****

**REPUBLIC OF CAMEROON**

PEACE-WORK-FATHERLAND

THE UNIVERSITY OF BUEA

P.O Box 63,

Buea, South West Region

Cameroon

Tel: (237) 3332 21 34/3332 26 90

Fax: (237) 3332 22 72

**FACULTY OF ENGINEERING AND TECHNOLOGY**

**Dean: Agbor Dieudonne Agbor PhD**

**Vice-Dean : Nde D. Nguti, PhD**

**Faculty Officer : Mme Ekpombang Patience Taku**

Date:

**CEF440 – TASK 2:**

**REQUIREMENT GATHERING**

**FOR A ROAD SIGN AND ROAD STATE MOBILE NOTIFICATION APPLICATION**

Presented by

**GROUP 4**

|  |  |  |
| --- | --- | --- |
| SN | NAME | MATRICULE NUMBER |
| 1 | DJOUMESSI NGUEFACK IVAN | FE21A174 |
| 2 | NJOYONG GODWILL | FE21A273 |
| 3 | TETUH WIBINAG ENGONWEI | FE21A322 |
| 4 | DJOUMESSI DONGMO RONSARD CARNEGIE | FE21A172 |
| 5 | GATCHUINNE MOMO MURIELLE CYNTHIA | FE21A200 |

**Valery Nkemeni, PhD**

**Course Supervisor**

Designing and Implementing of a Road Sign and Road State Mobile Notification Application.

Task 3: Requirement Analysis

Introduction

Requirements analysis, also known as requirements engineering, is a crucial phase in the development of a new product or the modification of an existing one. It involves understanding and defining the needs and expectations of users or stakeholders to ensure that the final product meets their requirements. This process typically requires the collaboration of a team with diverse skills and expertise.

One essential aspect of requirements analysis is the identification and documentation of user expectations. This involves gathering information about the desired functionalities, features, and qualities of the product. The requirements may encompass both functional aspects (what the product should do) and non-functional aspects (qualities such as performance, reliability, and usability).

Analysing the Requirements.

Requirement Prioritisation

Functional Requirements

Functional requirements are specific features and capabilities that the application should possess to fulfill its purpose effectively. During the brainstorming session, the internal stakeholders likely discussed and documented functional requirements.

We evaluated these features based on the following standards.

* Clarity
* Completeness
* Consistency
* Feasibility

Functional requirements are specific features and capabilities that the application should possess to fulfill its purpose effectively. During the brainstorming session, the internal stakeholders likely discussed and documented functional requirements such as

Higher priority Requirements

They are those gotten from surveys with high number of votes and interviews since they correspond to the ones directly gotten from the users additionally the ones gotten from retro-engineering.

1. **Real-time Traffic Updates**:

Integrate a feature that provides

* Real-time updates on construction zones
* Weather conditions
* Road closures, hazards, and checkpoints.
* Providing timely and accurate updates on the current state of the road
* Including information on road conditions and traffic congestion.

This could be achieved by tapping into various data sources such as government agencies, traffic cameras, and user reports.

1. **Location-Based Services**

* Utilize GPS data to provide location-based services
* allowing users to receive updates relevant to their current location

1. **Automatic Start and Stop**

Implement a feature that automatically starts and stops updating based on user preferences or when the app detects the user is driving.

1. **Display Road Signs**

Incorporate technology that recognizes and displays road signs in the app interface or overlay them on the video feed if applicable.

1. **User Feedback and Collaboration**

Allow users to provide feedback on the accuracy of updates and collaborate by reporting road conditions or incidents they encounter.

1. **Audio Notifications**

Implement audio notifications for important updates to minimize distractions. Users can customize the types of notifications they receive audibly.

1. **Current Speed, GPS Data, Date, and Time**

Embed relevant information such as current speed, GPS data, date, and time into the video file to provide additional context and ensure accuracy.

Medium priority Requirements

Addressing the medium priority requirements involves enhancing navigation features, optimizing routes, and improving data visualization. Here's how these requirements can be implemented.

1. **Navigation Assistance**

Provide turn-by-turn navigation assistance to users, guiding them along their chosen route. This can include voice prompts, visual cues, and lane guidance to help drivers navigate complex intersections.

1. **Data Visualization**

Implement data visualization techniques to present information such as traffic congestion, road closures, and construction zones in an intuitive and easy-to-understand manner. This could include color-coded maps, charts, and graphs to highlight relevant data points.

1. **Route Optimization**

Develop algorithms that analyze real-time traffic data to suggest the most efficient routes to users. This can help drivers avoid congestion, reduce travel time, and save fuel.

1. **Navigation Assistance with Rerouting Options**

Offer users the ability to reroute their journey in case of unexpected obstacles or changes in traffic conditions. This could be triggered automatically based on real-time updates or initiated by the user manually.

1. **Speed Camera Warnings**

Integrate a feature that alerts users to the presence of speed cameras along their route. This can help drivers adhere to speed limits and avoid potential fines.

1. **Exposure Compensation Feature**

Implement a feature that automatically adjusts the brightness levels of the app interface or camera feed to compensate for changes in lighting conditions. This ensures that the display remains visible and readable at all times, even in varying light conditions.

lower priority Requirements

Addressing the lower priority requirements involves adding supplementary features that enhance user convenience and interaction with the application. Here's how these requirements can be implemented:

1. **Hands-Free Reporting Capabilities**:

Develop a hands-free reporting feature that allows users to report incidents, such as accidents or road hazards, using voice commands or gesture controls. This ensures that users can report incidents safely while driving without being distracted.

1. **Integration of Existing Traffic Monitoring Systems**

Collaborate with existing traffic monitoring systems, such as government agencies or private companies, to integrate their data into the application. This can provide users with access to additional real-time traffic information and improve the accuracy of updates.

1. **User Incident Reporting**

Enable users to report incidents they encounter on the road, such as accidents, potholes, or debris. This crowdsourced data can supplement existing traffic information and help other users navigate more safely.

1. **Easy Sharing of Captured Video Footage on YouTube**

Implement a feature that allows users to easily share captured video footage of their journey on YouTube or other social media platforms. This can be done through integration with the respective platforms' APIs and providing a streamlined sharing process within the application.

* Non-Functional Requirements

High Priority Requirements

Performance:

* **Responsiveness:**
  + The app should load quickly and respond to user actions instantly to avoid frustration while driving. This is critical for safety reasons.
* **Low Latency Notifications:**
  + Real-time updates on road conditions need to be delivered with minimal delay to ensure users have the latest information for safe navigation.
* **Battery Efficiency:**
  + Since the app might run in the background for extended periods, it should be optimized to minimize battery consumption on mobile devices.

Meduim Priority Reaquirement

## Usability:

* **Intuitive Interface:**
  + The user interface (UI) should be clean, uncluttered, and easy to navigate, even for users with limited experience with smartphones. Icons and buttons should be clear and easily recognizable.
* **Minimal Distraction:**

Visual information and notification alerts should be designed to be clear and informative at a glance, minimizing the time users need to look away from the road.

* **Voice Control Options:**
  + Consider incorporating voice commands for hands-free interaction with the app, allowing users to access information or report incidents without taking their eyes off the road.

## Data Management:

* **Road Sign Information Source:**
  + Decide whether road sign data will be pre-loaded on the app or updated dynamically. Pre-loaded data might be suitable for static signs, while real-time updates are crucial for temporary signs or variable message boards.
* **Real-time Data Acquisition:**
  + Explore options for obtaining real-time road state data. This could involve integrating with existing traffic management systems, leveraging data from connected vehicles, or incorporating user reports.
* **Data Update Mechanism:**
  + Establish a system for updating the app's database with new signs, changes in road conditions, or any other relevant information. This might involve automatic updates or user-triggered refresh options.

Low priority requirement

## Security:

* **User Privacy:**
  + The app should collect minimal user data, ideally just location for notification purposes. This data should be anonymized and used only for the stated purpose of improving road safety.
* **Data Security:**
  + The app should implement robust security measures to protect user data from unauthorized access, hacking attempts, or malware. Secure data storage and transmission protocols are essential.

**Technical requirements Gathered**

The following are the technical requirements analyzed for the proper execution of this project:

**Frontend Framework for Mobile App:**

* **React Native**

A popular framework for building cross-platform mobile apps using JavaScript and React.

**Backend Frameworks for Mobile App:**

* **Express.js**

A fast and minimalist web application framework for Node.js.

* **Laravel**

A PHP framework for building scalable and feature-rich web applications.

**Database Frameworks for Mobile App**

* **Firebase**: A comprehensive backend platform by Google that offers real-time database, authentication, storage, and more.
* **MongoDB Stitch**: A serverless platform for MongoDB that provides data synchronization and backend services.
* **SQLite**: A lightweight and embedded database engine that is widely used in mobile app development.

**Modelling and System Design**

* StarUML

This is an open source software used for building usecase diagrams and modelling of the system

**Systems Requirements Specification (SRS) Document.**

**1. Introduction**

This document outlines the System Requirements Specification (SRS) for a mobile application designed to provide drivers and stakeholders with access to road sign information and real-time road state updates through notification.

**1.1. Purpose**

The application aims to improve driver awareness and safety by offering:

* A comprehensive database of road sign definitions and meanings.
* Real-time updates on traffic conditions, accidents, and weather hazards.
* Personalized notification preferences based on user needs.
* Seamless integration with popular navigation apps.

**1.2. Document Conventions**

* Bold text highlights important requirements or user stories.

**1.3. References**

* Google maps
* GPS
* Ministry of Transport
* Team Investigation

**2. Overall Description**

**2.1. Product Perspective**

The target users are drivers and stakeholders who want to:

* Enhance their understanding of road signs.
* Stay informed about current road conditions during their journeys.
* Receive timely alerts about potential hazards or delays.

**2.2. Product Functions**

The application will provide the following functionalities:

* **Road Sign Information:**
  + Searchable database of road signs commonly found on major highways.
  + Clear definitions and explanations for each road sign's meaning and significance.
  + Visual representations (icons or images) of each road sign for easy identification.
* **Real-time Updates:**
  + Integration with real-time data sources (traffic cameras, weather sensors, crowdsourced reports).
  + Display of traffic congestion information on a map, with severity levels and estimated delays.
  + Alerts about accidents, road closures, and weather hazards relevant to the user's location or planned route.
  + User-defined filtering options for alerts based on severity, type, and location.
* **Navigation Integration:**
  + Seamless integration with popular navigation apps and mapping platforms.
  + Sharing of road sign information and real-time road state data with navigation apps.
  + Ability to access this information directly within the user's preferred navigation app interface.
* **User Customization:**
  + Setting notification preferences for road signs, road conditions, and specific geographical areas.
  + Choosing the type of alerts received (visual, audio, text message).
  + Filtering information displayed on the app's main interface based on user preferences.
* **Search Functionality:**
  + Search function for finding specific road signs within the database by name, symbol, or keyword description.

**2.3. General Constraints**

* This application would be compatible with major smartphone and tablet operating systems (e.g., Android, iOS).
* The user interface should be intuitive and user-friendly for drivers of all ages and technical skill levels.
* The application should prioritize user privacy and collect minimal user data. Any data collected should be anonymized and used solely for improving the app's functionality.
* The application should be optimized for battery efficiency to minimize drain on mobile devices.

**3. Specific Requirements**

**3.1. Functional Requirements**

**3.1.1. User-friendly interface**

The requirement for a user-friendly interface is clear, emphasizing an intuitive and easy-to-navigate design. It covers key aspects for an optimal user experience, including intuitiveness, ease of navigation, and effortless access to information. It aligns with the objective of enhancing the user experience and remains consistent with this goal. Feasible within mobile app development, it can be achieved through design principles and best practices, user research, testing, and leveraging existing expertise. Overall, the requirement is well-defined, achievable, and sets a clear direction for interface design to ensure a positive user experience for drivers

**3.1.2. Real-time traffic updates**

The requirement for real-time traffic updates is clear, emphasizing the need for an intuitive interface. It covers the essential elements of delivering timely and accurate information, including congestion updates, accident notifications, and road closure alerts. The requirement is consistent with the objective of keeping drivers informed and is feasible through the utilization of various data sources and technology advancements. Overall, it is a well-defined and achievable requirement that ensures drivers can effortlessly access real-time traffic updates for informed decision-making and efficient navigation

**3.1.3 Route Optimization**

The requirement for route optimization is clear, emphasizing an intuitive interface. It covers the essential aspect of providing drivers with efficient routes, considering factors like traffic conditions and alternative routes. The requirement is consistent with the objective of enhancing driving efficiency and is feasible through the use of real-time data and GPS technology. Overall, it is a well-defined and achievable requirement that ensures drivers can effortlessly access optimized routes for a convenient and time-saving driving experience.

3.1.4 **Checkpoint Notifications**

The requirement for checkpoint notifications is clear, emphasizing the need for an intuitive interface. It covers the essential aspect of notifying drivers about upcoming checkpoints, enabling them to prepare and comply with requirements. The requirement is consistent with the objective of keeping drivers informed and can be feasibly implemented using data sources and technology advancements. Overall, it is a well-defined and achievable requirement that ensures drivers can effortlessly access checkpoint notifications, contributing to a seamless and informed driving experience.

4. External Interface Requirement

4.1 User Interfaces

* Intuitive and User-friendly Design: The UI should be clean, uncluttered, and easy to navigate for drivers of all ages and technical skill levels. This means using clear menus, recognizable icons, and large, easy-to-read fonts.
* Minimal Distraction While Driving: The UI should be designed to minimize distraction while driving. Information should be presented concisely and visually, requiring minimal time for users to glance at the screen.
* Customization Options: Offer user customization options for the UI. This could include:
  + Map View: Allow users to choose the level of detail displayed on the map and potentially switch between map views (e.g., standard, traffic-focused).
  + Notification Preferences: Users should be able to set notification preferences for different types of road signs, road conditions (accidents, closures), and geographical areas. They should also be able to choose the format of notifications (visual, audio, text message).
  + Information Filtering: Users should be able to filter information displayed on the main interface based on their preferences. This could involve filtering by road sign category, severity of road conditions, or specific geographical areas.

4.2 Software Interfaces

1. Programming Language:

* The choice of programming language depends on the target platforms (Android, iOS, or both) and desired functionalities. Here are some options:
  + Native Development:
    - Android: Java is our primary choice for native Android app development.
    - iOS: Swift is the preferred language for native iOS app development.
  + Cross-platform Development: Frameworks like React Native development using languages like JavaScript for apps that can run on both Android and iOS.

2. Integrated Development Environment (IDE):

* The choice of IDE is flexible and depends on developer preference. Popular options include:
  + Android Studio: The official IDE from Google for Android app development (supports Java and Kotlin).
  + Visual Studio Code: A versatile code editor with extensive plugin support, making it suitable for various languages and frameworks, including React Native and Flutter.

3. Design Tools:

* UI/UX design plays a crucial role in this app. Popular design tools to consider include:
  + Figma: A web-based design tool for creating user interfaces and prototypes.
  + Star UML:  A popular design tool for creating user interfaces, especially for mobile apps

4. External APIs and Data Sources:

* The app relies on external APIs to connect with real-time data sources and potentially integrate with navigation apps. These APIs will have their own documentation and specifications that need to be followed during development.

5. Version Control System:

* Using a version control system like Git is crucial for managing code changes, collaboration among developers, and easy rollback if needed. Popular options include GitLab and GitHub.

4.3 Hardware Interfaces

1. User Devices

* Smartphones and Tablets: These are the primary hardware platforms where the app runs. The app interacts with the device's functionalities like:
  + Operating System (OS): The app needs to be compatible with the target OS, most likely Android or iOS.
  + GPS: The app uses the device's GPS hardware to determine the user's location, allowing it to display relevant road sign information and real-time road state data.
  + Internet Connectivity: The app relies on internet connectivity to access real-time data sources (traffic cameras, weather sensors) and potentially download updates.
  + Display: The app utilizes the device's display to present road sign information, real-time road state data, and notifications to the user.
  + Speaker/Audio Output: The app might use the device's speaker to deliver audio notifications about road conditions or alerts.

2. External Data Sources (Indirect Hardware Involvement):

* Traffic Cameras: While the app doesn't directly control traffic cameras, it interacts with their data through APIs. Traffic cameras are the hardware that captures real-time traffic flow information.
* Weather Sensors: Similar to traffic cameras, the app retrieves weather data (temperature, precipitation, etc.) through APIs provided by weather monitoring networks. Weather sensors are the hardware that collects this environmental data.

5 Quality Attributes

5.1 Non-Functional Requirement

* **Performance and Responsiveness**

The requirement for performance and responsiveness is clear, emphasizing the need for a fast and responsive application. It covers the essential aspect of delivering a smooth user experience, including factors like fast loading times and the ability to handle concurrent users. The requirement is consistent with the objective of providing optimal performance and is feasible through technological advancements and scalable infrastructure. Overall, it is a well-defined and achievable requirement that ensures users can seamlessly interact with the application, resulting in a satisfying and reliable user experience.

* **Security and Privacy**

The requirement for security and privacy is clear, emphasizing the need for robust security measures to protect users' personal information and ensure data confidentiality. It covers essential aspects such as encryption, secure authentication, and adherence to privacy regulations. The requirement is consistent with the objective of safeguarding data and maintaining privacy, and it is feasible through the use of established security protocols and best practices. Overall, it is a well-defined and achievable requirement that ensures users' sensitive data is safeguarded, privacy is maintained, and trust is built among users regarding data security and privacy.

* **Audio Notifications**

The requirement for audio notification to enable hands-free interactions is clearly defined, emphasizing the need for a feature that allows users to interact with the application without using their hands. It covers the provision of audio notifications and supports voice commands or other hands-free methods. The requirement is consistent with the objective of providing a convenient and accessible user experience and is feasible through advancements in voice recognition and text-to-speech technologies. Overall, it is a well-defined and achievable requirement that enhances accessibility and user-friendliness by enabling hands-free interactions

* **Multi-language feature**

The requirement for a multi-language feature is clearly defined, highlighting the need for users to access the application in their preferred language. It covers language selection settings, translation capabilities, and localized content, ensuring a complete multi-language experience. The requirement is consistent with the objective of inclusivity and user-friendliness, and it is feasible through localization technologies and existing language resources. Overall, it is a well-defined and achievable requirement that enhances user experience by providing language options that cater to diverse language preferences.

Functional Decomposition and Work Breakdown.

A functional decomposition is the process of breaking down a complex problem, system or structure into a simpler and more understandable parts. In software engineering, functional decomposition helps creates a detailed visual representation of a system’s functionality known as the work breakdown structure.

Road sign, State and Notification app

Real-time Update

Road sign Information

Registration

Real-time data (Traffic camera)

Database

Registration form

Design maps to display traffic congestion

UI

Confirm Email

Functional Decomposition

Conclusion

In conclusion, the requirement analysis for a road sign recognition mobile application is essential for its successful development and deployment. Through meticulous examination of user needs, technological capabilities, regulatory requirements, and design constraints, a comprehensive understanding of the project scope can be achieved. This process enables the formulation of clear objectives, feature specifications, and system functionalities necessary for building an effective and user-friendly application. By conducting thorough requirement analysis, developers can lay a solid foundation for the development process, ensuring that the final product meets user expectations, adheres to safety standards, and delivers a seamless experience for drivers.

References

1. <https://www.brightwork.com/blog/project-requirements>
2. <https://www.geeksforgeeks.org/10-most-popular-requirement-gathering-tools-for-2024/>
3. <https://www.jamasoftware.com/requirements-management-guide/requirements-gathering-and-management-processes/what-is-requirements-gathering>
4. <https://www.altexsoft.com/blog/functional-requirements/>
5. <https://www.mindk.com/blog/requirements-gathering-process/>
6. <https://krazytech.com/projects/sample-software-requirements-specificationsrs-report-airline-database>