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

IMPORTANT NOTES

- Be sure to include all sections of the template in your document regardless whether you have something to write for each or not
  - If you do not have anything to write in a section, indicate this by the *N/A*, *void*, *none*, etc.
- Uniquely number each of your requirements for easy identification and cross-referencing
- Highlight terms that are defined in Section 1.3 (**Definitions, Acronyms, and Abbreviations**) with **bold**, *italic* or <u>underline</u>
- For Deliverable 1, please highlight, in some fashion, all (you may have more than one) creative and innovative features. Your creative and innovative features will generally be described in Section 2.2 (**Product Functions**), but it will depend on the type of creative or innovative features you are including.

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 are 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.

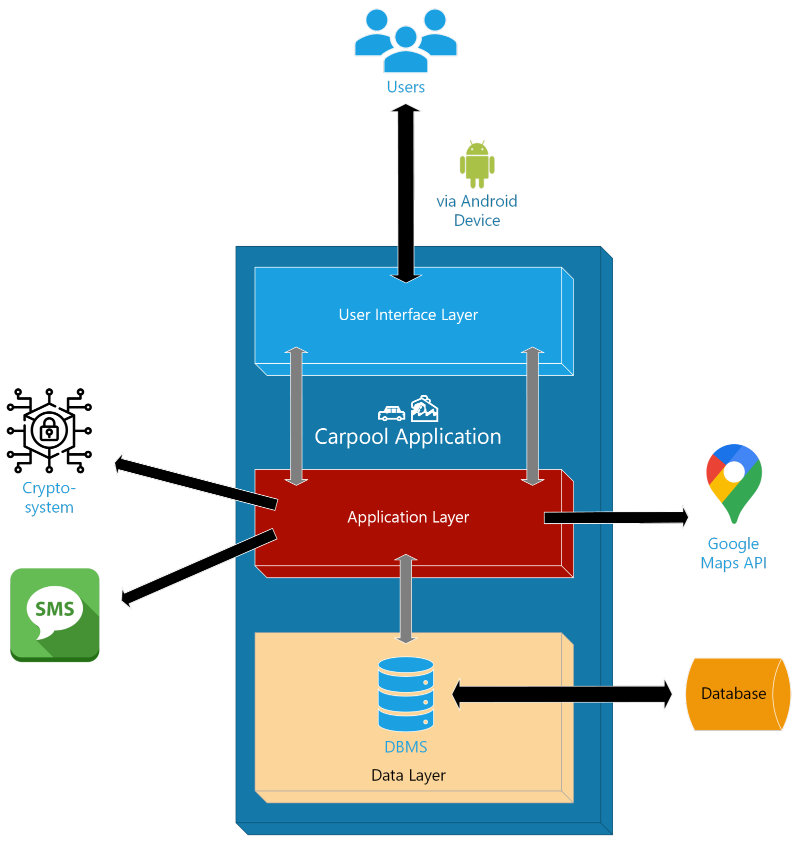

Figure 1: A block diagram showing the major layers of the carpooling application, interconnections, and external interfaces.

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

- Provide the use case diagram for the system being developed.
- You do not need to provide the textual description of any of the use cases here (these will be specified under "Highlights of Functional Requirements").

4 Highlights of Functional Requirements

- Specify all use cases (or other scenarios triggered by other events), organized by Business Event.

- For each Business Event, show the scenario from every Viewpoint. You should have the same set of Viewpoints across all Business Events. If a Viewpoint doesn't participate, write N/A so we know you considered it still. You can choose how to present this - keep in mind it should be easy to follow.
- At the end, combine them all into a Global Scenario.
- Your focus should be on what the system needs to do, not how to do it. Specify it in enough detail that it clearly specifies what needs to be accomplished, but not so detailed that you start programming or making design decisions.
- Keep the length of each use case (Global Scenario) manageable. If it's getting too long, split into sub-cases.
- You are *not* specifying a complete and consistent set of functional requirements here. (i.e. you are providing them in the form of use cases/global scenarios, not a refined list). For the purpose of this project, you do not need to reduce them to a list; the global scenarios format is all you need.
- Red text below is just to highlight where you need to insert a scenario - don't actually write it all in red.

Main Business Events: List out all the main business events you are presenting. If you sub-divided into smaller ones, you don't need to include the smaller ones in this list.

Viewpoints: List out all the viewpoints you will be considering.

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

BE1. Business Event Name #1

VP1. Viewpoint Name #1
Insert Scenario Here

VP2. Viewpoint Name #2
Insert Scenario Here

Global Scenario:
Insert Scenario Here

BE2. Business Event Name #2

VP1. Viewpoint Name #1
Insert Scenario Here

VP2. Viewpoint Name #2
Insert Scenario Here

Global Scenario:
Insert Scenario Here

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.

5.1.2 Style Requirements

LF-S1.

5.2 Usability and Humanity Requirements

5.2.1 Ease of Use Requirements

UH-EOU1.

5.2.2 Personalization and Internationalization Requirements

UH-PI1.

5.2.3 Learning Requirements

UH-L1.

5.2.4 Understandability and Politeness Requirements

UH-UP1.

5.2.5 Accessibility Requirements

UH-A1.

5.3 Performance Requirements

5.3.1 Speed and Latency Requirements

PR-SL1.

5.3.2 Safety-Critical Requirements

PR-SC1.

5.3.3 Precision or Accuracy Requirements

PR-PA1.

5.3.4 Reliability and Availability Requirements

PR-RA1.

5.3.5 Robustness or Fault-Tolerance Requirements

PR-RFT1.

5.3.6 Capacity Requirements

PR-C1.

5.3.7 Scalability or Extensibility Requirements

PR-SE1.

5.3.8 Longevity Requirements

PR-L1.

5.4 Operational and Environmental Requirements

5.4.1 Expected Physical Environment

OE-EPE1.

5.4.2 Requirements for Interfacing with Adjacent Systems

OE-IA1.

5.4.3 Productization Requirements

OE-P1.

5.4.4 Release Requirements

OE-R1.

5.5 Maintainability and Support Requirements

5.5.1 Maintenance Requirements

MS-M1.

5.5.2 Supportability Requirements

MS-S1.

5.5.3 Adaptability Requirements

MS-A1.

5.6 Security Requirements

5.6.1 Access Requirements

SR-AC1.

5.6.2 Integrity Requirements

SR-INT1.

5.6.3 Privacy Requirements

SR-P1.

5.6.4 Audit Requirements

SR-AU1.

5.6.5 Immunity Requirements

SR-IM1.

5.7 Cultural and Political Requirements

5.7.1 Cultural Requirements

CP-C1.

5.7.2 Political Requirements

CP-P1.

5.8 Legal Requirements

5.8.1 Compliance Requirements

LR-COMP1.

5.8.2 Standards Requirements

LR-STD1.

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.