas price data, information on mileage data, and terrain data to calculate and provide the amount of money it would take to reach a destination for users who drive many different types of gas vehicles; this allows them to use this app in many ways, such as, figuring out which gym may provide the most bang for their buck based on distance from their location.

2.2 Scope of Requirements

The scope of the proposed software, Greenway, is a mapping software that not only gives fuel efficient directions to the intended destination but provides the user with fuel cost calculations. The intention is to calculate fuel costs using gas price data, car mileage information and terrain information to show how much money it will take to get to the intended destination using the most fuel efficient route.

2.3 Stakeholders

2.3.1 Product Users

The product user is anyone who can drive and hence will be using the product. The product users have the most influence on the requirements of the application and its overall development since the purpose of the application is help these users choose an cost effective eco-friendly route for a journey.

2.3.2 Development Team

The development team is the group of people who are developing the application. They are the software developers who are developing the application and hence are an important stakeholder.

2.3.3 Teaching Staff/Instructors

The teaching staff and instructors are also important stake holders as they will be directly interacting with our product and assessing its usability. They are similar to investors and are required to be shown updates and hold influence on the requirements of the application and its development.

3 General System Description

This section provides general information about the system. It identifies the interfaces between the system and its environment and describes the user characteristics.

3.1 System Context

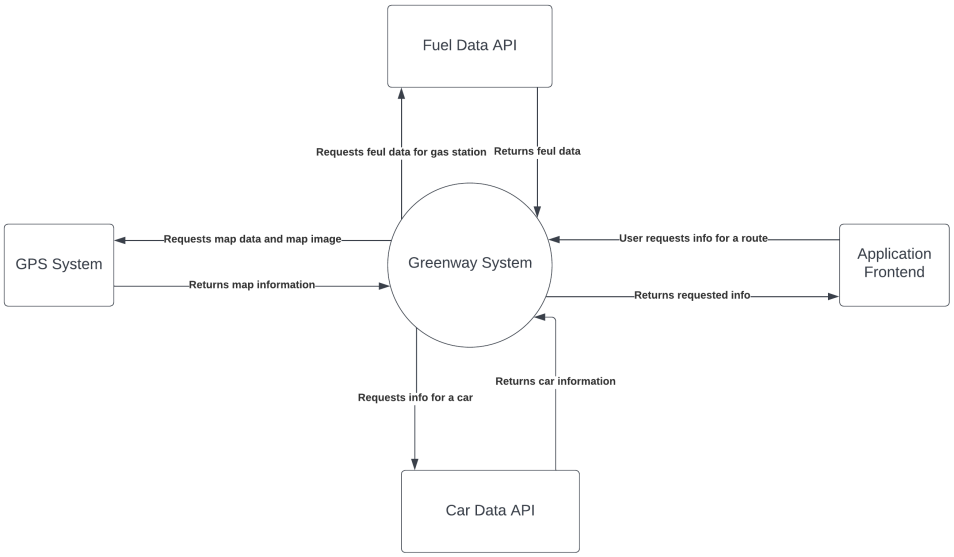

Figure 1: Context Diagram

- User Responsibilities:
  - Responsible for interacting with Application Frontend and receiving its output.
- Greenway System Responsibilities:
  - Responsible for calculating the total fuel price for trip with data from Feul Data API.
  - Responsible for interacting with GPS API to get all neccessary GPS data to present to user.
  - Responsible for interacting with Car Data API to get car information and process it.
- Application Frontend Responsibilities:
  - Responsible for allowing user to interact with application and displaying application output.

- Fuel Data API Responsibilities:
  - Responsible for sending Feul Station Data to any service requesting it.
- GPS System Responsibilities:
  - Responsible for sending any GPS data required to a service needing it.
- Car Data API Responsibilities:
  - Responsible for sending any Car data required to a service needing it.

3.2 User Characteristics

The users of this application are individuals who own gas powered vehicles and are looking for information on how much it will cost them in terms of fuel. This application will serve as a way of calculating the most fuel-efficient route taking into consideration elevation data, fuel mileage from an individuals car model, and the current gas prices. This users of app will have the ability to put in their cars information, giving them all the data they require to make an informed decision on if the destination they chose is worth going to.

4 Behavior

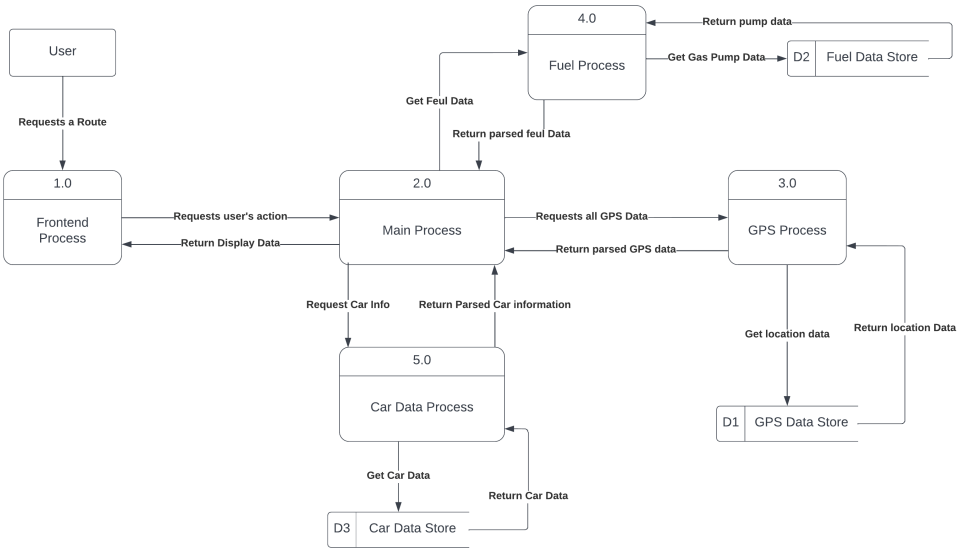

Figure 2: Application Behavior from a Data Flow Diagram

The Data flow diagram is used to demonstrate how the app will behave when data is provided to it through the user requesting a route. More specifically the frontend will access the main process with the request and then it will calculate everything it needs to with multiple different processes it accesses. Namely the Fuel Process will give it data regarding the fuel prices, GPS will give all positioning data and elevation along with anything else the program displays, and car data will provide car efficiency data. All of which will be used to calculate the final route and price to display on the front end.

5 Requirements

This section provides the functional requirements, the business tasks that the software is expected to complete, and the nonfunctional requirements, the qualities that the software is expected to exhibit.

5.1 Functional Requirements

FR1. The system shall allow the user a way to input a start location.

- Rationale: The application must know where the trip begins.
- Constants: None
- Undesired Event Handling: Display an error message and stop application.
- Normal Operation: The system is considered to be operating normally if the user can input a starting location for the trip.
- Priority: High

FR2. The system shall allow the user a way to input a final destination.

- Rationale: The application must know where the trip ends.
- Constants: None
- Undesired Event Handling: Display an error message and stop application.
- Normal Operation: The system is considered to be operating normally if the user can input a end location for the trip.
- Priority: High

FR3. The system shall allow the user a way to select any car model and make.

- Rationale: The system can use the users car model to internally calculate the fuel economy of the vehicle they will be driving in.
- Constants: None
- Undesired Event Handling: Assume the vehicle has the national average fuel economy of 8.9 L/100km.
- Normal Operation: The system is considered to be operating normally if the user can select any car by make/model and the system uses the corresponding fuel economy for its calculations.
- Priority: High

FR4. The system shall allow the user a way to input the fuel economy of their car.

- Rationale: The system can use a custom user-entered value to calculate the fuel economy.
- Constants: None
- Undesired Event Handling: Assume the vehicle has the national average fuel economy of 8.9 L/100km.
- Normal Operation: The system is considered to be operating normally if the user can input a numeric value for the fuel economy.
- Priority: High

FR5. The system shall have a way to update available car models from either user input or external sources.

- Rationale: The models available every year change and the system needs to account for newer cars entering the market.
- Constants: None
- Undesired Event Handling: If updates are unsuccessful, continue using the existing database of cars.
- Normal Operation: The system is considered to be operating normally if the user can update the fuel economy of a car.
- Priority: Low

FR6. The system shall have a map to show journey from start to final destination.

- Rationale: The user would appreciate having a visual of the route on a map.
- Constants: None
- Undesired Event Handling: Display a message stating Unable to display map.
- Normal Operation: The system is considered to be operating normally if the user can view a map of the overview of the route.
- Priority: Medium

FR7. The system shall display the gas stations along the route.

- Rationale: The user can easily see where to refuel if necessary.
- Constants: None
- Undesired Event Handling: Display an error message Unable to locate Gas Stations.
- Normal Operation: The system is considered to be operating normally if the map displays gas stations at their locations, indicated with a universal gas icon.
- Priority: Low

FR8. The system shall display gas prices at stations on route to destination.

- Rationale: A key feature of the system is to have real-time gas prices factor into the cost calculations.
- Constants: None
- Undesired Event Handling: Use most recent price available if real-time value is not available.
- Normal Operation: The system is considered to be operating normally if the map displays gas stations at their locations, along with the price listed at each station.
- Priority: High

FR9. The system shall have a way to calculate the most fuel efficient route.

- Rationale: The target user would like to minimize their fuel costs.
- Constants: None
- Undesired Event Handling: Display an error message Cannot calculate route.
- Normal Operation: The system is considered to be operating normally if the map displays an optimized route from the defined start and end points.
- Priority: High

FR10. The system shall have a way to collect elevation data along the suggested route.

- Rationale: The system must use elevation data to accurately determine fuel costs based on more uphill/downhill driving required.
- Constants: None
- Undesired Event Handling: Ignore the elevation if unavailable and assume flat drive.
- Normal Operation: The system is considered to be operating normally if different points on the map have appropriate elevations.
- Priority: High

FR11. The system shall have a way to calculate fuel consumption on different terrain elevations.

- Rationale: The system must use elevation data to accurately determine fuel costs based on more uphill/downhill driving required.
- Constants: None
- Undesired Event Handling: Ignore the elevation if unavailable and assume flat drive.

- Normal Operation: The system is considered to be operating normally if the fuel costs increase with more uphill routes and decrease with more downhill routes.
- Priority: High

FR12. The system must calculate the total cost of the journey.

- Rationale: The user must know how much their trip costs.
- Constants: None
- Undesired Event Handling: Display a message An error was encountered calculating the cost
- Normal Operation: The system is considered to be operating normally if there is an output displaying the total cost of the trip after performing the appropriate calculations.
- Priority: High

5.2 Non-functional Requirements

NFR1:
- *Description*: The system shall have a modern minimalist user interface.
- *Rationale*: Using popular modern design principles to create an appealing visual style for the system will make the system easy to use and attractive to users.
- *Fit Criterion*: The systems UI must follow modern minimalist design principles as defined in popular style guides. Users feedback will be used to assess the overall design of the interface.
- *Undesired Event Handling*: Given poor user feedback on interface design, user suggestions will be considered and overall interface design will be reevaluated.
- *Priority*: Medium

NFR2:
- *Description*: The system shall have a consistent style across the different pages (or interfaces) of the platform.
- *Rationale*: System will appear more professional. User experience will be more streamlined and less confusing if the system maintains a cohesive style.
- *Fit Criterion*: Each page/interface shall be checked against a list of common style/design elements that should remain consistent within the project. User feedback will be used to assess style consistency between interfaces.
- *Undesired Event Handling*: Given feedback on inconsistent style, alterations will be made to the interface to address specific feedback.
- *Priority*: Medium

NFR3:
- *Description*: The system shall have a user interface which is intuitive and easy to navigate.

- *Rationale*: Ease of use improves learnability and memorability of the system.
- *Fit Criterion*: User feedback will be used to assess ease of use. Metrics such as time taken for new user to complete sample task will be measured.
- *Undesired Event Handling*: Given poor ease of use metrics, the layout, simplicity, and flow of system will be reevaluated and alterations made.
- *Priority*: High

NFR4:
- *Description*: The system shall retain the users previous destinations.
- *Rationale*: Reduces time taken to select destination if user has frequently used destinations. Improves ease of use (NFR 3) by reducing time needed to complete task.
- *Fit Criterion*: Are the users previous destinations available to the user? Yes/No.
- *Undesired Event Handling*: If no, functionality should be implemented.
- *Priority*: Medium

NFR5:
- *Description*: The system shall have a guide (or short tutorial) for those who are using the application for the first time.
- *Rationale*: The guide/tutorial will introduce how to use all the features offered by the system.
- *Fit Criterion*: Several metrics measuring task completion will be compared: first time user without tutorial, first time user with tutorial, repeat user without tutorial. User feedback will also be obtained. Majority of users 80% should feel comfortable with system functionality after viewing the tutorial.
- *Undesired Event Handling*: Given poor learnability, tutorial/guide and overall layout and ease of use (NFR 3) shall be reevaluated and alterations made.
- *Priority*: High

NFR6:
- *Description*: The system shall hide the details of its implementation from the user.
- *Rationale*: The implementation (e.g. code, algorithms, etc.), is only relevant to the developers. Users are only concerned about the application functionality and outputs.
- *Fit Criterion*: During software testing, developers can test to see if abstraction was properly used to hide the details of the implementation of the system from the user.
- *Undesired Event Handling*: Given poor abstraction, details will be hidden or omitted from the user interface.
- *Priority*: High

NFR7:
- *Description*: The system shall be accessible to users with visual impairments.
- *Rationale*: The system shall be accessible to users with visual impairments.
- *Fit Criterion*: The system will be tested in conditions where the user is partially or completely visually impaired.
- *Undesired Event Handling*: Given poor accessibility, modifications to the system will be made to improve the usability of the system under these conditions.
- *Priority*: Medium

NFR8:
- *Description*: The system shall complete the route and fuel consumption calculation in a reasonable amount of time (¡5 seconds).
- *Rationale*: The user should not have to wait a very long time after they input their destination into the application.
- *Fit Criterion*: The time between the user input and the route calculation shall not exceed 5 seconds. The system calculation time should be comparable to the default google maps route calculation time.
- *Undesired Event Handling*: Given poor calculation time, algorithm time efficiency and other sources of lag should be considered and fixed (if possible).
- *Priority*: Low

NFR9:
- *Description*: The system shall calculate fuel consumption as accurately as possible.
- *Rationale*: We want our customers to know exactly how much fuel will be consumed during the trip. It is vital that the value be accurate so that they can plan their trip accordingly.
- *Fit Criterion*: In a panel of 50 users using the application, 80% of the users are satisfied with the calculation.
- *Undesired Event Handling*: N/A.
- *Priority*: Medium

NFR10:
- *Description*: The system shall be available for use 24 hours per day, 365 days per year.
- *Rationale*: Since this is a web app, it will be available for use at any time. Users will be able to use the application at any time of the day without any restriction.
- *Fit Criterion*: 90% of a panel of 10 users shall be able to use the application without it going offline.
- *Undesired Event Handling*: N/A.
- *Priority*: Low

NFR11:
- *Description*: The system shall be able to respond to invalid or unconventional inputs.
- *Rationale*: Ensures user error will not cause bugs or impede system functionality.
- *Fit Criterion*: Test cases with invalid inputs and outliers will be created to test this non-functional requirement.
- *Undesired Event Handling*: N/A.
- *Priority*: High

NFR12:
- *Description*: The system shall store the last 10 addresses that the user input and display it in the search bar.
- *Rationale*: For the users convenience, the last 10 search results shall be displayed in case the user wants to go to any of those locations again.
- *Fit Criterion*: After a new address is searched, that search result is added in the last 10 recent results. After 10 searches with different addresses, the oldest search result is overwritten.
- *Undesired Event Handling*: In case nothing shows up, the system shall just show a regular search bar with no history.
- *Priority*: Low

NFR13:
- *Description*: The system shall be able to handle an infinite amount of users.
- *Rationale*: This system shall stay available for anyone who is concerned about spending too much for a particular trip.
- *Fit Criterion*: As a local server will be hosting the web app, all users should be able to access the application without being blocked.
- *Undesired Event Handling*: In case nothing shows up, the system shall just show a regular search bar with no history.
- *Priority*: Medium

NFR14:
- *Description*: The system shall operate as long as possible.
- *Rationale*: This system shall stay up indefinitely for users who are concerned about how much gas will be used up during their trip.
- *Fit Criterion*: 9 out of 10 users shall be able to use the application without it being unusable.
- *Undesired Event Handling*: N/A
- *Priority*: Low

NFR15:
- *Description*: The system shall be able to run on all modern browsers.
- *Rationale*: Users with the system on many different platforms will be able to use it.

- *Fit Criterion*: Opening the HTML executable file with all different modern browsers will work.
- *Undesired Event Handling*: N/A
- *Priority*: Medium

NFR16:
- *Description*: The system shall have a simple installation process for all potential users.
- *Rationale*: This system shall have a simple installation guide provided with easy steps so that any user should be able to install the web app.
- *Fit Criterion*: 90 percent of a panel of 20 potentials users will be able to install the application without any hassle.
- *Undesired Event Handling*: In case of installation errors, they can refer to the documentation on GitHub about alternative ways to install.
- *Priority*: High

NFR17:
- *Description*: The system shall be available to download for the public via GitHub.
- *Rationale*: This system shall be distributed from the GitHub website for free for everyone.
- *Fit Criterion*: All users should be available to pull the repository and install the application in their personal device.
- *Undesired Event Handling*: N/A
- *Priority*: Medium

NFR18:
- *Description*: The systems code shall be well documented and clear to be easily understandable for any developer maintaining the code.
- *Rationale*: If a new developer went to work on the piece of the system, they should be able to easily understand what was previously worked on.
- *Fit Criterion*: When inspecting the code, comments will be found describing what that section does.
- *Undesired Event Handling*: If any documentation is missing, the developer of the code shall be contacted as soon as possible to provide the corresponding documentation.
- *Priority*: High

NFR19:
- *Description*: Supporting documentation shall be available on GitHub, which includes a users guide on how to use the application.
- *Rationale*: In order for users to understand how to download, launch the application, and access relevant data, there should be documentation to explain.

- *Fit Criterion*: When downloading the system, the supporting documentation will be found on the GitHub site.
- *Undesired Event Handling*: If any documentation is missing, the developer of the code shall be contacted as soon as possible to provide the corresponding documentation.
- *Priority*: Medium

NFR20:
- *Description*: The system shall be compatible to run between Windows, Mac, and Linux-based operating systems.
- *Rationale*: In order to support a wide variety of users, we need to allow users on the most commonly used operating systems to download and use the system.
- *Fit Criterion*: Downloading and running the files on all mentioned operating systems will show compatibility.
- *Undesired Event Handling*: If the application does not work on a certain operating system, a support tool will be provided to contact the developers to fix the issue.
- *Priority*: Medium

NFR21:
- *Description*: System should comply with the Google Maps Platform Terms of Service.
- *Rationale*: Mandatory compliance requirement when using the Google Maps Platform API.
- *Fit Criterion*: The system will be checked directly against each point in the terms of service.
- *Undesired Event Handling*: If the system, at any point, does not comply with the terms of service, the system will be changed to comply with the terms.
- *Priority*: High

6 Likely Changes

LC1: The addition of the application being able to use mileage data as well as gas tank size to calculate the most efficient route on a long journey, allowing for the least amount of stops to fill up gas.

LC2: The addition of the application taking data of electric cars to allow the use of not only gas station but charging stations for electric car users.

LC3: The application will be able to be used in all of Ontario with elevation and gas data for all areas.

LC4: Future changes should allow the application to be used with other apps other than Google Maps, such as Wayz or Apple Maps.

7 Unlikely Changes

ULC1: The application will not support other provinces or countries.

ULC2: The application will not be able to update car catalog information on its own and will require an update to the application when new cars are introduced.

ULC3: The application will not support hybrid vehicles as they require complex calculations to determine when the electric component of the vehicle is used and when the gas component is.

8 Traceability Matrices and Graphs

The purpose of the traceability matrices is to provide easy references on what has to be additionally modified if a certain component is changed. Every time a component is changed, the items in the column of that component that are marked with an "X" may have to be modified as well. Table 1 shows the dependencies of Functional Requirements and Non-Functional Requirements.

NFR IDs	FR IDs											
	FR1	FR2	FR3	FR4	FR5	FR6	FR7	FR8	FR9	FR10	FR11	FR12
NFR1	X	X	X	X		X	X	X				
NFR2	X	X	X	X		X	X	X				
NFR3	X	X	X	X		X	X	X				
NFR4						X						
NFR5	X	X	X	X		X	X	X				
NFR6					X				X	X	X	X
NFR7	X	X	X	X		X	X	X				
NFR8									X	X	X	X
NFR9				X								X
NFR10												
NFR11	X	X	X	X								
NFR12						X						
NFR13												
NFR14												
NFR15												
NFR16												
NFR17												
NFR18												
NFR19												
NFR20												
NFR21												

Table 1: Traceability Matrix: Functional Requirements to Non-Functional Requirements

References

Appendix — Reflection

There are many skills needed to accomplish the task, these skills however need to be distributed amongst the team as it makes it easier to complete the goals that have been set to acquire a finished product. The domain specific knowledge required ranges from understanding APIs to collecting, storing, and using data. The team requires the knowledge of how to collect the different types of data required, gas price data, elevation data of a specified area, car catalog data that contains information about gas mileage, and google maps data for fuel efficient route calculations. The intention is to use this data as a collective to perform calculations which would allow the user to find the cost of travel from one destination to another in the simplest manner; for this, an understanding of how the Google Maps API works, how the application will be developed, how the interface will be designed, and how everything will work simultaneously to provide the user with the best experience. These tasks are split amongst the team.

Jash — Collection of gas price data and how it will be stored and how the fuel calculation will be performed.

Priyansh — Collection of elevation specific data and how it will be stored and how the fuel calculation will be performed.

Utsharga — Collection of car catalog data and how it will be stored and how the fuel calculation will be performed.

Sharjil — Understanding of how the interface will be designed and how the fuel-efficient route will be calculated.

Bilal — Understanding of how the Google Maps API works and how the fuel-efficient route will be calculated.

Pranay — Understanding How the interface will be designed and how everything will work simultaneously.

There are many approaches to acquiring this knowledge; one approach is researching in the specific domain through forums and google searches. Another approach is getting hands on with the entity you would like to learn, this allows one to experience and interact with it, giving a greater understanding of how everything works.

Jash — The approach for collecting gas price data will be through the research domain, this allows him to find information about the gas prices as well as learn how to collect and store them in a specific way. The same approach applies to the fuel calculations as understanding how its done and factoring in the elevation data is going to be used.

Priyansh — The approach for collecting elevation data will be done through the research domain, this makes sure the data being collected is relevant to the information required

to perform the fuel calculations. The same approach applies to the fuel calculations as understanding how its done and factoring in the elevation data is going to be used.

Utsharga — The approach for collecting car catalog data will be done through the research domain, this allows him to collect all necessary data on all existing cars to be used in production. The same approach applies to the fuel calculations as understanding how its done and factoring in the elevation data is going to be used.

Sharjil — The design of the user interface will be done through the hand-on domain, this allows him to interact with the product and test to see what works and what doesnt. The calculations for the fuel-efficient route will be done through the research domain, this allows him to create and work on the algorithm required to perform the calculation.

Bilal — The Google Maps API will require the hand-on domain to be used to explore how it works and how it will be used for the purposes of this application, this will allow him to learn what to do and how to use it for our needs. The calculations for the fuel-efficient route will be done through the research domain, this allows him to create and work on the algorithm required to perform the calculation.

Pranay — The design of the user interface will be done through the hand-on domain, this allows him to interact with the product and test to see what works and what doesnt. To get everything working together, he will have to look into the hand-on domain to be able to test and see how all the different parts will need to be integrated together to allow for the product to function as intended.