**REPUBLIC OF CAMEROON**

Peace-Work-Fatherland

**UNIVERSITY OF BUEA**

P.O. Box 63.

Buea, South West Region

CAMEROON

Tel: (237)33322134/33322690

Fax: (273)33322272



**FACULTY OF ENGINEERING AND TECHNOLOGY**

**DEPARTMENT OF COMPUTER ENGINEERING**

**COURSE**: INTERNET PROGRAMMING AND MOBILE PROGRAMMING. **CEF 440**

**COURSE COORDINATOR**: DR VALERY NKEMENI

**TASK 3:**

**Requirement Analysis**

**GROUP 15 MEMBERS:**

1. ELAINE CLINTON NTONGWE-FE22A198
2. ETUNDI ZAMBO JOSIANE-FE22A212
3. FAI NJI JR BAHTINYUY-FE22A215
4. NEBA NELLY ACHA-FE22A255
5. NGWA NATHAN NINJECK-FE22A268

Second semester Task 3 report, (B.ENG) Degree in Computer Engineering.

MAY

2025

Contents

[**Introduction** 3](#_Toc197250767)

[**1.** **Review and Analysis of Requirements** 3](#_Toc197250768)

[**Completeness:** 3](#_Toc197250769)

[**Clarity:** 3](#_Toc197250770)

[**Technical Feasibility:** 3](#_Toc197250771)

[**Dependency Relationships:** 3](#_Toc197250772)

[**2.** **Identification of Issues** 4](#_Toc197250773)

[**Inconsistencies or Ambiguities:** 4](#_Toc197250774)

[**Missing Information:** 4](#_Toc197250775)

[**3.** **Requirement Prioritization** 4](#_Toc197250776)

[**High Priority(Must have):** 4](#_Toc197250777)

[**Medium Priority(Should have):** 4](#_Toc197250778)

[**Low Priority(could have):** 4](#_Toc197250779)

[**4.** **Requirement Classification** 4](#_Toc197250780)

[**Functional Requirements:** 5](#_Toc197250781)

[**Non-Functional Requirements:** 5](#_Toc197250782)

[**5.** **Software Requirements Specification (SRS)** 5](#_Toc197250783)

[**Introduction** 5](#_Toc197250784)

[**Requirements** 6](#_Toc197250785)

[**Functional Requirement:** 6](#_Toc197250786)

[**Non-Functional Requirements** 7](#_Toc197250787)

[**System interface** 8](#_Toc197250788)

[**Assumptions & constraints** 10](#_Toc197250789)

[**6.** **Requirement Validation** 10](#_Toc197250790)

# **Introduction**

Requirement analysis is an important phase in the system development lifecycle that involves the process of analyzing, validating and documenting software requirements to ensure clarity, feasibility and stakeholder alignment. This approach helps us to identify inconsistencies, classify requirements and prioritize them for development.

# **Review and Analysis of Requirements**

This process involves the examining of the gathered requirements to ensure its clarity, completeness, technical feasibility and dependency relationships.

## **Completeness:**

This ensures that all necessary functional and non-functional aspects are captured.  
The gathered requirements include:

* Real-time attendance recording using facial recognition and geofencing.
* User roles: Student, Instructor, Administrator.
* Face capture and matching using ML.
* Location validation via GPS.
* Mobile-first implementation.

## **Clarity:**

Here, we verify if all requirements are unambiguous and well defined. Most requirements are clearly stated. However, additional specifications are needed regarding:

* Supported device platforms (Android, iOS, both?)
* Authentication methods.
* Handling offline check-ins or GPS inaccuracies.

## **Technical Feasibility:**

This assesses if available technology can implement the requirements. Some examples include;

* Facial recognition: Feasible using mobile ML libraries (e.g., ML Kit, FaceNet, OpenCV).
* Geofencing: GPS-based geofencing is well-supported via APIs (Google Maps, MapKit).
* Data sync & storage: Cloud-based backend (Firebase, AWS) can ensure real-time sync and security.

## **Dependency Relationships:**

This identifies relationships between requirements. Some of these include;

* Geolocation validation must occur before face recognition-based check-in is permitted.
* Instructor reports depend on check-in data stored in the system.
* Course data management depends on admin input.

# **Identification of Issues**

We detected vague, absent and conflicting requirements which may have led to misinterpretation.

## **Inconsistencies or Ambiguities:**

Here we checked for contradictory and vague terms. Such as;

* What happens if a student’s GPS fails or their face is not recognized?
* Is recheck-in possible within a time window?
* Will attendance be per class session or once per day?

## **Missing Information:**

Here we identified gaps, some of which included;

* Specific ML models for facial recognition.
* Expected number of users to estimate scalability.
* Data privacy and security policies.
* Internet availability assumptions.

# **Requirement Prioritization**

It is the ranking of requirements to determine implementation order based on business value, risks and technical constraints.

**High Priority(Must have):** these are critical requirements

* Facial recognition for check-in
* GPS/geofencing check before check-in
* Real-time storage and verification of data
* Instructor and admin dashboards for attendance monitoring

**Medium Priority(Should have):** they are important, but not too urgent

* Historical data view for students
* Filtering tools by (course data and students)

**Low Priority(could have):** they have a low priority

* Notifications or alerts
* Offline caching of attendance
* Admin level management

# **Requirement Classification**

This involves the categorization of requirements into functional (what the system does) and non-functional (how the system performs) requirements.

## **Functional Requirements:**

* Students must check in using facial recognition.
* System must validate GPS location before allowing check-in.
* Instructors must be able to view attendance data in real time.
* Admin must manage users, courses, and permissions.
* The system must store and retrieve attendance history.
* System shall allow filtering attendance records by date, student and course
* System shall store and compare facial data for each student securely
* System shall allow students to view their attendance history

## **Non-Functional Requirements:**

* The check-in process must not exceed 5 seconds.
* The app must be available 99% of the time.
* User data must be securely stored and encrypted.
* The interface must be usable on Android and iOS devices.
* The system must scale to accommodate hundreds of users concurrently.
* All biometric data must be stored securely and comply with the data privacy regulations
* System must maintain at least 95% facial recognition accuracy.

# **Software Requirements Specification (SRS)**

## **Introduction**

The Student Attendance Management System is a software system designed to record and manage the attendance of students in educational institutions. Its purpose is to streamline (making work easier and more efficient) the attendance recording process, automate attendance tracking, and provide accurate attendance reports for teachers, students administrators, and parents. The target audience for the Student Attendance Management System primarily includes educational institutions such as colleges, universities . It is beneficial for teachers, administrators, and parents who need to monitor and manage student attendance efficiently.

Key features of a typical Student Attendance Management System may include:

* **Attendance Tracking:** The system allows teachers to record student attendance easily and accurately using various methods such as biometric scanning, barcode scanning, or manual entry.
* **Automated Attendance Calculation:** The system automatically calculates attendance percentages based on the recorded data, saving time and reducing errors associated with manual calculations.
* **Real-Time Monitoring**: Teachers, administrators, and parents can access real-time attendance information for individual students or entire classes. This feature enables them to identify patterns of attendance and take necessary actions promptly.
* **Reporting and Analytics**: The system generates comprehensive attendance reports and analytics, providing insights into attendance trends, patterns, and overall student attendance performance. These reports can be used for administrative purposes, evaluation, and decision-making. Overall, the Student Attendance Management System aims to streamline attendance management processes, improve accuracy, enhance communication between educational stakeholders.

## **Requirements**

These are the description of a features, functions, behaviors, or characteristics that a software system must possess or meet to satisfy the needs of its stakeholders. Requirements serve as the foundation for designing, developing, and testing software systems.

## **Functional Requirement:**

1. **User Registration:** The system should allow teachers, and student to register and create user accounts.
2. **User Authentication**
   * Students and instructors can securely log in and log out of the application.
3. **Facial Recognition Check-In**
   * The system captures and recognizes a student's face using a built-in camera and machine learning model.
   * Facial biometric data is compared securely to stored records for verification.
4. **Geofencing Validation**
   * The app uses GPS services to define a virtual boundary around the classroom.
   * The system validates that students are physically within the classroom boundary before allowing check-in.
5. **Attendance Check-In Process**
   * Students can check in by scanning their face
   * The check-in process completes within 5 seconds per student.
6. **Real-Time Attendance Monitoring** 
   * Instructors can view live attendance data for each course.
   * Attendance data can be filtered by **course**, **date**, or **student**.
7. **Attendance History** 
   * Students can view their personal attendance history and participation status for each registered course.
8. **Attendance Calculation:** The system should calculate attendance percentages based on recorded data and provide accurate attendance reports.
9. **Data Management**
   * The system securely stores facial biometric data.
   * Attendance records are stored and retrievable for analysis and reporting.
10. **Administrative Features**
    * Instructors/admins can generate attendance reports and statistics.
    * Role-based access control (different permissions for students vs instructors/admins).

## **Non-Functional Requirements**

1. **Performance**
   * The check-in process must complete in **≤ 5 seconds** per student.
   * The system should be able to handle a large number of concurrent users and provide efficient attendance recording and retrieval.
2. **Security**
   * Biometric data and personal data must be securely encrypted during storage and transmission
   * Multi-factor authentication for instructors and admins.
3. **Usability**
   * The app should have an intuitive and user-friendly interface for both students and instructors.
   * Support for both English and French
4. **Availability**
   * System uptime should be **≥ 99.5%** during class hours.
   * App should be available on both Android and iOS platforms.
5. **Scalability**
   * The system should be scalable to accommodate future growth in terms of the number of students, teachers, and courses.
6. **Maintainability**
   * The codebase should follow modular design patterns to facilitate easy updates and bug fixes.
   * System updates should require minimal downtime (≤ 10 minutes).
7. **Reliability**
   * The facial recognition model should have an accuracy of **≥ 95%**.
   * Failover mechanisms should be in place for critical services
8. **Battery and Resource Efficiency**
   * The app should minimize battery and data consumption, especially during geofencing and face recognition operations.
9. **Portability**

* The app should run on a wide range of smartphone hardware

For now, this concludes our SRS document. Further modeling and design of the required diagrams will be done in future works.

## **System interface**

The system interfaces define how different components of the application interact with each other and with external services. For this project, the core system interfaces are categorized into:

* User Interface (UI)
* Hardware Interface
* Software Interface
* Communication Interface

1. User Interface (UI)

The system provides different UIs based on user roles (Student, Instructor, Admin):

* Student Interface
* Login/Signup Screen: Email/password authentication or biometric login
* Face Registration Screen: Guides user to capture and upload a reference photo
* Attendance Check-In Screen: Captures live face image and gets location in real time
* Attendance History View: Displays attendance records filtered by course or date
* Instructor Interface
* Dashboard: Displays list of courses and enrolled students
* Attendance Monitoring: Real-time list of students who checked in per session
* Filtering Tools: Filter by date, student ID, or course code
* Export Feature: Export attendance reports (PDF/Excel)
* Admin Interface (optional)
* Manage users, permissions, and course assignments
* Monitor system status and logs

All interfaces will follow mobile-first design principles, ensuring responsive layouts, accessibility, and intuitive navigation.

1. Hardware Interface

* Camera Module
* Used for capturing student facial data
* Must support minimum 5MP resolution for accurate recognition
* Integrates with the facial recognition module
* GPS Module
* Accesses the device’s location services
* Continuously checks geofence boundary compliance
* Integrates with Google Maps API or other location services

1. Software Interface

* Operating System Interfaces
* Compatible with Android (8.0 and above) and iOS (13 and above)
* Interfaces with mobile OS services such as: Location Services
* Camera and Media APIs
* Permission manager for accessing camera, GPS, storage
* Facial Recognition Library
* Interfaces with TensorFlow Lite, OpenCV, or ML Kit
* Handles face detection, alignment, and embedding comparisons
* Geofencing API
* Uses Google Maps Geofencing API or similar
* Defines virtual boundaries (radius in meters) around classroom location
* Database Interface
* Firebase Fire store, MongoDB Atlas, or PostgreSQL through REST APIs Used to: Store user data (students/instructors), Store facial embedding, Log attendance records and device location
* Authentication Services
* Firebase Auth, OAuth2, or custom JWT-based authentication
* Role-based access control

1. Communication Interface

* API Communication
* RESTful APIs over HTTPS
* Handles:
* Check-in data transmission
* User registration/authentication
* Attendance retrieval and updates
* Secured with TLS encryption

## **Assumptions & constraints**

**Constraints**

Since Design is a requirement , here are constraints to design requirement

1. Platform compatibility: Ensure compatibility with iOS and Android mobile devices.
2. Scalability: Accommodate varying numbers of students and classes.
3. Security and privacy: Implement data encryption and comply with privacy regulations.
4. User-friendly interface: Provide intuitive navigation and clear instructions.
5. Performance: Ensure fast and responsive attendance recording.
6. Integration: Integrate with existing student information systems if required.
7. Maintainability: Design for modularity and easy maintenance.
8. Hardware constraints: The app has to work on mobile devices with front cameras, and the system needs to utilize built-in GPS hardware

**Assumptions**

Here are some assumptions that could be considered

1. Mobile Devices: It is assumed that students and instructors will have access to smartphones or other scanning devices capable of reading QR codes.
2. Internet Connectivity: The system assumes that there will be a reliable internet connection available for real-time data synchronization and communication with the central server.
3. User Authentication: The system assumes that students will have valid login credentials or unique identifiers to authenticate their attendance records.
4. Biometric Consent: Students will willingly provide consent to capture and process their facial biometric data as part of their enrollment in the attendance system.
5. Accurate GPS Coverage: GPS signal availability and accuracy will be sufficient within classroom environments (with minimal signal obstruction
6. Face Recognition Reliability: Environmental conditions (lighting, background, face visibility) will be sufficiently good to allow reliable face recognition
7. Data Privacy: The system assumes that appropriate measures will be in place to protect the privacy and security of student attendance data, adhering to relevant regulations and policies

# **Requirement Validation**

Ensuring requirements accurately reflect stakeholder needs through reviews, prototypes and feedback sessions.

**Stakeholder Engagement:**  
- Conducted surveys/interviews with students, instructors, and admins.  
- Received feedback on desired app features, GPS reliability, and UI preferences.

- Build mockups to visualize functionality.

**Validation Outcome:**  
- Confirmed user acceptance of facial recognition.  
- Raised concern over battery and GPS usage—optimization noted as a future consideration.  
- Agreed on importance of real-time feedback after check-in.