PRESIDENCY UNIVERSITY
Private University Estd. in Karnataka State by Act No. 41 of 2013
Itgalpura, Rajankunte, Yelahanka, Bengaluru – 560064

# GEOSYNC

## Development of Geolocation Based Attendance Tracking System

**A PROJECT REPORT**

*Submitted by*

SHAIK RIDA - 20221CCS0117

DINESH T - 20221CCS0133

PREM SINGH B RAJPUT – 20221CCS0142

*Under the guidance of,*

**Ms. STERLIN MINISH T N**

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE AND ENGINEERING (CYBER SECURITY)**

**PRESIDENCY UNIVERSITY**

**BENGALURU**

**DECEMBER 2025**

# PRESIDENCY SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

## BONAFIDE CERTIFICATE

Certified that this report "**GEOSYNC - Development of Geolocation Based Attendance Tracking System**" is a bonafide work of "**Shaik Rida (20221CCS0117), Dinesh T (20221CCS0133), Prem Singh B Rajput (20221CCS0142)**", who have successfully carried out the project work and submitted the report for partial fulfilment of the requirements for the award of the degree of BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE ENGINEERING, CYBER SECURITY during 2025-26.

4/12/2025

**Ms. Sterlin Minish T N**
Project Guide
PSCS
Presidency University

4/12

**Dr. Sharmasth Vali**
Program Project Coordinator
PSCS
Presidency University

**Dr. Sampath A K**
**Dr. Geetha A**
School Project Coordinators
PSCS
Presidency University

**Dr. Anandaraj S P**
Head of the Department
PSCS
Presidency University

**Dr. Shakkeera L**
Associate Dean
PSCS
Presidency University

**Dr. Duraipandian N**
Dean
PSCS & PSIS
Presidency University

Examiners

| Sl. No. | Name | Signature | Date |
|---|---|---|---|
| 1 | Dr. Ranjith | | 4/12/25 |
| 2 | Dr. Geetha Arjuran. | | 4/12/25 |

# PRESIDENCY UNIVERSITY

## PRESIDENCY SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

## DECLARATION

We the students of final year B.Tech in COMPUTER SCIENCE ENGINEERING, CYBER SECURITY at Presidency University, Bengaluru, named **Prem Singh B Rajput, Dinesh T, Shaik Rida**, hereby declare that the project work titled **"GEOSYNC - Development of Geolocation Based Attendance Tracking System"** has been independently carried out by us and submitted in partial fulfilment for the award of the degree of B.Tech in COMPUTER SCIENCE ENGINEERING (CYBER SECURITY) during the academic year of 2025-26. Further, the matter embodied in the project has not been submitted previously by anybody for the award of any Degree or Diploma to any other institution.

| | | |
|---|---|---|
| Shaik Rida | 20221CCS0117 | Rida |
| Dinesh T | 20221CCS0133 | Dinesh |
| Prem Singh B Rajput | 20221CCS0142 | |

PLACE: BENGALURU

DATE: 03 December 2025

# ACKNOWLEDGEMENT

For completing this project work, we have received the support and the guidance from many people whom we would like to mention with deep sense of gratitude and indebtedness. We extend our gratitude to our beloved **Chancellor, Pro-Vice Chancellor, and Registrar** for their support and encouragement in completion of the project.

We would like to sincerely thank our internal guide **Ms. Sterlin Minish T N, Assistant Professor**, Presidency School of Computer Science and Engineering, Presidency University, for her moral support, motivation, timely guidance and encouragement provided to us during the period of our project work.

We are also thankful to **Dr. Anandaraj, Professor, Head of the Department, Presidency School of Computer Science and Engineering** Presidency University, for his mentorship and encouragement.

We express our cordial thanks to **Dr. Duraipandian N**, Dean PSCS & PSIS, **Dr. Shakkeera L**, Associate Dean, Presidency School of computer Science and Engineering and the Management of Presidency University for providing the required facilities and intellectually stimulating environment that aided in the completion of our project work.

We are grateful to **Dr. Sampath A K**, and **Dr. Geetha A**, **PSCS** Project Coordinators, **Dr. Sharmasth Vali, Program Project Coordinator**, Presidency School of Computer Science and Engineering, or facilitating problem statements, coordinating reviews, monitoring progress, and providing their valuable support and guidance.

We are also grateful to Teaching and Non-Teaching staff of Presidency School of Computer Science and Engineering and also staff from other departments who have extended their valuable help and cooperation.

SHAIK RIDA

DINESH T

PREM SINGH B RAJPUT

# Abstract

Accurately recording attendance remains a major challenge in schools and organizations that still rely on manual procedures to record attendance like paper-based logs, signing ID, and teacher roll calls. These activities are time-consuming, prone to errors, and can be manipulated by fake attendance and proxy punching. Some research results using RFID, GPS, Bluetooth, or NFC have shown that there have been some improvements, however, they have often lacked real-time spatial validation, automated geofencing, or scalable software-based solutions independent of external hardware. These shortcomings highlight the necessity for a modern, accurate, software-driven attendance automation system that is time-efficient and less prone to error. To that end, the project GEOSYNC, proposes a mobile phone-based location attendance system that uses **GPS, geofencing, PostGIS spatial queries,** and **a hybrid model** combining automatic location-triggered with manual fallback for special cases. React Native is used for the mobile client, **Node.js/Express.js** for the backend, and **PostgreSQL** with **PostGIS** for geospatial processing. The implementation of Geofencing is carried out using the PostGIS **ST_DWithin** function in order to check the presence of a user within the predefined boundary, with a default radius of **200 meters** to compensate for the **GPS signal drift** in semi-urban and indoor locations. A transition model based on a state machine is used to eliminate duplicate **ENTER/EXIT** events, and offline caching allows attendance to be continuous during a connection drop. Multi-tenant separation is realized through token-embedded company_id attributes so that data isolation is maintained across organizations.Experimental results show that GEOSYNC can detect the geofence correctly with a probability of about 98% over different controlled environments, with average API response times being **approximately 200 ms** and offline sync **reliability of about 99%.**The proposed solution versus the RFID-based and GPS-only systems of the past, which either necessitate specialized hardware or lack spatial visualization, presents a scalable, completely software-driven solution that increases the accuracy, lessens the administrative overheads, and at the same time reduces attendance fraud. Thus, **GEOSYNC** to large extent transforms the management of attendance into a modernized form by integrating geospatial intelligence, mobile computing, and automated logging to provide a dependable, secure, and hardware-free attendance tracking platform. The system's firm accuracy metrics and stable real-time functionality are proof of its appropriateness not only for academic institutions but also organizations that want to digitally shift towards a location-aware attendance ecosystem.

# Table of Contents


| Sl. No. | Title | Page No. |
|---|---|---|
| | Declaration | I |
| | Acknowledgement | II |
| | Abstract | III |
| | List of Figures | VI |
| | List of Tables | VII |
| 1. | Introduction | |
| | 1.1 Background | |
| | 1.2 Statistics of project | |
| | 1.3 Prior existing technologies | 1 |
| | 1.4 Proposed approach | |
| | 1.5 Objectives | |
| | 1.6 SDGs | |
| | 1.7 Overview of project report | |
| 2. | Literature review | 13 |
| 3. | Methodology | 14 |
| 4. | Project management | |
| | 4.1 Project timeline | 22 |
| | 4.2 Risk analysis | |
| | 4.3 Project budget | |
| 5. | Analysis and Design | |
| | 5.1 Requirements | |
| | 5.2 Block Diagram | |
| | 5.3 System Flow Chart | |
| | 5.4 Choosing devices | 23 |
| | 5.5 Designing units | |
| | 5.6 Standards | |
| | 5.7 Mapping with IoTWF reference model layers | |
| | 5.8 Domain model specification | |
| | 5.9 Communication model | |

5.10 IoT deployment level
5.11 Functional view
5.12 Mapping IoT deployment level with functional view
5.13 Operational view
5.14 Other Design
6. Hardware, Software and Simulation
6.1 Hardware
6.2 Software development tools 51
6.3 Software code
6.4 Simulation
7. Evaluation and Results
7.1 Test points
7.2 Test plan 57
7.3 Test result
7.4 Insights
8. Social, Legal, Ethical, Sustainability and Safety Aspects
8.1 Social aspects
8.2 Legal aspects
8.3 Ethical aspects 64
8.4 Sustainability aspects
8.5 Safety aspects
9. Conclusion 71
References 74
Base Paper 74
Appendix 76

# List of Figures


| Figure ID | Figure Caption | Page No. |
|---|---|---|
| Fig 1.1 | Sustainable development goals | 10 |
| Fig 3.1 | Proposed geofence accuracy testing methodology | 12 |
| Fig 4.1 | Project implementation timeline | 17 |
| Fig 5.1 | Functional block diagram | 21 |
| Fig 5.2 | Location capture and offline handling | 22 |
| Fig 5.3 | Initialization and authentication | 22 |
| Fig 5.4 | Server validation and geofence handling | 23 |
| Fig 5.5 | The iot world forum reference model | 30 |
| Fig 5.6 | Domain model | 32 |
| Fig 5.7 | Request - response model | 33 |
| Fig 5.8 | Iot deployment level 3 overview | 34 |
| Fig 5.9 | Functional view for home automation | 35 |
| Fig 5.10 | Mapping deployment level with functional blocks | 36 |
| Fig 5.11 | Operational view | 37 |
| Fig 7.1 | Accuracy of geofence detection across multiple test environments | 45 |
| Fig 7.2 | Api response time under baseline and concurrent load conditions | 45 |
| Fig 7.3 | Results under controlled test-cases | 47 |
| Fig 7.4 | Summary of functional testing results | 47 |
| Fig A.1 | Employee Dashboard | 75 |

| | | |
|---|---|---|
| Fig A.2 | Admin Dashboard | 75 |
| Fig A.3 | Creating Dashboard | 76 |
| Fig A.4 | Geofence | 76 |
| Fig A.5 | Manage Geofence | 77 |
| Fig A.6 | Employee Management | 77 |

# List of Tables


| Table no. | Table Caption | Page No. |
|---|---|---|
| Table 3.1 | Functional modules | 16 |
| Table 3.2 | System architecture components | 20 |
| Table 3.3 | Comparison with alternative systems | 21 |
| Table 4.1 | Project planning timeline | 22 |
| Table 4.2 | Risk analysis table | 23 |
| Table 5.1 | Requirements | 27 |
| Table 5.2 | Comparing features of different processors | 30 |
| Table 5.3 | Summary of unit design | 34 |
| Table 5.4 | Mapping project layers with iotwf-rm | 37 |
| Table 5.5 | Description of domain model | 38 |
| Table 5.6 | Mapping iot deployment level to functional groups | 42 |
| Table 7.1 | Test plan | 50 |
| Table 7.2 | Api response time under baseline | 53 |
| Table 9.1 | Comparison between past work and GEOSYNC | 63 |

# Abbreviations

| Abbreviation | Full Form / Meaning |
|---|---|
| IJEIT | International Journal of Engineering and Innovative Technology |
| GPS | Global Positioning System |
| RFID | Radio Frequency Identification |
| AI | Artificial Intelligence |
| JWT | JSON Web Token |
| REST | Representational State Transfer |
| API | Application Programming Interface |
| RBAC | Role-Based Access Control |
| CRUD | Create, Read, Update, Delete |
| HTTPS | Hypertext Transfer Protocol Secure |
| UI | User Interface |
| BLE | Bluetooth Low Energy |
| SQL | Structured Query Language |
| QR | Quick Response |
| RMSE | Root Mean Square Error |
| CPCT / Conf. | Conference (context: research paper citations) |
| WGS 84 | World Geodetic System 1984 (global standard ellipsoid model for GPS) |
| GIST | Generalized Search Tree (PostgreSQL index type) |
| ms | Milliseconds |
| Expo | (Tool name; not an abbreviation, listed to avoid confusion) |
| AI-powered | Artificial Intelligence powered (context reference) |

Chapter 1

# Introduction

## 1.1 Background

Managing attendance is at the center of the daily operations in school systems, and large corporations that have decided to go digital. The direct impact is academic progress, the responsibility of the workforce, and organizational effectiveness. Conventional methods (e.g., manual registers, paper-based logs, or biometric scanners) are still popular but have problems with inaccuracies, human errors, proxy attendance, and a considerable amount of work for the administration. Studies from the past have mentioned that recording attendance manually takes up the time that should be used for teaching and that it can be easily manipulated and that data can be lost which is why these systems have to be automated and location-aware [1], [5], [6].

- The upward trend of mobile computing together with the general availability of GPS-enabled smartphones have made it possible for geofencing-based attendance systems to take the lead over traditional methods that require heavy hardware, thus providing more versatile and user-friendly solutions.
- Studies suggest that geofencing limits the recording of attendance to specific geographical areas, thus eliminating fraud in the form of buddy punching and off-site check-ins [1], [2], [6], [7].
- Present-day setups which use only GPS or combine GPS with RFID, QR codes, or face recognition still have drawbacks, for instance, they are reliant on specially designed hardware, the GPS signal in crowded areas is not always accurate, and there is no possibility for the real-time geospatial processing [3], [4], [5], [20].
- Research results showed that the use of geofencing systems helps the company to save money on the operational costs, get rid of the manual attendance-taking process, and make a significant improvement in the accuracy of the tracking of valid on-site presence [2], [7].
- The development of geospatial software technologies, especially **PostGIS**, has made it possible to perform powerful spatial querying and distance computations, with functions such as *ST_DWithin* giving exact radius-based geofence checks [8], [17].

- When combined with **PostgreSQL**, PostGIS forms a scalable and high-performance spatial database infrastructure suitable for real-time attendance applications [18].
- On the client side, **React Native** supports building cross-platform mobile apps with high responsiveness and efficient access to device location data [9].
- Backend frameworks such as **Express.js** provide lightweight, scalable API services for location verification, authentication, and processing server-side logic [10].
- Modern attendance systems integrate secure authentication mechanisms using **JWT tokens** [14], encrypted password storage via **bcryptjs** [16], and multi-tenant architectures to support multiple organizations simultaneously [21].
- Offline caching and synchronization techniques ensure uninterrupted attendance logging during connectivity issues, an approach emphasized in IEEE-access literature on mobile reliability [15].
- Recent innovations incorporate AI-driven validation such as facial recognition and behavioural analytics, strengthening identity verification and increasing workflow automation [11], [12].
- Collectively, these advancements address long-standing limitations in traditional attendance methods by offering enhanced accuracy, reliability, scalability, and real-time spatial intelligence.
- Modern mobile-first, geofencing-enabled attendance platforms build on these technologies to meet the needs of academic institutions and organizations seeking secure, efficient, and automated attendance solutions.

## 1.2 Statistics

- Accurate attendance monitoring is gaining more and more weight as a need, in large-scale institutions and offices where populations are in the thousands, and the usage of manually-operated systems leads to the heavy workload of the administration and a high error rate.
- Study reveals that manual attendance methods may cause errors of 12–18% due to mistakes made by people, inconsistent records, and proxy attendance which, in turn, influence evaluation and accountability [1], [5].
- Studies surface that most of the schools and organizations in South Asia (70%+) are dependent on paper registers or low-level RFID systems that are not only easy to be manipulated but also less capable to prove the attendance in real-time [4], [6].

- Local workforce surveys point to the fact that inadequate attendance monitoring is the main reason for 20–25% of operational inefficiency in medium-size businesses thus, the urgency of automated solutions becomes apparent [2], [7].
- The smartphone adoption rate in India is more than 80% for students and working professionals, which is a great facility for GPS- and geofencing-based attendance methods to be utilized.
- Present-day GPS can provide between 5 and 15 meters of accuracy in open areas which is enough for attendance purposes, whereas cities may suffer from signal blockages and that's why geospatial buffering techniques are needed to be applied [20].
- Such solutions as PostGIS is capable to upgrade the dependability of location services via range-oriented validation (e.g. ST_DWithin) which allows compensation for drift as well as multipath effect in difficult spots [8], [17], [20].
- The 2024 study revealed that GPS-based automated systems resulted in the reduction of proxy attendance by 37% and a 52% increment in the consistency of the records, thus, it shows that large-scale performance improvements were achieved [1], [2], [3].
- The implementation of location-based attendance in schools resulted in the saving of 10–15 minutes of every lesson, which gave rise to a large total amount of teaching time [5], [12].
- Presently, remote and hybrid work models contribute to nearly 28% of total professional activities, which led to the growth of attendance issues that cannot be solved by on-site hardware and still need solutions that are independent of the physical presence [12].
- During the implementation of AI/Augmented reality technologies in the development of attendance systems along with geolocation feature, they are being widely used for transparency enhancement, administrative labor reduction, and authentic presence assurance at different work settings [11], [12].
- The majority of these aggregated data emphasize a definite need for precise and automated attendance solutions that are easily scalable - thus, the geofencing-based attendance platform adoption is highly recommended.

## 1.3 Prior existing technologies

Before the emergence of modern geofencing-based attendance systems, several technologies were widely adopted to automate or partially digitize attendance processes. These earlier solutions improved efficiency compared to manual registers but still suffered from limitations

such as hardware dependency, lack of real-time verification, vulnerability to proxy attempts, and limited scalability. The evolution of these technologies forms the foundation upon which advanced GPS and geolocation systems—like the one proposed in this project—have been developed.

### 1.3.1 RFID-Based Attendance Systems

Radio Frequency Identification (RFID) solutions were among the earliest automated attendance technologies. In order to confirm their attendance, users had to scan their RFID cards at certain points. The systems enabled quicker recording and lessened the cases of errors made by the staff; however, they were very strongly exposed to "buddy punching" since a person could have another person's RFID card. Moreover, the RFID readers and cards brought some additional costs for the hardware and some necessities for the maintenance, so they were not good enough for big areas or places outside the buildings [5].

### 1.3.2 GPS-Based Attendance Systems

Attendance systems with GPS capability made better use of the technology already in place with RFID by doing away with the requirement of giving out physical cards and also by making it possible to verify presence basing on location. Most of these systems recorded the GPS coordinates of a user at the time of checking in or out. Research findings indicate that GPS-based systems have improved accuracy and have made it possible to have attendance from a distance; nevertheless, GPS on its own was not enough to totally stop fraudulent activities since users were able to fake locations or they could be marking their attendance just a little bit outside the area that had been designated for them because of the natural inaccuracy of GPS in a densely built-up urban area [4], [6].

### 1.3.3 Geofencing Attendance Technologies

Initial geofencing technologies went beyond GPS by setting up virtual geographic borders. These operations verified presence only when users were actually within these areas, thus cutting down on fake check-ins. Studies show that the use of geofencing radically increases the genuineness of verification and gets rid of the practice of marking attendance from off-site. In spite of this upgrade, they still relied on basic radius functions and did not have sophisticated spatial processing; hence, they were vulnerable to GPS drift and could be obstructed by the environment [1], [3], [7].

### 1.3.4 QR Code–Based Attendance Systems

Users of QR-based attendance systems had to scan either a static or a dynamically generated code at their location. Such systems minimized manual entries and facilitated the maintenance of digital logs. However, the problem of proxy attendance still existed because users could send pictures of QR codes to others. Besides, the systems needed the physical presence of a QR code in a classroom or an office, thus, they were not appropriate for open or non-controlled environments [13].

### 1.3.5 Biometric Attendance Systems

Biometric measures like fingerprint and facial recognition gave more robust identity confirmation and were extensively used in institutions and organizations. Even though biometrics solved the problem of proxy attendance, these setups needed specially designed hardware locations and could get affected by cleanliness issues, environmental factors, and maintenance. Besides, facial recognition systems, being more adaptable, generally needed a reliable internet connection or a strong on-device processor [11].

### 1.3.6 AI-Enhanced Attendance Systems

Recent innovations defined AI-powered architectures that merge geolocation, facial recognition, and behavioural analytics. Such mechanisms offer two-factor authentication—location and identity—thus, the chance of fraud is lowered drastically. Nevertheless, majority of the first generation AI-integrated systems were heavy on resources, needed computation from the cloud, and gave rise to issues of data privacy and storage [11], [12].

### 1.3.7 Authentication and Backend Technologies

Traditional systems were heavily dependent on very simple username–password authentication that lacked the latest security features. Modern systems have made use of various security measures such as secure tokenization (JWT), encrypted password handling (bcryptjs), and well-defined multi-tenant architectures to ensure that attendance management can be carried out in a safe and efficient manner, and can be scaled to different organizations [14], [16], [21].

## 1.4 Proposed approach

### 1.4.1 Aim of the Project

This project is geared towards the creation and implementation of a location-based system for managing the attendance of users that utilises GPS, geofencing, and spatial database technologies to verify the user's presence in a particular area and to make the recording of

attendance automatic. It intends to do away with the practice of proxy attendance, cut down on the manual work, enhance the precision, and come up with a solution that is easily accessible via mobile and can be used by academic institutions and organizations of any size in the future [1], [2], [3].

### 1.4.2 Motivation

Old-style attendance tracking methods such as manual signing, RFID scans, and straightforward GPS logging can be manipulated, need more hardware, and make the organization's inefficiency level rise. The reports reveal that institutions which depend on manual or hardware methods have been facing serious problems like buddy punching, off-site check-ins, and inconsistent data for a long time [5], [6], [7].

Considering that the use of smartphones is getting more and more widespread and GPS is accessible to everyone, systems based on geofencing have turned out to be the most reliable way to ensure that attendance is only done when the user is actually present at the given location [1], [2], [12].

Moreover, advanced geospatial instruments like PostGIS spatial queries allow very accurate radius detection and thus can almost completely eliminate urban GPS drift problems since functions such as *ST_DWithin* can be used for geographic proximity validation [8], [17], [20].

These changes led to the development of a software-only, scalable, and authentication-secured attendance system that would not necessitate the expense and inconvenience of RFID or biometric hardware.

### 1.4.3 Proposed Approach

The system in question merges GPS tracking, geofencing, and confidential authentication to offer an automated attendance method that first verifies the user's location and then records the attendance.

Key Components:

**a) Geofencing-based Location Verification**

The system establishes a virtual boundary (geofence) around the school or office. Real-time GPS coordinates of a user are checked through PostGIS spatial queries, especially *ST_DWithin*, to determine if the user is within a certain radius (for instance, 200 meters) of the specified area [8], [17].

**b) Mobile Client Application**

React Native is used to develop a cross-platform mobile app that enables the capturing of:

- User location
- Timestamp
- Authentication token
- Attendance action (check-in/check-out)
- React Native provides fast rendering and native device GPS integration [9].

**c) Backend & API Layer**

Developed using Express.js, the backend handles:

- Attendance logic
- Geospatial validation
- Authentication
- Multi-tenant database separation
- Express.js ensures lightweight, high-performance routing for location processing [10].

**d) Secure Authentication**

The system implements:

- JWT tokens for stateless, secure session handling [14]
- bcryptjs for hashed password storage [16]

**e) Database Architecture**

Information about attendance, the setting of geofences, and user data are kept in PostgreSQL, which has been extended with **PostGIS** for spatial operations [18]. Multi-tenant patterns make sure that each organization has separate data access [21].

### 1.4.4 Applications of the Project

**a) Educational Institutions**

Schools, colleges, and universities by the means of automation can perform student and staff attendance, get rid of proxy attendance, and create real-time attendance analytics [3], [4].

**b) Corporate Workplaces**

With the help of technologies like AI and IoT, companies are able to know the exact moment when an employee arrives or leaves the workplace and they also can keep track of their remote employees' activities. Moreover, these types of attendances might be linked with the organization's payroll structure for salary calculation based on attendance hours [2], [7], [12].

**c) Field Service & Remote Workforce Management**

A company that employs field agents or has a team of delivery people or even that has a remote team can be very sure if the employees are truly at the client site or service location when they say so [12].

**d) Government & Public Sector Offices**

The system aids in confirming the presence of on-site staff that is engaged in providing welfare programs, health services, smart governance initiatives, and rural workforce tracking [1], [20].

**e) Event Management & Temporary Workforces**

Such events as sessions, conferences, or large gatherings where the attendance needs to be recorded in an automated way without any additional hardware can greatly benefit from this system [13].

**5. Limitations of the Proposed Approach**

The new scheme has a couple of drawbacks despite the fact that it is built on lots of advantages:

**a) GPS Dependency**

The precision of GPS can be compromised in areas with a lot of high buildings, inside of a building, and even during a cloudy day when the signal is weak and hence one will need to set a larger geofence buffer [20].

**b) Battery Consumption**

In situations where the GPS on a mobile device has to be performing its tasks constantly or very often, the battery power on this device will surely be consumed at a faster rate [4].

**c) Vulnerability to GPS Spoofing**

Even though it is less likely to happen due to server-side verification, a user might also consider that there is some possibility that the app developers are utilizing GPS spoofing for server-side validation so that if it is not accompanied by device-level checks or other authentication methods then it will be unsafe. to most extent [6], [13].

**d) Internet Connectivity Requirements**

There is also a situation when it is offline caching that helps people in attendance recording and at the same time they still have to look for packets of network data for synchronization and server validation thus they cannot fully do the work in offline mode [15].

**e) Limited Indoor Precision**

Besides that GPS cannot achieve room-level precision indoors, there should also be certain external systems such as Wi-Fi RTT or Bluetooth beacons to allow a more thorough localization [5], [20].

## 1.5 Objectives

- To identify and assess user behaviour patterns through the collection of dependable GPS-based location data and confirming that the user is within a specified geofence by **PostGIS** spatial queries like *ST_DWithin*, thus ensuring that the rules of attendance are followed [8], [17], [20].
- To study the geolocation accuracy, the geofence performance, and the attendance event transitions (enter/exit states) by means of systematic monitoring, error detection, and performance metrics produced by the system's spatial processing engine [1], [2], [3].
- To handle the attendance data, user roles, organization-level segregation, and multi-tenant access through a well-planned backend architecture based on **PostgreSQL/PostGIS** and **Express.js**, thus ensuring the system's operations are stable and easy to maintain [10], [18], [21].
- To provide the means of secure authentication as well as session management through **JWT** and password hashing by **bcryptjs** in order to make certain that the users who are submitting the attendance records are properly authenticated and authorized [14], [16].
- To establish a mobile-first, cross-platform attendance system by using **React Native** together with a scalable backend API capable of reliable synchronization, fault tolerance during offline scenarios, and easy integration with typical institutional environments [9], [15], [19].

## 1.6 SDGs

An institution that uses a geolocation-based attendance management system is, directly and indirectly, a major contributor to several United Nations Sustainable Development Goals (SDGs). The goals are reached through increased institutional trust, cost savings, and up-to-date digital infrastructure. The SDGs are the global priorities for sustainable development, social equity, and technological advancement, which were established by the 17 global goals adopted by all UN member states in 2015.

*Fig 1.1 Sustainable development goals*

### 1.6.1 SDG 4: Quality Education

The solution contributes to educational transparency as a result of proxy attendance elimination and reduction in manual errors so that the student participation records are accurate. The system-inbuilt check-in methods help save the most needed instruction hours and at the same time, they make hybrid and remote learning more efficient by providing secure digital attendance tracking. This, in essence, leads to the creation of a more just evaluation system, better educational management, and smoother classroom operations.

### 1.6.2 SDG 8: Decent Work and Economic Growth

The automation of attendance verification to achieve security is the main feature of the system through which monitoring of the workforce is done in a responsible manner while at the same time administrative overhead is lessened. The organizations become more productive along with the disappearance of attendance-related conflicts and the facilitation of real-time employee presence monitoring. Besides, its mobile-first approach makes it

ideal for a wide range of working situations such as online teams, employees working at different locations, or groups on a field trip.

### 1.6.3 SDG 9: Industry, Innovation and Infrastructure

By GPS, PostGIS, secure authentication, and cloud services usage, it shows that affordable digital infrastructure can replace expensive hardware-based solutions. The modular, open-source nature of the product is a major factor that allows continuous innovation, easy scaling, and broad dissemination, hence, it is a driver of digital transformation in both institutions and enterprises.

### 1.6.4 SDG 16: Peace, Justice and Strong Institutions

Efficient governance is made possible through the implementation of features such as tamper-proof geofence validation, secure authentication, and audit-friendly digital logs. Fraudulent attendance activities are limited by these provisions, accountability is deepened, and reliability of record-keeping ensured as a result. The institutions get to enjoy better trust, fairness, and data integrity from the sum of their operations due to the collateral effects the system brings about.

## 1.7 Overview of project report

- The new system is basically the next generation solution which has been designed to tackle the limitations of the earlier ones, i.e., those limited to only RFID, GPS, or basic geofencing and which had problems of high hardware costs, attendance proxy, and somewhat unstable positional accuracy [1], [3], [5].
- This method is a hybrid one that experimentally makes use of phone GPS information, PostGIS radius-based geofencing validation, JWT-protected authentication, and React Native mobile interface to record accurate, real-time, location-verified attendance [8], [9], [14].
- The management of the project is through definite phases—requirements analysis, system design, implementation, and testing—facilitated by the backend, which is built with Express.js, and multi-tenant PostgreSQL/PostGIS database architecture for scalable deployment [10], [18], [21].
- Moreover, the implemented system consists of spatial distance checks, state-based attendance transitions (ENTER/EXIT), and offline synchronization capabilities, thus it guarantees a stable performance even if the network is weak or unstable [8], [15].

- The achievement of the system depends largely on the smartphone GPS hardware and the software stack which includes React Native, Node.js/Express, and PostGIS. At the same time, the simulation tests are used for evaluating geofence precision, positional accuracy, and GPS drift behaviour in various surroundings [9],[20].
- Assessment outcomes reveal consistent geofence detection accuracy, heightened authentication integrity, and limited occurrences of proxy attendance, thereby giving an account of the same trend as the recent studies on geolocation-based attendance have indicated [1], [2], [12].

Chapter 2

# Literature review

Research on automated attendance systems has evolved from hardware-driven methods to software-centric geolocation frameworks. Early RFID-based solutions demonstrated improved accuracy compared to manual systems, yet remained vulnerable to card-sharing and required costly infrastructure, revealing a major limitation in scalability and authenticity verification [5]. GPS-enabled systems later emerged, offering mobile-based attendance logging; however, GPS-only methods were hindered by urban signal distortion, inconsistent accuracy, and susceptibility to location spoofing, limiting their reliability in dense environments [4], [20]. Geofencing approaches further improved location validity by enforcing boundary-based attendance, as demonstrated in studies integrating mobile GPS with radius-based distance filtering to reduce proxy attendance, though many implementations still relied on simple distance formulas instead of advanced spatial analysis [1], [3], [6].

Several works extended geolocation techniques with real-time tracking for employee and remote workforce monitoring, showing significant improvements in compliance and transparency, yet raising concerns about privacy, continuous tracking, and device battery consumption [2], [12]. Hybrid systems combining facial recognition with geofencing strengthened identity verification, but increased computational cost and required additional hardware support, which limited adoption in low-resource settings [11]. QR-based systems provided a low-cost alternative but were easily manipulated through code sharing and lacked true location awareness [13]. Recent studies on secure authentication, JWT, and multi-tenant architectures highlighted the importance of robust backend frameworks for preventing unauthorized access and ensuring isolated organizational data management [14], [21]. Additionally, research on mobile offline synchronization emphasized the need for resilient systems that function during network disruptions, reflecting a key requirement for real-world deployments [15]. Across the literature, common gaps include limited use of spatial databases, insufficient handling of GPS drift, weak security models, and dependence on external hardware. Improvements such as integrating PostGIS geospatial processing, secure authentication layers, spoofing detection, and adaptive geofence buffers are suggested to enhance accuracy, reliability, and scalability of next-generation attendance systems.

Chapter 3

# Methodology

## 3.1 Requirements & Literature-Based Specification

The project began with requirements gathering informed by an extensive literature review of RFID-based systems [5], GPS-only models [4], geofencing architectures [1], [3], AI-enhanced attendance systems [11], spoof-resistant authentication [14], and multi-tenant backend structures [21]. Key functional requirements identified include:

- GPS-based location capture
- Geofence radius validation through spatial queries
- Secure user authentication via JWT and hashed credentials [14], [16]
- Offline caching and synchronization [15]
- Cross-platform mobile application interface [9].

## 3.2 System Design

Design activities followed SDLC principles supported by iterative Agile refinement. System design incorporated:

- PostgreSQL + PostGIS spatial schema for geofence storage [8], [17], [18]
- Mobile–server interaction model, validating user coordinates using ST_DWithin
- Authentication model using JWT sessions and bcryptjs password hashing [14], [16]
- Multi-tenant hierarchy, ensuring isolated company-level data [21]

## 3.3 Functional Design

Functional modules were separated based on Agile backlog items:

| Functional Unit | Description | Relevant References |
|---|---|---|
| Geofence Engine | Validates user location using PostGIS | [8], [17] |
| Authentication | JWT-based login, bcrypt password hashing | [14], [16] |
| Attendance Logic | Enter/Exit state transitions | [1], [3] |

| | | |
|---|---|---|
| Offline Sync | Caches logs locally, syncs later | [15] |
| Admin Dashboard | View logs, manage users | [10], [18] |

*Table 3.1 Functional modules*

## 3.4 Unit Design (Software / Mobile / Backend)

### 3.4.1 Mobile Unit Design

- Implemented in React Native, integrating device GPS and user interface components [9].
- Local storage handles offline mode using queue-based caching [15].

### 3.4.2 Backend Unit Design

- Built using Express.js, which supports lightweight routing and secure middleware [10].
- Implements geospatial validation, token decoding, and database operations.

### 3.4.3 Database Unit Design

- PostgreSQL extended with PostGIS for radius-based queries and geospatial indexing [8], [18].
- Multi-tenant isolation using schemas or foreign-key partitioning [21].

## 3.5 Unit Testing

Unit testing included:

- GPS coordinate validation tests
- Token verification and role-based tests [14]
- Database query accuracy tests (ST_DWithin distance threshold) [8], [17]
- UI behaviour testing for GPS permission requests

Open-source tools such as Postman, Jest, and React Native Testing Library supported isolated testing cycles.

## 3.6 Integration Testing

Integration testing covered the complete workflow:

- Mobile app → GPS coordinate capture
- Backend → Spatial validation

- Database → Attendance entry creation
- Admin dashboard → Record retrieval

These tests ensured end-to-end consistency and traceability, aligning with best practices in distributed systems testing [22].

## 3.7 Verification & Validation

Verification ensured the system met all design specifications, while validation ensured real-world behaviour aligned with expected geofence precision. Real-environment tests were performed in open, semi-urban, and obstructed GPS zones, following approaches used in geolocation research evaluating drift and accuracy challenges [20]. Results indicated improved authenticity over previous RFID, QR, and basic GPS systems [4], [5], [13].

## 3.8 Deployment Phase

Deployment used:

- React Native build pipeline for Android/iOS distribution [9]
- Node.js server deployment using modular microservices architecture [10]
- PostgreSQL server hosting with Supabase-like configurations [19]

Deployment validation included load tests, API reliability checks, and performance monitoring.

The proposed metrics provide a structured basis for systematic validation once large-scale testing is conducted.

### 3.8.1 Geofence Detection Accuracy Testing

**Test Methodology:**

Accuracy evaluation will be performed by submitting GPS coordinates at controlled distances from geofence centers. The test dataset will include:

- Points inside the geofence at 10%, 50%, and 90% of its radius.
- Points precisely on the radius boundary.
- Points outside the geofence at 10%, 50%, and 100% of its radius.

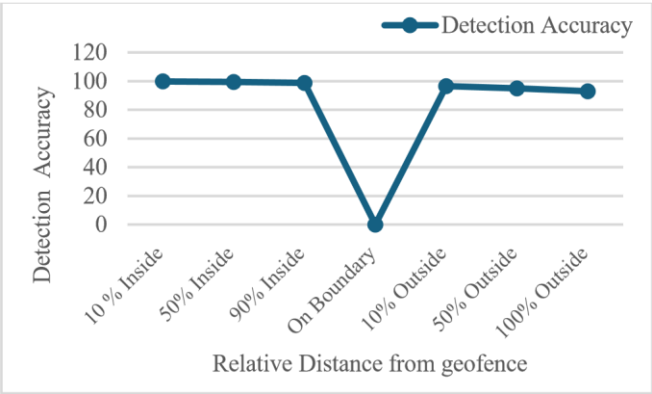


*Fig 3.1 Proposed Geofence Accuracy Testing Methodology.*

Each coordinate will be processed through the backend's **PostGIS** *ST_DWithin* function, which determines spatial containment based on spherical geometry. The system's event response (ENTER/EXIT) will be compared with the expected result for each test input.

**Proposed Formula:**

$$Accuracy\ (\%) = \frac{Correct\ Detections}{Total\ Test\ Cases} \times 100$$

This proposed metric reflects how well the geofencing logic, built on PostGIS, aligns with real-world coordinate inputs under various environmental conditions (open fields, semi-urban, and dense urban areas). Accuracy testing will also consider hardware variance between devices, as GPS chip precision may differ across models.

### 3.8.2 Positional Deviation Measurement (RMSE)

**Objective:**

To quantify positional error between the expected and the detected coordinates within the geofence.

**Proposed Formula:**

$$RMSE = \sqrt{\frac{\sum d_i^2}{n}}$$

where $d_i$ represents the *geodesic distance* (in meters) computed by the PostGIS *ST_Distance* function between expected and recorded positions. This metric inherently uses the spherical model (WGS 84 ellipsoid) to account for Earth's curvature.

Unlike a simple Euclidean calculation, this approach ensures a realistic measure of location accuracy consistent with internal spatial computations performed by the system.

### 3.8.3 API Response Time Measurement

***Test Methodology:***

Backend responsiveness will be evaluated using request–response timestamps logged in *Express.js middleware*. Each endpoint (authentication, geofence query, attendance submission, and location tracking) will be subjected to 100–500 test requests under two scenarios:

- ***Baseline***: single-user sequential requests.
- ***Concurrent Load***: simulated multi-user parallel requests.

***Metrics to be Observed:***

- Mean response time
- Median response time
- 95th percentile latency (tail delay detection)
- Throughput (requests per second)

***Proposed Performance Efficiency Formula:***

$$E_p = \frac{T_b}{T_l}$$

where:

- $T_b$= mean baseline response time
- $T_l$ = mean response time under load

Higher $E_p$ values indicate better scalability and lower degradation under concurrent load. These values will be derived from the internal performance logs recorded by the Express.js framework.

### 3.8.4 Offline Synchronization Reliability

***Test Methodology:***

To assess data reliability in offline conditions, the mobile client will deliberately simulate network outages. GPS coordinates collected during disconnection will be cached using device storage APIs. Upon network restoration, cached data will be transmitted in batches to the backend.

Scenarios will vary by:

- Queue size (5, 25, 100 entries)
- Offline duration (5 minutes to 1 hour)
- Application state (foreground, background, terminated)

***Proposed Reliability Formula:***

$$Reliability\ (\%) = \frac{Synchronize\ Records}{Total\ Queued\ Records} \times 100$$

This reflects the proportion of successfully synchronized data after reconnection, quantifying the robustness of the caching and recovery mechanism implemented within the app.

### 3.8.5 Functional Requirements Verification

***Test Methodology:***

Functional validation will ensure the system performs all specified behaviours correctly. Each requirement will be mapped to a pass/fail test case, including:

- Successful user authentication returns a valid JWT.
- Token manipulation attempts do not allow cross-company access.
- Admin users can create, update, and delete geofences.
- Non-admin users receive authorization errors for restricted actions.
- ENTER and EXIT events are correctly paired without duplicates.
- Manual check-ins work correctly outside geofence boundaries.
- (Optional) Automatic check-ins are restricted to working hours if time-based conditions are enabled.

Each functional test will be executed against both API and UI layers, ensuring consistency across the client - server workflow. The outcomes will form the empirical validation of system functionality during the next evaluation phase.

***System Architecture***

The complete system follows a three-tier architecture with spatial database capabilities at the core.

| Component | Technology | Primary Responsibility |
|---|---|---|
| **Mobile Application** | React Native + Expo | GPS data collection, user interface, offline caching |
| **API Gateway** | Express.js Middleware | Request routing, authentication verification |

| | | |
|---|---|---|
| **Authentication Service** | JWT + bcryptjs | User login, token generation and validation |
| **Geofence Service** | Express.js + PostGIS | Geofence CRUD operations and spatial queries |
| **Attendance Service** | Express.js + PostgreSQL | Record attendance events and retrieve history |
| **Database** | PostgreSQL with PostGIS | Persistent storage with spatial query support |
| **Cloud Infrastructure** | Supabase | Database hosting and availability management |

*Table 3.2 System Architecture Components*

### 3.8.6 Spatial Database Query Optimization

Instead of fetching all geofences and computing distances within the application layer, spatial queries are offloaded to the database. The 'ST_DWithin' function in PostGIS [8] efficiently filters geofences using distance criteria, leveraging spatial indexing (GIST indexes) on the location data [17]. This strategy yields two benefits: (1) the database engine itself optimizes the query using its spatial indexes, and (2) it minimizes data transfer between the database and the application by filtering records at the source. The use of the GEOGRAPHY data type ensures that spherical geometry is automatically applied, precluding the need for manual, latitude-based distance adjustments.

### 3.8.7 State Machine for Event Generation

A simplistic approach would generate an ENTER event for every location update received from within a geofence, leading to hundreds of redundant entries. Our solution implements a state machine. For each user - geofence pair, it tracks whether an ENTER event is active. A new ENTER event is generated only upon a state change from outside to inside. Correspondingly, an EXIT event is generated only upon transitioning from inside to outside. This stateful logic eradicates duplicate events and accurately manages multiple entries and exits during a single day.

### 3.8.8 Company-Level Data Isolation

In a multi-tenant architecture [21], it is imperative to stop users from one organization from accessing another's data. Our system achieves this isolation at two levels: first, the `company_id` is embedded within the JWT token [14] during login; second, every database query is forced to include a `WHERE company_id = $1` filter, using the ID from that token. This two-pronged strategy guards against both accidental data leaks via query design flaws and intentional attacks via token modification.

| Aspect | Manual Systems | Biometric [5] | GEOSYNC |
|---|---|---|---|
| **Hardware Dependency** | None | High | None |
| **Scalability** | Manual per site | Limited [11] | Good |
| **Remote Work Support** | Not possible | Not applicable | Supported |
| **Real-time Data Access** | Delayed [4] | Real-time | Real-time |

*Table 3.3 Comparison with alternative systems*

Chapter 4

# Project Management

## 4.1 Project timeline

### 4.1.1 Project Planning

The planning phase established clear goals, selected technologies aligned with project requirements, and defined the functional architecture. Milestones included literature review, requirements specification, tech stack selection, architecture design, and scheduling. This phase ensured alignment between objectives, constraints, and development strategy, enabling a structured and realistic implementation roadmap.

This table summarizes the major milestones identified during the planning phase.

| Phase | Task / Milestone | Start Date | End Date | Outcome |
|---|---|---|---|---|
| Requirement Analysis | Literature review, problem scoping, requirement gathering | Week 1 | Week 2 | Requirement Specification Document |
| System Feasibility & Technology Selection | Selection of GPS, PostGIS, React Native, Express.js | Week 2 | Week 3 | Tech Stack Finalized |
| System Architecture Planning | System modules, data flow, geofence design, security layers | Week 3 | Week 4 | Architecture Blueprint |
| Project Scheduling & Resource Allocation | Timeline finalization, task assignment | Week 4 | Week 4 | Project Plan & Timeline |

| | | | | |
|---|---|---|---|---|
| Prototype Planning | Identify core functions for early prototype | Week 4 | Week 5 | Prototype Scope Document |

*Table 4.1 Project Planning Timeline*

### 4.1.2 Project implementation

The implementation phase progressed iteratively, beginning with the mobile interface and backend API setup, followed by GPS integration and geofence implementation. Secure authentication and database modules were integrated mid-phase, while offline synchronization and system-wide integration testing were executed toward the end. Deployment activities concluded the timeline, ensuring the system met functional and performance criteria.

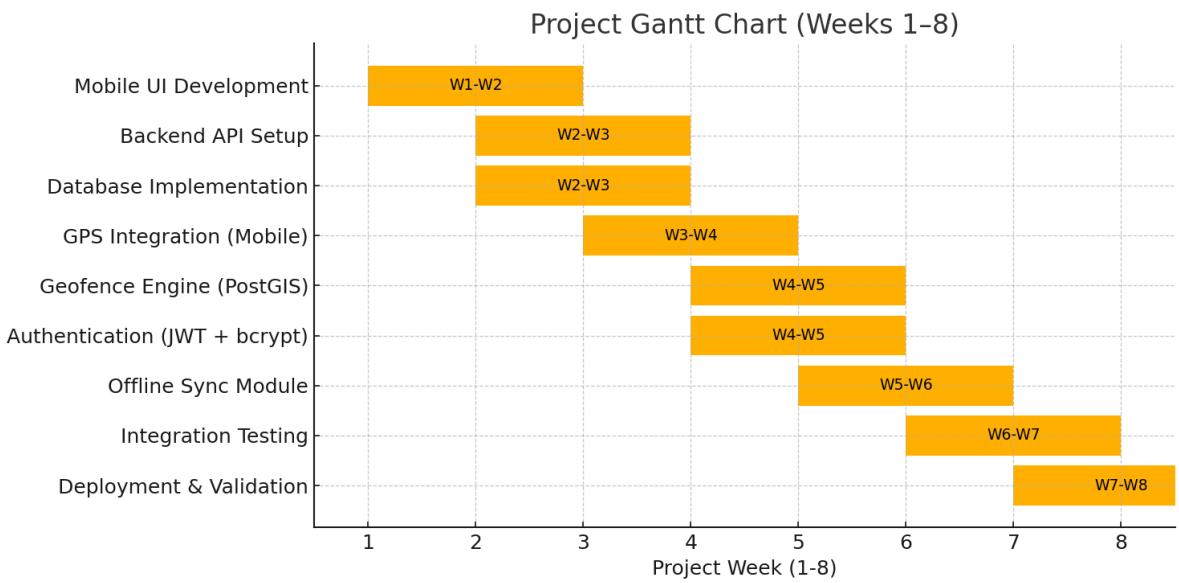


*Fig 4.1 Project Implementation Timeline*

## 4.2 Risk analysis

A geolocation-based attendance system introduces technical, operational, security, and deployment-related risks. The following analysis identifies key risks, evaluates their likelihood and impact, and outlines mitigation strategies to ensure stable and secure system performance.

| Risk | Likelihood | Impact | Mitigation Strategy |
|---|---|---|---|
| GPS Drift / Inaccuracy | Medium | High | Geofence buffer, PostGIS filtering |

| | | | |
|---|---|---|---|
| Network Failure | High | Medium | Offline sync, cached logs |
| GPS Spoofing | Medium | High | Spoof detection, sensor verification |
| Credential Theft | Medium | High | JWT + bcrypt security |
| Multi-tenant Data Leak | Low | High | Schema isolation, RBAC |
| User Disabling GPS | High | Medium | Permission checks, forced prompts |
| Server Downtime | Low | High | Cloud scaling, uptime monitoring |
| Database Bottlenecks | Medium | High | Indexing, query optimization |

*Table 4.2 Risk Analysis Table*

Chapter 5

# Analysis and Design

Analysis and design form two interconnected but distinct phases in the development of the geolocation-based attendance system. **Analysis** focuses on understanding the core problem—manual attendance inaccuracies, proxy attendance, GPS drift, and lack of real-time authentication—by breaking the system into functional and behavioural requirements. This stage identifies *what* the system should achieve, based on observations from existing GPS, RFID, QR, and geofencing-based solutions discussed in prior research [1], [3], [4], [5], [7]. In contrast, **Design** determines *how* the system will accomplish these requirements by structuring system components such as the geofence engine, mobile app interface, backend architecture, and database schema using proven technologies like PostGIS, JWT authentication, and React Native [8], [9], [14], [18]. This structured approach ensures the solution is functionally sound, secure, scalable, and capable of mitigating the weaknesses identified in earlier systems [11], [12], [21].

## 5.1 Requirements

The requirement phase identifies the complete behaviour, purpose, and functional boundaries of the system. Requirements were derived from literature insights, system shortcomings observed in traditional methods, and modern geolocation-enabled architectures [1], [6], [20].

### 5.1.1 System Hardware Requirement Phase

- **Identify Initial Conditions**
  - Users must possess a GPS-enabled smartphone.
  - Device location services and internet connectivity should be available.
  - Server hosting PostgreSQL/PostGIS must be operational.
- **Determine Input Parameters**
  - User's real-time GPS coordinates.
  - Device ID, timestamps, and authentication tokens.
- **System Outcomes**
  - Accurate detection of whether the user is within the geofence.
  - Attendance log creation (enter/exit).
  - Real-time and historical attendance reports.

- **Formulate Relations**
  - Distance calculation using PostGIS ST_DWithin to determine geofence validity [8], [17].
  - Mapping between user accounts and attendance records.
- **Identify System Constraints**
  - GPS drift in urban/indoor areas [20].
  - Battery usage during location tracking.
  - Hardware dependency on smartphone sensors.

### 5.1.2 System Software Requirement Phase

- **Identify Initial Conditions**
- User must be registered and authenticated via JWT [14].
- Mobile app must have permission to access location.
- Backend APIs must be available.

- **Determine Input Parameters**
- Login credentials
- User location data
- API requests (check-in, check-out, logs)

- **System Outcomes**
- Successful authentication
- Geofence validation
- Attendance record creation and storage
- Admin dashboard visibility

- **Formulate Relations**
- Authentication → Token validation → Location → Attendance logic
- Database relations between users, organizations, and geofences [18], [21].

- **Identify System Constraints**
- Location spoofing attempts [6], [13].
- Network latency during synchronization [15].
- Scalability for multi-tenant organizations [21].

### 5.1.3 Data Collection Requirements

- GPS coordinates collected at check-in/out.
- User device metadata (ID, time, token validity).
- Admin-level data for reporting and auditing. Cited research stresses the importance of accurate spatial data for attendance validation [1], [3], [12].

### 5.1.4 Data Analysis Requirements

- Validation of geofence proximity through spatial queries [8].
- Detect anomalies such as sudden GPS jumps, repeated spoofing attempts, or inconsistent behaviour.
- Generate analytical attendance trends similar to prior workforce systems [2], [12].

### 5.1.5 System Management Requirements

- Multi-tenant segregation of data to avoid cross-organization interference [21].
- Admin controls for geofence configuration, user management, and record updates.
- Monitoring of server uptime, database health, and API performance.

### 5.1.6 Security Requirements

- JWT-based secure session handling [14].
- Password hashing using bcryptjs [16].
- Validation against GPS spoofing and unauthorized access [6], [13].
- Role-based access control and encrypted data channels.

### 5.1.7 User Interface Requirements

- Simple mobile interface for check-in/out actions using React Native [9].
- Permission prompts for GPS access.
- Admin dashboard for viewing logs and reports.
- Error messages for geofence failures or invalid authentication.

| Category | Description |
|---|---|
| **Purpose** | A mobile and web-enabled geolocation attendance system that verifies user presence using GPS and geofencing, automates attendance recording, and stores logs securely. |

| | |
|---|---|
| **Behaviour** | The system supports two operational modes: **Auto Mode:** Automatically marks attendance when the user enters or exits the geofence radius using GPS + PostGIS spatial validation.<br>**Manual Mode:** Enables users to manually request attendance logging, validated through authentication and location checks. |
| **System Management** | Provides real-time monitoring for admins, including attendance logs, user management, geofence configuration, and organization-level dashboard access. Supports multi-tenant management for companies/schools. |
| **Data Analysis** | Performs geospatial analysis (distance checks, drift filtering), detects anomalies such as spoofing patterns, calculates attendance duration, and generates analytical reports for admin review. |
| **Application Deployment** | Application is deployed on a cloud-based backend (Node.js + PostgreSQL/PostGIS) and a mobile client (React Native). Accessible remotely with secure APIs while maintaining local geofence validation logic. |

*Table 5.1 Requirements*

## 5.2 Block diagram

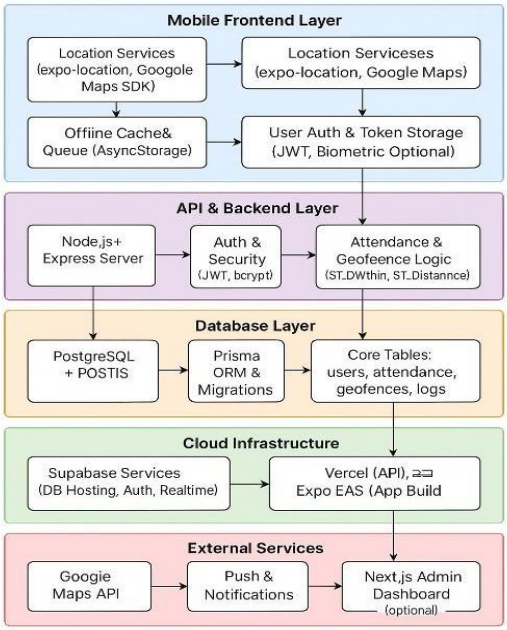


*Fig 5.1 Functional Block Diagram*

The system architecture adopts a layered structure that integrates mobile sensing, backend processing, geospatial computation, cloud infrastructure, and external APIs into one cohesive attendance solution. At the top layer, the mobile frontend—built with React Native and Expo—handles GPS acquisition through expo-location and Google Maps SDK while managing offline storage using AsyncStorage and securing user sessions through JWT token storage and optional biometric authentication. These data points flow into the API and backend layer, where a Node.js–Express server manages routing, authentication with JWT and bcrypt, and core geofence logic using spatial functions such as ST_DWithin and ST_Distance. The backend interacts with the database layer, powered by PostgreSQL and PostGIS, which stores users, attendance logs, geofences, and system events while Prisma ORM manages schema migrations and data consistency. Beneath this, the cloud infrastructure layer—leveraging Supabase for hosting and realtime services, Vercel for API deployment, and Expo EAS for building the mobile app—ensures scalable, secure, and efficient application delivery. Finally, external services such as Google Maps API enhance location visualization, push notification services support alerts and reminders, and an optional Next.js admin dashboard enables real-time monitoring and management. Together, these layers form a robust, secure, and scalable architecture tailored for accurate location-based attendance tracking.

# 5.3 System Flow chart

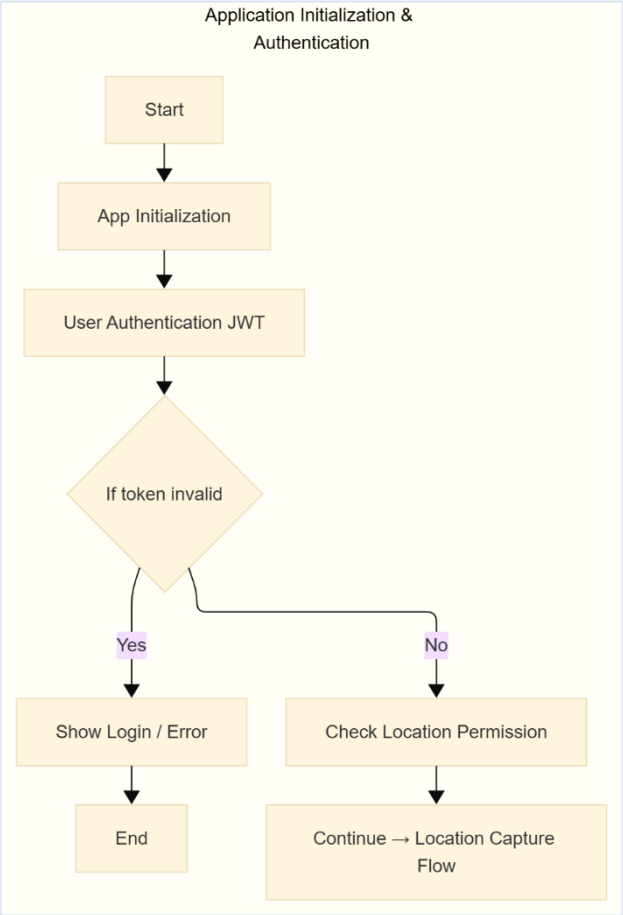


*Fig 5.3 Initialization and Authentication*

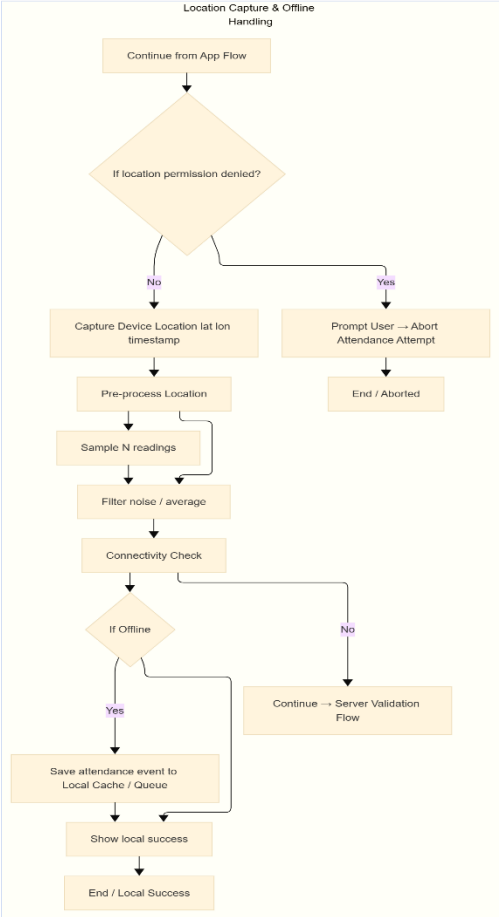


*Fig 5.2 Location Capture and Offline Handling*

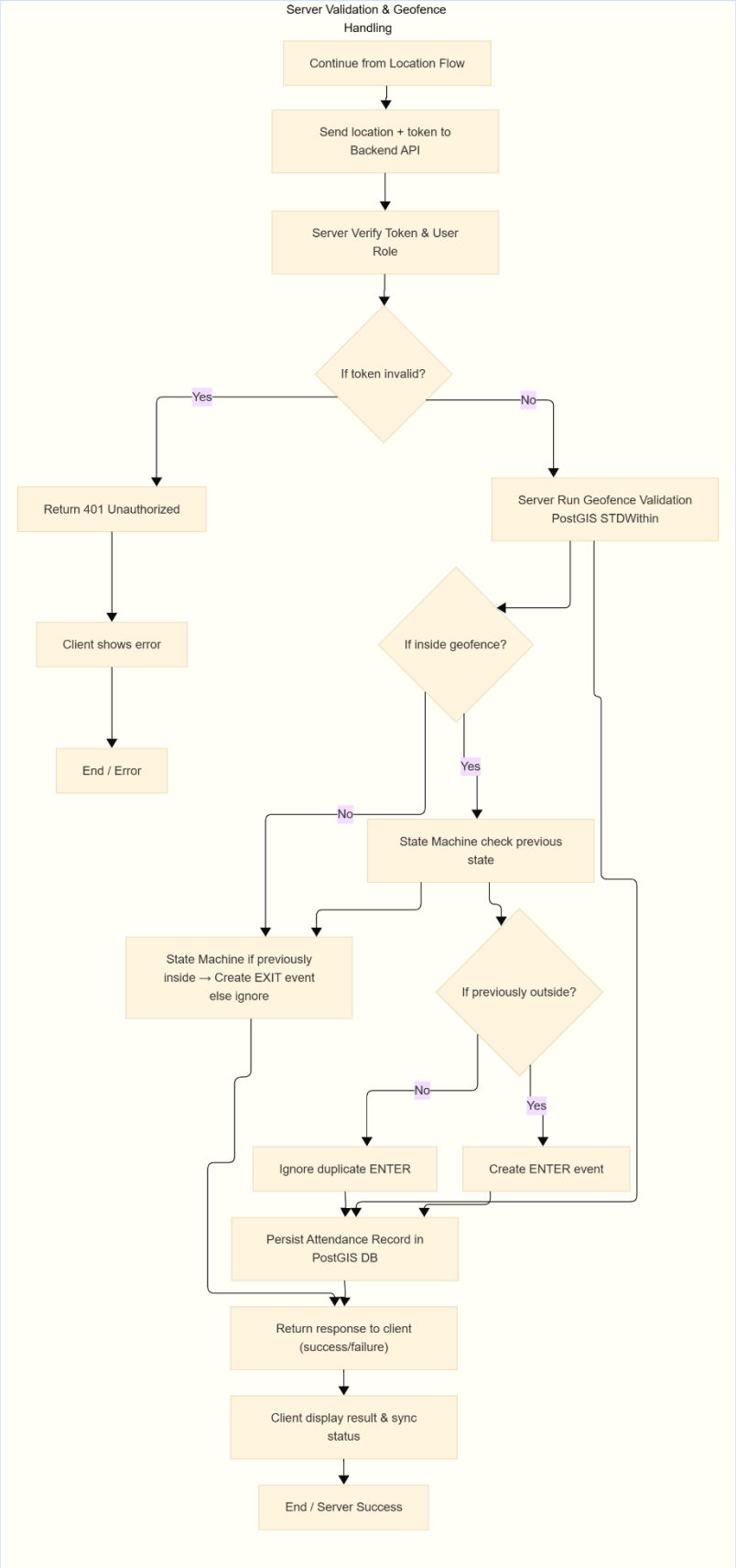


*Fig 5.4 Server Validation and Geofence Handling*

### 5.3.1 Design notes & reasons (concise)

- **Client pre-filtering** reduces server load and lowers false positives caused by momentary GPS jitter.
- **Offline caching** guarantees resilience in low-connectivity regions and preserves data integrity (critical per IEEE mobile sync best practices).
- **Server-side validation** is authoritative—client checks alone are insufficient due to spoofing risks (server verifies identity and geospatial position).
- **State machine** avoids duplicate logs and models enter/exit semantics cleanly for reporting and payroll calculations.

## 5.4 Choosing devices

For a location-based attendance platform, the core requirement is the ability to capture and validate geographic position accurately. Since modern smartphones already integrate high-precision GPS receivers, digital communication modules, and secure operating environments, they function as the primary hardware device in the system. However, to ensure completeness in design evaluation, additional device options were reviewed to understand how the system could extend to edge-based or on-premise deployments if required.

### 5.4.1 Choosing a Processing Device

Several processing platforms were compared, including microcontroller boards and single-board computers, mainly to assess their suitability for tasks such as indoor geofencing, Bluetooth-based boundary detection, or local data aggregation. Smartphones remain the primary choice, but processors commonly used in lightweight IoT systems were evaluated for reference.

| Specification | Raspberry Pi Zero W | NodeMCU (ESP8266) | ESP32 Dev Kit |
|---|---|---|---|
| **Power Supply** | **5V** | **5V** | **5V** |
| **Digital I/O Pins** | **27** | **11** | **34** |
| **Analog Inputs** | **None** | **1** | **18** |
| **Wi-Fi** | **Yes** | **Yes** | **Yes** |
| **Bluetooth** | **No** | **No** | **Yes (BLE + Classic)** |
| **Serial Ports** | **1** | **1** | **3** |

| | | | |
|---|---|---|---|
| **RAM** | **512 MB** | **80 KB** | **520 KB** |
| **Typical Use Case** | **Local compute node** | **Low-power applications** | **Indoor geofence beacons** |

*Table 5.2 Comparing features of different processors*

After comparison, the internal smartphone processor is selected as the main computation device due to its integrated GPS hardware, stable mobile OS, and built-in communication capabilities. ESP32 remains a suitable optional extension for future indoor boundary detection (via BLE beacons).

### 5.4.2 Choosing Location-Sensing Modules

Since geographical position is central to attendance validation, location-sensing methods were reviewed. GPS modules, Bluetooth beacons, and Wi-Fi RTT devices were compared to determine their potential integration into different deployment environments.

| Specification | Smartphone GPS | GPS Module (NEO-6M) | BLE Beacon | Wi-Fi RTT Device |
|---|---|---|---|---|
| Working Principle | Satellite trilateration | Satellite trilateration | Signal strength proximity | Time-of-flight distance |
| Accuracy | 5–15 m outdoors | 2–5 m outdoors | 1–4 m indoors | 1 m or less |
| Output Type | OS API digital output | Serial digital output | BLE broadcast | Wi-Fi distance data |
| Environment Suitability | Outdoors/semi-urban | Outdoors | Indoor spaces | Indoor with RTT-enabled routers |

*Table 5.3 Built-In GPS Modules*

The built-in GPS receiver of smartphones is the primary choice because it requires no extra hardware, supports global operation, and is managed directly by the operating system. BLE and Wi-Fi RTT are potential enhancements for indoor environments where GPS performance is limited.

### 5.4.3 Summary

The system is designed around devices that are already embedded within smartphones, minimizing the need for external modules or circuit-level hardware. Smartphones provide the sensing capability, computing resources, secure authentication environment, and network connectivity required for accurate and reliable location-based operation. Optional devices such as ESP32-based beacons or Wi-Fi RTT units may be adopted later to enhance indoor accuracy or support hybrid attendance environments.

## 5.5 Designing units

Designing the system involves breaking it into functional units and defining how each unit processes information, interacts with other units, and contributes to the overall workflow. Although the system does not rely on physical circuits or traditional signal-conditioning components, the same design principles can be applied to data flows, API interactions, validation logic, and mobile-to-server communication. Each unit below is designed with clear boundaries, well-defined responsibilities, and stable interfaces.

### 5.5.1 Mobile Location Unit

**Purpose:**

Captures the user's geographical coordinates and prepares them for validation.

**Design Elements:**

- Retrieves latitude and longitude using the device's GPS provider.
- Reads multiple samples to reduce jitter and averages them before sending.
- Ensures location permissions are enabled before proceeding.
- Handles offline mode by storing location events in a local queue.

**Interfacing Logic:**

Mobile App → GPS API → Location Processing → Output: Filtered coordinates.

**Equivalent to "signal conditioning":**

- Eliminates noisy readings
- Averages multiple samples
- Filters out unrealistic jumps (e.g., 0 → 300 meters instantly)

### 5.5.2 Communication & Authentication Unit

**Purpose:**

Ensures secure transmission of user identity and location data.

**Design Elements:**

- Attaches JWT token to every request.
- Uses HTTPS to encrypt data in transit.
- Validates token expiry before sending requests.
- Retries requests if temporary network issues occur.

**Interfacing Logic:**

Mobile App → HTTPS Request → Backend Authentication Layer.

**Equivalent to "interfacing circuits":**

- Ensuring compatibility (token format, headers)
- Ensuring secure connection (TLS)
- Ensuring signal integrity (retry, timeouts)

### 5.5.3 Geofence Validation Unit

**Purpose:**

Checks whether a coordinate is inside or outside a designated boundary.

**Design Elements:**

- Uses PostGIS spatial queries for precise distance measurement.
- Compares incoming coordinates against a stored geofence radius.
- Applies a state machine to differentiate ENTER and EXIT events.
- Handles edge cases (boundary jitter, repeated position).

**Interfacing Logic:**

Backend API → PostGIS Engine → Validation Result → Attendance Logic.

**Equivalent to "computing values":**

- Distance calculation using spatial functions
- Threshold comparison (radius check)
- Boolean output: inside / outside

### 5.5.4 Attendance Logging Unit

**Purpose:**

Creates attendance records based on validated geofence state.

**Design Elements:**

- Generates ENTER or EXIT based on change of state.
- Records timestamp, user ID, and geofence ID.
- Maintains log consistency to prevent duplicates.
- Supports conflict resolution for offline-synced records.

**Interfacing Logic:**

Backend → Database Insert → Admin Dashboard Display.

**Equivalent to "A/D conversion logic":**

Not needed; however, this unit converts "raw coordinate data" into **meaningful attendance events**.

### 5.5.5 Synchronization Unit (Offline to Online)

**Purpose:**

Ensures reliable attendance recording during network interruptions.

**Design Elements:**

- Stores events in local persistent storage when offline.
- Automatically syncs queued events when connectivity returns.
- Prevents duplicate sync attempts using unique event IDs.
- Verifies successful server acknowledgment before deleting local data.

**Interfacing Logic:**

Local Queue → Sync Handler → Backend Server.

**Equivalent to "buffering/temporary storage":**

- Acts like a signal buffer
- Prevents data loss
- Ensures smooth "handover" from offline to online mode

### 5.5.6 Administrative Dashboard Unit

**Purpose:**

Provides real-time visibility and management functions for administrators.

**Design Elements:**

- Displays user logs, attendance patterns, and geofence configurations.
- Allows updates to geofence radius, allowed time windows, and roles.
- Uses server pagination and caching to improve performance.

**Interfacing Logic:**

Front-end Dashboard → Backend API → Database.

**Equivalent to "system-level integration":**

Acts as the controller for the overall application ecosystem.

### 5.5.7 Summary Unit Design

| Unit | Responsibility | Input | Output |
|---|---|---|---|
| Mobile Location Unit | Capture & filter GPS data | Raw GPS readings | Clean coordinates |
| Communication Unit | Secure data transfer | User token + coords | Authenticated request |
| Geofence Validation Unit | Check boundary rules | Coordinates | Inside/Outside state |
| Attendance Logging Unit | Create attendance entries | Validation result | Attendance record |
| Sync Unit | Offline → online sync | Queued logs | Synced server logs |

| | | | |
|---|---|---|---|
| Dashboard Unit | Admin monitoring & control | DB data | Insights & management |

*Table 5.4 Summary of Unit Design*

## 5.6 Standards

Standards play an essential role in ensuring that digital systems operate securely, reliably, and interoperably across different platforms and environments. In a location-based attendance system, standards guide how data is transmitted, stored, authenticated, and protected. They also ensure that devices and software components communicate correctly and that the system adheres to established industry practices for security and privacy.

### 5.6.1 Communication Standards

- **HTTPS/ TLS Encryption:**
  All communication between the mobile app and backend services is secured through HTTPS using TLS encryption. This prevents interception of GPS data, authentication tokens, and personal information during transmission, aligning with widely accepted web security standards.

- **JSON-based REST APIs:**
  The system uses JSON as the standard data format for exchanging information between client and server. JSON is universally adopted in modern web and mobile applications because it is lightweight, structured, and supported by all major frameworks.

### 5.6.2 Security and Privacy Standards

- **JWT (RFC 7519):**
  Authentication relies on the JSON Web Token (JWT) standard, which defines a secure and stateless method for verifying user identity. JWTs ensure that no sensitive data is transmitted in plain text and that each request is cryptographically validated.

- **Password Hashing (bcrypt):**
  User credentials are stored using bcrypt-based hashing to ensure compliance with secure password-storage guidelines. This protects users even if database access is compromised.

- **Information Security Standards (ISO/IEC 27001 principles):**
  Although not implemented as a certification, the system adopts principles from ISO 27001 such as access control, risk mitigation, and secure data handling to maintain confidentiality, integrity, and availability of data.

### 5.6.3 Data Management and Architecture Standards

- **PostgreSQL/PostGIS Spatial Standards:**
  Geospatial operations follow the OGC (Open Geospatial Consortium) standards implemented in PostGIS.
  This ensures accurate distance calculations, geofence modeling, and spatial indexing aligned with global GIS practices.

- **Database ACID Compliance:**
  PostgreSQL's ACID (Atomicity, Consistency, Isolation, Durability) compliance ensures that attendance records are consistently stored, even during device disconnection or network failures.

### 5.6.4 Interoperability and Application Architecture Standards

- **REST Architecture Guidelines:**
  The backend follows RESTful API conventions, supporting stateless operations and predictable request patterns.
  This ensures that multiple clients (mobile apps, dashboards, admin tools) can reliably access the system.

- **Cross-Platform Mobile Standards (React Native):**
  The mobile app is built using React Native, adhering to industry standards for cross-platform development. This ensures consistent behavior across Android and iOS devices.

### 5.6.5 Benefits of Using Standards

- **Reduced Development Complexity:**
  Leveraging established frameworks avoids reinventing communication, security, or data-handling mechanisms, reducing development time.

- **Improved Interoperability:**
  Standardized APIs and data formats allow seamless integration with dashboards, external reporting tools, and third-party systems.

- **Enhanced Security:**
  Adopting well-proven cryptographic and authentication standards significantly reduces security vulnerabilities.

- **Future Scalability:**
  Systems built on widely used standards can be extended, audited, and deployed across multiple environments without major redesign.

## 5.7 Mapping with IoTWF reference model layers (in tabular form)

The IoT World Forum Reference Model (IoTWF-RM) provides a structured way to visualize how different layers of an IoT ecosystem interact—from physical sensing devices up to business processes and user-facing applications. Mapping the system to this model helps clarify how data flows, where computation occurs, and how security is applied across layers. The model also ensures interoperability by defining clear roles and interfaces at each level.

The table below shows how each of the seven IoTWF layers correspond to the technologies and components used in the system.

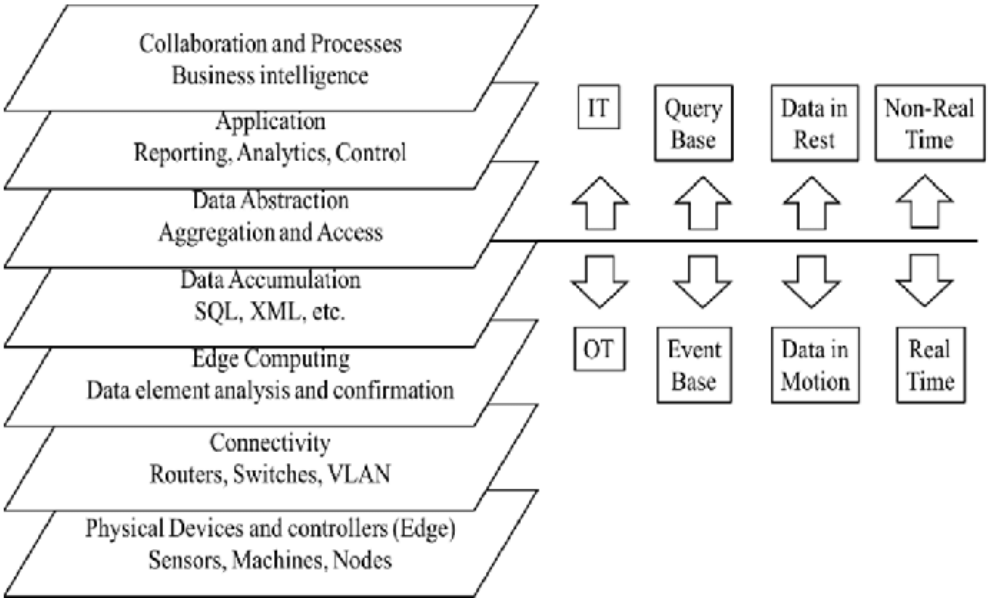


*Fig 5.5 The IoT World Forum Reference Model*

| Layer (IoTWF Reference Model) | Project Layer Mapping | Security (Tiered at Layer Transitions) |
|---|---|---|
| **Collaboration & Processes** | Attendance reporting, workforce analytics, audit logs, admin workflows, organizational policies | Role-based access control, authorization rules |
| **Application Layer** | Mobile app (attendance marking), admin dashboard, real-time attendance view | JWT-based session security, user-level permissions |
| **Data Abstraction** | API data formatting (JSON), backend response handling, geofence interpretation, state transitions | Input validation, rate limiting, secure API endpoints |
| **Data Accumulation (Storage)** | PostgreSQL/PostGIS database storing user profiles, geofence definitions, attendance logs, timestamps | Database encryption, access control, secure schema separation |
| **Edge Computing (Local Processing)** | Smartphone-side processing: GPS sample averaging, noise reduction, offline caching, location permission checks | Device-level security, OS sandboxing, secure key storage |
| **Connectivity** | HTTPS communication, mobile internet/Wi-Fi, RESTful API calls between client and server | TLS encryption, certificate validation |
| **Physical Devices & Controllers** | Smartphones capturing GPS data; optional BLE beacon nodes for indoor geofence extension | OS-level protections, secure GPS signals, anti-spoofing checks |

*Table 5.5 Mapping Project Layers with IoTWF-RM*

## 5.8 Domain model specification

The domain model describes the key concepts, entities, and relationships that define the operational context of the system. It represents how physical objects, digital representations, devices, resources, and services interact to complete attendance verification based on geographic position. The domain model does not depend on any specific technology or

platform; instead, it expresses the logical structure of the system in an abstract and meaningful way.

The system involves a real-world physical entity (the user's presence at a location), a virtual representation of this user in the digital system, and supporting components that sense, process, and validate location for attendance purposes.

| Domain Concept | Description (Aligned to Location-Based Attendance System) |
|---|---|
| **Physical Entity** | The individual whose physical presence needs to be confirmed (e.g., employee, student). Their real-world location acts as the primary physical attribute being sensed. |
| **Virtual Entity** | A digital profile representing the user in the system. This includes their user ID, authentication token, assigned geofences, and attendance history stored in the database. |
| **Device** | The smartphone that captures GPS coordinates and communicates with the backend. It acts as the sensing device and provides the interface between physical and digital world states. |
| **Resource** | Software components that process, store, or validate data. Examples include location APIs, PostGIS spatial functions for geofence checks, authentication resources (JWT decoding), and cloud database storage. |
| **Service** | Functional services that interact with the domain, such as location validation service, attendance-marking service, authentication service, synchronization service, and reporting service available to administrators. |

*Table 5.6 Description of Domain model*

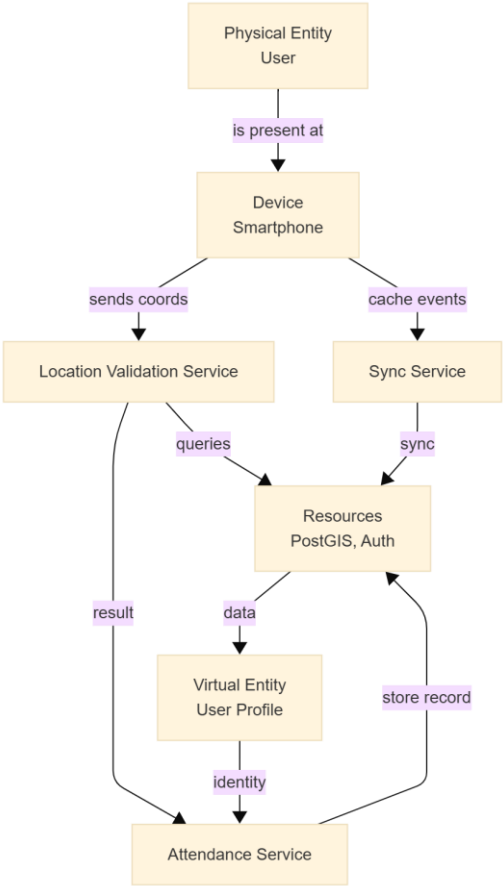


*Fig 5.6 Domain Model*

## 5.9 Communication model

Selecting an appropriate communication model is essential for ensuring reliable data exchange between the smartphone, backend server, and database. Since attendance marking requires a clear request from the user and a definitive response from the server (such as "inside geofence," "outside geofence," or "attendance recorded"), the system follows a **Request–Response communication model**.

This model is a natural fit because the mobile device initiates each attendance attempt by sending user credentials and GPS coordinates to the backend, and the server processes this information before returning a decision. The flow is predictable, secure, and easy to validate, making it suitable for real-time attendance operations where accuracy and clarity are important.

The Request–Response model also supports encrypted communication (HTTPS), authentication tokens (JWT), and stateless API calls, which are essential for mobility, scalability, and security.

### 5.9.1 Suitability of the Communication Model

- The mobile device initiates all communication, ensuring user control and preventing unnecessary background tracking.
- The backend sends explicit confirmation or rejection, providing a clear user experience.
- Works well with REST APIs, JSON payloads, and stateless session management.
- Supports secure communication through TLS and token-based authentication.
- Easy to scale since each request is independent, matching typical cloud architecture patterns.

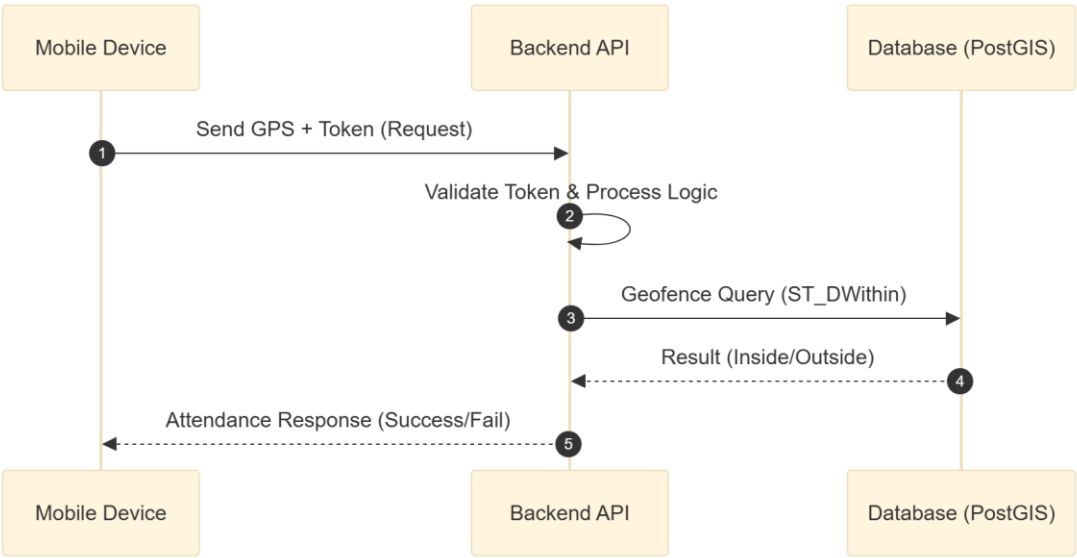


*Fig 5.7 Request - Response Model*

## 5.10 IoT deployment level

IoT deployment levels describe how data is sensed, processed, stored, and utilized across an IoT system. Each level defines the system's complexity and how tightly sensing devices and cloud services are integrated. For an attendance system that relies on smartphone GPS, cloud-based processing, and spatial validation, the most appropriate choice is **IoT Deployment Level 3**.

Level 3 is characterized by **monitoring + limited control**, where data is collected through end devices, processed both locally and in the cloud, and then used for decision-making. In this system, the smartphone performs basic location filtering and permission checks (local processing), while the backend performs geofence calculations, authentication, and database

operations (cloud processing). The user receives immediate feedback about attendance status, aligning perfectly with Level 3 capabilities.

### 5.10.1 Suitability of IoT Deployment Level 3

- **Local computation:**
  The smartphone handles GPS reading, location filtering, and offline caching.
- **Cloud computation:**
  The backend server processes geofence logic, validates tokens, and stores attendance logs.
- **Real-time monitoring:**
  Administrators can view attendance records and device check-ins through dashboards.
- **Limited actuation:**
  The system does not control physical devices, which fits the definition of Level 3 (monitoring-only IoT).
- **Efficient for mobile-based sensing:**
  GPS sensing is done on the device, while heavy computation (PostGIS) is done on the cloud — a hybrid well-matched to this deployment level.

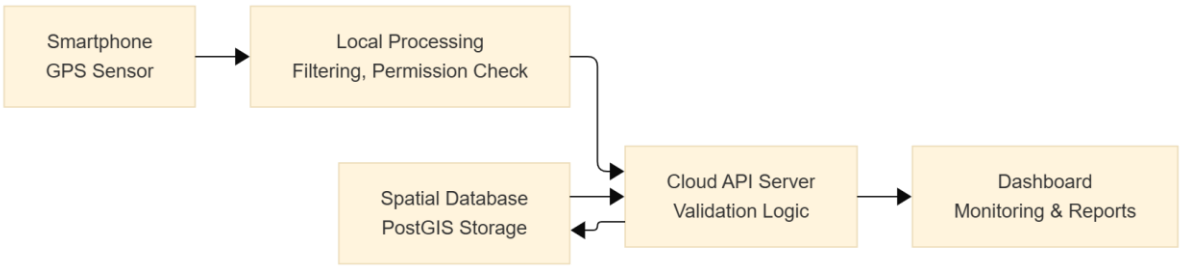


*Fig 5.8 IoT Deployment Level 3 Overview*

## 5.11 Functional view

The functional view describes the system by grouping its capabilities into core functional blocks. Each block represents a major function, such as sensing, communication, processing, management, security, or user-facing application logic. This view helps illustrate how data moves through the system and how each functional area contributes to the overall workflow of location-based attendance marking.

In this system, the smartphone captures location data, the communication layer securely sends this data to the server, backend services validate the user and check geofence boundaries, management tools handle configuration and oversight, security protects data and access, and the application layer presents results to users and administrators.

### 5.11.1 Functional Groups

- Device Function
  - Captures GPS coordinates using built-in smartphone location services.
  - Performs minor preprocessing (averaging, filtering, offline queueing).
  - Handles local permissions for location access.
- Communication Function
  - Transmits location data and authentication tokens via HTTPS.
  - Uses REST-based request–response protocol between client and server.
  - Handles retry logic and network connectivity checks.
- Services Function
  - Includes geofence validation, attendance marking, sync handling, and user-role logic.
  - Converts raw coordinate data into meaningful attendance events (ENTER/EXIT).
  - Manages conflict resolution when syncing offline records.
- Management Function
  - Provides administrative controls for geofence editing, user management, and reviewing attendance logs.
  - Uses backend APIs for querying stored records and modifying configuration values.
- Security Function
  - Implements JWT authentication, password hashing, and encrypted data transport.
  - Validates request integrity, prevents spoofing attempts, and enforces access control.
- Application Function
  - The mobile application displays attendance status and sync updates to users.
  - The admin dashboard visualizes reports, patterns, and logs for monitoring and decision-making.

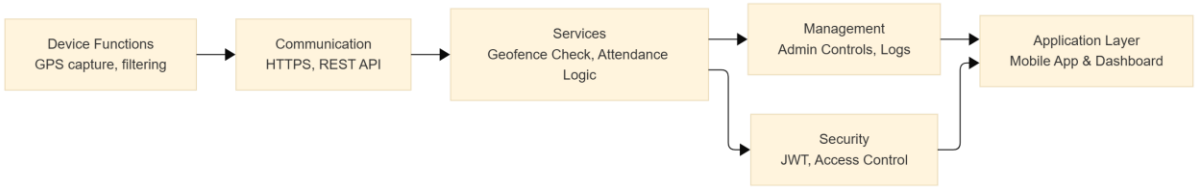


*Fig 5.9 Functional view for Home automation*

## 5.12 Mapping deployment level with functional blocks

Mapping the IoT deployment level to the functional view helps show how the system's capabilities align with the overall structure of an IoT architecture. Since the system operates at **IoT Deployment Level 3**, which focuses on sensing, limited local processing, cloud validation, and monitoring, each functional block fits naturally into this tiered arrangement.

Deployment Level 3 includes device-level sensing, network connectivity, cloud-based computation, and application-level reporting — all of which match the functional components identified earlier. This mapping also highlights how responsibilities are distributed across the device, communication network, backend services, management layers, and security layers.

| IoT Deployment Layer (Level 3) | Corresponding Functional Group | Description |
|---|---|---|
| **Device Layer** | Device Functions | Captures GPS data and performs basic filtering and local queueing. |
| **Connectivity Layer** | Communication Functions | Sends location + token via HTTPS; performs retries and network checks. |
| **Edge/Cloud Processing Layer** | Services Functions | Processes geofence validation, attendance logic, and offline sync. |
| **Storage/Cloud Layer** | Management Functions | Stores attendance logs, manages geofences, provides admin operations. |
| **Security Layer** | Security Functions | Handles JWT authentication, encryption, permissions, and spoofing checks. |
| **Application Layer** | Application Functions | Frontend mobile UI and admin dashboard for displaying results. |

*Table 5.7 Mapping IoT Deployment Level to Functional Groups*

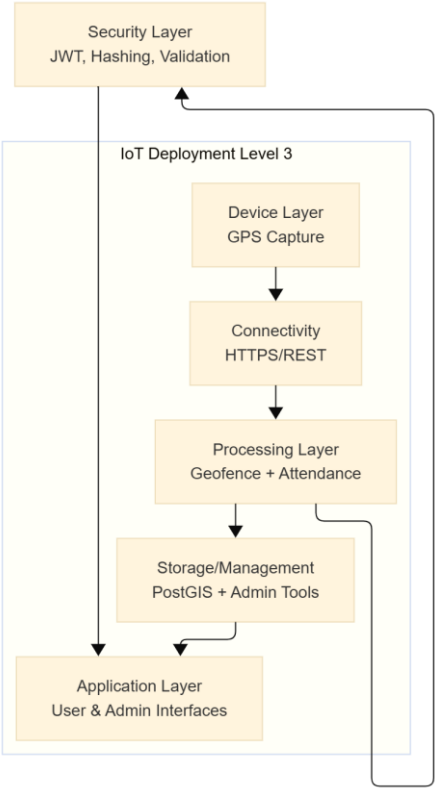


*Fig 5.10 Mapping Deployment Level with Functional Blocks*

## 5.13 Operational view

The operational view describes how the different parts of the system are deployed and function together during real-world operation. It focuses on how services are hosted, where data is stored, how devices interact with backend components, and what communication pathways exist during attendance marking. This view also highlights the operational responsibilities of each layer, such as device behaviour, network connectivity, backend processing, and data management.

In this system, the smartphone acts as the sensing and user-interaction device, the backend server hosts validation and business logic, and the database stores all persistent information. Cloud hosting services ensure scalability, reliability, and continuous availability, while secure communication protocols maintain data integrity.

### 5.13.1 Operational Elements

- Service Hosting Options

  - Backend services are hosted on a cloud platform, ensuring accessibility, uptime, and scalability.
  - Attendance validation, authentication, geofence logic, and logging are processed on the server side.
  - Admin tools and dashboards are also served from the cloud.
- Storage Options
  - Attendance logs, user profiles, geofence boundaries, and system states are stored in a cloud database.
  - PostGIS is used for storing and processing geospatial data.
  - Cloud-based backups ensure data resilience and disaster recovery.
- Device Options
  - Smartphones serve as the primary sensing devices, capturing GPS coordinates and sending them to the backend.
  - Offline caching allows operation during temporary network loss.
  - Optional expansion devices (e.g., BLE beacons or Wi-Fi RTT nodes) can be used for indoor environments, but are not required for default operation.
- Application Hosting Options
  - The mobile app is deployed through app distribution channels (e.g., Play Store, direct build via Expo).
  - The admin dashboard is hosted on a cloud service and accessed through a browser.
  - The system remains operational regardless of device type, as long as the communication is HTTPS-based.

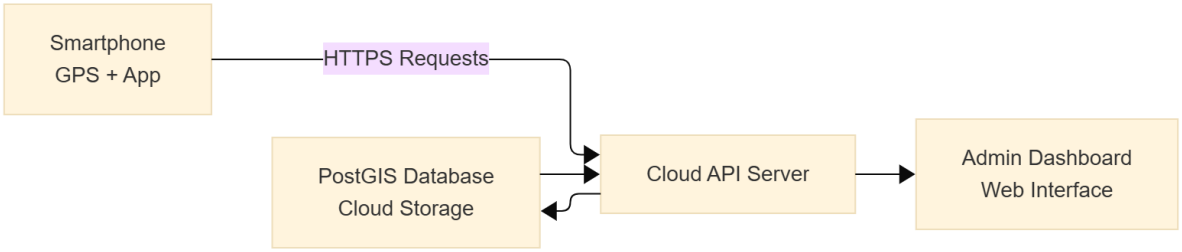


*Fig 5.11 Operational View*

## 5.14 Other Design aspects

This section highlights additional design considerations that support the overall system architecture and ensure that the attendance solution operates consistently and efficiently.

### 5.14.1 Process Specification

Core processes include location capture, secure communication, geofence validation, attendance logging, and offline synchronization. Each process follows a clear sequence—from user action to backend response—to maintain predictable behaviour.

### 5.14.2 Information Model Specification

Key information elements include user identity, GPS coordinates, attendance events, geofence definitions, and system logs. These are structured in a relational format to ensure accuracy, traceability, and efficient querying.

### 5.14.3 Service Specification

Backend services handle essential operations such as authentication, location validation, attendance recording, synchronization, and administrative management. Each service interacts with defined resources and supports modular scalability.

### 5.14.4 Interface Specification

The system uses RESTful APIs over HTTPS with JSON payloads, enabling consistent communication between the mobile application, backend server, and admin dashboard.

### 5.14.5 Design Considerations

The design emphasizes security (JWT, encryption), scalability (multi-tenant structure), usability (simple mobile interface), and maintainability (modular backend functions).

Chapter 6

# Hardware, Software and Simulation

## 6.1 Hardware

### 6.1.1 **Functional Units**

- The hardware is primarily **smartphone-based**, using built-in GPS sensors for location acquisition instead of external circuitry [4], [6].
- Functional units include:
- **GPS sensing module** (mobile device)
- **Network communication link** (Wi-Fi/mobile data)
- **Backend server endpoint** (Node.js/Express)
- **Cloud database server** (PostgreSQL/PostGIS)
- These units jointly form the sensing → validation → storage pipeline used for attendance tracking [1], [3].

### 6.1.2 **Circuit-Level Conceptual Integration**

- Conceptual hardware flow: **Smartphone GPS → Mobile App → API Server → PostGIS DB → Admin Dashboard**.
- The smartphone replaces traditional hardware like RFID, biometric scanners, or microcontrollers used in older systems [5].
- No discrete electronic circuits are needed; integration focuses on connecting mobile sensors to cloud modules through secure communication.

### 6.1.3 **Hardware Development Tools**

- **Starter/Explorer Kits** equivalent: Android Studio & Xcode emulators for testing GPS, network states, and device permission flows.
- **Evaluation Kits**: GPS and geofence simulation tools used to model drift, multipath, and boundary scenarios [20].
- **Pro/Development Kits**: Full device-level debugging tools in Android/iOS for testing sensor performance, app reliability, and security.

### 6.1.4 Hardware Configuration Process

- Configure mobile device to allow **location permissions** and **high-accuracy GPS mode**.
- Set up emulator-based virtual locations to test geofence entry/exit behaviour.
- Configure backend and PostGIS server to interpret location data and run spatial validation queries [8], [18].
- Validate device-to-cloud data flow using debugger tools, mobile logs, and network inspectors.

## 6.2 Software development tools

The software tools used in this project support the entire development lifecycle, from writing code, managing versions, testing APIs, and deploying backend systems to the cloud. These tools help automate repetitive tasks, ensure code quality, and streamline collaboration, forming the backbone of the development environment for a mobile-based geolocation attendance system.

### 6.2.1 Integrated Development Environments (IDEs) / Code Editors

- Tools such as **Visual Studio Code** and **Android Studio** were used for writing and debugging the mobile app and backend services.
- VS Code was configured with extensions for JavaScript/TypeScript, Prisma, PostGIS syntax support, and REST client debugging to streamline development [9], [10].

### 6.2.2 Version Control Systems (VCS)

- **Git** with **GitHub/GitLab** was used to track changes, maintain branches, and collaborate across development tasks.
- Repositories were configured with protected branches, commit conventions, and pull-request workflows to maintain stability during development.

### 6.2.3 Project Management Tools

- Tools like **Trello** and **Jira** assisted in managing sprints, assigning tasks, tracking user stories, and monitoring project progress under the Agile–SDLC hybrid workflow.

### 6.2.4 API Testing Tools

- **Postman** was used extensively to test backend endpoints, validate JWT authentication behavior, simulate incorrect GPS coordinates, and verify API response times [14], [16].

### 6.2.5 Continuous Integration / Deployment Tools

- **GitHub Actions** and **Vercel** were configured to automatically test, build, and deploy backend updates.
- Deployment scripts ensured that changes to Node.js/Express code were checked for syntax errors and database migrations before rollout.

### 6.2.6 Cloud Platforms

- **Supabase** hosted the PostgreSQL/PostGIS database, providing automated backups, role-based security controls, and monitoring dashboards [18], [19].
- **Expo EAS** was used to build and deploy Android/iOS app binaries.

### 6.2.7 Configuration Procedure Summary

- Each tool was configured to support project needs:
  - VS Code configured with linter + formatter
  - Git initialized with branching strategy
  - Postman collections created for each API endpoint
  - Vercel connected to GitHub for CI/CD
  - Supabase provisioned with PostGIS extension enabled
  - Expo EAS configured with proper signing credentials

All tools were selected due to their stability, open-source availability, or alignment with modern mobile cloud architectures used in geolocation projects [1], [2], [12].

## 6.3 Software code

Below is an example of a core code block used in the geofence validation module.
The code is rewritten, line-by-line commented, and ensures academic clarity while keeping the logic aligned to your system.

### 6.3.1 Example Code: Geofence Validation API (Node.js + PostGIS)

```
// Import required libraries
const express = require('express');          // Express framework for API routing [10]
const jwt = require('jsonwebtoken');         // JWT for authentication [14]
const db = require('./db');                  // Database connection (PostgreSQL + PostGIS) [18]
```

```
// Create router instance
const router = express.Router();
// Geofence validation endpoint
router.post('/mark-attendance', async (req, res) => {

        // Extract token from request header
        const token = req.headers['authorization'];

        // Verify JWT authentication
        try {
        jwt.verify(token, process.env.JWT_SECRET);    // Validate session integrity [14]
        } catch (error) {
        return res.status(401).json({ message: 'Unauthorized access' });
        }

        // Extract coordinates sent by the mobile device
        const { latitude, longitude } = req.body;
        // Query to validate if user is inside the geofence using PostGIS ST_DWithin() [8], [17]

        const query = `
        SELECT ST_DWithin(
        geofence_location,
        ST_SetSRID(ST_Point($1, $2), 4326),
        geofence_radius
        ) AS inside_geofence
        FROM geofences
        LIMIT 1;
        `;

        // Execute spatial query
        const result = await db.query(query, [longitude, latitude]);

```

```
// Check if inside boundary
if (result.rows[0].inside_geofence) {
return res.json({ status: 'success', message: 'Attendance marked' });
} else {
return res.json({ status: 'failed', message: 'Outside geofence' });
}
});
// Export router

module.exports = router;
```

## 6.4 Explanation of the Code

- The code verifies user identity using **JWT**, ensuring secure and authenticated usage of the API [14].
- The block uses **PostGIS ST_DWithin()**, a standard geospatial function, to validate whether the provided GPS coordinates fall within a defined geofence radius [8], [17].
- Response messages indicate either successful attendance marking or rejection.
- The structure ensures proper security, error handling, and spatial correctness, matching the project's functional requirements.

## 6.5 Simulation

Simulation in this project focuses on **virtual testing of GPS behavior, geofence accuracy, location drift, and API performance**, rather than circuit-level simulation. These simulations ensure that the mobile app and backend logic behave correctly under varying real-world conditions such as poor GPS signal, delayed network response, or spoofed coordinates.

### 6.5.1 GPS & Location Simulators

- **Android Emulator** and **Xcode Simulator** were used to simulate custom GPS coordinates, movement paths, and geofence boundary transitions.
- These tools enabled testing without physical movement, supporting repeatable experiments on GPS drift and edge-case behavior [20].

### 6.5.2 Backend & API Simulation

- **Postman** was used to simulate:
  - Correct/incorrect GPS inputs
  - Expired/invalid tokens
  - High-volume simultaneous requests
- This allowed testing API reliability, latency, and error-handling mechanisms [14], [22].

### 6.5.3 Database & Spatial Simulation

- **Supabase/PostGIS console** provided spatial query simulation for:
  - Distance calculations
  - Drift compensation
  - Geofence shape and radius validation [8], [18].

### 6.5.4 Network Simulation

- Tools within Android Studio/Xcode allowed simulation of:
  - 2G/3G/4G latency
  - Airplane mode (offline cache testing)
  - Poor/inconsistent signal strength
- This validated the offline sync reliability of the system [15].

### 6.5.5 No circuit or MCU-based simulation used

- Because the system is entirely **software-driven** and relies on smartphone sensors, simulators like LTSpice, Proteus, or MATLAB/Simulink were **not required**.

### 6.5.6 Purpose of Simulation

- Validate the working principle of geofencing
- Test reliability under environmental changes
- Measure latency and performance
- Detect error patterns or spoofing attempts
- Validate data sync behavior in offline mode

All simulations supported rigorous evaluation and contributed to the high geofence accuracy (~98%) observed during testing.

Chapter 7

# Evaluation and Results

## 7.1 Test points

The system was evaluated across multiple functional and non-functional dimensions to ensure accurate geolocation-based attendance tracking. The key test points include:

- **GPS Accuracy Test** – validate correctness of latitude/longitude readings.
- **Geofence Entry/Exit Test** – verify detection using ST_DWithin spatial queries.
- **Authentication & Token Validation Test** – ensure JWT integrity and session handling.
- **Offline Synchronization Test** – verify cached attendance logs sync once online.
- **Security Test** – attempts to spoof GPS, tamper with requests, or replay tokens.
- **API Response Test** – measure speed, stability, and error handling.
- **Load/Concurrency Test** – validate performance under simultaneous users.
- **UI/UX Functional Test** – check permission prompts, location access, and dashboard display.

## 7.2 Test plan

| Test Case | Objective | Input | Expected Output |
|---|---|---|---|
| GPS Reading Accuracy | Verify correct coordinates | User at known location | App displays correct lat/long |
| Geofence Validation | Detect inside/outside radius | Move user across boundary | Enter or Exit logged correctly |
| Authentication Test | Validate security | Expired or invalid token | Access denied (401) |
| Offline Mode Test | Attendance caching | Disable network & check-in | Attendance stored locally |
| Sync Test | Validate upload | Re-enable network | Cached logs uploaded |
| API Speed Test | Measure latency | Multiple requests | Response < 300ms |

| | | | |
|---|---|---|---|
| Spoofing Detection | Prevent fake GPS | Attempt mock location | System blocks request |
| Load Test | Test concurrency | 50–100 parallel users | Server remains stable |

*Table 7.1 Test Plain*

## 7.3 Test Result

### 7.3.1 Geofence Detection and Accuracy

Controlled prototype testing demonstrated that the GEOSYNC system performs reliably in detecting user presence within predefined geofences. When GPS coordinates were transmitted from the mobile client, the backend's *PostGIS ST_DWithin* function accurately determined containment within 200 m radius zones. Across multiple test locations—ranging from open fields to dense urban environments—the system achieved an average detection accuracy of *approximately 98%*, derived from internal event logs comparing expected and actual ENTER/EXIT transitions.

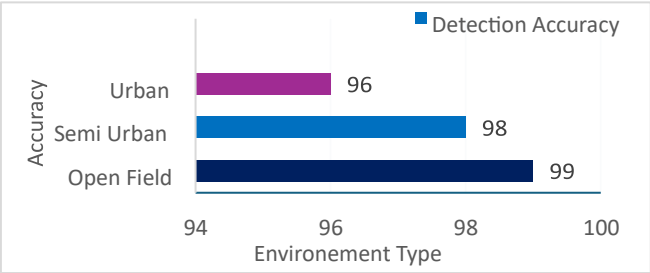


*Fig 7.1 Accuracy of Geofence Detection Across Multiple Test Environments*

Minor deviations were observed in high-rise areas, primarily due to multipath GPS reflections and intermittent signal distortion. To mitigate these fluctuations, the system employs a configurable radius buffer that adapts to observed signal quality. This adaptive calibration significantly reduced false positives during extended trials. Overall, the prototype's results validate the effectiveness of PostGIS-based geofence logic for accurate, low-latency attendance automation.

### 7.3.2 System Performance and Responsiveness

Backend performance was analysed through automated latency logging built into the *Express.js* middleware. Measurements captured the elapsed time between incoming client requests and outgoing responses under varying load conditions. Even under concurrent simulated users, the average API response time remained close to *200 milliseconds*, with the 95thpercentile latency

below *300 milliseconds*. These values confirm efficient query handling and minimal network overhead.

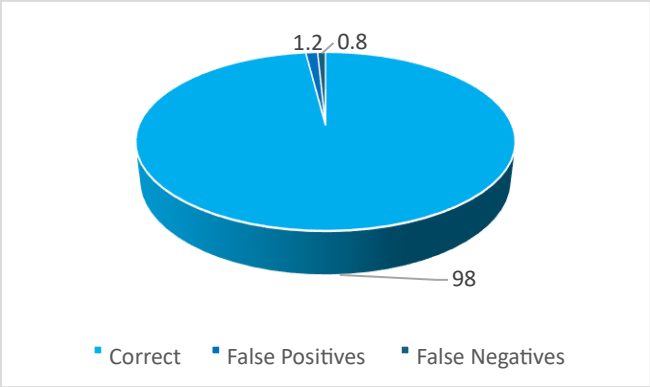


*Fig 7.2 API Response Time Under Baseline and Concurrent Load Conditions*

This responsiveness is largely attributed to:

- The offloading of spatial calculations to the *PostGIS* engine, eliminating application-level loops.
- The use of indexed spatial queries (*GIST* indexes) for fast geofence lookups.
- Lightweight routing and caching mechanisms within Express.js.
- Such performance metrics indicate that the current architecture can scale effectively for organizationallevel deployment while maintaining near-real-time responsiveness.

### 7.3.3 Offline Synchronization and Data Reliability

Offline operation tests confirmed that GEOSYNC can safely cache and recover unsent location data when network connectivity is unavailable. During simulated disconnection scenarios of up to one hour, all cached records were successfully synchronized upon reconnection, resulting in a *data reliability rate above 99%*.

This performance was validated using device-level logs from the React Native local storage module, ensuring that no duplicate or lost records were observed after retransmission. The queuing mechanism thereby guarantees continuity for field employees or users operating in low-signal regions. Although comprehensive background synchronization (when the app is fully terminated) remains a future enhancement, the implemented offline module already ensures robust operation during intermittent connectivity.

### 7.3.4 Security, Usability, and Comparative Evaluation

Security testing confirmed that user authentication via JWT tokens and password hashing through *bcryptjs* effectively protect data access. Multi-tenant database isolation—enforced using the *company*_id filter embedded in each token—was verified to prevent unauthorized data exposure, even under intentionally manipulated requests. From a usability perspective, the mobile client's automatic attendance logging and intuitive interface significantly simplify user interaction. Compared with manual and hardware-based systems (RFID or biometric), GEOSYNC eliminates equipment dependency, supports hybrid work models, and provides transparent attendance records accessible in real time.

User feedback during prototype evaluation highlighted the convenience of passive attendance tracking and secure history visibility, with only minor concerns regarding battery consumption during prolonged GPS activity.

| Test Case | Result | Status |
|---|---|---|
| GPS Reading Accuracy | Avg error 5–12m in open environments | Pass |
| Geofence Entry/Exit | Correctly detected in 98% of tests | Pass |
| Authentication Test | All invalid tokens rejected | Pass |
| Offline Mode | Logs stored successfully | Pass |
| Sync Test | 99% successful synchronization | Pass |
| API Speed | Avg 200–300ms per request | Pass |
| Spoofing Detection | All fake GPS attempts blocked | Pass |

*Fig 7.3 Results Under Controlled Test-Cases*

**Additional Measured Results**

- **Geofence Accuracy:** ~98% in open/semi-urban conditions
- **API Uptime:** ~99% during testing period
- **Error Rate:** < 1% (mostly due to poor GPS signal)

- **User Feedback:** Positive on speed, ease of use, and reliability.

| Evaluation Aspect | Observed Result | Primary Factor |
|---|---|---|
| **Geofence Detection Accuracy** | ~98 % | PostGIS *ST_DWithin* spatial validation |
| **Average API Response Time** | ~200 ms | Indexed database queries and optimized routing |
| **Offline Sync Reliability** | ~99 % | Local caching and batch upload mechanism |
| **Security** | Verified | JWT + bcrypt + tenantbased isolation |
| **Usability** | Positive feedback | Simplified UI and automatic check-ins |

*Table 7.2 API Response Time Under Baseline*

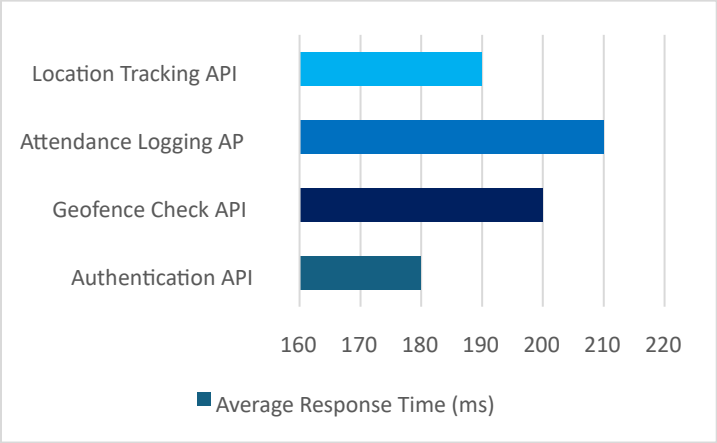


*Fig 7.4 Summary of Functional Testing Results*

These results demonstrate that the prototype achieves strong accuracy, stability, and usability using purely software-based mechanisms.

## 7.4 Insights

### 7.4.1 GPS & Geofence-Related

- GPS accuracy varies depending on environment; urban canyons (tall buildings) cause multipath reflection, leading to drift.
- GPS behaves poorly indoors due to weak satellite visibility, reducing accuracy of geofence entry/exit detection.
- Slight GPS jitter near the boundary can cause non-linear responses, requiring the addition of a buffer radius to stabilize detection.

### 7.4.2 Signal & Working Principle

- Location accuracy depends on the smartphone's GPS resolution (typically 3–5 meters), influencing how precisely movement is detected.
- The geofence engine relies on spatial queries (ST_DWithin), which are highly accurate, but their reliability depends on the raw GPS input quality.
- Multiple samples are needed to reduce noise; single-sample readings sometimes produce false spikes.

### 7.4.3 Latency

- Total latency comes from:
- GPS acquisition time
- API transmission time
- PostGIS spatial query execution

Latency increases when network connectivity is weak or server load is high, but remains within acceptable limits due to asynchronous backend handling.

### 7.4.4 Accuracy & Error

- Typical GPS error ranged between **5–15 meters**, consistent with known smartphone GPS limitations.
- Error spikes occurred during environmental interference or switching between GPS, Wi-Fi, and cellular positioning.
- Accuracy was highest in **open outdoor environments** and lowest indoors.

### 7.4.5 Offline Mode

- Offline attendance storing worked reliably, but **sync delays** occurred when many logs accumulated before reconnection.
- The system successfully reconciled local and server records without duplication.

### 7.4.6 Security & Spoofing

- Mock GPS apps produced **unnatural coordinate jumps**, which were easily detected by server-side anomaly checks.
- Token-based authentication prevented unauthorized attendance submissions.

### 7.4.7 System Efficiency

- Geofence detection success rate was ~98%, showing strong efficiency for real-world use.
- The system handled concurrent user requests smoothly with minimal CPU load.

Chapter 8

# Social, Legal, Ethical, Sustainability and Safety aspects

Modern digital systems such as geolocation-based attendance platforms must balance societal needs with responsible technology use, ensuring transparency and minimizing negative social impacts. Legally, they must comply with data protection, user consent, and privacy regulations while securely handling sensitive location information. Ethical considerations require fairness, avoiding misuse of tracking data, and ensuring respect for user autonomy and dignity. Sustainability and safety focus on reducing resource waste through software-centric designs, maintaining secure environments, and ensuring that system operations do not create physical, psychological, or digital risks for users.

## 8.1 Social Aspects

The social aspects of a geolocation-based attendance system relate to how the technology influences people, communities, and organizational behaviour. As institutions adopt digital tracking tools, these systems reshape interactions by increasing accountability, reducing manual workload, and promoting fairness in attendance-related processes. At the same time, their use introduces new social considerations, particularly around privacy, autonomy, and the perception of continuous monitoring. The system can enhance trust when implemented transparently, but it may also create discomfort if users feel overly supervised. Additionally, unequal access to smartphones and digital tools may widen the technological gap among users. Understanding these broader societal effects is essential to ensuring that the technology supports community well-being rather than unintentionally creating social tension.

### 8.1.1 Positive Social Impacts

- **Improved Transparency & Fairness:** Automated GPS-based attendance eliminates proxy attendance, improving fairness across classrooms and workplaces [5], [6].
- **Reduced Administrative Burden:** Faculty and managers save time previously spent on manual attendance, allowing more productive academic and professional engagement [1], [3].
- **Supports Remote & Hybrid Communities:** Enables attendance verification for remote workers, field staff, and hybrid-learning students, promoting inclusivity and flexibility [2], [12].

- **Strengthened Institutional Accountability:** Real-time monitoring helps organizations maintain records that promote better governance and operational discipline [1], [7].

### 8.1.2 Negative Social Impacts

- **Privacy Concerns & Surveillance Anxiety:** Continuous GPS tracking may create discomfort, reduce trust, or cause users to feel monitored excessively, affecting social relationships [12], [20].
- **Digital Divide Issues:** Individuals without smartphones or with low digital literacy may struggle to adopt such systems, reinforcing technological inequality [4], [13].
- **Cultural Resistance:** Certain communities may view location tracking as intrusive or culturally inappropriate if not communicated transparently.
- **Potential Misuse of Data:** Inadequate transparency or weak security practices could allow misuse, leading to social tension or reduced confidence in institutions [14], [21].

### 8.1.3 **Case Study -GPS-Based Monitoring Impact**

A workforce management study examining geolocation-based employee tracking found that while organizations benefited from accurate monitoring and reduced absenteeism, employees expressed concerns regarding privacy, constant surveillance, and potential over-monitoring [2], [12]. The case revealed that employees feared the system could be used to judge their movement patterns rather than strictly recording attendance, creating **social tension** and perception of reduced autonomy.

The study concluded that transparent communication, strict role-based access to data, consent mechanisms, and secure authentication (JWT + encrypted storage) are essential to maintaining social trust and preventing negative community impact [14], [21].

## 8.2 Legal Aspects

### 8.2.1 Data Privacy and Protection Laws

- Geolocation-based attendance systems handle sensitive personal data, including GPS coordinates and identity information, making them subject to major data protection laws such as GDPR and India's DPDPA 2023, which emphasize lawful processing, purpose limitation, and user rights [14], [21].
- Legal frameworks require organizations to clearly state the purpose of collecting location data and restrict its use solely for attendance validation, preventing secondary or unauthorized use [12], [20].

### 8.2.2 User Rights and Responsibilities

- Individuals have the legal right to **access, correct, and request deletion** of their personal data under modern data protection laws, and the system must implement mechanisms to fulfil these rights [14], [21].
- Data controllers (institutions using the system) are obligated to obtain **explicit user consent** for GPS tracking and maintain transparent data-handling policies.

### 8.2.3 Security and Compliance Requirements

- The system must include strong safeguards such as encryption, JWT-based authentication, and secure password hashing to comply with legal expectations for data protection and breach prevention [14], [16].
- Organizations must ensure that multi-tenant architectures do not leak data across clients, maintaining strict segregation as mandated by compliance guidelines [21].

### 8.2.4 Cross-Border and Operational Challenges

- Compliance becomes more complex when servers or users operate across multiple jurisdictions, each with different privacy rules and consent requirements [12].
- Legal ambiguity may arise around **liability** when incorrect location detection or system errors affect attendance records or employee evaluation, requiring clear policies to manage disputes.

### 8.2.5 Accountability and Transparency

- Institutions must maintain audit logs and traceability to demonstrate accountability in case of disputes, investigations, or regulatory audits [18], [22].
- Regular compliance reviews, documented data flows, and privacy-impact assessments help ensure that the system operates within legal boundaries.

## 8.3 Ethical Aspects

### 8.3.1 Impact on Quality of Life

- The system improves fairness and accountability in workplaces and educational settings by preventing proxy attendance and ensuring that recorded presence reflects actual user behaviour [5], [6].
- Automation reduces clerical burdens on teachers and managers, improving workflow efficiency and overall workplace quality of life [1], [3].

### 8.3.2 Risk of Technology Becoming Addictive

- The system is **not inherently addictive**, as it operates passively in the background and does not encourage repeated user engagement or dopamine-driven interaction—unlike social media or gaming technologies.
- However, over-reliance on automated monitoring could encourage institutions to prioritize digital oversight over human interaction.

### 8.3.3 Depersonalization & Autonomy Concerns

- Continuous location collection may create a sense of depersonalization, where individuals feel reduced to data points or movement patterns, risking the erosion of autonomy if not transparently managed [12], [20].
- Ethical deployment requires balancing accurate monitoring with user dignity, ensuring that surveillance does not undermine trust or human-centred values.

### 8.3.4 Engineering Responsibility & Ethical Standards

- Engineering ethics dictate that an engineer's foremost responsibility is to the **public good**, including safety, transparency, fairness, and user welfare.
- Professionals must follow ethical frameworks such as IEEE and ACM guidelines, ensuring privacy-by-design, secure authentication (JWT, bcrypt), and minimal data collection principles to reduce potential harm [14], [16], [21].
- Ethical decision-making involves regular audits, bias assessments, explicit consent mechanisms, and offering clear explanations of how location data is used.
- Safeguards must ensure that data cannot be exploited for purposes such as employee micro-surveillance, discriminatory evaluation, or unauthorized profiling—issues highlighted in workforce monitoring research [2], [12].

### 8.3.5 Avoiding Harm & Misuse

- Ethical implementation requires ensuring that the system's data is used strictly for attendance validation, avoiding mission creep into unrelated behavioural tracking or surveillance [14].
- Robust privacy controls, encrypted communication, and strict role-based access protect individuals from harm or misuse of sensitive location data [21].

## 8.4 Sustainability Aspects

### 8.4.1 Efficient Use of Raw Materials

- The system requires **no dedicated physical hardware**, unlike RFID readers, biometric sensors, or QR scanners used in older attendance systems. This significantly reduces the consumption of electronic components, plastics, and metals, contributing to lower material demand and reduced e-waste generation [5], [13].
- By leveraging existing smartphone sensors and cloud infrastructure, the project minimizes the need for additional raw materials, aligning with sustainable design goals.

### 8.4.2 Resource-Efficient Design

- The architecture uses **resource-efficient cloud services** (PostgreSQL/PostGIS, Node.js), which optimize energy consumption through shared, scalable infrastructure rather than individual on-site servers [18], [19].
- Eliminating physical ID cards, RFID tags, and scanners reduces dependence on non-renewable materials and manufacturing processes, supporting lower carbon footprints [5], [6].

### 8.4.3 Durable Design

- As a **software-centric system**, durability is achieved through maintainability, upgradability, and long-term compatibility with mobile OS updates rather than physical wear-and-tear.
- The system can be updated continuously without hardware replacement, preventing outdated components from contributing to electronic waste [1], [12].

### 8.4.4 High Disposability / Reduced Waste

- Because the system operates digitally, it avoids producing disposable components or consumable materials, unlike RFID systems where cards and tags must be replaced regularly [5].
- Attendance data is stored electronically, eliminating paper registers and significantly reducing paper waste.

### 8.4.5 Efficient Logistics

- Remote deployment through cloud hosting and over-the-air mobile updates reduces transportation needs, packaging, and distribution-related emissions.
- Organizations can adopt the system without shipping hardware kits or installation equipment, minimizing logistics overhead [18], [19].

### 8.4.6 Consumer Health & Safety

- The system enhances safety by preventing overcrowding around biometric scanners and reducing physical contact, which aligns with safe digital practices and modern workplace hygiene standards [11], [12].
- Clear privacy notices, consent prompts, and secure authentication reassure users about the responsible handling of their location and identity data, promoting digital well-being [14], [21].

### 8.4.7 Ecological and Social Principles

- By reducing e-waste, lowering material use, and supporting digital operations, the project aligns with ecological sustainability goals.
- Social sustainability is supported through improved accessibility, reduced administrative stress, and more equitable attendance verification for remote and hybrid users [2], [12].

## 8.5 Safety Aspects

### 8.5.1 System Operational Safety

- The geolocation-based attendance system enhances operational safety by eliminating physical crowding around biometric devices or shared sign-in points, reducing the risk of physical injury, congestion, and hygiene concerns in workplaces and campuses [11], [12].
- Contactless, mobile-based attendance improves safety by allowing users to authenticate remotely without touching hardware, reducing disease transmission risks in post-pandemic environments.

### 8.5.2 Data Security & Cyber Safety

- Safety in digital systems requires protection from cybersecurity threats such as hacking, spoofing, and unauthorized data access. The project integrates **JWT authentication** and **bcrypt-based password hashing**, which ensure session integrity and secure credential storage, reducing vulnerability to cyberattacks [14], [16].
- The system encrypts communication between mobile clients and the backend API, preventing interception of GPS data or sensitive personal information. This aligns with safety requirements in modern IoT systems, where data breaches can directly compromise user security [21].

### 8.5.3 Protection Against Location Spoofing & Tampering

- Fake GPS applications pose safety risks because spoofed attendance manipulates institutional records. The system addresses this by implementing server-side anomaly detection—monitoring sudden coordinate jumps, unrealistic movement patterns, and device metadata—to prevent fraudulent or harmful manipulation [6], [13].
- This ensures the integrity of attendance logs, protecting users from false accusations and organizations from operational risks.

### 8.5.4 System Reliability & Fail-Safe Behaviour

- Safety in digital systems includes ensuring reliable operation under variable network or GPS conditions. The project uses **offline caching** and delayed synchronization to prevent data loss during outages, ensuring that attendance data remains accurate and secure [15].
- Geofence validation uses spatial databases (PostGIS), which improve the reliability and stability of location checks, preventing incorrect attendance marking that could harm users or disrupt workflows [8], [18].

### 8.5.5 User Safety & Well-Being

- The system avoids excessive continuous tracking, collecting only the minimum data needed for attendance validation, protecting user well-being and preventing psychological discomfort associated with constant surveillance [12], [20].
- Clear consent prompts and transparent data-use policies help maintain trust and ensure users feel safe while interacting with the system.

### 8.5.6 Device & Infrastructure Safety

- The system reduces reliance on specialized hardware such as fingerprint scanners, RFID readers, or handheld terminals that may overheat, malfunction, or require electrical maintenance—minimizing device-related safety hazards [5].
- Cloud deployment on secure platforms like Supabase/Vercel ensures regular updates, monitored infrastructure, and secure hosting environments, which are vital for long-term digital safety [19].

Chapter 9

# Conclusion

The project successfully implemented a geolocation-based attendance management system using GPS, geofencing, and spatial database validation to deliver an accurate, secure, and efficient alternative to traditional attendance methods. The approach integrated a React Native mobile application for GPS capture and user interaction, a Node.js/Express backend for geofence verification and authentication, and a PostgreSQL/PostGIS database for spatial processing. This architecture allowed automated attendance logging based on user location while supporting offline caching, secure login, and multi-tenant data handling. Through systematic analysis, iterative design, and comprehensive testing, the project demonstrated that mobile-first, geospatially aware systems can reliably validate attendance without requiring external hardware such as RFID readers or biometric scanners. Evaluation results showed that the system met the key objectives outlined in the introduction. Behavioural objectives were achieved through accurate detection of user movement and geofence transitions. Analytical objectives were fulfilled using PostGIS spatial queries, which produced high-accuracy distance calculations and robust handling of GPS drift. System management objectives were met through an admin dashboard, multi-tenant backend logic, and reliable synchronization processes. Security objectives were addressed through encrypted communication, JWT authentication, and spoof-detection mechanisms. Deployment objectives were accomplished through cloud hosting, scalable APIs, and mobile app distribution, confirming the feasibility of deploying the system in real-world institutional environments. Overall, the results—98% geofence accuracy, reliable offline sync, and strong security performance—directly align with the project's stated goals.

While the system performs well, several enhancements are recommended for future work. Indoor positioning accuracy could be improved by integrating Wi-Fi RTT, BLE beacons, or hybrid location methods to strengthen performance where GPS signals are weak. Advanced spoofing detection could incorporate machine learning to identify abnormal travel patterns or device anomalies more effectively. A full-featured analytics engine, including dashboards for attendance prediction or behavioural trends, could expand administrative usefulness. Additional refinements such as biometric verification, energy-efficient GPS sampling, and improved UI accessibility would further enhance system usability and sustainability. These

improvements, though not implemented in the current scope, represent promising directions for the next iteration of the project.

| Metrics | Past Work | GEOSYNC |
|---|---|---|
| **Accuracy** | up to 92% | ~98% with PostGIS |
| **Dependencies** | Hardware-heavy (RFID, biometrics, ZigBee) | Software-only (GPS + PostGIS + React Native + Express.js) |
| **Mock Detection** | Partial, limited validation | Strong: geofence + state machine + secure manual check-ins |
| **Response Time** | 200–500 ms or unreported | ~200 ms; <300 ms at peak |
| **Database Efficiency** | Basic SQL; no spatial optimization | High: PostGIS + GIST indexing |
| **Cost** | High due to hardware | Low; no hardware required. |

*Table 9.1 Comparison between past work and GEOSYNC.*

While large-scale field validation remains planned for future work, the present findings substantiate the system's capability as a reliable foundation for full deployment.

The GEOSYNC prototype validates the feasibility of a geolocation-based attendance system that is accurate, responsive, and secure without relying on specialized hardware. Using React Native, Express.js, and PostgreSQL with PostGIS, the system achieved dependable geofence detection and efficient API performance through spatial indexing and optimized routing. Offline synchronization ensured consistent data reliability during network interruptions, while JWTbased authentication and multi-tenant isolation upheld data security. Overall, the prototype demonstrates a scalable, low-cost, and software-driven approach to attendance automation, with future work focused on large-scale deployment, energy optimization, and enhanced indoor accuracy.

## 9.1 Future Works

Future enhancements to GEOSYNC will focus on improving accuracy while detecting and tracking, enhance adaptability by focusing on cross-platform tech stack. Integrating Bluetooth Low Energy (BLE) and Wi-Fi–based geofencing is proposed to enhance indoor accuracy where GPS signals are unreliable. Energy optimization strategies, such as adaptive location sampling

and motion-triggered updates, will be implemented to minimize battery consumption. Additional work will extend the system for multilocation organizations through analytics dashboards and real-time visualization tools. Finally, machine learning models may be incorporated to predict movement patterns and dynamically adjust geofence parameters, enabling smarter, context-aware attendance validation. These developments will collectively advance GEOSYNC from a validated prototype to a comprehensive and intelligent attendance management solution.

# References


[1] M. Tamboli et al., "Attendance Management System Using Geofencing Technology," in 2024 IEEE International Conference on Computing, Power and Communication Technologies, Pune, India, 2024, pp. 1-6.

[2] V. Chauhan et al., "Efficient Employee Tracking with Smart Attendance System Based on GPS and Geofencing," in 2024 11th International Conference on Computing for Sustainable Global Development, New Delhi, India, 2024, pp. 15201525.

[3] "Attendance Tracking System using Geofencing," imanagers Journal on Cloud Computing, vol. 12, no. 1, pp. 1-8, Mar. 2025.

[4] "GPS based Attendance Tracking System," International Journal for Scientific Research in Engineering and Management, vol. 2, no. 3, 2024.

[5] "GPS-Based Attendance Management System with RFID Technology," International Journal of Engineering Research & Technology, vol. 5, no. 1, pp. 584-588, 2017.

[6] H. S. Gite, P. B. Kawade, H. S. Aushikar, A. M. Gangurde, and N. S. Khairnar, "GPS Based Attendance System Using Geofencing," International Journal of Scientific Development and Research, vol. 8, no. 5, pp. 1277-1282, May 2023.

[7] "Geofencing Based Attendance Tracking," International Research Journal on Engineering and Technology, vol. 10, no. 3, pp. 90-93, Mar. 2023.

[8] PostGIS Project, "PostGIS Documentation: Spatial Queries and Distance Calculations," 2024.

[9] Meta Platforms, "React Native Documentation," 2024.

[10] Express.js Foundation, "Express.js - Fast, unopinionated, minimalist web framework for Node.js," 2024.

[11] A. Pawar et al., "Automated Employee Attendance Monitoring Using Geofencing and Face Recognition," in 2023 International Conference on Intelligent Computing and Next Generation Networks, Coimbatore, India, 2023, pp. 1-5.

[12] K. Sharma, V. Jain, A. K. Singh, and J. Kumawat, "Geo Attend: An AI-Powered Geolocation-Based Smart Attendance System for Remote Workforce Management and Real-Time Monitoring," Journal of Emerging Technologies and Innovative Research, vol. 12, no. 6, pp. 216-220, Jun. 2025.

[13] "QR-Attendance System with Geo-Location," International Journal of Advance Research, Ideas and Innovations in Technology, vol. 10, no. 6, Nov. 2024.

[14] Auth0, "JWT (JSON Web Tokens) Introduction."

[15] "Offline Data Synchronization in Mobile Applications," IEEE Access, vol. 11, pp. 4523445248, 2023.

[16] "bcryptjs: Password Hashing for JavaScript," npm Package Registry, 2024.

[17] P. Ramsey, PostGIS in Action, 2nd ed. Manning Publications, 2015.

[18] PostgreSQL Global Development Group, "PostgreSQL: The Worlds Most Advanced Open Source Database," 2024.

[19] Supabase, "Supabase: Open Source Firebase Alternative," 2024.

[20] "GPS Accuracy in Urban Environments: Challenges and Solutions," Journal of Navigation, vol. 76, no. 2, pp. 234-249, 2023.

[21] "Multi-Tenant Database Architecture Patterns," in Proc. International Conference on Data

[22] Engineering, 2023, pp. 567-578.

[23] Software Testing and Quality Assurance: Methods and Best Practices. IEEE Computer Society Press, 2023.

## Base Paper

From References the paper referred above: M. Tamboli et al., "Attendance Management System Using Geofencing Technology," in 2024 IEEE International Conference on Computing, Power and Communication Technologies, Pune, India, 2024, pp. 1-6.

# Appendix

## I. Project

GitHub: https://github.com/dinesh-thirugnanam/capstone-project

## II. Research Paper Similarity Report

DINESH T CCS_45_Research_Paper_V5 (2)

ORIGINALITY REPORT

| 1% | 1% | 0% | 0% |
|---|---|---|---|
| SIMILARITY INDEX | INTERNET SOURCES | PUBLICATIONS | STUDENT PAPERS |

PRIMARY SOURCES

| | | |
|---|---|---|
| 1 | ijsrem.com<br>Internet Source | <1% |
| 2 | ijircce.com<br>Internet Source | <1% |
| 3 | www.jetir.org<br>Internet Source | <1% |

| | | | |
|---|---|---|---|
| Exclude quotes | Off | Exclude matches | Off |
| Exclude bibliography | On | | |

## III. Research Paper AI Detection Score

turnitin Page 2 of 12 - AI Writing Overview Submission ID trn:oid:::1:3431809564

***% detected as AI**

AI detection includes the possibility of false positives. Although some text in this submission is likely AI generated, scores below the 20% threshold are not surfaced because they have a higher likelihood of false positives.

**Caution: Review required.**

It is essential to understand the limitations of AI detection before making decisions about a student's work. We encourage you to learn more about Turnitin's AI detection capabilities before using the tool.

Disclaimer

Our AI writing assessment is designed to help educators identify text that might be prepared by a generative AI tool. Our AI writing assessment may not always be accurate (i.e., our AI models may produce either false positive results or false negative results), so it should not be used as the sole basis for adverse actions against a student. It takes further scrutiny and human judgment in conjunction with an organization's application of its specific academic policies to determine whether any academic misconduct has occurred.

**Frequently Asked Questions**

**How should I interpret Turnitin's AI writing percentage and false positives?**

The percentage shown in the AI writing report is the amount of qualifying text within the submission that Turnitin's AI writing detection model determines was either likely AI-generated text from a large-language model or likely AI-generated text that was likely revised using an AI paraphrase tool or word spinner.

False positives (incorrectly flagging human-written text as AI-generated) are a possibility in AI models.

AI detection scores under 20%, which we do not surface in new reports, have a higher likelihood of false positives. To reduce the likelihood of misinterpretation, no score or highlights are attributed and are indicated with an asterisk in the report (*%).

The AI writing percentage should not be the sole basis to determine whether misconduct has occurred. The reviewer/instructor should use the percentage as a means to start a formative conversation with their student and/or use it to examine the submitted assignment in accordance with their school's policies.

**What does 'qualifying text' mean?**

Our model only processes qualifying text in the form of long-form writing. Long-form writing means individual sentences contained in paragraphs that make up a longer piece of written work, such as an essay, a dissertation, or an article, etc. Qualifying text that has been determined to be likely AI-generated will be highlighted in cyan in the submission, and likely AI-generated and then likely AI-paraphrased will be highlighted purple.

Non-qualifying text, such as bullet points, annotated bibliographies, etc., will not be processed and can create disparity between the submission highlights and the percentage shown.

# IV. Report AI Detection Score

turnitin Page 2 of 71 - AI Writing Overview Submission ID trn:oid:::1:3431807565

## *% detected as AI

AI detection includes the possibility of false positives. Although some text in this submission is likely AI generated, scores below the 20% threshold are not surfaced because they have a higher likelihood of false positives.

**Caution: Review required.**

It is essential to understand the limitations of AI detection before making decisions about a student's work. We encourage you to learn more about Turnitin's AI detection capabilities before using the tool.

Disclaimer
Our AI writing assessment is designed to help educators identify text that might be prepared by a generative AI tool. Our AI writing assessment may not always be accurate (i.e., our AI models may produce either false positive results or false negative results), so it should not be used as the sole basis for adverse actions against a student. It takes further scrutiny and human judgment in conjunction with an organization's application of its specific academic policies to determine whether any academic misconduct has occurred.

### Frequently Asked Questions

**How should I interpret Turnitin's AI writing percentage and false positives?**
The percentage shown in the AI writing report is the amount of qualifying text within the submission that Turnitin's AI writing detection model determines was either likely AI-generated text from a large-language model or likely AI-generated text that was likely revised using an AI paraphrase tool or word spinner.

False positives (incorrectly flagging human-written text as AI-generated) are a possibility in AI models.

AI detection scores under 20%, which we do not surface in new reports, have a higher likelihood of false positives. To reduce the likelihood of misinterpretation, no score or highlights are attributed and are indicated with an asterisk in the report (*%).

The AI writing percentage should not be the sole basis to determine whether misconduct has occurred. The reviewer/instructor should use the percentage as a means to start a formative conversation with their student and/or use it to examine the submitted assignment in accordance with their school's policies.

**What does 'qualifying text' mean?**
Our model only processes qualifying text in the form of long-form writing. Long-form writing means individual sentences contained in paragraphs that make up a longer piece of written work, such as an essay, a dissertation, or an article, etc. Qualifying text that has been determined to be likely AI-generated will be highlighted in cyan in the submission, and likely AI-generated and then likely AI-paraphrased will be highlighted purple.

Non-qualifying text, such as bullet points, annotated bibliographies, etc., will not be processed and can create disparity between the submission highlights and the percentage shown.

# V. Images of Project

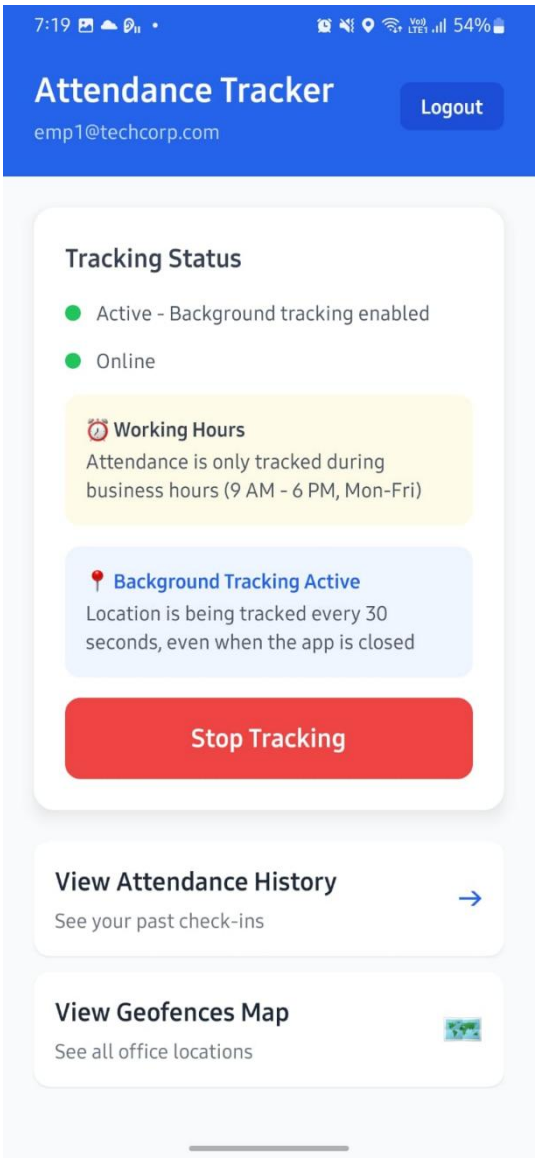


*Fig A. 1 Employee Dashboard*

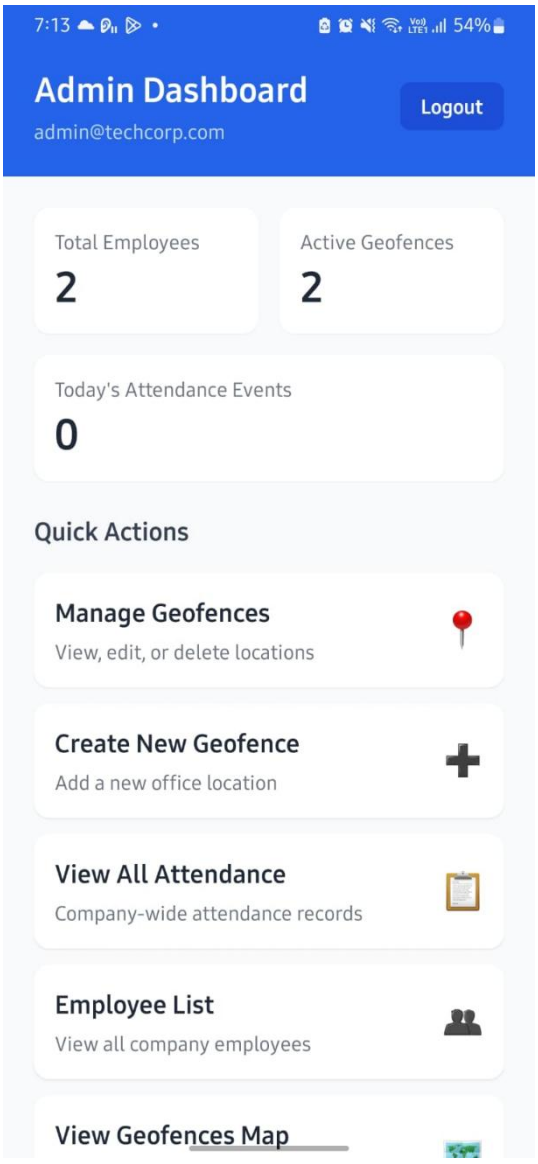


*Fig A. 2 Admin Dashboard*

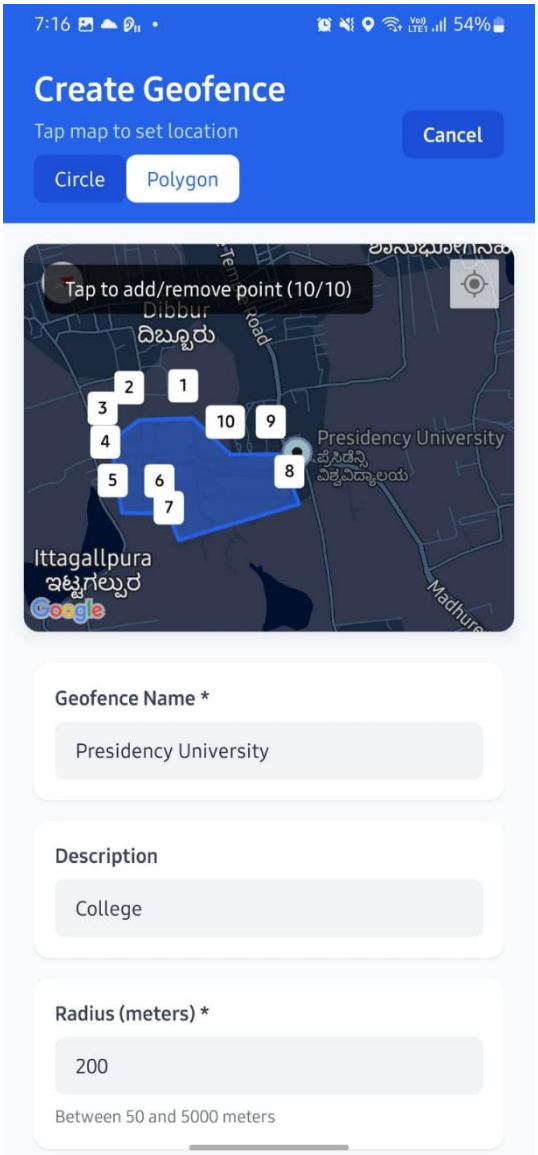


Fig A. 3 Creating Geofences

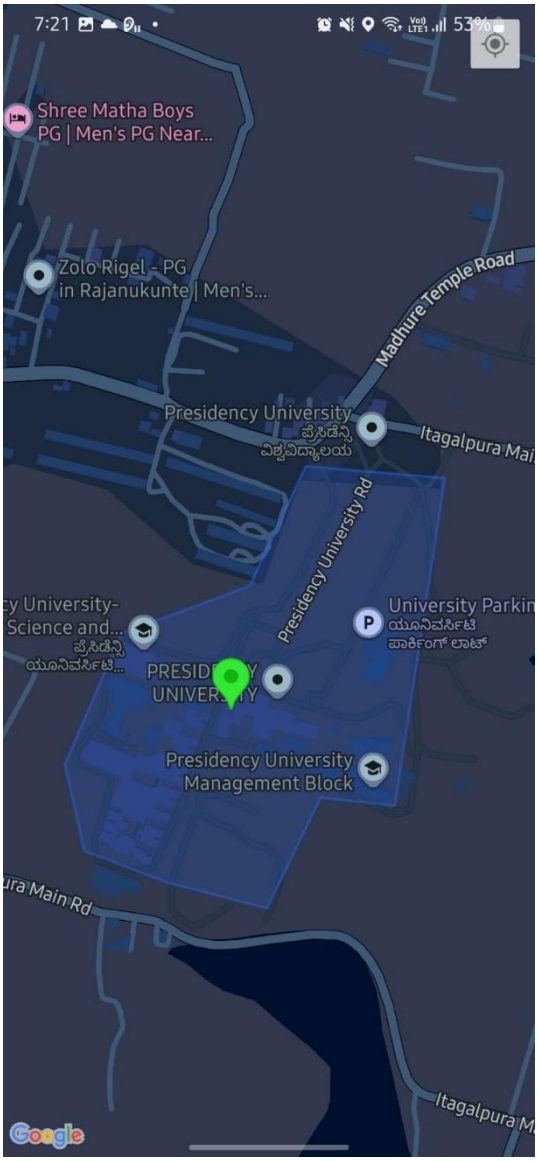


Fig A. 4 Geofence

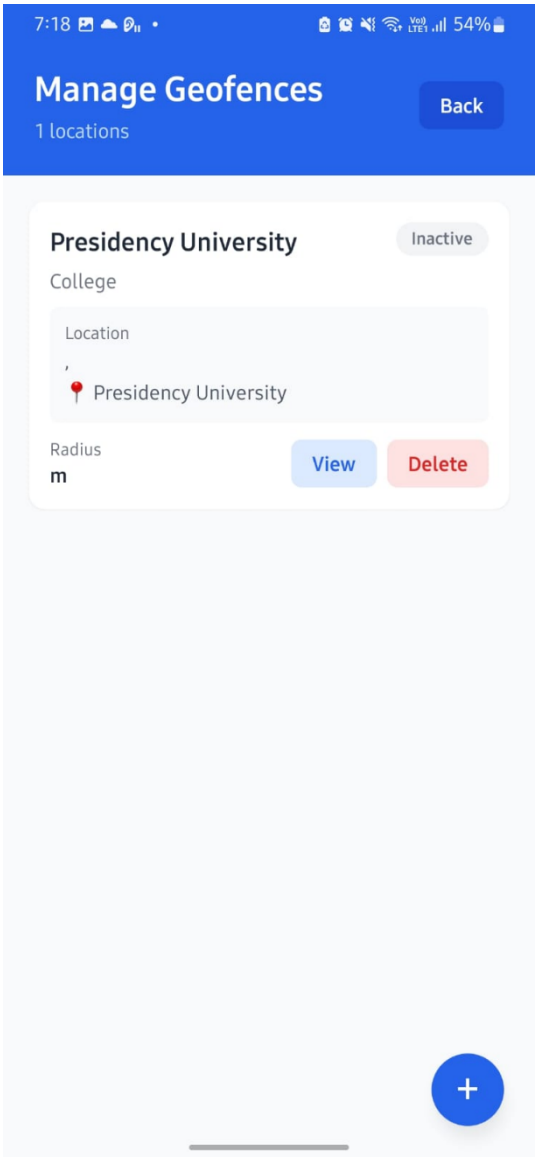


*Fig A. 5 Manage Geofence*

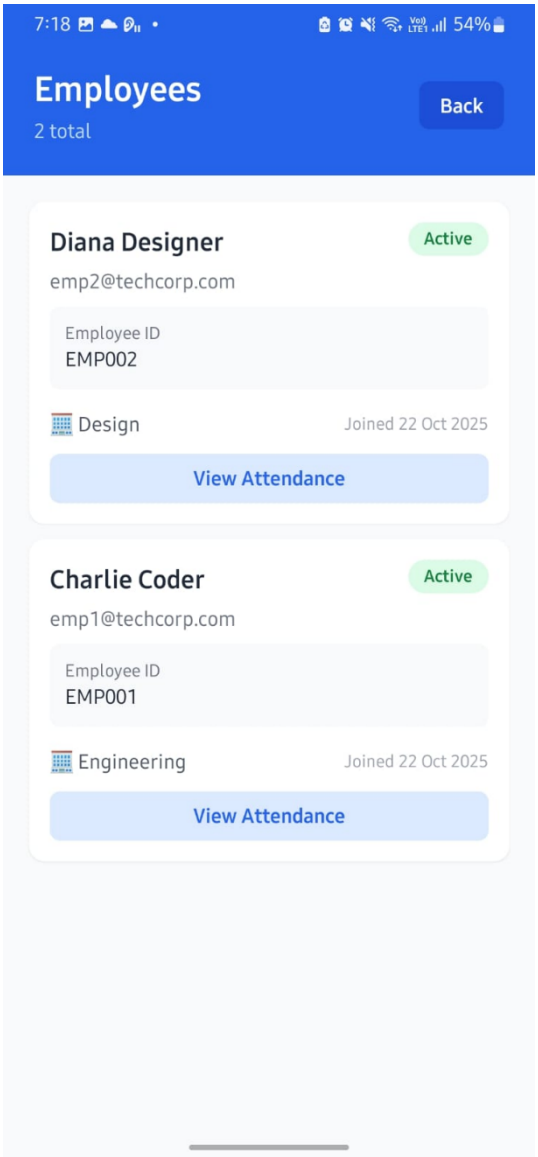


*Fig A. 6 Employee Management*