Deliverable #1 Template : Software Requirement Specification (SRS)

SE 3A04: Software Design II – Large System Design

February 6, 2025

Tutorial Number: T01
Group Number: G3 Gx
Group Members:

- Group Member Name (as listed in Avenue)
- You do not need to use student #s or macid (keep those private).

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

This SRS will describe the software requiremnets for GeoLens, a community-driven application for identifying locations based on images, descriptitons and discussions. This document outlines the system's overall purpose, the context for building the application, functional and non-functional requirements and user iteraction.

1.1 Purpose

This document serves as a guideline for developers, designers, QA engineers, project managers and other stakeholders, ensuring a clear understanding of the system, including it's objectives, users, expected behavior, and technical constraints. As the project progresses and changes, the document will provide a baseline from which these changes can be compared. This SRS may also aid in risk identification early in the project lifecycle by outlining dependencies and constraints, allowing managers to prevent risks.

1.2 Scope

GeoLens is a community and AI driven application designed to identify locations from images. It consists of several individual software products described below.

The **Forum** UI is the primary interface where users interact with GeoLens. It allows users to upload images of locations for identification, view responses, and engage with the community. If the forum UI can provide ease of use and an interactive experience it will encourage community engagement and bring traffic to the application. The interface facilitates discussions, displays aggregated predictions, and contains gamification elements such as leaderboards. It does not perform image analysis or identification itself but serves as the user's gateway to the platform.

The **Orchestrator** is responsible for handling submitted images, delegating tasks to experts (AI, geography experts, and community aggregators), and returning the final result to users. It preprocesses images to optimize identification accuracy and ensures smooth communication between different system components. The back end does not analyze images directly but functions as the orchestration layer that efficiently processes requests.

The **Host DBMS** stores user account information, historical user interactions, reputation scores, previously identified locations and leaderboards. It ensures that the system retains valuable data, in a structured manner that is efficiently accessible by other system components. The database should be highly secure, as it contains sensitive user information. The database does not process images or make identifications but acts as the central location for all platform data.

Identify the software product(s) to be produced, and name each (e.g., Host DBMS, Report Generator, etc.) Explain what the software product(s) will do (and, if necessary, also state what they will not do). Describe the application of the software being specified, including relevant benefits, objectives, and goals. Be consistent with similar statements in higher-level specifications (e.g., the system requirements specification), if they exist

1.3 Definitions, Acronyms, and Abbreviations

- Provide the definitions of all terms, acronyms, and abbreviations required to properly interpret the SRS.
- This should be in alphabetical order.

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

- Describe what the remainder of the document/SRS contains.
(e.g. "Section 2 discusses...Section 3...")

2 Overall Product Description

- This section should describe the general factors that affect the product and its requirements.
- It does not state specific requirements.
- It provides a *background* for those requirements and makes them easier to understand.

2.1 Product Perspective

- Put the product into perspective with other related products, i.e., context
- If the product is independent and totally self-contained, it should be stated here
- If the SRS defines a product that is a component of a larger system, then this subsection should relate the requirements of that larger system to the functionality of the software being developed. Identify interfaces between that larger system and the software to be developed.
- A block diagram showing the major components of the larger system, interconnections, and external interfaces can be helpful

2.2 Product Functions

- Provide a *summary* of the major functions that the software will perform.
  - **Example**: An SRS for an accounting program may use this part to address customer account maintenance, customer statement, and invoice preparation without mentioning the vast amount of detail that each of those functions requires.
- Functions should be organized in a way that makes the list of functions understandable to the customer or to anyone else reading the document for the first time
- Present the functions in a list format - each item should be one function, with a brief description of it
- Textual or graphical methods can be used to show the different functions and their relationships
  - Such a diagram is not intended to show a design of a product, but simply shows the logical relationships among variables

2.3 User Characteristics

- Describe those general characteristics of the intended users of the product including educational level, experience, and technical expertise
- Since there will be many users, you may wish to divide into different user types or personas

2.4 Constraints

- Provide a general description of any constraints that will limit the developer's options

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.

2.6 Apportioning of Requirements

- Identify requirements that may be delayed until future versions of the system

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. The system shall have a minimalistic and clean UI.

Rationale:

LF-A2. All images in the system must be high-quality with the resolution well-adjusted.

LF-A3. The system shall use a consistent and legible font.

5.1.2 Style Requirements

LF-S1. The system shall have both a light and dark mode option.

LF-S2. The system must follow a gamified design and have visually appealing elements (ex. Leaderboard, user profile).

LF-S3. The system must scale to the size of the screen.

LF-S4. The system must use consistent spacing and padding between elements.

5.2 Usability and Humanity Requirements

5.2.1 Ease of Use Requirements

UH-EOU1. The system's buttons shall be big and bright in colour.

UH-EOU2. The system shall allow users to complete the identification of a location within four steps.

UH-EOU3. The system shall allow users to report incorrect results through a feedback tool in the application.

5.2.2 Personalization and Internationalization Requirements

UH-PI1. The system must be able to support multiple languages.

UH-PI2. The system shall allow the user to select between the metric or imperial system to display information.

5.2.3 Learning Requirements

UH-L1. The system shall have a basic tutorial for navigating through the features, which automatically executes the first time a user opens the app and is available at all times within the app.

UH-L2. The system shall provide short descriptions when the user presses and holds key features, that automatically fade after a short period.

UH-L3. The system shall have a demo that introduces the user to the app features and allows them to test it out before having to create an account.

5.2.4 Understandability and Politeness Requirements

UH-UP1. The system shall hide information and aspects of the app construction that are necessary for the user to interact with.

UH-UP2. The system shall provide positive engagement with the user when they successfully identify a location.

5.2.5 Accessibility Requirements

UH-A1. The system must be compatible with screen readers.

UH-A2. The system shall allow built-in zoom-in and zoom-out features.

UH-A3. The system shall provide high-contrast options that are colorblind-friendly.

5.3 Performance Requirements

5.3.1 Speed and Latency Requirements

PR-SL1. The system shall have a basic app response time of no longer than 3 seconds.

PR-SL2. The system shall return location identification results within 5 seconds.

5.3.2 Safety-Critical Requirements

PR-SC1. The system must securely encrypt all user data.

PR-SC2. The system must not return locations that reveal people's addresses or sensitive information.

5.3.3 Precision or Accuracy Requirements

PR-PA1. The system must successfully return the general location of the image processed to within 25 km of the actual area.

PR-PA2. The system shall have a priority setting to assess the accuracy of each of the API's results.

5.3.4 Reliability and Availability Requirements

PR-RA1. The system must maintain an uptime of 99%, except during routine maintenance, patch updates, and unexpected situations (ex. power outage).

PR-RA2. The system shall save and backup the user's progress continuously.

5.3.5 Robustness or Fault-Tolerance Requirements

PR-RFT1. The system shall be able to handle incorrect inputs or inputs of wrong formats.

PR-RFT2. The system must retry a failed image input up to three times before alerting the user of an error.

5.3.6 Capacity Requirements

PR-C1. The system must be able to support at least 10000 concurrent users during peak usage hours.

5.3.7 Scalability or Extensibility Requirements

PR-SE1. The architecture of the system must be modular and allow APIs to be integrated without major refactoring.

5.3.8 Longevity Requirements

PR-L1. The system shall have quarterly updates for new and changed locations.

5.4 Operational and Environmental Requirements

5.4.1 Expected Physical Environment

OE-EPE1. The system should be able to process images in various outdoor conditions.

5.4.2 Requirements for Interfacing with Adjacent Systems

OE-IA1. The system must be able to send and receive geolocation data with all the connected APIs.

5.4.3 Productization Requirements

OE-P1. N/A

5.4.4 Release Requirements

OE-R1. The system must be compatible with Android 14 (API level 34).

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.