

# Deliverable #1 Template : Software Requirement Specification (**SRS**)

SE 3A04: Software Design II – Large System Design

**Tutorial Number:** T03

**Group Number:** G8

**Group Members:**

- Adam Mak
- Eric Chen
- Justin Ho
- Ahmad Hamadi
- Kevin Ishak
- Jonathan Jiang

# 1 Introduction

## 1.1 Purpose

The purpose of this **SRS** is to define the functionality and characteristics that the carpool app should have. It should provide a detailed overview of all the processes and systems that needs to be present in the final product. It describes the system and user interaction with the application. Intended audience for the **SRS**:

- Developers
- UI/UX Designers
- Product Managers
- Engineers
- System Architects
- Quality Assurance Testers

## 1.2 Scope

The software product is a taxi carpool service. It will provide a communication environment for passengers to carpool via taxis. This service is not meant to be replacement for taxi services (i.e. Uber-like services).

The objective of the software (local taxi system) being implemented is to transport users from one destination to another using a carpool method. This software allows user to enter a destination along with some optional search criteria that they want to arrive at. The software will then match the user with a taxi nearby which may or may not carry other individuals who are travelling in the same general direction. This benefits both the users, taxi driver as well as the environment by as it helps reduce costs, whether it be the fare for the ride or the reduction in environmental pollutants by the means of carpooling.

There are no higher-level specifications that exist, thus, any lower-level specifications must be consistent with this document, as it is the highest-level specification.

## 1.3 Definitions, Acronyms, and Abbreviations

Abbreviation/Acronym	Definition
API	Application Programming Interface
Carpool	To collectively share a ride (either to help reduce the cost of transit or as an act of being more environmentally “green”)
DBMS	Database Management System
GoPlay Store	The default Android distribution platform
SMS	Software Requirements Specification
SRS	Simple Message Service

## 1.4 References

- Provide a complete list of all documents referenced elsewhere in the **SRS**.
- Identify each document by title, report number (if applicable), date, and publishing organization.
- Specify the sources from which the references can be obtained.
- Order this list in some sensible manner (alphabetical by author, or something else that makes more sense).

## 1.5 Overview

In the order of occurrence, the remainder of the **SRS** contains an overview of the system, which includes product perspective, product functions, user characteristics, constraints, and dependencies. The **SRS** will have a descriptive model, which will be represented visually in the form of a use-case diagram. Furthermore, the document also includes various system-specific requirements which consist of functional and non-functional requirements. Finally, at the end of the document, there is a record which represents the division of labour of all contributors to the project.

## 2 Overall Product Description

### 2.1 Product Perspective

The carpool app product is independent and self-contained, as it does not build upon any existing products/applications. However, the standalone system uses and owns interfaces to and from external systems, such as hardware, users, databases, and other **API**'s, to provide essential functionality to the app. Figure 1 shows how these external systems will interact in a block diagram.

The carpooling app is developed for Android devices, providing users owning these devices with easy access to this service. Also, the app's user interface interacts with an Android device to allow users to see information and interact with the application through a touchscreen.

The app sends and receives data from a database that stores information about users, rides, route information, and other related data. This is managed by a **DBMS**.

The carpooling app uses the Google Map **API** to provide maps and location-related services such as route planning and estimating travel times. The **API** provides information about the locations of users and destinations, as well as real-time traffic data, to help drivers make more informed decisions about their routes.

To protect sensitive information such as passwords, payment information, transmitted messages, and other confidential data, the carpooling app uses a cryptosystem to encrypt this information before it is stored in the **DBMS**. When the data is needed, it is decrypted and used by the app. This ensures that information is protected from unauthorized access.

The carpooling app uses **SMS** functionality to provide communication between users to provide updates, confirm ride details, and more. For instance, when a ride is requested, the app may send a message to the driver to notify them of the request, and then another message to the passenger to confirm the ride details.

### 2.2 Product Functions

- App should compare carpool price with non carpool price and display the money saved as a point system.
- App should allow carpoolers to setup/join other carpoolers using the app.
- When requesting a carpool, carpoolers should be able to filter through different travel options (i.e. sort by closest cab/highest rated cab driver). The app should return a list of cab options according to the specification.
- Carpoolers looking to set up a carpool should be able to input the taxi ID , destination, maximum number of other carpoolers, as well as potential auxiliary information. They system should post this info and allow other carpoolers to join.
- Users should be able to create and delete accounts, which the system should store.
- The system should check if a user that wants to request/setup a carpool is a registered user. It should reject unregistered users.

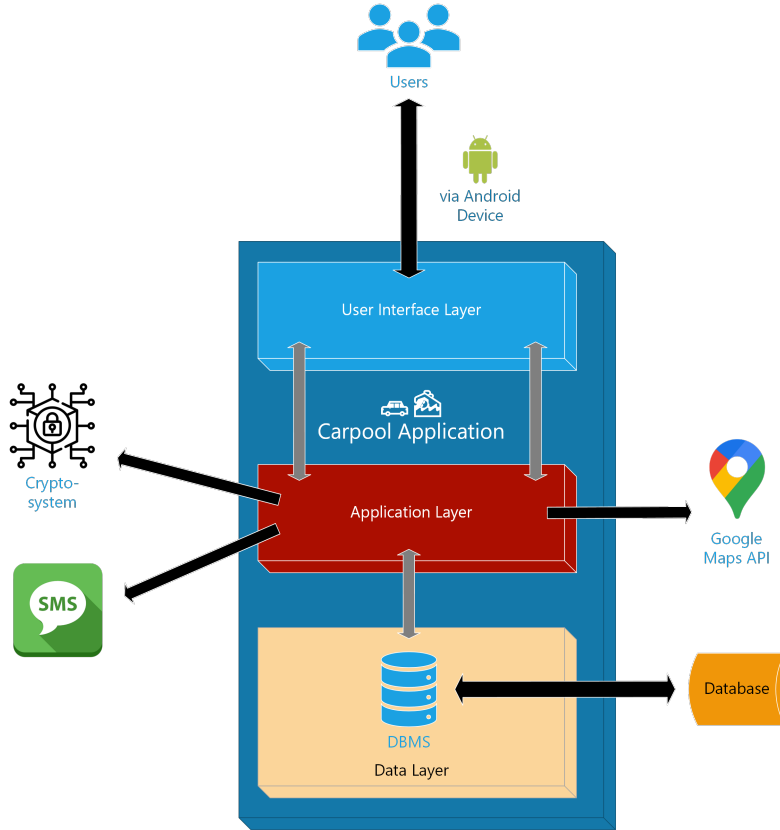


Figure 1: A block diagram showing the major layers of the carpooling application, interconnections, and external interfaces.

- App should have a reward system that allows users to obtain free rides based on their accumulated cost savings.

### 2.3 User Characteristics

Users include both passengers and drivers, of which age and education level may vary. The users of the product are expected to have a basic knowledge of technology and are familiar with smartphones, which the app is developed for.

### 2.4 Constraints

- Legal and regulatory restrictions: The developer needs to ensure that the application complies with the local laws and regulations.
- Technical limitations: The developer may face limitations with regards to the technology used to build the application, such as compatibility with different platforms, scalability, and security.
- Competition: They need to differentiate their product and offer unique features.
- Budget and resources: The availability of financial and human resources can impact the scope and timeline of the project.

## 2.5 Assumptions and Dependencies

The carpool app will be available for installation/download through the Apple and Google Play store for users to use. It is assumed that Apple/Google Play will streamline our app upon the necessary guidelines being met (I.e., Safety, Privacy, Performance, Business, Design, Legal).

It is assumed that the Google Maps **API** will be available to use with an 100% uptime which our carpool app will utilize to plan its routes and navigate users around cities. If the Google Maps **API**, for some extenuating circumstances, is unable to provide the services for route navigation, then another GIS **API** system may have to be utilized (e.g., OpenLayers, ESRI, etc. . . ).

A primary telecommunication company will be managing the **SMS** traffic of the carpool app as the app makes use of **SMS** functionality to inform users about the status of their ride. If for some reason, any of the telecommunication companies listed within this **SRS** document disbands, terminates the contract or breach some sort of confidentiality between both parties, and/or something similar in nature, then the **SRS** might have to change accordingly to fill up the vacant **SMS** management company.

It is assumed that the carpool app will be systematically be calculating the cost of the trip per person on a reduced per mileage basis.

## 2.6 Apportioning of Requirements

The following requirements are usually delayed until later versions to meet the initial launch deadline and budget, or to focus on delivering the most valuable features to users first.

1. Advanced features: These are features that are not essential to the core functionality of the app, but can be added later as the app evolves.
  - For example, an in-app tipping feature, or integration with a rewards program.
2. Non-critical performance optimizations: Improving the app's performance can be a continuous process, and some optimizations can be deferred until later versions.
  - For example, optimizations that improve the app's speed, reliability or efficiency, but are not essential to its basic functionality.
3. Complex user interfaces: Advanced user interfaces can be time-consuming to develop and may be deferred in favor of a simpler interface in the initial release.
  - For example, a more sophisticated design that includes augmented reality, or a chatbot interface.
4. Integrations with third-party systems: Integrating the app with other systems can be a significant effort, and this work may be deferred until later versions.
  - For example, integration with a payment system, or with a local transit system.
5. Support for older platforms: Supporting older platforms can be time-consuming, and this work may be deferred until later versions or only supported for critical security and bug fixes.
  - For example, supporting older versions of the iOS or Android operating systems.

It is important to prioritize the most critical requirements, such as the core functionality of booking a ride, and ensuring that the app is easy to use. Other requirements can be deferred until future versions to meet the initial launch deadline and budget.

### 3 Use Case Diagram

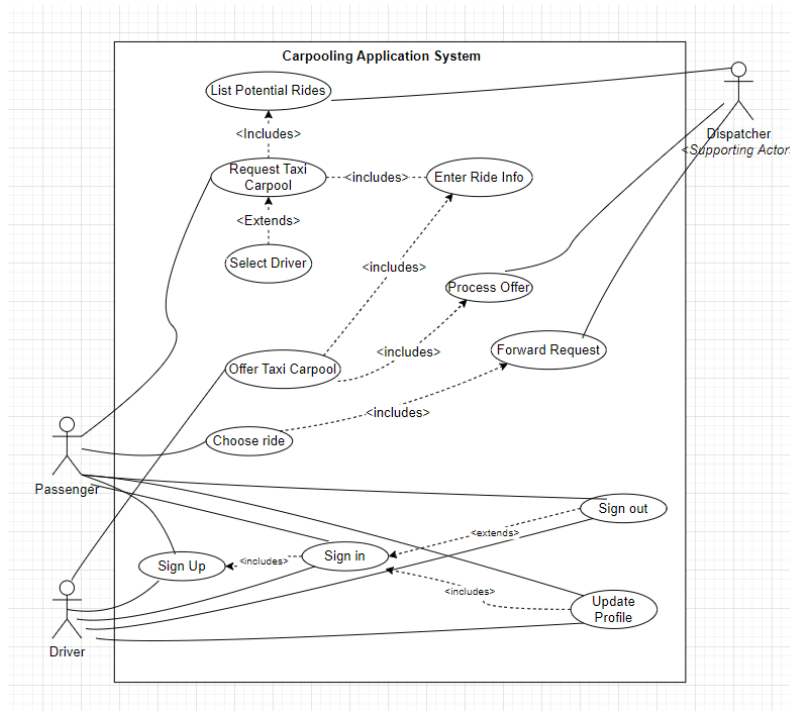


Figure 2: Use case diagram of the carpooling application.

### 4 Highlights of Functional Requirements

The business events we will present are:

- BE1.** Partake in a ride
- BE2.** Leave a rating on the app
- BE3.** Setup/Update user profile/account information
- BE4.** Cancel a ride
- BE5.** Login/Logout of an account
- BE6.** Edit Payment Information

The viewpoints we will consider are:

- VP1.** Passenger
- VP2.** Driver

**Interpretation:** Specify any liberties you took in interpreting business events, if necessary.

- BE1.** Partake in a ride
  - VP1.** Passenger

- $S_1$ : System asks user for a pickup location and a destination point.
- $E_1$ : Passenger inputs a pickup location and a destination point.
- $S_2$ : System asks if passenger would like to carpool with nearby users travelling in the same general direction.
- $E_2$ : Passenger decides whether to carpool with nearby users.
  - $E_{2.1}$ : User inputs their intent to carpool with nearby users. **⟨Branch BE1.VP1.S3.1⟩**
  - $E_{2.2}$ : User inputs their intent to not carpool with nearby users. **⟨BE1.VP1.1.S3.2⟩**
- $S_3$ : System calculates the most suitable path to take to get from pickup location to destination location and display the total cost of travel.
  - $S_{3.1}$ : System calculates a one-way route from pickup to destination location with no stops.
  - $S_{3.2}$ : System calculates a suitable route from pickup to destination with the capability of other users partaking in a carpool ride.
- $E_3$ : User decides whether to take the ride or not.
  - $E_{3.1}$ : User decides to reject the ride. **⟨Branch EXIT⟩**
  - $E_{3.2}$ : User decides to accept the ride. **⟨Branch to BE1.VP1.2.S1⟩**

#### VP2. Driver

- $S_1$ : System notifies driver to pick up passenger.
- $E_1$ : Driver picks up the passenger and drops them off at their destination.

#### Global Scenario:

- $S_1$ : System asks user for a pickup location and a destination point.
- $E_1$ : Passenger inputs a pickup location and a destination point.
- $S_2$ : System asks if passenger would like to carpool with nearby users travelling in the same general direction.
- $E_2$ : Passenger decides whether to carpool with nearby users.
  - $E_{2.1}$ : User inputs their intent to carpool with nearby users. **⟨Branch BE1.VP1.S3.1⟩**
  - $E_{2.2}$ : User inputs their intent to not carpool with nearby users. **⟨BE1.VP1.1.S3.2⟩**
- $S_3$ : System calculates the most suitable path to take to get from pickup location to destination location and display the total cost of travel.
  - $S_{3.1}$ : System calculates a one-way route from pickup to destination location with no stops.
  - $S_{3.2}$ : System calculates a suitable route from pickup to destination with the capability of other users partaking in a carpool ride.
- $E_3$ : User decides whether to take the ride or not.
  - $E_{3.1}$ : User decides to reject the ride. **⟨Branch EXIT⟩**
  - $E_{3.2}$ : User decides to accept the ride. **⟨Branch to BE1.VP1.2.S1⟩**
- $S_4$ : System notifies driver to pick up passenger.
- $E_4$ : Driver picks up the passenger and drops them off at their destination.

#### BE2. Leaving a rating on the app

#### VP1. Passenger

- $S_1$ : System attempts to authenticate the passenger and confirm that passenger has taken a ride previously.  
 $S_{1.1}$ : System successfully authenticates the passenger and displays a list of applicable (previous carpool partners) passengers to leave a review for **⟨Branch to BE3.VP2.1.⟩**  
 $S_{1.2}$ : System fails to authenticate the passenger. **⟨Branch EXIT⟩**
- $E_1$ : Passenger selects an individual to leave a rating for.
- $S_2$ : System receives response and prompts passenger for a message/rating for the selected individual. System waits for user response.
- $E_2$ : Passenger responds to the system  
 $E_{2.1}$ : Passenger cancels the review process  
 $E_{2.2}$ : Passenger inputs a valid response (I.e. gives a rating and a potential message response) to the system
- $S_3$ : System calculates the most suitable path to take to get from pickup location to destination location and display the total cost of travel.  
 $S_{3.1}$ : System receives an invalid response (e.g., passenger cancels review process) **⟨Branch EXIT⟩**  
 $S_{3.2}$ : System receives a valid response and updates the individual's profile with the new rating
- $E_3$ : Passenger is returned to ratings screen.

**VP2. Driver**

N/A

**Global Scenario:**

- $S_1$ : System attempts to authenticate the passenger and confirm that passenger has taken a ride previously.  
 $S_{1.1}$ : System successfully authenticates the passenger and displays a list of applicable (previous carpool partners) passengers to leave a review for **⟨Branch to BE3.VP2.1.⟩**  
 $S_{1.2}$ : System fails to authenticate the passenger. **⟨Branch EXIT⟩**
- $E_1$ : Passenger selects an individual to leave a rating for.
- $S_2$ : System receives response and prompts passenger for a message/rating for the selected individual. System waits for user response.
- $E_2$ : Passenger responds to the system  
 $E_{2.1}$ : Passenger cancels the review process  
 $E_{2.2}$ : Passenger inputs a valid response (I.e. gives a rating and a potential message response) to the system
- $S_3$ : System calculates the most suitable path to take to get from pickup location to destination location and display the total cost of travel.  
 $S_{3.1}$ : System receives an invalid response (e.g., passenger cancels review process) **⟨Branch EXIT⟩**  
 $S_{3.2}$ : System receives a valid response and updates the individual's profile with the new rating
- $E_3$ : Passenger is returned to ratings screen.

**BE3. Setup/Update user profile/account information**

**VP1. Passenger**

- $S_1$ : System prompts passenger for personal information (Full name, phone number, email).
- $E_1$ : Passenger inputs their information.
- $S_2$ : System stores personal information of user and allows carpool services under the account.

**VP2. Driver**



N/A

**Global Scenario:**

- $S_1$ : System prompts passenger for personal information (Full name, phone number, email).
- $E_1$ : Passenger inputs their information.
- $S_2$ : System stores personal information of user and allows carpool services under the account.

**BE4. Cancel a ride**

**VP1. Passenger**

- $S_1$ : System re-asks user if they want to cancel the ride.
- $E_1$ : User indicates that they are sure they want to cancel the ride.
- $S_2$ : System notifies the driver that the ride has been canceled.

**VP2. Driver**

N/A

**Global Scenario:**

- $S_1$ : System re-asks user if they want to cancel the ride.
- $E_1$ : User indicates that they are sure they want to cancel the ride.
- $S_2$ : System notifies the driver that the ride has been canceled.

**BE5. Login/logout of an account**

**VP1. Passenger**

- $S_1$ : System prompts passenger for login information (username and password).
- $E_1$ : Passenger inputs their information.
- $S_2$ : System checks if username and password matches any accounts information.
  - $S_{2.1}$ : Customers login info matches an account and they are granted access.
  - $S_{2.2}$ : Customers login does not match an account **⟨Branch to VP1.S1⟩**

**VP2. Driver**

N/A

**Global Scenario:**

- $S_1$ : System re-asks user if they want to cancel the ride.
- $E_1$ : User indicates that they are sure they want to cancel the ride.
- $S_2$ : System notifies the driver that the ride has been canceled.

**BE6. Edit payment information**

**VP1. Passenger**

- $S_1$ : System authenticates user login and displays current payment method information (e.g., cards/email/PayPal).
- $E_1$ : User selects payment method desired to modify.
- $S_2$ : System allows for modification of payment method.
- $E_1$ : User inputs new information.
- $S_2$ : System saves edited information to the database.

**VP2. Driver**

N/A

**Global Scenario:**

- $S_1$ : System authenticates user login and displays current payment method information (e.g., cards/email/PayPal).
- $E_1$ : User selects payment method desired to modify.
- $S_2$ : System allows for modification of payment method.
- $E_1$ : User inputs new information.
- $S_2$ : System saves edited information to the database.

**BE7.** Redeem a free ride

**VP1.** Passenger

- $S_1$ : System checks if the user has a free ride available.  
 $S_{1.1}$ : System confirms that the user has a free ride available. System prompts user for a response with a display message: “Do you want to redeem a free ride?”  
 $S_{1.2}$ : System confirms that the user does not currently have a free ride available.  
**⟨Branch EXIT⟩**
- $E_1$ : User responds to the system’s prompt.  
 $E_{1.1}$ : User responds with “No” and decides not to redeem a free ride. **⟨Branch EXIT⟩**  
 $E_{1.2}$ : User responds with “Yes” and would like to redeem a free ride.
- $S_2$ : System consumes the free ride coupon and removes the coupon (remove one if the user has more than one free-ride coupon) from the user’s list of coupons/promotions available.
- $E_2$ : User proceeds with confirming their travel route. **⟨Branch to BE1⟩**

**VP2.** Driver

N/A

**Global Scenario:**

- $S_1$ : .

## 5 Non-Functional Requirements

- For each non-functional requirement, provide a justification/rationale for it.

**Example:**

SC1. *The device should not explode in a customer’s pocket.*

**Rationale:** Other companies have had issues with the batteries they used in their phones randomly exploding [insert citation]. This causes a safety issue, as the phone is often carried in a person’s hand or pocket.

- If you need to make a guess because you couldn’t really talk to stakeholders, you can say “We imagined stakeholders would want...because...”
- Each requirement should have a unique label/number for it.
- In the list below, if a particular section doesn’t apply, just write N/A so we know you considered it.

### 5.1 Look and Feel Requirements

#### 5.1.1 Appearance Requirements

LF-A1. The app should have a modern aesthetic to it, conforming to a standardized colour scheme which is to be used throughout the app. The choice of colour should be consistent all throughout the app and the must provide enough vibrance amongst each other so that in no shape of form gives off a “dull” image.

**Rationale:** Having a modern aesthetic and consistent color scheme help create a more engaging and enjoyable user experience while also reinforcing the brand and establishing a sense of professionalism. The vibrance also makes the app more accessible for those with visual impairments.

- LF-A2. The app should be organized by related content with little to no cluttering of text/images to provide for visually appealing and easy to read navigation. The cluttering of text/images should follow a basis of having at minimum 5 vw/vh between unrelated images/text bodies and 2.5 vw/vh for related ones.

**Rationale:** Organizing content and following a reasonable minimum spacing guideline help create a visually appealing and user-friendly experience that is easy to navigate and accessible to all users.

- LF-A3. The font family will remain consistent throughout the interface of the app. The size of the font will be increased amongst reasonable intervals and there shall be no difference in text size which varies ever so slightly. There shall not be font size of something of the sort: 13.5, 13.7 and 13.9. Instead, a font size with more appropriate intervals will be an interval of 10, 12, 16 as it provides an easy way to distinguish the size of the text and remain consistent all throughout the app.

**Rationale:** Consistent font sizes make it easier for users to read and understand the text. If there are variations in font size, it can cause confusion and make it difficult for users to read and comprehend the content.

### 5.1.2 Style Requirements

- LF-S1. The app shall have a monochromatic appearance, with appropriate font size and font type.

**Rationale:** See rationale for LF-A1.

## 5.2 Usability and Humanity Requirements

### 5.2.1 Ease of Use Requirements

- UH-EOU1. The app should be relatively easy to learn for beginners and should require no more than 3 minutes of tutorial time to learn. A tutorial will be provided upon creating of a new account that helps navigate the user through the app if they so choose to.

- UH-EOU2. All accessibility and navigation buttons should be intuitive and easy to find. Likewise, the content should be organized in a way so that finding related content is relatively easy and intuitive.

- UH-EOU3. The app should be accessible, and convenient for the general public to use. The app shall allow users to create an account through various forms of media (e.g. Gmail OAuth, **SMS** Login, Apple Login, etc...) so that users with only one source of media are not limited.

**Rationale:** The requirement for all accessibility and navigation buttons to be intuitive and easy to find ensures that users can easily navigate the app's interface and access the content they need. Organizing related content in an easy and intuitive way improves user experience and minimizes frustration when searching for information.

### 5.2.2 Personalization and Internationalization Requirements

- UH-PI1. The app shall have the basic US English language implemented in its initial run and can take on various languages afterwards.

**Rationale:** Initial target audience is based in US and Canada, whose primary languages are both English. Internationalization will be considered in later iterations of the application, in which other languages may

### 5.2.3 Learning Requirements

- UH-L1. The app should not require users to have any pre-existing knowledge beside knowing where the user wants to get to.

**Rationale:** An application that does not require pre-existing knowledge is more accessible to users with varying levels of technical expertise, which leads to greater accessibility and market audiences.

### 5.2.4 Understandability and Politeness Requirements

- UH-UP1. The carpool app should not make use of any vulgar speech in any dialogue/informational sections throughout the app.  
**Rationale:** The carpool app should be appropriate to use for all ages and should maintain a degree of professionalism. Vulgar speech can limit the target audience and impair reputation.
- UH-UP2. Has buttons labelled with different actions the user can take.  
**Rationale:** It should be clear to users what they are doing when interacting with the app.

### 5.2.5 Accessibility Requirements

- UH-A1. App must have a text-to-speech functionality for visually impaired users.  
**Rationale:** Visually impaired users are unable to see the user interface, and thus they are not able to use the app's functionality even though they are just as capable of using the carpool's services. Text-to-speech increases accessibility for those that are visually impaired.

## 5.3 Performance Requirements

### 5.3.1 Speed and Latency Requirements

- PR-SL1. The app should run reliably fast with a responsive user interface with minimal lag. The latency shall not exceed 200ms when transitioning between different pages on the app and that new content should be displayed within the 200ms.  
**Rationale:** A slow user interface risks users losing patience, becoming frustrated, and tending towards competitors. This is vice-versa with consistently high-responsive user interface. Google recommends aiming for a latency of no more than 200ms, and that it is consistent for all users.<sup>1</sup>
- PR-SL2. The app must also calculate a suitable travel route for carpool users within 5 seconds of matching.  
**Rationale:** Similar rationale to PR-SL1. However, since calculating routes take up much more computational power than displaying context on a screen, a time of 5 seconds accounts for that while maintaining a degree of efficiency.

### 5.3.2 Safety-Critical Requirements

- PR-SC1. The system should provide information about the correct vehicle for their trip. This includes the make/model of the car that the driver is driving, the license plate attributed with the car, and the location of the car/driver when they are ready to receive passengers.  
**Rationale:** Ensuring correct information helps to ensure that passengers are getting into the correct vehicle with a verified driver, and not unauthorized or unsafe vehicles/drivers. This prevents instances of fraud, theft, or other safety concerns.
- PR-SC2. Route should not be recalculated when driver is in motion. Travel route should only be able to be changed/updated accordingly if there's a faster route (e.g., due to traffic or something) and that the car has been stationary for at least 15 seconds.  
**Rationale:** This helps to ensure that the driver remains focused on the road and does not become distracted by changes to the travel route.

### 5.3.3 Precision or Accuracy Requirements

- PR-PA1. The locations of pick-up/destinations should be accurate within 50m.  
**Rationale:** Inaccurate destinations cause inconveniences for all parties participating in the carpool, such as having to walk further than expected or missing the carpool altogether because the taxi's location is inaccessible from the user's location.
- PR-PA2. Estimated times of pickup/arrival should be accurate within 5 minutes.  
**Rationale:** Inaccurate pickup/arrival times cause unnecessary delays (early arrival delays the

taxi and late arrival delays the user). This may turn people away from using the app as this is an inconvenience to all parties participating in the carpool.

### 5.3.4 Reliability and Availability Requirements

PR-RA1. The app should functionally run with an uptime of 100%.

**Rationale:** While it is not safety-critical to have the app crash or go on maintenance, any downtime directly affects any users that are looking for carpools, in a carpool. This can also cause confusion and complications between users and the taxi company in terms of payment. Furthermore, it may turn customers away from using the app if it is not reliable. Many of today's leading SaaS vendors offer at least 99.5% availability.<sup>2</sup>

### 5.3.5 Robustness or Fault-Tolerance Requirements

PR-RFT1. In the event of an internet outage, the system should prevent users from using the app, and should Continue displaying any committed trips.

**Rationale:**

PR-RFT2. In the event the system goes down, any ongoing trips should continue.

**Rationale:**

### 5.3.6 Capacity Requirements

PR-C1. The functional capacity of the app should scale with the number of users.

**Rationale:**

PR-C2. The application installation size shall not exceed 40 MB.

**Rationale:**

### 5.3.7 Scalability or Extensibility Requirements

PR-SE1. The system shall adapt to any increase in user influx while maintaining performance.

**Rationale:**

### 5.3.8 Longevity Requirements

PR-L1. The application will be operational until the service is no longer required by a minimum population of users.

**Rationale:**

## 5.4 Operational and Environmental Requirements

### 5.4.1 Expected Physical Environment

OE-EPE1. Expected physical environments include urbanized areas which have paved roads and cellular/wireless connectivity.

**Rationale:**

### 5.4.2 Requirements for Interfacing with Adjacent Systems

OE-IA1. All the tech stack used in the development of the carpool app is expected to interface correctly with all other components of our app. In extreme cases where any of the interfaces (e.g. Google Maps **API**/MongoDB) exhibits severely poor performance, it may be recommended to swap out the interface with another one similar (e.g. OpenLayers/MySQL).

**Rationale:**

### 5.4.3 Productization Requirements

OE-P1. System should must be tested on multiple devices before deployment.

**Rationale:**

OE-P2. System should be accessible to customers In the **GoPlay** store.

**Rationale:**

### 5.4.4 Release Requirements

OE-R1. The application shall be released to the **GoPlay** store.

**Rationale:**

## 5.5 Maintainability and Support Requirements

### 5.5.1 Maintenance Requirements

MS-M1. In the event that maintenance/shutdown is required, users must be informed via the app's home-page a week in advance.

**Rationale:**

MS-M2. New feature updates should not conflict with previous versions of the app and can be pulled by updating through the App/Google Play store when users decide to update the app - This can also be set to automatic for automatic updates.

**Rationale:**

### 5.5.2 Supportability Requirements

MS-S1. Forward Compatibility with Android devices in newer versions.

**Rationale:** Naturally, the general population will update their devices when newer versions of Android releases. This should not break or crash the application, as people seldomly rollback versions just to use the app.

MS-S2. The repo to the project's main repository shall be made public to allow for improvements and suggestions from other developers.

**Rationale:** Along with users, other developers can provide valuable feedback, particularly things that are too technical for a user to notice. Having more feedback and improvement ideas help support the app with bug fixes, missing features, and so forth.

MS-S3. The source code shall be fully documented through Javadoc comments and shall adhere to correct coding styles.

**Rationale:** Having coding standards make it easier for the project to be maintained and extended. Consistent coding styles and thorough comments allow anyone involved in the project to look at any part of the code, comprehend it, and change it regardless of when it was written during the project. It will also reduce code complexity, which in turn is more likely to be better safeguarded against bugs.

### 5.5.3 Adaptability Requirements

MS-A1. N/A

## 5.6 Security Requirements

### 5.6.1 Access Requirements

SR-AC1. Users can only access their accounts after entering their own set password and email.

**Rationale:** Sensitive information are stored in accounts and should remain confidential to the user. Unwanted access can lead to a violation of LR1. and unwanted use of the app.

### 5.6.2 Integrity Requirements

SR-INT1. N/A

### 5.6.3 Privacy Requirements

SR-P1. User's private information will always be kept private and only accessible by the user and certain components of the application.

**Rationale:** By assuring users that their personal information will be kept confidential and secure, it builds trust between the users and the software application, leading to increased user adoption and retention.

SR-P2. All personal data stored in the database shall be encrypted, and data shall only be decrypted once the user has authorized access to view or modify the information.

SR-P3. **Rationale:** This adds another layer of security to ensure any unauthorised access of the database will not be able to retrieve any sensitive information about users.

### 5.6.4 Audit Requirements

SR-AU1. Internal audits will be conducted once every month per fiscal period and external audits will be conducted at random intervals throughout the fiscal year to check for discrepancies.

**Rationale:** By conducting internal and external audits, the software application can demonstrate its commitment to compliance with regulatory requirements and best practices, build trust with users and stakeholders, and reduce the risk of financial and reputational losses by detecting and acting on discrepancies.<sup>3</sup>

### 5.6.5 Immunity Requirements

SR-IM1. N/A

## 5.7 Cultural and Political Requirements

### 5.7.1 Cultural Requirements

CP-C1. Documentation shall include an inclusion and diversity statement to ensure all users have the right to access and utilize the website.

**Rationale:** As the target audience is made up of a range of different social and ethnic backgrounds, genders, ages, etc., the app should demonstrate a commitment to building an inclusive and welcoming product for all people.

### 5.7.2 Political Requirements

CP-P1. N/A

## 5.8 Legal Requirements

### 5.8.1 Compliance Requirements

LR-COMP1. The application shall not violate any Canadian or American laws. For instance, all data will be encrypted so that personal information privacy is preserved.

**Rationale:** By complying with laws such as the Personal Information Protection and Electronic Documents Act (PIPEDA) in Canada, the application can avoid any legal liabilities and potential lawsuits.<sup>4</sup>

LR-COMP2. Payment and Income logs shall be stored for tax purposes and accounting.

**Rationale:** The logs help ensure compliance with tax laws, such as the Internal Revenue

Service (IRS) in the United States and the Canada Revenue Agency (CRA) in Canada. By keeping detailed records of all financial transactions, the software application can provide an accurate record of income and expenses, and thus tax returns.<sup>5</sup>

### 5.8.2 Standards Requirements

LR-STD1. N/A

## A Division of Labour

Include a Division of Labour sheet which indicates the contributions of each team member. This sheet must be signed by all team members.

Team Member	Contribution
Adam Mak	
Eric Chen	
Justin Ho	
Ahmad Hamadi	
Kevin Ishak	
Jonathan Jiang	