|  |  |  |
| --- | --- | --- |
| **P.O BOX 63 Buea** |  | **REPUBLIC OF CAMEROON** |
| **Buea, Cameroon** | **PEACE-WORK-FATHERLAND** |
| **Email:** info@ubuea.cm | **MINISTRY OF HIGHER**  **EDUCATION** |
|  |  |

**FACULTY OF ENGINEERING AND TECHNOLOGY**

**Department Of Computer Engineering**

Course Code: **CEF 440 – INTERNET PROGRAMMING(J2EE) AND MOBILE PROGRAMMING**

Course Instructor**: Dr Nkemeni Valery**

**DESIGN AN IMPLEMENTATION OF A MOBILE APPLICATION FOR CAR FAULT DIAGNOSIS**

**TASK 2: REQUIREMENTS GATHERING**

**GROUP 25**

## **2.1. Introduction**

This report outlines the requirements gathered for the development of a mobile application that uses Artificial Intelligence to help car owners identify vehicle faults through camera-based recognition of dashboard warning lights and analysis of engine sounds. The app aims to provide real-time diagnostics, repair suggestions, and educational support to users without requiring external hardware.

## **2.2 Stakeholder Identification**

Identifying stakeholders is crucial for understanding who will use, develop, or be impacted by the app.

|  |  |  |
| --- | --- | --- |
| **Stakeholder** | **Role** | **Responsibility** |
| Product Owner | Client / Sponsor | Defines app vision, features, and goals |
| Software Developer | Team | Implements the system frontend and backend |
| AI/ML Specialist | Team | Handles dashboard image and sound detection models |
| End Users | Car owners / drivers | Use the app to scan and monitor vehicle health |
| UI/UX Designer | Team member | Designs intuitive user interfaces |
| Tester | QA role or Developer | Ensures app meets functional and non-functional specs |

## **2.3 Requirement Gathering Techniques**

To ensure complete and validated requirements, we used a combination of techniques:

|  |  |  |
| --- | --- | --- |
| **Technique** | **Description** | **Application in Project** |
| Interviews | Direct discussions with car users & mechanics | Asked what features they expect in a car diagnosis app |
| Brainstorming | Team session | Identified core features: OBD scanning, sound detection, dashboard scanning |
| Surveys | Google Form survey to car owners | Gathered info on common user issues and expectations |
| Competitive Analysis | Studied apps like Torque, Car Scanner | Identified key features and usability gaps to improve |
| Observation | Watched users scan OBD with existing apps | Noted pain points in Bluetooth setup, scan results |
| Research & Reverse Engineering | Explored how existing apps work internally | Helped define backend/API and BLE data expectations |

## **2.4 Data Gathering**

The following raw data points were collected:

- Top 5 most common car issues (from interviews)

- Devices used for OBD (Bluetooth vs Wi-Fi)

- User expectations from scan results

- Frequency of using scan apps

- Preference for visual/auditory diagnostics

- Interest in real-time alerts

## **2.5 Data Cleaning**

Raw data from surveys and interviews were cleaned to remove irrelevant, duplicated, or unclear inputs.

|  |  |  |
| --- | --- | --- |
| **Raw Input** | **Cleaned Format** | **Notes** |
| “I just want something that shows errors fast” | Real-time fault code display (DTC) | Converted into a clear functional requirement |
| “My OBD never connects” | OBD Bluetooth connectivity must be reliable | Highlighted as a pain point |
| “I use Car Scanner but hate the ads” | Preference for ad-free or freemium model | Design consideration |
| Multiple conflicting opinions on dashboard UI | Consolidated through consensus | Prioritized usability and clarity |

## **2.6 User Reluctance Assessment**

We anticipated and evaluated potential user reluctance issues:

|  |  |  |
| --- | --- | --- |
| **Type** | **Concern** | **Mitigation Strategy** |
| Trust | Users may distrust an app that touches car diagnostics | Include disclaimers, offline scan logs, and transparent permissions |
| Complexity | Users may find setup or scanning confusing | Simplified UI and tooltips for each step |
| Performance | Fear of app lag during scans or ML usage | Optimize ML models (TensorFlow Lite), use local processing, and show progress indicators |
| Privacy | Concerns about sound/camera access | Request permissions only when necessary, explain usage clearly |
| Adoption | Users may prefer apps they already know | Add unique features (e.g., sound detection) and real-time alerts as a competitive edge |

## 2.7 Conclusion

This requirement gathering phase ensured a user-driven, technically feasible, and innovative vision for the app. The use of diverse techniques (interviews, surveys, reverse engineering) helped gather clear, realistic, and implementable requirements that will serve as the backbone for analysis and design in later stages.