

Deliverable #1: Software Requirement Specification (SRS)

SE 3A04: Software Design II – Large System Design

Tutorial Number: T03

Group Number: G8

Group Members: Hashim Bukhtiar, Jaden Moore, James Ariache, Olivia Reich, Omar Abdelhamid

- Hashim Bukhtiar
- Jaden Moore
- James Ariache
- Olivia Reich
- Omar Abdelhamid

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 underline
- 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

The SRS is a structured document that outlines the functional and non-functional requirements of the RideRecon software system. It serves as a blueprint for developers, testers, and stakeholders, ensuring a clear understanding of the system to be built. This SRS will provide visibility over software requirements for RideRecon, a taxi rideshare application. This document will discuss the purpose of RideRecon, the scope of the application, user characteristics, product requirements, and use case diagrams.

1.1 Purpose

The document focuses on software requirements, user characteristics, and use cases for RideRecon. The purpose of this SRS is to define the software's objectives, scope, and functionalities. This document is intended for internal RideRecon stakeholders, including but not limited to, project managers, developers, domain experts, and RideRecon team members/investors. No prior readings are required.

1.2 Scope

RideRecon, the car identification application, will allow users to upload an image and text about a vehicle that they want to identify, consulting up to four "Experts" who will use all or some of the user's input to determine the make and model of the car that has been depicted.

Users are required to register an account on RideRecon in order to access the car identification service. The service includes four experts: "RIS", "G8M", "4oE" and "vAI". The "RIS" expert performs a reverse image search with the user's given image to identify the car. The "G8M" expert is a trained machine learning model by the developers that utilizes an optimized database of car images and their corresponding makes and models to identify the user's car. The "4oE" expert is the Large Language Model GPT-4o-mini, accessed through OpenAI's API, which will utilize both the image and the text description to identify the car. Finally, the "vAI" expert is also an LLM, this one hosted on Google Cloud Platform using Vertex AI, specifically the Gemini 1.0 Flash Model, also accessed through its API.

RideRecon's objective is not only to identify users' vehicles, but also to deliver other innovative and creative features. For example, once the car has been identified, 4oE will provide an interesting fact about the car, adding more depth and intrigue to it, as well as provide sources and listings for where to purchase the car, allowing the user to go one step further and save time if they are interested in making that vehicle their own.

One of the goals of the software is to increase users, as more users will lead to more cars being identified and added to the optimized database of G8M, and thus better car identification overall, which is even better for future users. Another goal would be to monetize its user base by offering premium features like better expert access, exclusive community forums, or partnerships with car-related businesses, thereby creating additional revenue streams beyond the core car identification service. This diversification will improve long-term sustainability and profitability.

1.3 Definitions, Acronyms, and Abbreviations

LLM: Large Language Model. A sophisticated artificial intelligence that can understand, generate, and translate human language, often used for tasks like text generation, question answering, and translation.

RIS: Reverse Image Search. This is the idea of using an image as a search query in a browser to find results, rather than using text as traditional browser searches as performed.

G8M: Group 8 Model. This is the model trained by the developers of RideRecon to identify cars. It will utilize an optimized dataset of car images and makes and models.

4oE: GPT-4o-mini. This is an LLM that uses text and images to determine the make and model of the user's car.

vAI: Vertex AI, Gemini 1.0 Flash. This is the LLM that uses images to determine the make and model of the user's car.

Finalizer: The component of the system that consolidates and finalizes the output, including resolving conflicts between experts and outputting the fun fact about the car and where to purchase it.

1.4 References

- 1 “AvtoPix: Car Id - apps on Google Play,” Google, https://play.google.com/store/apps/details?id=ru.egoroffsoft.avtopix&hl=en_CA&pli=1 (accessed Feb. 14, 2025).
- 2 “CARSSNAP - car model identify - apps on Google Play,” Google, https://play.google.com/store/apps/details?id=com.mm999.car&hl=en_CA (accessed Feb. 14, 2025).
- 3 M. Nadir, “Left handedness and mobile user experience,” Medium, <https://medium.com/design-bootcamp/left-handedness-and-mobile-user-experience-a3728c72f880> (accessed Feb. 14, 2025).
- 4 Written By Jennifer Gaskin Published: Jan 13, About Jennifer Gaskin A veteran of newsrooms and agencies, and A. J. Gaskin, “Color blind design guidelines: A comprehensive guide,” Venngage, <https://venngage.com/blog/color-blind-design/> (accessed Feb. 14, 2025).
- 5 L. Nikolov, “What’s the difference between API latency and API response time?,” Product Blog Sentry, <https://blog.sentry.io/whats-the-difference-between-api-latency-and-api-response-time/#:~:text=Just%20to%20have%20a%20number,to%20abandon%20the%20application%2Fwebsite> (accessed Feb. 15, 2025).
- 6 “Overview of app launch times,” New Relic Documentation, <https://docs.newrelic.com/docs/mobile-monitoring/new-relic-mobile/get-started/introduction-app-launch-times/#:~:text=Mobile%20app%20launch%20time%20benchmarks,take%20less%20than%201.5%20seconds> (accessed Feb. 15, 2025).
- 7 R. Gordon, “Image recognition accuracy: An unseen challenge confounding today’s AI,” MIT News — Massachusetts Institute of Technology, <https://news.mit.edu/2023/image-recognition-accuracy-minimum-viewi> (accessed Feb. 15, 2025).
- 8 C. Staff, P. Sayer, M. K. Pratt, and M. Bayer, “What is an SLA? Best practices for service-level agreements,” CIO, <https://www.cio.com/article/274740/outsourcing-sla-definitions-and-solutions.html> (accessed Feb. 15, 2025).
- 9 “How many users is realistic?,” Developer Nation Community, <https://www.developernation.net/blog/how-many-users-is-realistic/> (accessed Feb. 15, 2025).
- 10 “Implement fallback with API gateway,” API7.ai, <https://api7.ai/blog/fallback-api-resilience-design-patter> (accessed Feb. 15, 2025).
- 11 “Amazon S3 pricing - cloud object storage - AWS,” <https://aws.amazon.com/s3/pricing/> (accessed Feb. 15, 2025).
- 12 IBM, “What is cloud storage?,” IBM, <https://www.ibm.com/think/topics/cloud-storage> (accessed Feb. 15, 2025).
- 13 IBM, “What is network optimization?,” IBM, <https://www.ibm.com/think/topics/network-optimization> (accessed Feb. 15, 2025).
- 14 D. S. Sahin Ahmed, “Why does machine learning model performance degrade, and how can we detect and prevent it?,” Medium, <https://medium.com/@sahin.samia/why-does-machine-learning-model-performance> (accessed Feb. 15, 2025).
- 15 “Adaptive preprocessing and segmentation for a ...,” NDT.net, <https://www.ndt.net/article/ctc2012/papers/307.pdf> (accessed Feb. 15, 2025).
- 16 F. June, “Model quantization 1: Basic concepts,” Medium, https://medium.com/@florian_algo/model-quantization-1-basic-concepts-860547ec6aa9 (accessed Feb. 15, 2025).

- 17 A. Kareliya, "Firebase vs. AWS: Which one to select for your app project in 2025?," Radixweb, <https://radixweb.com/blog/firebase-vs-aws> (accessed Feb. 15, 2025).
- 18 "Android 9 PIE: Platform," Android Developers, <https://developer.android.com/about/versions/pie> (accessed Feb. 15, 2025).
- 19 "Developer policy center," Developer Policy Center, <https://play.google/developer-content-policy/> (accessed Feb. 15, 2025).
- 20 "Privacy checklist : android developers," Android Developers, <https://developer.android.com/privacy-and-security/about> (accessed Feb. 15, 2025).
- 21 L. S. Branch, "Consolidated federal laws of Canada, Personal Information Protection and Electronic Documents Act," Personal Information Protection and Electronic Documents Act, <https://laws-lois.justice.gc.ca/eng/acts/p-8.6/> (accessed Feb. 15, 2025).
- 22 "Legal text," General Data Protection Regulation (GDPR), <https://gdpr-info.eu/> (accessed Feb. 15, 2025).
- 23 C. Masha Komnenic CIPP/E, "Mobile app terms and conditions template," Termly, <https://termly.io/resources/templates/app-terms-and-conditions/> (accessed Feb. 15, 2025).
- 24 "ISO/IEC 27001:2022," ISO, <https://www.iso.org/standard/27001> (accessed Feb. 15, 2025).
- 25 V. Kananda, "What is AES 256 encryption & how does it work?," Progress Blogs, <https://www.progress.com/blogs/use-aes-256-encryption-secure-data> (accessed Feb. 15, 2025).
- 26 "What is OWASP? What is the OWASP Top 10?," Cloudflare, <https://www.cloudflare.com/learning/security/threats/owasp-top-10/> (accessed Feb. 15, 2025).
- 27 "Does Canada have a version of the DMCA takedown?," DMCA Protection & Takedown Services, <https://www.dmca.com/FAQ/Does-Canada-have-a-version-of-the-DMCA-Takedown> (accessed Feb. 15, 2025).

1.5 Overview

Section 2 discusses the overall product description talking about the product perspective, product functions, user characteristics, assumptions and dependencies, and apportioning of requirements.

Section 3 contains the Use Case Diagram for the use case scenario of using the RideRecon app.

Section 4 contains the highlights of functional requirements talking about main business events and viewpoints.

Section 5 contains the Non-Functional Requirements talking about Look and Feel Requirements, Usability and Humanity Requirements, Performance Requirements, Operational and Environmental Requirements, Maintainability and Support Requirements, Security Requirements, Cultural and Political Requirements and Legal Requirements.

Lastly, **Section A** contains the Division of Labour.

2 Overall Product Description

2.1 Product Perspective

RideRecon is a mobile app for car identification that is developed to be compatible with the Android platform. Similar to products such as AvtoPix [1] and CarsSnap [2], it will allow car identification through images while also enhancing search capabilities by also allowing identification through textual descriptions. The product will allow users to create a profile and edit account information linked to that profile. This includes the creation of virtual car collections using car images submitted in the app.

Once an image is selected by the user, the product will have an interface that allows it to interact with different reverse image search technologies. Specifically, it will be able to access a Vertex AI image model and the Reverse Image Search Tool via Google.

If provided textual input by the user, the product will have an interface to interact with Google Search. This interface will also be used to search for relevant car facts that can be returned to the user upon identifying pivotal car identification information.

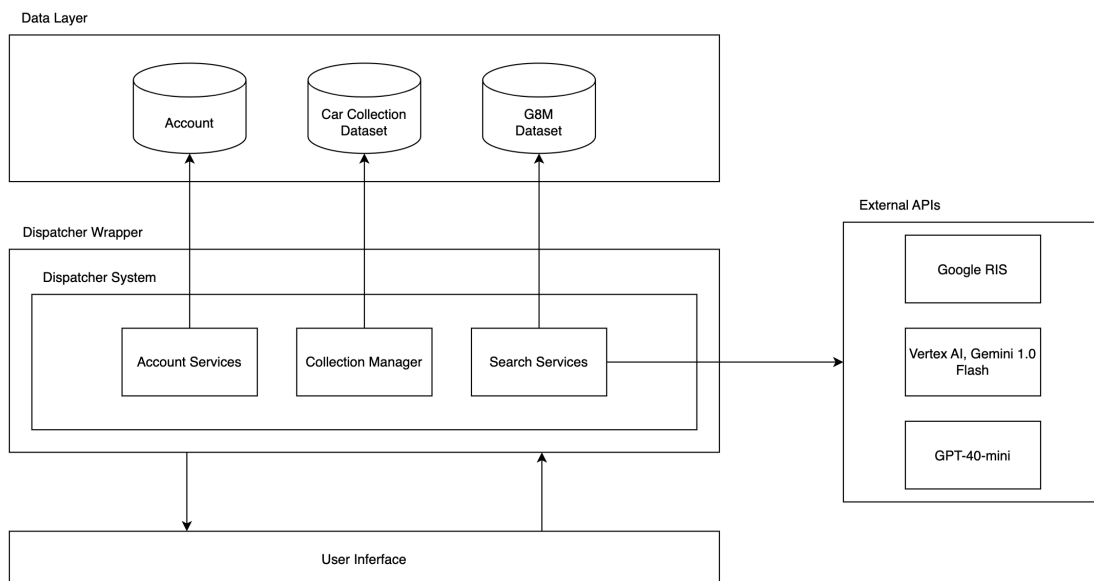


Figure 1. System Diagram

2.2 Product Functions

There will be 3 modules in the product. Each module focuses on different major functions within the product, which have been defined in the table below.

Modules	Functions
Account Services	<ul style="list-style-type: none">• Create an account• Login and logout of account• Update account information• Account recovery<ul style="list-style-type: none">– allow user to reset password if forgotten• Authenticate account<ul style="list-style-type: none">– verifies contact information only done during account creation or recovery
Search Services	<ul style="list-style-type: none">• Image search<ul style="list-style-type: none">– allow user to request car identification through an image• Text search<ul style="list-style-type: none">– allow user to request car identification through text descriptors• Present car information<ul style="list-style-type: none">– displays all relevant identification information– displays ‘fun fact’ information• Confirm car identification<ul style="list-style-type: none">– allow user to confirm whether the identified information matches the car• Add car to collection<ul style="list-style-type: none">– allow user to add identified car to specified ‘Car Collection’
Collection Manager	<ul style="list-style-type: none">• Create new collection<ul style="list-style-type: none">– allow user to create new sub-class in their car collection• View collection<ul style="list-style-type: none">– allow user to select and view a car collection

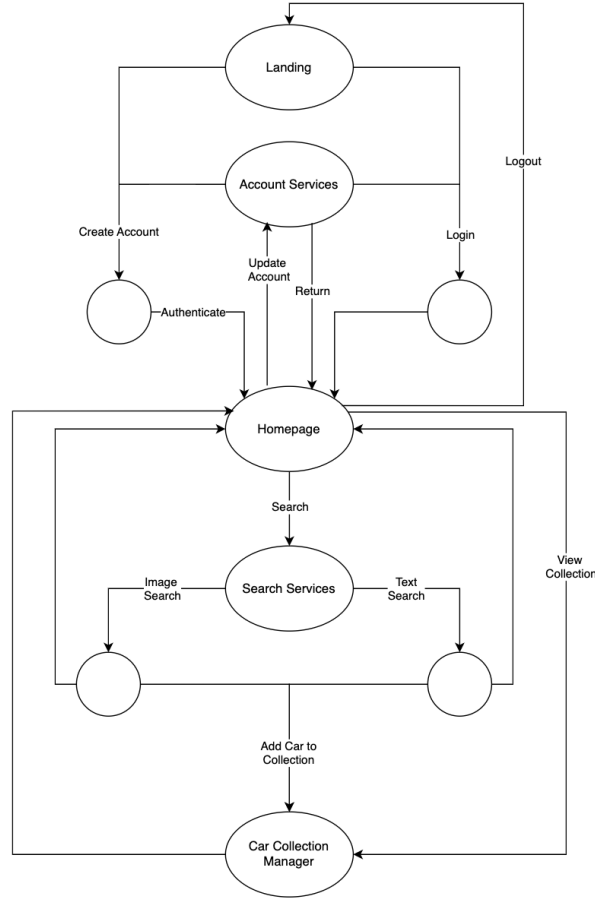


Figure 2. State Diagram

2.3 User Characteristics

- **Educational Level:** Users must have basic literacy skills which includes the fundamental ability to understand information through reading, writing, listening, and speaking.
- **Technical Expertise:** Users should have a basic understanding of how to operate and navigate an Android smartphone, including downloading apps, as this application is designed specifically for the Android platform.
- **Technical Experience:** Users do not require any prior experience or knowledge with cars in order to utilize the app. Aside from the basic technical expertise outlined above, no extra experience will be required to navigate and utilize the features of the app as it will prioritize an intuitive design.

2.4 Constraints

- Provide a general description of any constraints that will limit the developer's options
- The system must have an active **internet connection** to perform car identification.
- The app must be compatible with **Android 9.0 (Pie) and above**.
- The device must have at least **2GB of RAM** for smooth operation.
- Captured images must have a **minimum resolution of 720p** to ensure accurate car identification.
- The AI model may not recognize **damaged or heavily modified cars** accurately.

- The system must comply with **privacy laws** regarding the storage and processing of user-uploaded images.

2.5 Assumptions and Dependencies

- List any assumptions you made in interpreting what the software being developed is aiming to achieve
- List any other assumptions you made that, if it fails to hold, could require you to change the requirements
 - **Example:** An assumption may be that a specific operating system will be available on the hardware designated for the software product. If, in fact, the operating system is not available, the SRS would then have to change accordingly.
- **Assumptions:**
 - Users will have stable internet access while using the app.
 - The AI model can identify most common car brands and models.
 - Dealerships will provide accurate and up-to-date pricing.
- **Dependencies:**
 - The AI model relies on an external **machine learning API** to process images.
 - The system depends on a third-party **reverse image search API** (e.g., Google Lens, TinEye) for validation.
 - Pricing and specifications are sourced from **external dealership databases**.

2.6 Apportioning of Requirements

- Identify requirements that may be delayed until future versions of the system
- **Offline Mode:** The app currently requires internet access; an offline model will be explored in future versions.
- **Multiple Car Detection:** Initially, the app will support one car per image; future updates may allow multiple cars.
- **Personalized Car Recommendations:** The first version won't suggest similar cars, but this may be added later.

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").

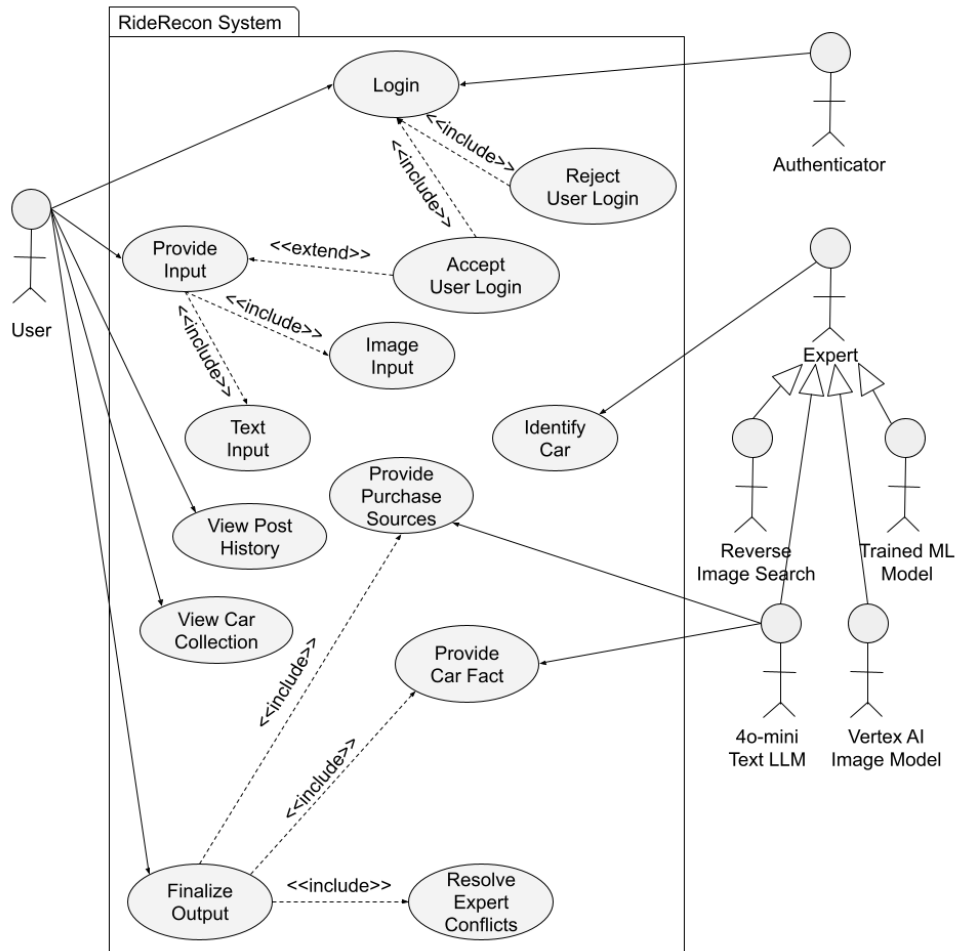


Figure 3. Use Case Diagram

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.

- BE1. Create Account
- BE2. Authenticate User
- BE3. Upload Text and/or Image as Input
- BE4. Compare Expert Answers
- BE5. Present Final Output With Identification Information
- BE6. Add to or Remove From Car Collection

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

- VP1. Users
- VP2. Customer Support (RideRecon)
- VP3. Marketing (RideRecon)
- VP4. Accounting (RideRecon)
- VP5. Dealership

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

- NA.

BE1. Create an Account #1

Preconditions: The user is not logged in, thus the log-in screen is displayed

VP1. User #1

Main Success Scenario

1. User opens the RideRecon app
2. The system responds by showing the user the log-in screen interface.
3. User selects create account
4. The system displays a screen to request information from the user relating to their account.
5. The user provides information to make the account and selects the option to confirm the account creation.
6. The system ensures the user's information is valid and grants them access to the main functionalities of the system.

Secondary Scenario

- 3i. User selects "sign in as guest"
 - 3i.1 User selects "sign in as guest"
 - 3i.2 System grants user access to RideRecon app in guest mode.
- 5i. User cancels the operation
 - 5i.1 User cancels the operation
 - 5i.2 System returns them to the log-in menu
- 6i. The system detects the user's information is invalid
 - 6i.1 The system detects the user's information is invalid
 - 6i.2 The system returns an message to the user stating the information is invalid.
 - 6i.3 The user acknowledges the message.
 - 6i.4 The system returns the user to the information request screen in step 5.
- 6ii. An error occurs and the system is unable to validate the user's information.
 - 6ii.1 An error occurs and the system is unable to validate the user's information.
 - 6ii.2 The system returns an message to the user stating an error has occurred.
 - 6ii.3 The user acknowledges the error.

- 6ii.4 The system returns the user to the information request screen in step 5.
- VP2. Customer Support #2
 - 5i. User selects get support option to speak with customer support regarding signup
 - 5i.1 User selects get support option to speak with customer support regarding signup
 - 5i.2 System redirects the user to a location outside of the app where the user can get in touch with our contractor's customer support line.
 - 6i. An error occurs and the user's information cannot be validated.
 - 6i.1 An error occurs and the user's information cannot be validated.
 - 6i.2 The system sends a message to the user asking if they want to submit a crash report for analysis.
 - 6i.3 The user selects yes or no.
 - 6i.4 If yes is selected, the system sends a crash report of the error to the system maintainers. If no is selected, nothing happens.
 - 6i.5 The system returns the user to the information request screen in step 5.
- VP3. Marketing #3
 - N/A
- VP4. Accounting #4
 - N/A
- VP5. Dealership #5
 - N/A

Global Scenario:

Main Success Scenario

1. User opens the RideRecon app
2. The system responds by showing the user the log-in screen interface.
3. User selects create account
4. The system displays a screen to request information from the user relating to their account.
5. The user provides information to make the account and selects the option to confirm the account creation.
6. The system ensures the user's information is valid and grants them access to the main functionalities of the system.

Secondary Scenario

- 3i. User selects "sign in as guest"
 - 3i.1 User selects "sign in as guest"
 - 3i.2 User gains access to the system in guest mode.
- 5i. User cancels the operation
 - 5i.1 User cancels the operation
 - 5i.2 System returns them to the log-in menu
- 5ii. User selects get support option to speak with customer support regarding signup
 - 5ii.1 User selects get support option to speak with customer support regarding signup
 - 5ii.2 System redirects the user to a location outside of the app where the user can get in touch with our contractor's customer support line.
- 6i. The system detects the user's information is invalid
 - 6i.1 The system detects the user's information is invalid
 - 6i.2 The system returns an message to the user stating the information is invalid.
 - 6i.3 The user acknowledges the message.
 - 6i.4 The system returns the user to the information request screen in step 5.
- 6ii. An error occurs and the system is unable to validate the user's information.
 - 6ii.1 An error occurs and the system is unable to validate the user's information.
 - 6ii.2 The system returns an message to the user stating an error has occurred. Alongside the message, the system prompts the user to submit a crash report to the system maintainers for analysis.
 - 6ii.4 The user selects either yes or no to submitting a crash report.

- 6ii.5** If the user selected yes, a crash report of the error is submitted to the system maintainers. If no is selected, nothing happens.
- 6ii.6** The system returns the user to the information request screen in step 5.

BE2. Authenticate User #2

Pre-Condition: The user must have an existing RideRecon account and access to the internet. The system must be able to connect to the authentication server to verify login credentials.

VP1. Users #1 Main Success Scenario

1. User opens the RideRecon app.
2. System displays the log-in screen.
3. User enters their email/username and password.
4. System verifies credentials against the database.
5. If credentials are valid, the system grants access to the user and loads the homepage.

Secondary Scenario

- 4i. User enters incorrect credentials.
- 4ii. System detects invalid credentials and displays an error message.
- 4iii. User is given the option to retry or reset their password.
- 4iv. If the user retries, they are redirected back to the login screen.
- 4v. If the user selects password reset, they are taken to the password recovery process.
- 5i. System is unable to connect to the authentication server.
- 5ii. System displays a message informing the user of a connection issue.
- 5iii. User is given the option to retry logging in.
- 6i. User selects "Remember Me" during login.
- 6ii. System securely stores authentication tokens to allow automatic login on future app openings.
- 7i. User selects "Sign in with Google/Apple ID".
- 7ii. System redirects user to the respective authentication service.
- 7iii. If authentication is successful, user is granted access to the app.

VP2. Customer Support (RideRecon) #2

Main Success Scenario

1. User selects the "Need Help?" option on the login screen.
2. System displays support options, including FAQs and a contact form.
3. User submits an inquiry regarding login issues.
4. System sends the inquiry to customer support.

Secondary Scenario 4i. System provides automated troubleshooting suggestions before allowing the user to submit a ticket.

- 4ii. If an error prevents login, system suggests clearing cache or checking for updates.

VP3. Marketing (RideRecon) #3

N/A

VP4. Accounting (RideRecon) #4

N/A

VP5. Dealership #5

N/A

Global Scenario:

Main Success Scenario

1. User opens the RideRecon app.
2. System displays the log-in screen.
3. User enters login credentials.
4. System verifies credentials.
5. If valid, user is granted access to the app.

Secondary Scenario

- 4i. User enters incorrect credentials and is prompted to retry or reset their password.
- 4ii. System encounters a connectivity issue and informs the user.
- 4iii. User opts to remember their credentials for future logins.
- 4iv. User logs in using third-party authentication (Google/Apple ID).

BE3. Upload Text and/or Image as Input #3

Pre-Condition: The user must be authenticated in the system and must have an active internet connection. The user must either provide an image of the car or enter a textual description. If using an image, the device must have a functioning camera or access to stored images.

VP1. Users #1

Main Success Scenario

- 1. The user selects an image OR enters text describing a car.
- 2. If an image is provided, the system checks its quality: If acceptable, the system processes the image. If blurry, the system prompts the user to retake the photo.
- 3. If text is provided, the system extracts key information and performs a database search.
- 4. The system provides instant feedback while processing.
- 5. The user receives results, including car specifications and pricing (if available).
- 6. The system sends the input data to the Experts for processing.

Secondary Scenario

- 6i. User provides an image, but it is blurry or unclear.
- 6ii. System detects poor image quality and prompts the user to re-upload a clearer image.
- 6iii. User does not have an image and enters text instead.
- 6iv. System prompts the user to refine their text description if it is too vague.
- 6v. The Experts detect that the provided image and text do not match the same car model.
- 6vi. System prompts the user to confirm or edit the text description to match the uploaded image.

VP2. Customer Support (RideRecon) #2

NA

VP3. Marketing (RideRecon) #3

NA

VP4. Accounting (RideRecon) #4

NA

VP5. Dealership #5

- 9i. The system determines that the dealership database lacks a match for the provided input.
- 9ii. The dealership can request to update the system database to ensure their listings appear in future searches.

Global Scenario:

Main Success Scenario

- 1. User opens the RideRecon app on their device.
- 2. System requires the user to log in and displays the login fields.
- 3. User enters login credentials.
- 4. System authenticates the user.
- 5. System prompts the user to upload an image or enter a text description of the car they want to identify.
- 6. User uploads an image or enters a text description.
- 7. System verifies the uploaded image quality and processes it.
- 8. System processes the text description using language processing and matches potential vehicles.
- 9. System sends the input data to the Experts for processing.

Secondary Scenario

- 6i. User provides an image, but it is blurry or unclear.
- 6ii. System detects poor image quality and prompts the user to re-upload a clearer image.
- 6iii. User does not have an image and enters text instead.
- 6iv. System prompts the user to refine their text description if it is too vague.
- 9i. The Experts detect that the provided image and text do not match the same car model.
- 9ii. System prompts the user to confirm or edit the text description to match the uploaded image.

BE4. Compare Expert Answers #4

VP1. User #1

Main Success Scenario

- 1. User provides the information of a car for the system to process.
- 2. System feeds input to all of its experts
- 3. Experts provide the system with their answer on information about the car.
- 4. The finalizer recognizes the experts agree on what kind of car the user inputs.
- 5. The finalizer processes the data from the experts for output.

Secondary Scenario

- 4i. The finalizer recognizes that not all of the experts agree on what kind of car has been input into the system.
 - 4i.1 The finalizer recognizes that not all of the experts agree on what kind of car has been input into the system.
 - 4i.2 Using conflict resolution methods, the finalizer will determine which experts to accept information from.
 - 4i.3 If a certain threshold of certainty is reached by the finalizer, the finalizer will use information from the selected experts in output processing. If not, the system will attempt to provide information to the experts that will refine their outputs and the system returns to step 3. in the process.
 - 4i.4 After a pre-defined maximum amount of attempts to refine the solution is reached, the system will notify the user that it was unable to find the car that was input.
 - 4i.5 The user acknowledges the message.
 - 4i.6 The system returns itself to the state where the user can provide input for identifying a car.

VP2. Customer Support #2

- 4i. Should the experts and finalizer fail to determine what car was input by the user, the system will prompt the user with the option to send feedback.
 - 4i.1 Should the experts and finalizer fail to determine what car was input by the user, the system will prompt the user with the option to send feedback.
 - 4i.2 The user can select yes or no to this prompt.
 - 4i.3 If the user selects no, nothing happens and the system is returned to a state where the user can provide input for identifying a car again. If the user selects yes, the system will prompt them for their feedback and any other important information.
 - 4i.4 The user inputs their feedback.
 - 4i.5 The system thanks the user for their feedback and returns them to a state where they can provide input for identifying a car again.

VP3. Marketing

N/A

VP4. Accounting

N/A

VP5. Dealership

N/A

Global Scenario:

Main Success Scenario

1. User provides the information of a car for the system to process.
2. System feeds input to all of its experts
3. Experts provide the system with their answer on information about the car.
4. The finalizer recognizes the experts agree on what kind of car the user inputs.
5. The finalizer processes the data from the experts for output.

Secondary Scenario

- 4i. The finalizer recognizes that not all of the experts agree on what kind of car has been input into the system.
 - 4i.1 The finalizer recognizes that not all of the experts agree on what kind of car has been input into the system.
 - 4i.2 Using conflict resolution methods, the finalizer will determine which experts to accept information from.
 - 4i.3 If a certain threshold of certainty is reached by the finalizer, the finalizer will use information from the selected experts in output processing. If not, the system will attempt to provide information to the experts that will refine their outputs and the system returns to step 3. in the process.
 - 4i.4 After a pre-defined maximum amount of attempts to refine the solution is reached, the system will notify the user that it was unable to find the car that was input. Alongside this message will be an option to send feedback to the system maintainers about the experience.
 - 4i.5 The user acknowledges the message and chooses not to give feedback.
 - 4i.5.i.1 The user decides to give feedback.
 - 4i.5.i.2 The system will prompt the user for their feedback and any other important information.
 - 4i.5.i.3 The user inputs their feedback.
 - 4i.5.i.4 The system thanks them for their response.
 - 4i.6 The system returns itself to the state where the user can provide input for identifying a car.

BE5. Finalize and Present Output With Identification Information #5

Pre-Condition: All experts have processed input and obtained their final answer for the make and model of the car.

VP1. Users #1

Main Success Scenario

1. User opens the RideRecon app on their device.
2. System requires the user to log in and displays the login fields.
3. User enters login credentials.
4. System authenticates the user.
5. System prompts the user to upload an image and text description of the car they want to identify.
6. User uploads an image with some text description of their car.
7. The input data is sent to the Experts for processing.
8. All Experts come to the same conclusion about the make and model of the car.
9. The Finalizer displays the make and model of the car, as well as the fun fact and where to purchase the car.

Secondary Scenario

- 6i. User doesn't have both forms of input.
- 6ii. System prompts the user to provide the missing form of input.
- 6iii. Available form of input is given to the Experts which will do their best to determine the car.
- 8i. The Experts come to different conclusions about the make and model of the car.
- 9i. The Finalizer displays all Expert answers, with a recommendation on which is most likely based on how many Experts came to the same conclusion.
- 9ii. The Finalizer also obtains fun facts and purchase information about ALL cars that the Ex-

perts concluded on.

VP2. Customer Support (RideRecon) #2

NA

VP3. Marketing (RideRecon) #3

NA

VP4. Accounting (RideRecon) #4

NA

VP5. Dealership #5

9i. The given dealership was not chosen as a source of purchase.

9ii. They can contact RideRecon to ensure that they are more likely to be chosen as the preferred purchase source.

Global Scenario:

Main Success Scenario

1. User opens the RideRecon app on their device.
2. System requires the user to log in and displays the login fields.
3. User enters login credentials.
4. System authenticates the user.
5. System prompts the user to upload an image and text description of the car they want to identify.
6. User uploads an image with some text description of their car.
7. The input data is sent to the Experts for processing.
8. All Experts come to the same conclusion about the make and model of the car.
9. The Finalizer displays the make and model of the car, as well as the fun fact and where to purchase the car.

Secondary Scenario

- 6i. User doesn't have both forms of input.
- 6ii. System prompts the user to provide the missing form of input.
- 6iii. Available form of input is given to the Experts which will do their best to determine the car.
- 8i. The Experts come to different conclusions about the make and model of the car.
- 9i. The Finalizer displays all Expert answers, with a recommendation on which is most likely based on how many Experts came to the same conclusion.
- 9ii. The Finalizer also obtains fun facts and purchase information about ALL cars that the Experts concluded on.

BE6. Add to Car Collection #6

Pre-Condition: The user must be logged into their account.

VP1. Users #1

Main Success Scenario

1. User accesses their car collections through the icon on the homepage.
2. User chooses the add to car collection icon.
3. System prompts the user to choose a car from their past identification history.
4. User selected an identified car.
5. System prompts the user to choose a car collection.
6. User selects an existing car collection to add the identified car to.
7. The input data is sent to the Experts for processing.
8. System provides a review of changes to car collection and prompts the user for verification.
9. User verifies changes and returns to homepage.

Secondary Scenario

- 1i. User chooses to add a car to a collection immediately after identification. Skip VP1.2 & VP1.3.
- 2i. No past identified car. Return to homepage.
- 8i. User declines changes. No modification made to car collection and return to homepage.

VP2. Customer Support (RideRecon) #2

NA

VP3. Marketing (RideRecon) #3

NA

VP4. Accounting (RideRecon) #4

NA

VP5. Dealership #5

NA

Global Scenario:

Main Success Scenario

1. User accesses their car collections through the icon on the homepage.
2. User chooses the add to car collection icon.
3. System prompts the user to choose a car from their past identification history.
4. User selected an identified car.
5. System prompts the user to choose a car collection.
6. User selects an existing car collection to add the identified car to.
7. The input data is sent to the Experts for processing.
8. System provides a review of changes to car collection and prompts the user for verification.
9. User verifies changes and returns to homepage.

Secondary Scenario

- 1i. User chooses to add a car to a collection immediately after identification. Skip VP1.2 & VP1.3.
- 2i. No past identified car. Return to homepage.
- 8i. User declines changes. No modification made to car collection and return to homepage.

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 application must use company colours that align with the ones used in their branding.

Rationale: The user should feel as if the application is closely tied to the company that contracted us and their mission statements.

LF-A2. The application should not use harsh and overly vibrant colours

Rationale: If the application is hard on the eyes to use, the user will have a worse experience overall and be adverse to using the application.

LF-A3. The application should be consistent in how it associates colours with certain buttons or actions.

Rationale: When a user goes to do actions across the application, the colour of buttons relating to

certain prompts like "yes" or "no" should remain consistent across all of those prompts. For example, if a "yes" button uses the colour green, then any other time "yes" comes up it should also be green.

LF-A4. Users should be able to quickly distinguish what is and what is not a button.

Rationale: The user should be able to quickly understand the options available to them without needing a lot of effort or research.

5.1.2 Style Requirements

LF-S1. The system must scale its UI so that it can fit to the size of the screen device.

Rationale: Device screens come in all shapes and sizes, thus the application should be able accommodate all kinds of screen sizes.

LF-S2. Additional options or menus should be hidden under one to a small handful of dropdown menus.

Rationale: The user's screen should not be cluttered with an abundance of buttons that makes the application hard to navigate.

LF-S3. The system should adhere to general practices for making applications for android devices

Rationale: Since the system is made for android devices, we need to follow general practices for making android applications to fit with the styles of other android applications.

LF-S4. When an expert is mentioned in the system, if it was called by an API call, the system should use the expert's branding appearance when displaying information from it.

Rationale: When a user interacts with the system in a way that displays information from the experts, if the branding from the given expert is used, it will be easily identifiable by the user and will help them gain trust in the information they are receiving.

5.2 Usability and Humanity Requirements

5.2.1 Ease of Use Requirements

UH-EOU1. The system should only require a maximum of two inputs from the user to produce an output.

Rationale: The user should not need to go through an extensive process to identify a car. They should be able to take a picture, type in a model, etc. and receive the results they are looking for.

UH-EOU2. The system should be usable with only one hand.

Rationale: There are many times in a user's daily life where only one hand is available to them. Thus, making the app usable with only one hand means users would be able to use the app even in those moments.

5.2.2 Personalization and Internationalization Requirements

UH-PI1. The system should have a variety of language options to choose from.

Rationale: With a variety of users comes a variety of languages that are spoken. Thus, to improve the user's experience, we should give them a wide range of languages to choose from so they can use the app with the language they are most comfortable with.

UH-PI2. The system should be easy to use by both right and left handed people.

Rationale: Users have a preference whether to use their phone with their left or right hand, thus the system should be comfortable to use regardless of which hand is used. Furthermore, there are occasions when users are not able to use one hand or the other, so making the application only feel comfortable to use with a specific hand could make the application feel unwieldy [3].

UH-PI3. The system should be usable on a lower device brightness.

Rationale: Users will not always have their device at full brightness, so if the device becomes harder to use at these low brightness settings the user will find the system more tedious to work with.

5.2.3 Learning Requirements

- UH-L1. Users should be able to start identifying a car within their first 5 minutes of logging into the application.
Rationale: The process of inputting the information required to identify a car should be minimal and should not be complex. Since coming across a car may be a quick experience, it should not take the user a long time to learn how to use the system effectively.

5.2.4 Understandability and Politeness Requirements

- UH-UP1. Any icons used in the system without explanation should be universally understood by users or easy to learn.
Rationale: An icon that can be quickly understood or universally known will make it easier for users to understand what our system does.

5.2.5 Accessibility Requirements

- UH-A1. The system should be able to accommodate users with colour blindness.
Rationale: Approximately 360 Million people are colourblind [4]. Thus, as to not alienate a large demographic of users, the interface should be designed with these users in mind.
- UH-A2. The system should be compatible with accessibility screen readers.
Rationale: Users with vision or comprehension disabilities might use screen readers to better navigate the devices and applications that they are engaging with. Thus, to make the system accessible to those users it would be necessary to make the system friendly to common screen readers.

5.3 Performance Requirements

5.3.1 Speed and Latency Requirements

- PR-SL1. The response time of all core APIs should be under 1 second for 90% of requests and must not exceed 5 seconds.
Rationale: Research on mobile application UX shows that users expect instantaneous responses, with 1 second or less considered a high-performing API response time. At 2 seconds, delays become noticeable, and beyond 5 seconds, users are likely to abandon the interaction or log off the app[5].
- PR-SL2. A cold start (first launch or after app closure) must be under 5 seconds, while a hot start (reopening from background memory) must be under 1.5 seconds.
Rationale: Google's Android App Performance Best Practices indicate that mobile apps should launch within 5 seconds for cold starts and under 1.5 seconds for hot starts to maintain user engagement. A slow start-up time increases drop-off rates, as users may assume the app is unresponsive or broken[5][6].
- PR-SL3. Uploading a photo should take less than 2 seconds under normal mobile network conditions.
Rationale: Google's Mobile UX Research states that 90% of mobile users abandon an interaction if an image takes longer than 5 seconds to load. Since RideRecon relies on image uploads for vehicle identification, ensuring fast upload speeds is critical to maintaining user engagement.
- PR-SL4. Identifying a car must take under 3 seconds for 80% of requests.
Rationale: Users expect AI-driven systems to return results quickly, particularly for image recognition tasks, where they anticipate near-instant feedback. The 3-second goal aligns with industry benchmarks for real-time image classification models. If external APIs take longer than 5 seconds, users will get frustrated[6].

5.3.2 Safety-Critical Requirements

- PR-SC1. N/A

5.3.3 Precision or Accuracy Requirements

- PR-PA1. The system must achieve an accuracy of at least 90% in correctly identifying car make and model.
Rationale: High identification accuracy is crucial for user trust. Machine learning models trained on large, diverse datasets should reach this threshold, but real-world accuracy should be measured using benchmark tests on real user images. If accuracy is below this level, model retraining or additional expert validation is necessary.
- PR-PA2. The confidence score of each expert’s prediction must be included with the result.
Rationale: Since multiple experts provide input, some may be more confident than others. Showing confidence scores allows users to make informed decisions instead of blindly trusting the system.
- PR-PA3. The image search expert (RIS) should correctly match at least 95% of input images when the car exists in the database.
Rationale: Reverse image search should work reliably for known cars. If the car is not found, the system should clearly indicate this rather than returning a false match[7].

5.3.4 Reliability and Availability Requirements

- PR-RA1. The system must maintain a minimum uptime of 99.5% over any rolling 30-day period.
Rationale: Availability is critical for user retention. An uptime of 99.5% aligns with industry best practices for consumer applications, meaning downtime does not exceed 3.6 hours per month. If the system becomes unreliable, users may abandon it in favour of alternatives[8].
- PR-RA2. The system must be able to handle at least 2000 concurrent users without significant degradation in performance.
Rationale: Ensuring the system can handle high traffic loads prevents server crashes due to excessive load, increased response times negatively affecting user experience, and poor scalability as the user base grows[9].
- PR-RA3. The system must detect external API failures automatically to prioritize internal processing and notify users of limited functionality.
Rationale: Dependence on third-party APIs introduces potential points of failure. A fallback mechanism ensures that core functionalities, such as image recognition and classification, remain operational while providing users with clear explanations instead of vague errors. This enhances system resilience, preventing a complete breakdown if an API becomes slow or unresponsive[10].

5.3.5 Robustness or Fault-Tolerance Requirements

- PR-RFT1. The system must detect invalid, corrupt, or unsupported image uploads and provide users with actionable guidance.
Rationale: Users may attempt to upload incorrect file types, oversized images, or corrupted files, leading to failed identification attempts.
- PR-RFT2. The system must detect ambiguous text descriptions.
Rationale: User-submitted descriptions such as “blue car” may be too vague for accurate identification.

5.3.6 Capacity Requirements

- PR-C1. The system must support storage for at least 1 million images, with an average image size of 5MB.
Rationale: As user activity increases, a scalable storage solution is required to ensure historical searches remain accessible. Cloud storage solutions such as AWS S3 or Google Cloud Storage should be considered to provide cost-effective scalability without impacting performance[11][12].
- PR-C2. The system must optimize bandwidth efficiency to ensure that total data usage does not exceed 3MB per processed request on average.
Rationale: Since many users will access RideRecon from mobile devices, excessive data usage can lead

to long load times and increased costs for users with limited data plans. The system should implement techniques to minimize bandwidth consumption[13].

5.3.7 Scalability or Extensibility Requirements

PR-SE1. The system must follow SOLID design principles and implement modular architecture to support future enhancements.

Rationale: A well-structured and modular codebase allows for easy modifications and feature expansion without breaking existing functionality. By adhering to SOLID principles, we can integrate new expert models or improve the system without requiring major code refactoring.

PR-SE2. The system must support dynamic addition and removal of AI experts without requiring a full system redeployment.

Rationale: As AI models improve over time, the system should allow for seamless integration of new expert modules. This ensures RideRecon remains competitive by using the latest advancements in AI-based object recognition.

PR-SE3. The system should support multi-platform accessibility, allowing for future deployment on web and iOS platforms.

Rationale: Although RideRecon is initially developed for Android, future expansion to web and iOS platforms will increase accessibility.

5.3.8 Longevity Requirements

PR-L1. The system must support periodic model retraining and dataset updates to maintain high identification accuracy.

Rationale: Machine learning models degrade over time as new car models enter the market, and retraining the models will ensure that identification remains accurate and up to date[14].

PR-L2. The system should be designed to support future AI model improvements.

Rationale: As AI research advances, more efficient architectures may become industry standards. The system should be flexible enough to integrate these advancements without requiring major architectural changes.

PR-L3. The source code and documentation must be maintained with clear version control and developer guidelines for the duration of development.

Rationale: Proper documentation and version control ensure that future developers can understand and modify the system efficiently, preventing code degradation and technical debt.

5.4 Operational and Environmental Requirements

5.4.1 Expected Physical Environment

OE-EPE1. The system must function in a variety of lighting conditions for image capture, including low-light and high-glare environments.

Rationale: Users may take photos of vehicles in dimly lit parking lots, bright sunlight, or reflective surfaces, which can impact image quality. The system should use adaptive preprocessing techniques such as contrast adjustment to improve recognition accuracy across different lighting conditions[15].

OE-EPE2. The system must be designed to function efficiently on mobile devices with limited hardware capabilities.

Rationale: Not all users will have high-end smartphones. The application should be optimized for devices with lower processing power, low RAM, and slow internet connections. Techniques such as model quantization should be used to ensure a smooth user experience[16].

5.4.2 Requirements for Interfacing with Adjacent Systems

OE-IA1. The system must support integration with third-party APIs, including Google Vision API, OpenAI, and other AI-based identification services.

Rationale: RideRecon depends on multiple AI models and image recognition APIs to provide accurate results. The architecture must allow seamless communication between these external services, ensuring modularity and flexibility for potential future integrations.

OE-IA2. The system must allow interoperability with cloud storage services for secure image storage and retrieval.

Rationale: Users may want to save their past identifications or access reports. The system should integrate with cloud storage providers to securely manage image uploads and retrievals while ensuring data privacy[17].

5.4.3 Productization Requirements

OE-P1. N/A

5.4.4 Release Requirements

OE-R1. The application must be compatible with Android 9.0 and above.

Rationale: Ensuring compatibility with older but still widely used Android versions allows broader accessibility. Android 9.0 provides sufficient support for modern AI and image processing capabilities without compromising performance[18].

OE-R2. The system must undergo beta testing with at least 10 users before public release.

Rationale: A beta testing phase ensures that real-world feedback is collected before full deployment. This helps identify issues and unexpected performance bottlenecks, reducing the likelihood of post-launch failures.

OE-R3. The application must comply with Google Play Store policies, including data privacy and API usage guidelines.

Rationale: To ensure the application is accepted and remains available in the Play Store, it must adhere to Google's developer guidelines regarding privacy policies, data handling, and API rate limits. Violating these guidelines could lead to the app being delisted[19].

5.5 Maintainability and Support Requirements

5.5.1 Maintenance Requirements

MS-M1. The system must receive monthly software updates to address bugs, improve performance, and enhance security.

Rationale: Regular maintenance updates ensure the system remains stable, secure, and efficient. Monthly updates allow us to patch vulnerabilities, optimize code performance, and introduce incremental improvements based on user feedback.

MS-M2. The system must support rollback functionality to revert to a stable version in case of a failed update.

Rationale: Software updates sometimes introduce unintended bugs or compatibility issues. Implementing a rollback mechanism ensures that users can revert to a previously stable version while we can address the issue, minimizing disruptions to service availability.

5.5.2 Supportability Requirements

MS-S1. The application must include an in-app FAQ and troubleshooting guide for common issues.

Rationale: Providing users with an easily accessible FAQ section reduces reliance on customer support. A structured troubleshooting guide can assist users in resolving common problems without requiring direct assistance.

- MS-S2. The system must provide a support ticketing mechanism for users to report unresolved issues.
Rationale: Some issues may require our direct intervention. A support ticketing system allows users to submit reporting ensuring the prioritization of user-reported problems.

5.5.3 Adaptability Requirements

- MS-A1. The system must be designed to support future updates with minimal changes to core functionality.
Rationale: As AI models and image recognition technologies improve, the system should accommodate improvements without requiring extensive rework. Using modular design principles ensures that new features or optimizations can be implemented easily.
- MS-A2. The system must be configurable to adjust recognition parameters based on different regions and datasets.
Rationale: Vehicle models and identification patterns may vary across countries and regions. The system should allow customization of recognition parameters to improve accuracy in different environments.

5.6 Security Requirements

5.6.1 Access Requirements

- SR-AC1. Users should only be able to access an account if they have the correct login credentials.
Rationale: This is to ensure a user has legitimate access to an account. This also ensures that there is no "master account" or backdoor that could be exploited to get into a user's account.
- SR-AC2. The system must ask for access to the user's camera if they want to use the camera functionality.
Rationale: It is imperative to ask for the user's consent to use the camera in the application so they understand what features of their phone the system is gaining access to.
- SR-AC3. The system must ask for access to the user's photo library if they want to upload a photo of a car for identification.
Rationale: It is imperative to ask for the user's consent to use photos from their photo library in the application so they understand what features of their phone the system is gaining access to.

5.6.2 Integrity Requirements

- SR-INT1. An account's history or car collection can only be manipulated if a user is logged into the account.
Rationale: Outside actors that do not have access to a user's account should not be able to manipulate data specific to that account.

5.6.3 Privacy Requirements

- SR-P1. The system should comply with Android's Privacy Checklist[20].
Rationale: Since this is an Android application, following their best practices for application development will ensure the system complies with Android's standards for privacy.

5.6.4 Audit Requirements

- SR-AU1. N/A

5.6.5 Immunity Requirements

- SR-IM1. When an external tool is used for development, ensure it is reliable and trustworthy.
Rationale: Some external tools could be compromised or contain malicious code. While this cannot be completely avoided, risk can be mitigated by using tools that are trusted and have a good reputation of safety and reliability.

SR-IM2. The application should be resistant to malicious input from the user.

Rationale: The user may have tactics of penetrating security systems such as SQL injections. Thus, the system should be resistant to malicious input like this.

5.7 Cultural and Political Requirements

5.7.1 Cultural Requirements

CP-C1. The system is able to identify a car, regardless of which side the drivers side is on.

Rationale: Countries around the world do not always drive on the same side of the road. In some countries, drivers drive on the left side of the road while others drive on the right. Depending on the country, the car has been designed with the drivers side on the left or the right side of the car. Thus, our system should be able to handle either kinds of these cars for users across the world.

5.7.2 Political Requirements

CP-P1. N/A

5.8 Legal Requirements

5.8.1 Compliance Requirements

LR-COMP1. The system must comply with PIPEDA (Personal Information Protection and Electronic Documents Act) for handling user data in Canada.

Rationale: RideRecon collects and processes user-submitted images and metadata. Under PIPEDA, personal information must be collected with user consent, securely stored, and only used for stated purposes. The system must include a privacy policy detailing data collection, storage, and sharing practices[21].

LR-COMP2. The system must comply with GDPR (General Data Protection Regulation) if accessible in the European Union.

Rationale: If RideRecon is available in the EU, it must adhere to GDPR principles, including user consent, data minimization, and the right to be forgotten. The system should allow users to request data deletion and should not store personally identifiable information beyond what is necessary[22].

LR-COMP3. The application must comply with Google Play Developer Distribution Agreement.

Rationale: To be approved and remain listed on the Google Play Store, RideRecon must follow Google's data usage policies, user consent requirements, and restricted API guidelines. Failure to comply could result in app suspension or removal[19].

LR-COMP4. The system must include a legally adequate Terms of Service and Privacy Policy accessible from within the app.

Rationale: Users must be informed about their rights, data usage policies, and liability limitations before using the app. The Terms of Service should outline acceptable use, intellectual property rights, disclaimers, and dispute resolution mechanisms[23].

5.8.2 Standards Requirements

LR-STD1. The system must use industry-standard encryption protocols (e.g., AES-256) for data transmission and storage.

Rationale: To protect user-uploaded images and metadata, the system must encrypt sensitive data during transmission and at rest. This ensures compliance with security best practices and helps prevent data breaches[24][25].

LR-STD2. The system must not log personally identifiable information in plaintext storage or logs.

Rationale: To comply with privacy laws and security best practices, logs should not store sensitive user data in plaintext. Any logging mechanism should mask or anonymize sensitive data before storage to reduce exposure in case of a breach adhering to OWASP best practices[26].

LR-STD3. The system must adhere to fair use and copyright laws when handling third-party images.

Rationale: Users may upload images sourced from third-party websites or copyrighted materials. RideRecon should not process or store copyrighted content without permission, and it must include disclaimers clarifying user responsibility for uploaded images, to adhere to DMCA laws(The Notice and Notice regime in Canada)[27].

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.





Hashim Bukhtiar	Jaden Moore	James Ariache	Olivia Reich	Omar Abdelhamid
1.1, 1.2, 1.3, 1.4, 1.5 Section 3 BE5 in Section 4	5.3, 5.4, 5.5, 5.8 BE2 in Section 4	5.1, 5.2, 5.6, 5.7 BE1 in Section 4 BE4 in Section 4	2.1, 2.2, 2.3 BE6 in Section 4	2.4, 2.5, 2.6 BE3 in Section 4
				Omar Hassan

Table 1: Division of Labour