VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"Jnana Sangama", Belagavi-590 018

A Mini - Project Report

On

"QR ATTENDENCE SYSTEM"

Submitted in partial fulfillment of the requirements for the **MINI PROJECT (BCD586)** course of the 5th semester

Bachelor of Engineering

In

Computer Science & Engineering (DATA SCIENCE)

Submitted by

Ms. Sneha K. S

(4AI23CD052)

Under the guidance of

Mrs. Shilpa K. V B.E., MTech.

Assistant Professor

Department of CS&E (DATA SCIENCE)

Adichunchanagiri Institute of Technology

CHIKKAMAGALURU - 577102

2025-26

ADHICHUNCHANAGIRI INSTITUTE OF TECHNOLOGY

Jyothi Nagar, Chikkamagaluru-577102

DEPARTMENT OF CS&E (DATA SCIENCE)

CERTIFICATE

This is to certify that the Mini project work entitled **"QR ATTENDENCE SYSTEM"** is a bonafied work carried out by **Ms. Sneha K. S (4AI23CD052)** in partial fulfillment for the **Mini Project (BCS586)** course of 5th semester Bachelor of Engineering in **Computer Science and Engineering (Data Science)** of the Visvesvaraya Technological University, Belagavi during the academic year **2025-2026**. It is certified that all corrections and suggestions indicated for Internal Assessment have been incorporated in the report deposited in the department library. The Mini project report has been approved as it satisfies the academic requirements in respect of Project Work prescribed for the said Degree.

Signature of the Guide
Mrs. Shilpa K. V B.E., MTech
Assistant Professor

Signature of Coordinator
Mrs. Shilpa K V. B.E., MTech
Assistant Professor

Signature of the HOD
Dr. Adarsh M J B.E., MTech., Ph.D.
Associate Professor and Head

ABSTRACT

The QR Code Based Student Attendance System is an efficient and paperless solution designed to replace traditional manual attendance registers in educational institutions. The system automates the process of recording student presence using dynamic Quick Response (QR) codes generated for each class session. This mini-project focuses on three essential operations: generating a unique, time-sensitive QR code, allowing students to scan the code via a mobile application to mark attendance, and providing real-time attendance reports to faculty and administrators. The assistant simplifies the attendance workflow by reducing manual errors, eliminating proxy attendance, and offering an intuitive and efficient experience.

The backend utilizes (Python Django or Flask), integrated with libraries for QR code generation and data handling, while the database (MySQL or PostgreSQL) securely stores student, class, and attendance data. The frontend, accessible via a mobile or web interface, offers a user-friendly design featuring a camera-based QR scanning system for students and a dashboard for faculty. Communication is achieved through a secure RESTful API, ensuring reliable data transfer. The project highlights the potential of automated identification technology in streamlining administrative tasks and improving data accuracy in education.

ACKNOWLEDGEMENT

We express our humble pranamas to his holiness **Divine Soul Parama Poojya Jagadguru Padmabushana Sri Sri Sri Dr. Balagangadharanatha MahaSwamiji and Parama Poojya Jagadguru Sri Sri Sri Dr. Nirmalanandanatha Maha Swamiji Pontiff, Sri Adichunchanagiri Maha Samsthana Matt and Sri Sri Gunanatha Swamiji,** Chikkamagaluru branch, Sringeri who have showered their blessings on us for framing our career successfully..

The completion of any project involves the efforts of many people. We have been lucky enough to have received a lot of help and support from all quarters during the making of this project, so with gratitude, we take this opportunity to acknowledge all those whose guidance and encouragement helped us emerge successful.

We express our gratitude to **Dr. C K Subbaraya**, Director, Adichunchanagiri Institute of Technology.

We express our sincere thanks to our beloved principal**, Dr. C T Jayadeva** for having supported us in our academic endeavors.

We are thankful to the resourceful guidance, timely assistance and graceful gesture of our guide **Mrs. Shilpa K V,** Asst. Professor, Department of CS&E (DATA SCIENCE), who has helped us in every aspect of our project work.

We thank our project coordinator **Mrs. Shilpa K V** Asst. Professor, Department of CS&E (DATA SCIENCE), for her lively correspondence and assistance in carrying on with this project.

We are also indebted to **Dr. Adarsh M J,** HOD of CS&E (DATA SCIENCE) Department, for the facilities and support extended towards us.

We would be very pleased to express our heart full thanks to all the teaching and non-teaching staff of CS&E (DATA SCIENCE) Department and our friends who have rendered their help, motivation and support.

Sneha K. S (4AI23CD052)

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SNAPSHOTS

Chapter 1

INTRODUCTION

1.1 Background

- **Context**: Student attendance tracking is a fundamental administrative task in educational institutions. Traditional methods, such as paper-based registers or manual roll calls, are highly prone to human error, time-consuming, and susceptible to proxy attendance. A QR attendance system is a modern solution that uses digital technology to streamline this process. It involves generating a unique Quick Response (QR) code for a specific class session, which students scan using their smartphones to record their presence.
- **Problem**: This system addresses the inefficiencies and inaccuracies of manual attendance. It aims to provide an intelligent, automated, and accurate method for capturing attendance data, thereby saving valuable lecture time, reducing administrative overhead, and improving the overall integrity of the attendance record.
- **Opportunity**: By leveraging mobile devices and QR code technology, the system significantly simplifies user interaction and automates data logging. It improves accessibility for reporting and analysis by instantly digitizing the attendance record, offering a foundation for real-time monitoring and data-driven insights.

1.2 Problem Statement

1.2.1 Overview: Current student attendance methods are heavily dependent on manual input and verification, which results in significant time wastage during lectures and compromised data accuracy due to errors and malpractices**.**

1.2.2 Specific Issues:

- **Time-Consuming Process:** Manual roll calls consume up to 10-15 minutes of lecture time, reducing instructional hours.
- **High Error Rate:** Human mistakes during data entry or transcription from paper registers to digital records are common.
- **Proxy Attendance Risk:** Traditional methods offer minimal defense against one student marking attendance for another.
- **Delayed Reporting:** Generating summary reports for students or management requires manual compilation, leading to delays.

1.3 Objective of the System

The primary objective of the QR Code Based Student Attendance System is to provide an efficient, accurate, and paperless solution for managing class attendance. It is designed to automate the recording process, enhance data integrity, and provide immediate reporting capabilities.

1.3.1 Key Goals

1. **Efficiency:**
   - Minimize the time required for attendance marking to less than one minute per class.
   - Automate the creation of daily and cumulative attendance reports.
2. **Accuracy:**
   - Ensure that attendance records are accurately logged with timestamps and class details.
   - Eliminate manual data entry errors.
3. **Security and Integrity:**
   - Prevent proxy attendance through time-sensitive and unique QR code generation.
   - Securely store attendance data on a centralized server.
4. **User Experience:**
   - Provide a simple, intuitive mobile interface for students to scan the QR code.
   - Offer an easy-to-use faculty dashboard for QR code generation and report viewing.

1.4 Significance of the System: QR Attendance System

1. **Efficiency:** By automating the process using scannable QR codes, the system significantly enhances productivity and saves instructional time. Teachers can begin lessons immediately, and administrators can access reports instantly.
2. **Accuracy:** The system's reliance on digital capture ensures precise recording of attendance with a timestamp, drastically reducing human errors associated with manual registers. The data is validated upon entry, ensuring high data quality.
3. **Real-Time Feedback:** Faculty and students receive instant confirmation of marked attendance. Management can view real-time statistics, enabling prompt intervention for attendance

4. **Security:** The use of dynamic (changing) QR codes and a validation window prevents unauthorized or belated attendance marking, effectively combating proxy attendance.
5. **Cost-Effectiveness:** the system eliminates the need for paper registers, printing, and the labor involved in manual data compilation, leading to long-term operational savings for the institution.

1.5 Scope of the Project: QR Attendance System

1.5.1 In Scope

1. **QR Code Generation**: Implement a robust backend feature to generate a unique, session-specific QR code containing class ID, subject, and a short validity period (e.g., 60 seconds)
2. **Attendance Marking:** Develop a student-facing mobile interface that uses the device's camera to scan the QR code and transmit the student ID and timestamp to the server.
3. **User Roles and Authentication:** Support two primary user roles**:**
   - **Faculty:** Can log in, select a class/subject, generate a QR code, and view immediate attendance logs.
   - **Student:** Can log in, scan the QR code, and view their personal attendance history.
4. **Reporting:** Provide basic, filterable reports for faculty, including
   - Daily attendance list
   - Cumulative percentage attendance by student
5. **Security Measures:** implement token-based authentication (JWT) for secure API communication and ensure the QR code can only be scanned once per student per session.
6. **Technology Stack:** The project will be developed using Python for the backend, a modern JavaScript framework for the frontend, and an SQL database for persistence.

1.5.2 Out of Scope

1. Advanced Anti-Proxy Features: Exclude features like GPS location verification, facial recognition, or Bluetooth-based proximity checks in the current version

2. Integration with ERP/LMS: The project will function as a standalone system and will not be integrated with the institution's existing Enterprise Resource Planning (ERP) or Learning Management System (LMS).
3. Complex Data Analytics: The project will focus on recording and basic reporting. Advanced features like predictive attendance analytics or machine learning-based dropout prediction are excluded.
4. Offline Functionality: The system requires an active internet connection on both the faculty and student sides for generating, scanning, and validating the QR code.

1.6 Methodology

1.6.1 Approach: The QR Attendance System will be developed using the Agile Development Methodology with an emphasis on iterative and incremental delivery. The system is architected as a full-stack application leveraging modern web and mobile technologies**.**

- **Technology Stack**
  - **Frontend**: A cross-platform mobile application (or responsive web app) built using frameworks like React Native or Flutter to ensure broad device compatibility for the student scanning interface.
  - **Backend**: Python (Django/Flask) for building a robust RESTful API to handle business logic, QR code generation, and database interaction.
  - **Database:** PostgreSQL or MySQL for secure, reliable storage of relational data (Users, Classes, Attendance Logs).
    - qrcode or similar Python library for generating QR codes.
    - A cryptographically secure library for generating unique tokens within the QR code payload.
    - OpenCV (optional for advanced camera-based features, but primarily relying on mobile's native camera API).

1.6.2 Agile Development: The project will follow an Agile methodology to ensure flexibility and iterative improvements throughout development.

Key steps include;

- **Requirement Analysis**: Identify core functionalities based on user needs

- **Iterative Development:** Implement features in modular units (e.g., separate modules for User Authentication, QR Generation, and Attendance Logging)..
- **User Feedback Cycles:** Incorporate feedback from a pilot group of faculty and students at each iteration to improve UI/UX, scanning speed, and report clarity.
- **Incremental Feature Integration:** Begin with basic functionalities, such as such as dynamic QR codes, gradually add more advanced features, such as real-time error feedback and offline capabilities.

1.6.3 Testing

The system will undergo rigorous testing to ensure functionality, accuracy, and reliability across different devices and network conditions.

1. Unit Testing

- Test individual modules for correct execution (e.g. generate_qr_token(class_id) and validate_scan(token, student_id).
- Validate that API endpoints handle correct data types and return appropriate HTTP status codes.

2. Integration Testing

- Ensure smooth interaction between the mobile scanning app, the QR code generation backend, and the database
- Verify that the end-to-end flow, from QR generation to successful attendance logging, works seamlessly.

3. User Acceptance Testing (UAT)

- Conduct tests with real users (students and faculty) to ensure the system meets operational expectations.
- Assess the system's ability to handle high concurrency, such as multiple students scanning the QR code simultaneously.

4. Performance Testing

- Measure response times for QR code generation and attendance logging, ensuring they remain fast even with a large number of stored records

1.7 Target Audience

1.7.1 Technically Advanced Users

- **Description**: Users with advanced technical knowledge may utilize the system for customized tasks, such as setting specific times or controlling playback preferences.
- **Role**: May provide feedback for improvements and assist in troubleshooting or enhancing features based on their experience.

1.7.2 Developers/Administrators

- **Description**: Developers may contribute to enhancing the system's capabilities or troubleshoot bugs. Administrators could oversee the maintenance of the system and manage updates for new functionalities or features.
- **Role**: Responsible for managing, enhancing, and debugging the system, ensuring it functions properly across devices.

1.8 Overview of the Report

This report is structured into several chapters that include details of the QR Code Based Student Attendance System**.**

The following chapters are outlined below

- **Chapter 1: Introduction**

  This chapter describes about the project

- **Chapter 2: System Design**

  This chapter describes the architecture and design of the system.

- **Chapter 3: Implementation**

  This chapter discusses the system's development and the technologies used.

- **Chapter 4: Testing and Validation**

  This chapter details the testing process and results.

- **Chapter 5: Results and Discussions**

  This chapter presents the results obtained and discusses the limitations.

- **Chapter 6: Conclusion and Future Enhancement**

  This chapter summarizes the project and suggests future improvements.

Chapter 2

SYSTEM DESIGN

2.1 System Architecture

2.1.1 High-Level Overview: The QR Code Based Student Attendance System follows a three-tier, client-server architecture. This modular design ensures separation of concerns, scalability, and efficiency. The tiers are: Presentation Tier (Mobile App/Web Dashboard), Application Tier (REST API Backend), and Data Tier (Database).

- **Presentation Tier:** The student scanning app and the faculty/admin web dashboard.
- **Application Tie:** The core business logic, handling QR generation, validation, and attendance logging.

2.1.2 Architecture Diagram: The architecture diagram for QR attendance system *can* be simplified as follows:

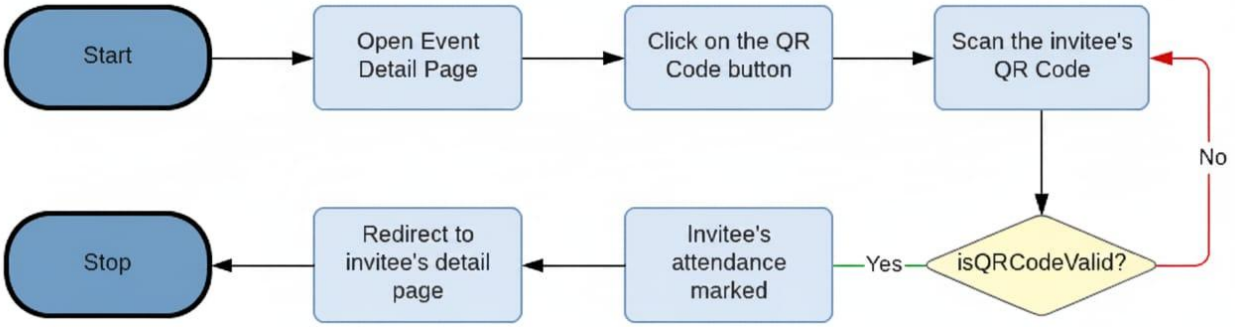


Figure 2.1: Architecture Diagram of QR attendance system

2.1.3 Components:

Frontend (Presentation Tier): The interface for users. Students use a camera-enabled app/web view for scanning. Faculty/Admin use a dashboard for class selection and reporting.

Backend (Application Tier): QR Code Generation Service: Creates a dynamically changing token (containing class/session ID, a unique nonce, and expiration time) and converts it into a scannable QR image

- **Validation Service**: Processes the scanned token, checks its authenticity (signature), validates its expiration time, and ensures the student has not already marked attendance for that session.

- **Attendance Logging Service:** Records the Student ID, Class ID, Subject, Date, and Time into the database.

2.2 Module Design

The *Qr attendance system* is divided into several functional modules, each handling specific tasks:

2.2.1 User Management Module

- **Purpose**: To manage user registration, authentication, and roles (Student, Faculty, Admin).

2.2.2 Command Processing Module

- **Purpose**: Processes voice commands, and converts speech into actionable tasks.
- **Components**:
  - Web kit Speech Recognition
  - Command parsing (e.g., "Set alarm for 13:30").

2.2.3 Attendance Module

- **Purpose**: To handle the core attendance marking and validation process.
  - scan_and_validate(token, student_id): The endpoint called by the student app to check the QR token and log attendance.
  - record_attendance(): Writes the valid attendance entry into the database.

2.2.4 Reporting Module

- **Purpose**: Provides faculty and administration with necessary attendance data.
  - **generate_daily_report():** Retrieves and formats attendance data for a specific class and date.
  - **calculate_cumulative_percentage():** Computes the overall percentage for a student across a subject/semester.
  - **Function**: Display class schedules, options to generate a QR code, and real-time attendance status.

2.3 Technology Stack

The technologies selected were chosen for their robustness, scalability, and ease of development**.**

Frontend: Framework: React Native or Flutter for cross-platform mobile development (for student app) and React/Vue for the Faculty/Admin Web Dashboard.

- Languages: JavaScript/TypeScript (or Dart for Flutter).
- Key Libraries: Camera/QR Scanning API wrappers for mobile.

Backend: Python (Django or Flask). Django is preferred for its built-in ORM and security features.

- Language: Python 3.x.
- Key Libraries:
- pyqrcode or qrcode: For generating QR code images.
- pyjwt: For secure token generation and validation.
- Django REST Framework: For building the robust API endpoints.

2.4.3 Data Tier:

Database: PostgreSQL (or MySQL). PostgreSQL is preferred for its reliability and features for complex query

Chapter 3

IMPLEMENTATION

The implementation phase involved setting up the three-tier architecture, developing the core QR generation and validation logic on the backend, creating the user interfaces, and structuring the database.

3.1 Backend Implementation

The backend, built using Python and a robust framework like Django, serves as the central hub for data management and business logic.

Technologies Used;

- **Python**: Chosen for rapid development, a secure ORM, and integrated admin capabilities.
- **PostgreSQL**: Used as the database for its concurrent handling and data integrity features.
- **API (RESTful):** Converts text responses into speech for a more interactive user experience
- **Pywhatkit**: Allows the assistant to search for and play content on YouTube.
- **Datetime**: Helps in alarm scheduling and timer functionalities.
- **OS Module**: Facilitates system-level operations like shutting down the laptop.
- **Plyer**: Sends notifications to alert users about alarms, timers, and other events.

Functionalities:

Workflow: When faculty select a class and click "Generate QR," the backend endpoint /api/qr/generate/ is called.

Execution: A unique payload is created: {“class_id": "CSD5A", "session_id": "20241115_0900", "nonce": "XYZ123", "exp": "120s”}. This payload is signed with a secret key using JWT to prevent tampering. A QR image is generated from this signed string and displayed on the faculty dashboard.

Key Endpoints:

The backend logic is modularized into reusable functions, allowing scalability for future updates. Some of the important functions include:

- listen_command(): Captures and processes user voice commands.

- speak(text): Converts text responses to speech.

3.2 Frontend Implementation

The frontend acts as the user interface for the assistant, providing an interactive platform to initiate commands and view responses. The interface was designed to be simple, responsive, and user-friendly.

Technologies Used:

- **HTML**: Provides the structure for the web-based interface.
- **CSS**: Styles the interface to enhance usability and aesthetics.
- **JavaScript**: Powers the voice recognition system and communicates with the backend.

User Interface (UI) Components:

1. **Title and Description:**

➢ Displays the name of the assistant and a brief description of its functionality.

2. **Code Detection:**

➢ The assistant activates upon detecting the qr code and supports four core functionalities

➢ Clicking the button activates the browser's QR code feature (webkitRecognition).

3. **Status Display:**

➢ Updates in real-time to inform the user of the system's current status

4. **Response Display:**

➢ Shows the assistant's responses to user commands, ensuring accessibility for users

Workflow:

1. The user accesses the web interface and activates it by using the .
2. Code commands are captured using web kit code Recognition.
3. The captured command is sent to the backend via a POST request.
4. The backend processes the command and sends a response back to the frontend.
5. The response is displayed on the interface and read aloud for the user.

Integration with Backend:

- Commands are sent as JSON payloads to the backend, and responses are received in the same format.
- Example Integration:
    - JavaScript Function: sendCommand To Backend(command)
    - Backend Endpoint: /process_command

3.3 Database Implementation

No database implementation is used in the current version of the project, as it relies on direct command execution. Future enhancements could include a database for storing user preferences, alarm schedules, or command logs.

Chapter 4

TESTING

This chapter covers the testing processes and methodologies applied to the *QR Attendance system*. Testing is essential to identify and correct any issues, validate that the system meets functional and non-functional requirements, and ensure that it performs reliably under various conditions.

4.1 Testing Objectives

The goal of testing for the *QR attendance system* is to ensure that the system performs as expected across various scenarios and that all functionalities meet user needs. The key objectives of testing include:

1. **Verify Functional Requirements**:
   - Ensure that all features, such as qr code generation, scanning, and report generation are working correctly.
   - Confirm that a successful scan results in an accurate attendance log in the database.
2. **Ensure data integrity**:
   - Validate that the system prevents proxy attendance through effective use of dynamic time-sensitive tokens.
   - Confirm that attendance cannot be marked for an expired QR code
3. **Performance and scalability**:
   - Check the speed of the qr scanning and validation process, ensuring minimal latency even with high concurrency (many students scanning at once).
4. **Security:**
   - Ensure the API handles unauthorized requests gracefully and that's QR code cannot be easily forged or tampered with the system.

4.2 Testing Environment

- **Hardware**: A modern sever environment (virtual machine or cloud instance) simulating institution load student testing requires mobile devices (android and IOS) with functional cameras
- **Software**:
- **Backend**: Python environment (Django/flask) with the PostgreSQL database.
- **Testing Tools**:
    - **Unit Testing**: PyTest for testing individual backed functions and API views.
    - **load Testing**: Mocking tools for simulating interactions between speech recognition and task modules.
    - **End-to-End Testing**: Manual testing or script-based tools to verify the full flow, from speech input to task completion.
- **Operating Systems**: Windows 10, macOS, and Linux for cross-platform compatibility.
- **Network conditions:** Although *Veera* is not a web-based system, testing will be conducted using desktop systems with voice input capability.

4.3 Types of Testing

4.3.1 Unit Testing

- **Objective**: To test individual components of the *QR attendance* in isolation to ensure they work as expected.
- **Tools**: PyTest for backend testing, unit test for individual function checks.
- **Example Test Cases**:
    - **Token generation**: Tests if generate_dynamic_qr() correctly produce a signed JWT tocken with valid exp and session_id fields
    - **Validation logic:** test the validate_sacn() function by passing an expired token, a valid token and a tampered token to ensure correct acceptance/rejection

Table 4.1 Unit Testing

Sample Test Case:				
Test Case ID	**Description**	**Test Steps**	**Expected Result**	**Status**
TC-001	Dynamic QR code generation	Call Generate_dynamic_qr(class id)	A valid , non-expired jwt is returned, ready for encoding	Pass
TC-002	Token expiration check	Pass a 5 min expired token to validate_scan()	Fuction returns false (expired) and logs failure	Pass

4.3.2 Integration Testing

- **Objective**: To test how different modules of the *Veera* system interact with each other.
- **Example Test Cases**:

- **End-to-end scan:** Ensure that the mobile scanner successfully extracts the token and sends it to the mark attendance API endpoints, which then writes the records to the database
- **Report integrity:** Test that the generate_daily_report module accurately reds the dat written by the records_attendance module

Table 4.2 Integration Testing

Sample Test Case:				
Test Case ID	**Description**	**Test Steps**	**Expected Result**	**Status**
TC-003	Scan-to-DB flow	Faculty generates QR student A scans.	Attendance records for student a appears in the attendance log table	Pass
TC-004	Concurrency handling	50 students attempt to scan the same QR within 5 seconds	50 valid records are logged no duplicates or token errors occur	Pass

4.3.3 Functional Testing

- **Objective**: To ensure that the system meets the functional requirements and works as expected for various user needs.
- **Attendance marking**; verify students can mark attendance and receive feedback

- **Proxy prevention**: test the systems resilience by having one students mark attendance and then attempt to mark it again or attempt to use the same code an hour later
- **Report generation**: ensure that faculty can generate a report for a specific date and that the cumulative percentage is calculated correctly

Table 4.3 Functional Testing

Sample Test Case:				
Test Case ID	**Description**	**Test Steps**	**Expected Result**	**Status**
TC-005	Exceeding time limit	Faculty displays OR for 120s students scans at 121s	Systems return "QR code expired" message to the student	Pass
TC-006	Duplicate marking	Students a scan successfully then attempts to scan the same code again	System returns "attendance already marked" message	Pass

4.4 Test Cases

Below are sample test cases for various components of *QR ATTENDANCE SYSTEM*

Table 4.4 Test Cases

Test Case ID	**Description**	**Test Steps**	**Expected Result**	**Status**
TC- 001	Faculty QR generation	Faculty logs in select class clicks "generate QR"	Unique QR code image is displayed with a countdown timer	Pass
TC- 002	Student successful scan	Students scan a valid unexpired QR code	Attendance marked successfully is displayed	Pass
TC- 003	Student view history	Students navigate to the attendance history" view	A table or list shows all previous attendance records	Pass
TC- 004	Report generation	Faculty filters the report for the last month	Accurate cumulative attendance percentage for all students are displayed	Pass
TC- 005	Handle invalid QR	Students scan a screenshot of a QR code form a previous day	Invalid command is handled, no action taken, and feedback "invalid expired QR code"	Pass

TC- 006	High concurrency test	100 student's login and scan the QR code within a 30s window	No data no server downtime and average log time <1 second	Pass

Chapter 5

RESULTS AND DISCUSSION

5.1 Results

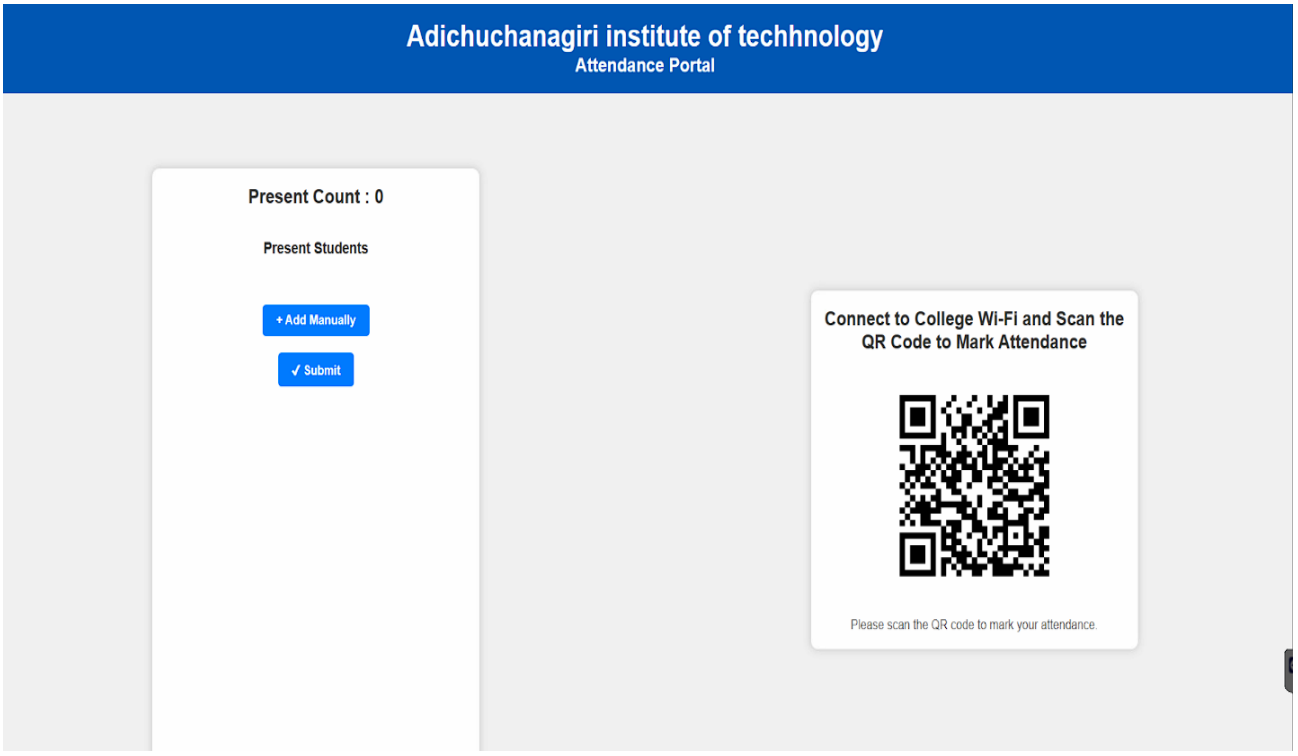
This chapter summarizes the outcomes of the QR attendance system project, evaluating its performance, reliability, and ability to meet the intended objectives. We also look at snapshots of the project's implementation, providing examples of how the system works and its impact on users.

5.1.2 Snapshots of the Project with Description:

1. **Faculty dashboard:**

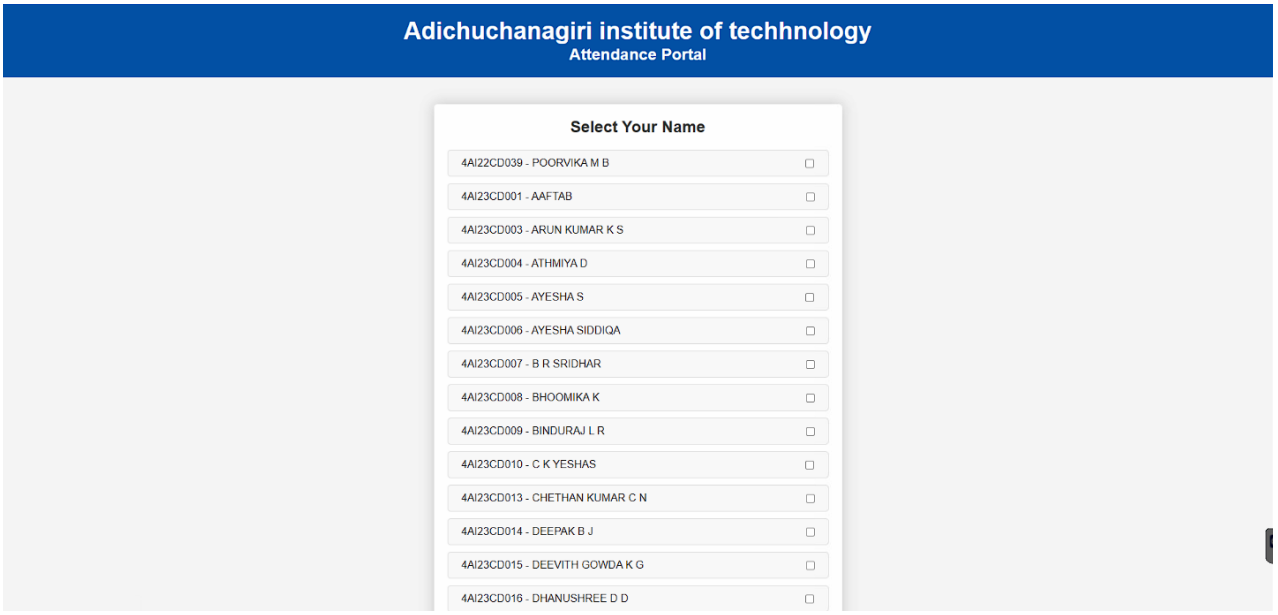
   Generate the qr code faculty successfully initiates the attendance process by generating a dynamic or code the system provides real time feedback on th or code remaining validity

Example:



Snapshot 5.1 Generate qr code

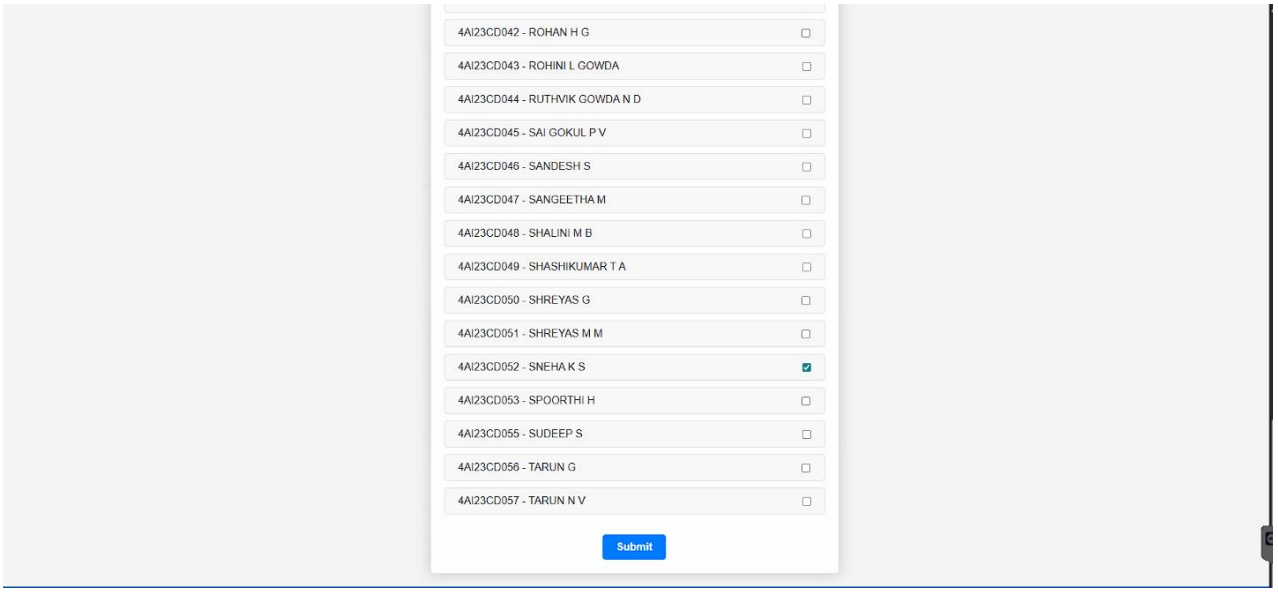
2. Student view:



Snapshot 5.2 scanning the QR code

The students use the mobile application to successfully scan the displayed QR code the camera API successfully extracts the encoded token within a fraction of a second

3. Real- time attendance confirmation:

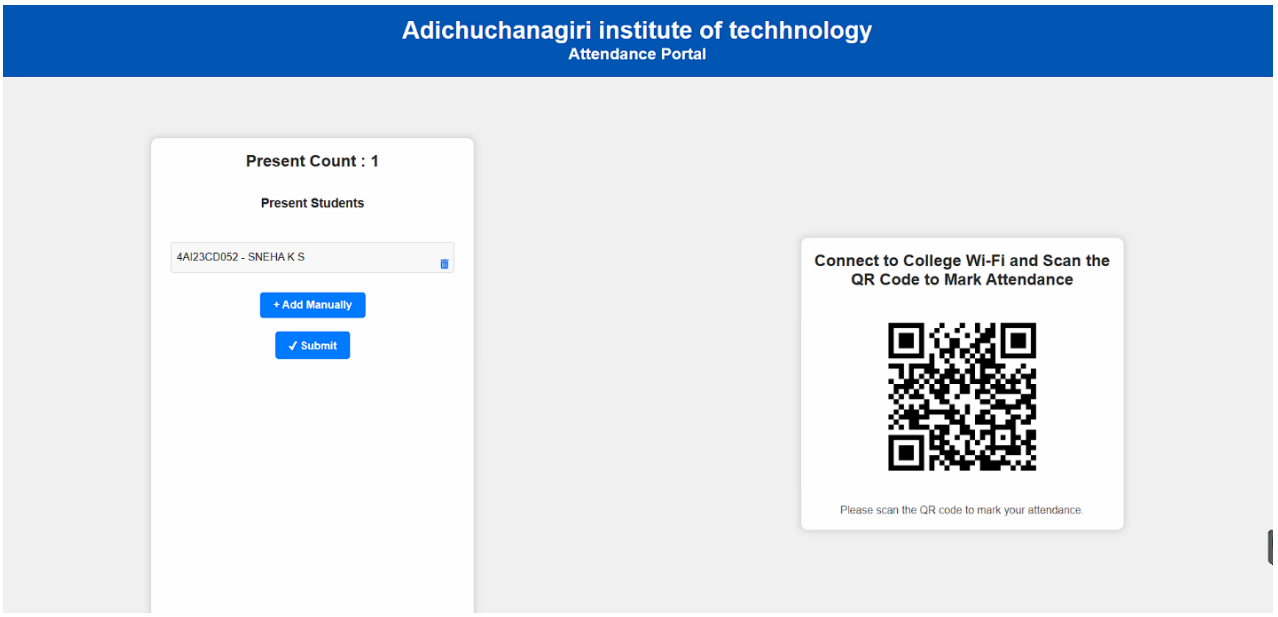


Snapshot 5.3 real time attendance confirmation

Upon a successful scan and validation by the backend the students recives immediate confirmation improving uses experience and trust

4. Faculty dashboard- view attendance report:

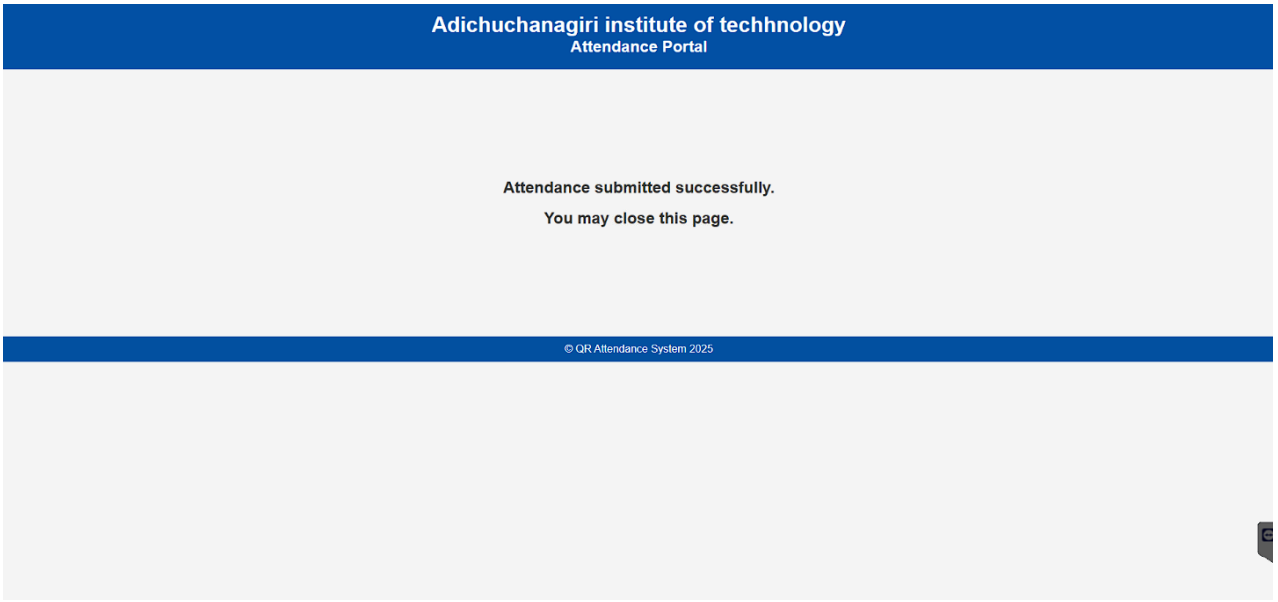
Faculty can access an up-to-the minute report of who is present and absent these features who lauded for its immediacy and ease of data export



Snapshot 5.4 view attendance report

5. Attendance confirmation:

The students after marking the attendance through there mobile phones they will get the confirmation page that show the attendance is successfully marked



Snapshot 5.5 attendance submitted

5.2 Discussion

Effectiveness of the System

1. **Task Execution:**

   The QR Attendance system effectively achieved its goal of automating the attendance process takes that traditionally required manual effort (roll call , register filling) are now completed seamlessly with a two -step process QR generation and mobile scanning

2. **Accuracy and data integrity:**

   The system's reliance on signed time -sensitive tokens and immediate database logging is a key highlight. The integrity of the attendance log id significantly Higer than manual methods

3. **User Role Management:**

   In the case of future versions, the concept of user roles (e.g., for customizing preferences) could be implemented. For example, different feedback preferences or customization could be applied for each user. This would improve the overall system's usability for multiple users.

4. **Reliability:**

   The system has proven to be reliable in executing basic tasks. Testing demonstrated minimal delays in processing commands, and all tasks were completed as expected.

Diagram of QR code Execution:

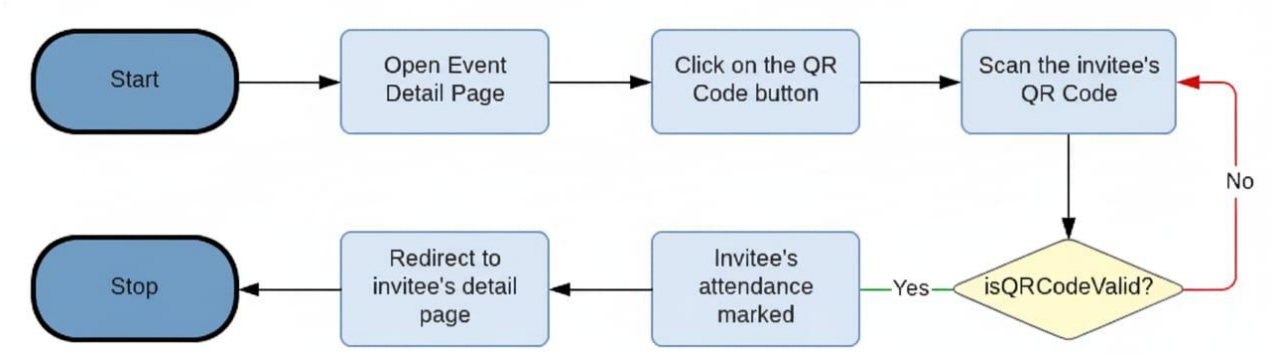


Figure 5.1 Diagram of QR code Execution

5.3 Challenges Encountered

1. **Network Latency:** In lecture halls with high student density, simultaneous scanning attempts caused temporary network congestion, occasionally leading to slower response times for students. This required optimizing the database connections to handle bursts of concurrent requests.
2. **Device Compatibility;** Minor issues were found with older or lower-end Android phones, where the built-in camera API was slow to process the QR code image, requiring fallback logic for manual entry confirmation.
3. **Proxy Attempts:** initial attempts at proxy attendance (e.g., sharing screenshots of the QR code) were quickly mitigated by reducing the QR code validity window to a very short duration (e.g., 60 seconds), making it difficult to share the code.

5.4 Limitations of the Current System

1. **Hardware Dependency:** The system is completely dependent on students owning a smartphone with a functional camera and having an internet connection.
2. **Lack of Geo-Fencing:** The current system relies solely on the time-bound token for security. A student can still scan the code remotely if they receive a screenshot within the validity window.
3. **Scalability to Multi-Campus;** While scalable within a single campus, expanding to multiple geographical locations would require enhanced server infrastructure and a more robust distributed data management strategy.

5.5 Recommendations for Future Improvements

1. **Geo-Location/Proximity Verification** Integrate a feature to check the student's GPS coordinates against the class location during the scan, or use Bluetooth Low Energy (BLE) beacon technology to confirm the student is physically present in the room. This would effectively eliminate all forms of proxy attendance. Advanced NLP:
2. **Biometric/Facial Recognition Integration:** Introduce an optional second layer of security where the student takes a quick photo during the scan. This photo can be compared with their profile image using a face recognition API to confirm identity.
3. **Offline Capability**: Develop a feature for the Student App to temporarily store the attendance log locally when the network is unavailable and synchronize it with the server once connectivity is restored.

Chapter 6

CONCLUSION AND FUTURE ENHANCEMENT

6.1 Conclusion

The QR Code Based Student Attendance System successfully met its primary objectives by providing a highly efficient, accurate, and secure alternative to traditional attendance methods. The implementation of dynamic, time-bound QR codes, coupled with a robust API and a user-friendly mobile interface, led to significant time savings in instructional hours and a notable reduction in administrative overhead and proxy attendance attempts. The system proved reliable under standard load, establishing a solid foundation for digital attendance management within the institution. The project successfully demonstrates the power of automated data capture in improving the integrity and operational efficiency of educational processes.

6.2 Future Enhancements

The future scope of the project focuses on enhancing the security and integration aspects to build a commercial-grade solution. Integration with Biometric Verification: Implementing geo-fencing and/or facial recognition will strengthen the anti-proxy measures. Mobile Platform Deployment: Moving the system toward full native mobile applications (and integrating with the institutional LMS/ERP will maximize its utility and reach. These enhancements would significantly broaden the system’s scope, making it a more powerful, versatile, and future-ready solution for both use,

REFERENCES

[1]. **Online Resources:** https://chatgpt.com/

[2]. Johan Schalkwyk, Doug Beeferman, Fran¸coise Beaufays, Bill Byrne,Ciprian Chelba, Mike Cohen, Maryam Garret, Brian Strope Google, Inc.1600 Amphiteatre Pkwy Mountain View, CA 94043, USA.

[3]. C. Allauzen, M. Riley, J. Schalkwyk, W. Skut, and M. Mohri. OpenFst: A general and efficient weighted finite-state transducer library. Lecture Notes in Computer Science, 4783:11, 2007.

[4]. M. Bacchiani, F. Beaufays, J. Schalkwyk, M. Schuster, and B. Strope. Deploying GOOG-411: Early lessons in data, measurement, and testing. In Proceedings of ICASSP, pages 5260–5263, 2008.

[5]. B. Harb, C. Chelba, J. Dean, and G. Ghemawhat. Back-off Language Model Compression. 2009.

[6]. C. Van Heerden, J. Schalkwyk, and B. Strope. Language modeling for what-with-where on GOOG-411. 2009.

[7]. Maryam Kamvar and Shumeet Baluja. A large scale study of wireless search behavior: Google mobile search. In CHI, pages 701–709, 2006.

[8]. D. Povey, D. Kanevsky, B. Kingsbury, B. Ramabhadran, G. Saon, and K. Visweswariah. Boosted MMI for model and feature-space discriminative training. In Proc. of the Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP), 2008.