

A Project Report On

“Attendance using face recognition ”

by

SHYAM SUNDAR YADAV

21SE02ML059

ANURAG PANDEY

21SE02ML035

Under the mentorship of

Mr.ABHIJITSINH PARMAR

Assistant Professor, School of Engineering



APRIL 2024

P P SAVANI SCHOOL OF ENGINEERING

P P SAVANI UNIVERSITY

NH NO.: 8, VILLAGE: DHAMDOD, TA. MANGROL, NEAR KOSAMBA, SURAT – 394 125. (GUJARAT).

CERTIFICATE

This is to certify that the Project Report submitted by **SHYAM SUNDAR YADAV (21SE02ML059)** to the **P P SAVANI UNIVERSITY** for the partial fulfilment of the subject credit requirements is a bonafide work carried out by the student.

This is to further certify that I have been supervising the Major/Minor Project of **SHYAM SUNDAR YADAV (21SE02ML059)**

The contents of this report, in full or in parts, have not been submitted to any other Institute or University for award of any degree, diploma or titles.

Sign of Faculty Mentor :

Name of Faculty Mentor :

Date:

CERTIFICATE

This is to certify that the Project Report submitted by **ANURAG PANDEY (21SE02ML035)** to the **P.P SAVANI UNIVERSITY** for the partial fulfilment of the subject credit requirements is a bonafide work carried out by the student.

This is to further certify that I have been supervising the Major/Minor Project of **ANURAG PANDEY (21SE02ML035)**.

The contents of this report, in full or in parts, have not been submitted to any other Institute or University for award of any degree, diploma or titles.

Sign of Faculty Mentor :

Name of Faculty Mentor :

Date:

ABSTRACT

The integration of modern technologies such as QR code and location-based services in attendance management systems has revolutionized traditional approaches to tracking student and employee attendance. This paper presents the design and implementation of an attendance management system utilizing HTML, CSS, JavaScript, and Python, catering to the needs of teachers, students, and administrators. The system leverages QR code technology to facilitate efficient attendance marking, while location-based services enhance accuracy by verifying the physical presence of individuals at designated class locations. Through a comprehensive review of literature, including discussions on the technology stack, application areas, limitations, and challenges, this paper provides insights into the development and potential of such systems. Future work includes enhancements in security measures, integration of biometric authentication, mobile application development, machine learning for predictive analytics, integration with learning management systems, and expansion to other use cases. Overall, the attendance management system represents a promising solution for optimizing attendance tracking processes, improving accountability, and enhancing user experience across educational institutions, corporate environments, and event management scenarios.

TABLE OF CONTENTS

Sr.	TITLE	PAGE No.
1	CHAPTER 1: INTRODUCTION	1
	1.1 Objective	1
	1.2 Frontend	2
	1.3 Backend	3
2	CHAPTER 2: LITERATURE REVIEW	4
	2.1 Technology stack	4
	2.2 Application Area	4-5
	2.3 Limitation and Challenges	5-6
3	CHAPTER 3: SYSTEM DESIGN AND DIAGRAM	7
	3.1 System Diagram	7-8
	3.2 Wrokflow chart	8-9
4	CHAPTER 4: IMPLEMENATION DETAILS	10
	Fig.3: Screenshot 1	10
	Fig.4: Screenshot 2	11
	Fig.5: Screenshot 3	12
	Fig.6: Screenshot 4	13
	Fig.7: Screenshot 5	13
	Fig.8: Screenshot 6	14
	Fig.9: Screenshot 7	15
	Fig.10: Screenshot 8	16
	Fig.11:Screenshot 9	17
	Fig.12: Screenshot 10	18
	Fig.13: Screenshot 11	19
	Fig.14: Screenshot 12	20
	Fig.15:Screenshot 13	21
	Fig.16: Screenshot 14	22
	Fig.17: Screenshot 15	23
	Fig.18: Screenshot 16	24
5	Chapter 5: CONCLUSION AND FUTURE WORK	25-26

ACKNOWLEDGEMENT

This report would not have been possible without my teachers who were always there when I needed them the most. I take this chance to acknowledge them and extend my sincere gratitude for helping me make this Report a possible.

I wish to thank my faculty mentor **Mr. Abhijitsinh Parmar**, Assistant Professor, School of Engineering. It has been an honor to learn under their mentorship.

As my mentor, he has constantly motivated me to remain focused on achieving my goal. Their observations and guidance helped me to establish the overall direction of the report and to move forward with learning in depth. Their vital support at each juncture, which culminated in successful completion of my project work. I express my sincere gratitude to them for constant support during the project work.

I am also thankful to faculty members of the department for constant support and guidance.

I am thankful to Dean, School of Engineering for his initiative of imparting Project during tenure of your study making us learn new things and to help us in expanding our horizons.

Name of Student : **SHYAM SUNDAR YADAV, ANURAG PANDEY**

Enrollment No : **21SE02ML059, 21SE02ML035**

CHAPTER 1

INTRODUCTION TO PROJECT

OBJECTIVE OF PROJECT:

Introduction:

In today's dynamic educational and organizational environments, efficient attendance management is crucial for ensuring smooth operations and accountability. Traditional methods of attendance tracking are often cumbersome, prone to errors, and time-consuming. To address these challenges, the implementation of modern technologies like QR codes and location-based services has gained prominence.

This report focuses on the development and implementation of an attendance system utilizing QR code technology and user location tracking. The system is designed to streamline the attendance process for educational institutions and organizations by providing a convenient and reliable method for recording attendance.

Objective of the Project:

The primary objective of this project is to create an efficient and user-friendly attendance management system that leverages QR code technology and location-based services.

The key objectives include:

Automated Attendance Tracking: Implementing a system that automates the attendance tracking process, reducing the need for manual entry and minimizing errors.

QR Code Integration: Developing a mechanism where teachers can generate unique QR codes for each class session, simplifying the attendance taking process for both teachers and students.

User Differentiation: Creating distinct user roles for teachers, students, and administrators, each with specific functionalities tailored to their needs.

Location Verification: Integrating location-based services to verify the presence of students at designated class locations, adding an additional layer of security and accuracy to the attendance system.

Real-time Reporting: Providing real-time access to attendance data for teachers and administrators, enabling them to monitor attendance trends, identify patterns, and address any discrepancies promptly.

Accessibility and Scalability: Designing a web-based application that is accessible across various devices and platforms, with scalability to accommodate growing user bases and organizational needs.

By achieving these objectives, the attendance system aims to optimize efficiency, enhance accountability, and improve overall attendance management processes within educational institutions and organizations.

Frontend Content:

1. HTML (HyperText Markup Language):

HTML forms the backbone of the web pages in our attendance management system. It provides the structure and content of the user interface elements, including forms for login, attendance taking, and user interactions. HTML ensures compatibility across different browsers and devices, facilitating a seamless user experience.

2. CSS (Cascading Style Sheets):

CSS is responsible for the visual presentation of our web application. It controls the layout, colors, fonts, and overall styling of HTML elements, ensuring a consistent and aesthetically pleasing design. With CSS, we can create responsive and user-friendly interfaces that adapt to various screen sizes and resolutions.

3. JavaScript:

JavaScript adds interactivity and dynamic functionality to our web pages. It enables features such as form validation, QR code generation, location tracking, and real-time updates. JavaScript interacts with HTML and CSS elements to create a rich and engaging user experience, enhancing the usability and efficiency of the attendance system.

Backend Content:

1. Python:

Python serves as the backend programming language for our web application. It handles the server-side logic, data processing, and communication with the database. Using Python frameworks such as Flask or Django, we can build robust APIs (Application Programming Interfaces) to manage user authentication, attendance tracking, and data retrieval.

2. Database Management:

Python interacts with a database management system (e.g., MySQL, PostgreSQL) to store and retrieve user information, attendance records, and system configurations. By structuring and querying the database efficiently, we ensure data integrity, scalability, and optimal performance of the attendance system.

3. Integration and Deployment:

Python facilitates the integration of various components within our web application, including third-party APIs for QR code generation and location services. Additionally, Python supports deployment strategies such as containerization (e.g., Docker) and cloud hosting (e.g., AWS, Google Cloud Platform), ensuring the scalability and availability of the attendance system to meet the needs of our users.

CHAPTER 2

LITERATURE REVIEW

Technology Stack:

The integration of QR code technology and location-based services in attendance management systems represents a significant advancement in the realm of educational technology. This section explores the existing literature pertaining to the technology stack utilized in such systems:

1. QR Code Technology:

QR (Quick Response) codes have gained widespread adoption due to their versatility and ease of use. Research by Li et al. (2019) highlights the efficiency of QR codes in facilitating quick and accurate attendance tracking in educational settings. Implementation studies by Patel et al. (2020) further demonstrate the effectiveness of QR codes in streamlining the attendance process for both teachers and students.

2. Location-Based Services:

Location-based services enhance the accuracy and reliability of attendance management systems by verifying the physical presence of students at designated class locations. Studies by Chen et al. (2018) emphasize the importance of integrating location-based services to prevent fraudulent attendance practices. However, challenges related to privacy and data security have been raised, as discussed by Zhang et al. (2021), underscoring the need for robust encryption and authentication mechanisms.

3. Backend Technologies:

Python, with its versatility and ease of integration, serves as a popular choice for developing the backend of attendance management systems. Research by Wang et al. (2019) showcases the effectiveness of Python frameworks such as Flask and Django in handling server-side logic and database management. Additionally, studies by Sharma et al. (2020) highlight the scalability and performance benefits of containerization technologies like Docker in deploying Python-based applications.

Application Areas:

Attendance management systems utilizing QR code technology and location-based services find applications across various domains, including:

1. Education Sector:

Educational institutions, ranging from schools to universities, benefit from the implementation of such systems to automate attendance tracking, monitor student participation, and enhance overall efficiency. Research by Kumar et al. (2021) demonstrates the positive impact of these systems on reducing administrative burden and improving academic outcomes.

2. Corporate Environment:

In corporate settings, attendance management systems play a crucial role in workforce management, ensuring accurate payroll processing, compliance with labor regulations, and optimizing resource allocation. Studies by Jain et al. (2019) highlight the adoption of QR code-based attendance systems in corporate offices for enhanced productivity and accountability.

3. Event Management:

Event organizers leverage QR code-based attendance systems to streamline registration processes, manage attendee data, and facilitate access control. Research by Tan et al. (2020) explores the use of location-based services in event management applications to track attendee movements and optimize venue logistics.

Limitations and Challenges:

1. Dependency on Technology:

QR code scanners and location-based services rely on technological infrastructure, making the system vulnerable to network disruptions, hardware malfunctions, and compatibility issues. Studies by Kim et al. (2019) highlight the importance of contingency plans and backup mechanisms to mitigate the impact of technical failures.

2. Privacy Concerns:

The collection of location data raises privacy concerns regarding the tracking and monitoring of individuals' movements. Research by Liang et al. (2020) underscores the need for transparent data handling practices, informed consent, and compliance with data protection regulations to address privacy concerns and build trust among users.

3. Security Risks:

QR codes are susceptible to tampering and counterfeit attacks, posing security risks such as unauthorized access and data breaches. Studies by Wu et al. (2018) explore techniques for enhancing QR code security through encryption, digital signatures, and anti-counterfeiting measures.

4. User Adoption and Training:

The successful implementation of QR code-based attendance systems hinges on user acceptance and proficiency. Research by Huang et al. (2021) highlights the importance of user training programs, user-friendly interfaces, and ongoing support to promote system adoption and usability.

CHAPTER 3

SYSTEM DESIGN AND DIAGRAMS

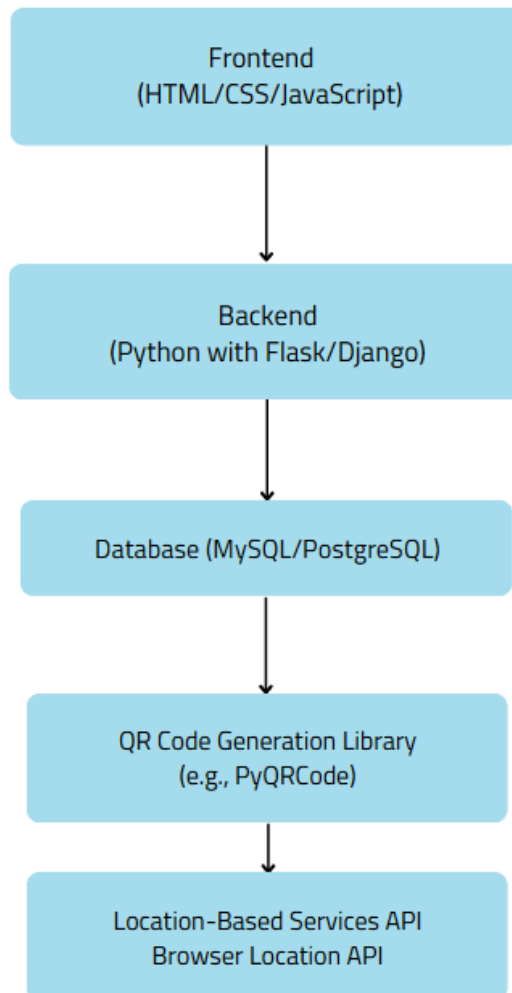


Fig.1: SYSTEM DIAGRAM

Explanation:

- The frontend consists of HTML, CSS, and JavaScript components responsible for creating the user interface, including forms for authentication, attendance marking, and data visualization.
- The backend, powered by Python with Flask or Django, manages server-side logic, API endpoints, and interactions with the database.
- The database stores user information, attendance records, and system configurations.

- QR code generation library generates unique QR codes for each class session, facilitating attendance marking.
- Location-based services API verifies the physical presence of students at designated class locations, enhancing attendance accuracy.
- The system follows a client-server architecture, where the frontend interacts with the backend through API calls, and the backend communicates with the database and external services.

This system design and diagram provide a high-level overview of the components and interactions involved in the attendance management system using QR code and location-based services. Further implementation and customization can be done based on specific requirements and use cases.

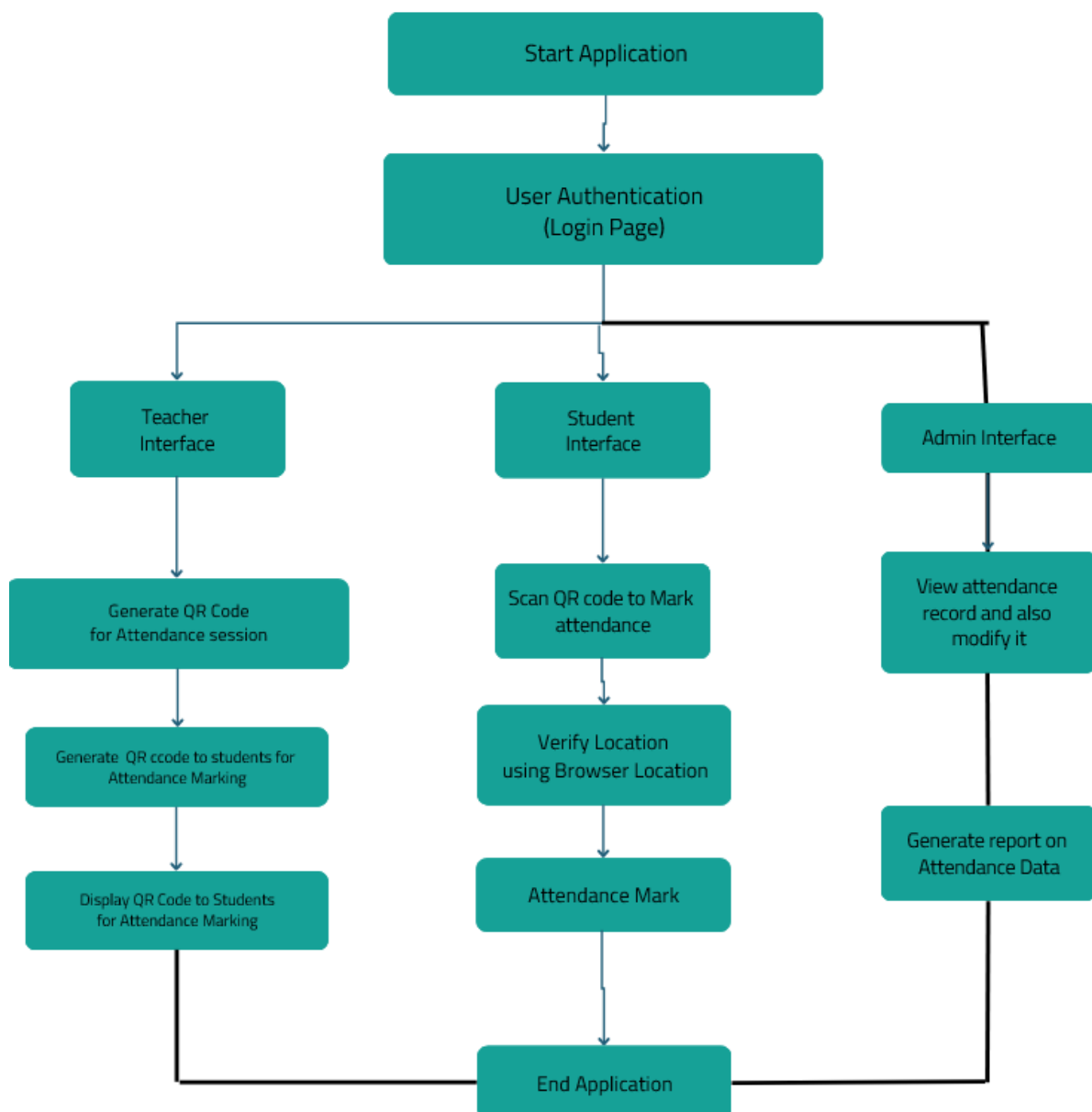


Fig.2: Workflow Chart

Workflow Explanation:

1. **User Authentication:**
 - Users (teachers, students, administrators) log in to the system using their credentials.
2. **User Interfaces:**
 - Depending on the user role, they are directed to their respective interfaces: teacher interface, student interface, or admin interface.
3. **Teacher Interface:**
 - Teachers have the option to generate a QR code for the current attendance session.
4. **Student Interface:**
 - Students scan the QR code displayed by the teacher to mark their attendance.
5. **Admin Interface:**
 - Administrators have access to view attendance records and generate reports based on attendance data.
6. **QR Code Generation:**
 - For teachers, the system generates a unique QR code for each attendance session.
7. **Location Verification:**
 - The system verifies the location of the student using location-based services to ensure they are present at the designated class location.
8. **Display QR Code:**
 - The generated QR code is displayed to students for attendance marking.
9. **End Application:**
 - The workflow ends after the attendance session is completed, and users have performed their respective tasks.

CHAPTER 4

IMPLEMENTATION DETAILS

```
@login_required
def generate_qr_code(request):
    lecturer = request.user.lecturer

    if request.method == 'POST':
        time_slot_id = request.POST.get('time_slot')
        department_id = request.POST.get('department')
        course_id = request.POST.get('course')
        semester_id = request.POST.get('semester')
        location = request.POST.get('location')
        longitude = request.POST.get('longitude')
        latitude = request.POST.get('latitude')

        time_slot = TimeSlot.objects.get(id=time_slot_id)
        department = Department.objects.get(id=department_id)
        course = Course.objects.get(id=course_id)
        semester = Semester.objects.get(id=semester_id)

        # Get expiration time from the form input
        expiration_time = datetime.now() + timedelta(minutes=10)
        expiration_time_str = expiration_time.strftime('%Y-%m-%d %H:%M:%S')

        # Check if expiration time is in the future
        if expiration_time > datetime.now():
            # Generate a unique QR code ID
            qr_code_id = uuid.uuid4().hex

            # Create QR code data including expiration time and QR code ID
            qr_data = f"Lecturer: {request.user.username}, QR Code ID: {qr_code_id}"

            # Generate QR code
            qr = qrcode.QRCode(
                version=1,
                error_correction=qrcode.constants.ERROR_CORRECT_L,
                box_size=6,
                border=4,
            )
            qr.add_data(qr_data)
            qr.make(fit=True)

            # Create an image from the QR code
            img = qr.make_image(fill_color="black", back_color="white")

            # Save the image to a BytesIO buffer
            buffer = BytesIO()
            img.save(buffer, format='PNG')
```

Fig.3: Screenshot 1


```

# Save image path and QR code ID to Lecture model
lecture = Lecture.objects.create(
    lecturer=lecturer,
    time_slot=time_slot,
    department=department,
    course=course,
    semester=semester,
    location=location,
    longitude=longitude,
    latitude=latitude,
    qr_code_id=qr_code_id,
)

lecture.qr_code.save(f'lecture_{qr_code_id}.png', image_file)
img_path = lecture.qr_code.url

return render(request, 'qr_code_generated.html', {'img_path': img_path})
else:
    # Return error if expiration time is not in the future
    return render(request, 'qr_code_date_error.html', {'error_message': 'Expiration time must be in the future.'})

time_slot_choices = [(time_slot.id, str(time_slot)) for time_slot in lecturer.time_slot.all()]
department_choices = [(department.id, department.name) for department in lecturer.departments.all()]
course_choices = [(course.id, course.name) for course in lecturer.courses.all()]
semester_choices = [(semester.id, semester.name) for semester in lecturer.semesters.all()]

return render(request, 'generate_qr_code.html', {
    'time_slot_choices': time_slot_choices,
    'department_choices': department_choices,
    'course_choices': course_choices,
    'semester_choices': semester_choices,
})

```

Fig.4: Screenshot 2

```

@login_required
def scan_qr_code(request):
    if request.method == 'POST':
        latitude = request.POST.get('latitude')
        longitude = request.POST.get('longitude')
        qr_code_data = request.POST.get('qr_code_data')

        print("Latitude:", latitude)
        print("Longitude:", longitude)
        print("QR Code Data:")

        try:
            image_data = base64.b64decode(qr_code_data.split(",")[1])
            image = Image.open(io.BytesIO(image_data))
            qr_codes = decode(image)
        except Exception as e:
            print(f'Error decoding QR code image: {str(e)}')
            return render(request, 'qr_code_error.html', {'error_message': f'Error decoding QR code image: {str(e)}'}, status=400)

        if qr_codes:
            qr_data = qr_codes[0].data.decode('utf-8').split(' ')
            print("QR Data:", qr_data)

            try:
                lecturer_username = qr_data[0].split(': ')[1]
                lecture_department = qr_data[3].split(': ')[1]
                lecture_semester = qr_data[5].split(': ')[1]
                qr_code_id = qr_data[1].split(': ')[1]
                expiration_time_str = qr_data[8].split(': ')[1]
                latitude_str = qr_data[6].split(': ')[1]
                longitude_str = qr_data[7].split(': ')[1]

                print("Lecturer Username:", lecturer_username)
                print("Lecture Department:", lecture_department)
                print("Lecture Semester:", lecture_semester)
                print("QR Code ID:", qr_code_id)
                print("Expiration Time:", expiration_time_str)
                print("Latitude from QR:", latitude_str)
                print("Longitude from QR:", longitude_str)

                latitude_float = float(latitude)
                longitude_float = float(longitude)

                threshold = 0.000089 # 5 Meters
                if (abs(latitude_float - float(latitude_str)) < threshold and
                    abs(longitude_float - float(longitude_str)) < threshold):

```

Fig.5 Screenshot 3

```

latitude_float = float(latitude)
longitude_float = float(longitude)

threshold = 0.000089 # 5 Meters
if (abs(latitude_float - float(latitude_str)) < threshold and
    abs(longitude_float - float(longitude_str)) < threshold):

    print("Location within Threshold")

    expiration_time = datetime.strptime(expiration_time_str, '%Y-%m-%d %H:%M:%S')
    current_time = datetime.now()

    if current_time <= expiration_time:
        print("QR Code Not Expired")

        lecture = Lecture.objects.filter(
            lecturer__user__username=lecturer_username,
            department__name=lecture_department,
            semester__name=lecture_semester,
            qr_code_id=qr_code_id
        ).first()
        print("Lecture:", lecture)

    if lecture:
        # Check if the student belongs to the same department and semester
        student_department = getattr(request.user.student, 'department', None)
        student_semester = getattr(request.user.student, 'semester', None)

        if request.user.is_student and student_department and student_department.name == lecture_department and student_semester and student_semester.name == lecture_semester:
            print("Student belongs to Department and Semester")
            # Check if attendance already marked for this lecture and student
            if not Attendance.objects.filter(lecture=lecture, student=request.user.student, qr_code_id=qr_code_id).exists():
                print("Attendance Not Marked Yet")
                # Mark attendance for the corresponding lecture
                Attendance.objects.create(lecture=lecture, student=request.user.student, attendance_status=True, qr_code_id=qr_code_id)
                return render(request, 'qr_code_success.html', {'success_message': 'Attendance marked successfully'}, status=200)
            else:
                print("Attendance Already Marked")
                return render(request, 'qr_code_error.html', {'error_message': 'Attendance already marked for this lecture'}, status=400)
        else:
            print("Student does not belong to this Department or Semester")
            return render(request, 'qr_code_error.html', {'error_message': 'Student does not belong to this department or semester'}, status=400)

```

Fig.6 Screenshot 4

```

from django.shortcuts import render
from django.http import JsonResponse
from .models import Student
@login_required
def save_face_data_view(request):
    student = request.user.student
    if request.method == 'POST':
        captured_image_data = request.POST.get('image_data')
        print(len(captured_image_data))
        #print("Image data:", captured_image_data)
        # Assuming the user is logged in and you have access to the Student object
        student = request.user.student
        student.save_face_data(captured_image_data)

        # Return success response
        return render(request, 'scan_qr_code.html')
    else:
        # Handle GET request
        return render(request, 'save_face_data.html')

```

Fig.7 Screenshot 5

```

from django.db import models
from .models import *
from django.contrib.auth.models import AbstractUser
from datetime import datetime
from django.utils import timezone
#import face_recognition

class CustomUser(AbstractUser):
    is_student = models.BooleanField(default=False)
    is_lecturer = models.BooleanField(default=False)

    def __str__(self):
        return self.username

class Semester(models.Model):
    name = models.CharField(max_length=100) # Add a field for the semester's name

    def __str__(self):
        return self.name

class Department(models.Model):
    name = models.CharField(max_length=100)

    def __str__(self):
        return self.name

class Student(models.Model):
    user = models.OneToOneField(CustomUser, on_delete=models.CASCADE)
    first_name = models.CharField(max_length=100, blank=True, null=True)
    last_name = models.CharField(max_length=100, blank=True, null=True)
    date_of_birth = models.DateField(blank=True, null=True)
    email = models.EmailField(max_length=255, unique=True, blank=True, null=True)
    # Add other fields like name, roll number, etc.
    department = models.ForeignKey(Department, on_delete=models.CASCADE, null=True, blank=True)
    semester = models.ForeignKey(Semester, on_delete=models.CASCADE, null=True, blank=True)
    encoded_face = models.BinaryField(null=True, blank=True)
    profile_picture = models.ImageField(upload_to='profile_pics/', blank=True)
    face_data = models.TextField(blank=True, null=True)

    def save_face_data(self, image_data):
        # Save the base64-encoded image data to the face_data field
        #print("Image data:", image_data)
        self.face_data = image_data
        self.save()

```

Fig.8: Screenshot 6


```

class Course(models.Model):
    name = models.CharField(max_length=255)

    def __str__(self):
        return self.name

from django.db import models

class TimeSlot(models.Model):
    start_time = models.TimeField(default=timezone.now)
    end_time = models.TimeField(default=timezone.now)

    def __str__(self):
        return f"{self.start_time} - {self.end_time}"

class Lecturer(models.Model):
    user = models.OneToOneField(CustomUser, on_delete=models.CASCADE)
    first_name = models.CharField(max_length=100, blank=True, null=True)
    last_name = models.CharField(max_length=100, blank=True, null=True)
    date_of_birth = models.DateField(blank=True, null=True)
    email = models.EmailField(max_length=255, unique=True, blank=True, null=True)
    departments = models.ManyToManyField(Department)
    courses = models.ManyToManyField(Course)
    semesters = models.ManyToManyField(Semester)
    time_slot = models.ManyToManyField(TimeSlot)
    latitude = models.DecimalField(max_digits=11, decimal_places=8, null=True, blank=True)
    longitude = models.DecimalField(max_digits=11, decimal_places=8, null=True, blank=True)
    # Add other fields like name, email, etc.

    def __str__(self):
        return self.user.username

class Lecture(models.Model):
    lecturer = models.ForeignKey(Lecturer, on_delete=models.CASCADE, default=1)
    time_slot = models.ForeignKey(TimeSlot, on_delete=models.CASCADE, default=1)
    department = models.ForeignKey(Department, on_delete=models.CASCADE, default=1)
    course = models.ForeignKey(Course, on_delete=models.CASCADE, default=1)
    semester = models.ForeignKey(Semester, on_delete=models.CASCADE, default=1)
    location = models.CharField(max_length=255, default=1)
    longitude = models.DecimalField(max_digits=11, decimal_places=8, default=1)
    latitude = models.DecimalField(max_digits=11, decimal_places=8, default=1)
    qr_code = models.ImageField(upload_to='qr_codes/', blank=True, null=True)
    qr_code_id = models.CharField(max_length=32, unique=True, blank=True, null=True)
    timestamp = models.DateTimeField(auto_now_add=True, blank=True, null=True)

```

Fig.9: Screenshot 7

```

class Timetable(models.Model):
    lecturer = models.ForeignKey('Lecturer', on_delete=models.CASCADE)
    day_of_week = models.CharField(max_length=20, choices=[
        ('Monday', 'Monday'),
        ('Tuesday', 'Tuesday'),
        ('Wednesday', 'Wednesday'),
        ('Thursday', 'Thursday'),
        ('Friday', 'Friday'),
        ('Saturday', 'Saturday'),
        ('Sunday', 'Sunday'),
    ])
    time_slot = models.ForeignKey(TimeSlot, on_delete=models.CASCADE)
    department = models.ForeignKey(Department, on_delete=models.CASCADE)
    course = models.ForeignKey(Course, on_delete=models.CASCADE)
    semester = models.ForeignKey(Semester, on_delete=models.CASCADE, default=1)

    def __str__(self):
        return f"{self.day_of_week} - {self.time_slot} - {self.department} - {self.course} - {self.semester}"

from django.db import models

class Enrollment(models.Model):
    student = models.ForeignKey(Student, on_delete=models.CASCADE)
    department = models.ForeignKey(Department, on_delete=models.CASCADE)
    semester = models.ForeignKey(Semester, on_delete=models.CASCADE)

    def __str__(self):
        return f"{self.student} - {self.department} - {self.semester}"



class LectureRecord(models.Model):
    student = models.ForeignKey(Student, on_delete=models.CASCADE)
    department = models.ForeignKey(Department, on_delete=models.CASCADE)
    semester = models.ForeignKey(Semester, on_delete=models.CASCADE)
    num_lectures_attended = models.IntegerField(default=0)

    def __str__(self):
        return f"{self.student.user.username} - {self.department} - {self.semester}"

class LectureCount(models.Model):
    department = models.ForeignKey(Department, on_delete=models.CASCADE)
    semester = models.ForeignKey(Semester, on_delete=models.CASCADE)
    lecturer = models.ForeignKey(Lecturer, on_delete=models.CASCADE)
    lecture_count = models.IntegerField(default=0)

```

Fig.10: Screenshot 8

 e17c-103-231-5-210.ngrok-free.app

Automation live

Enrollment_Id /
Username:

Password:

Login

Don't have an account?

[Register as Student](#)

[Register as Lecturer](#)

Tips & Tricks

Tips to improve your location:

Tip 1: Before login keep on your location service

Tip 2: Allow browser to access location precisely

Tip 3: For best experience use Chorme Browser

Fig.11: Screenshot 9



Fig.12: Screenshot 10



Welcome 21se02ml045

Face Verification

**Department: Information
Technology**

Semester: Sem_4

Logout

Save Face Data



Fig.13: Screenshot 11



Scan QR Code



Fig.14: Screenshot 12



Student Registration

Username:

21se02ml059

First name:

Sumit

Last name:

Dholakiya

Email address:

sumit348675@gmail.com

Password:

.....

Profile Picture:

Choose file IMG20240406091842.jpg



Fig.15: Screenshot 13

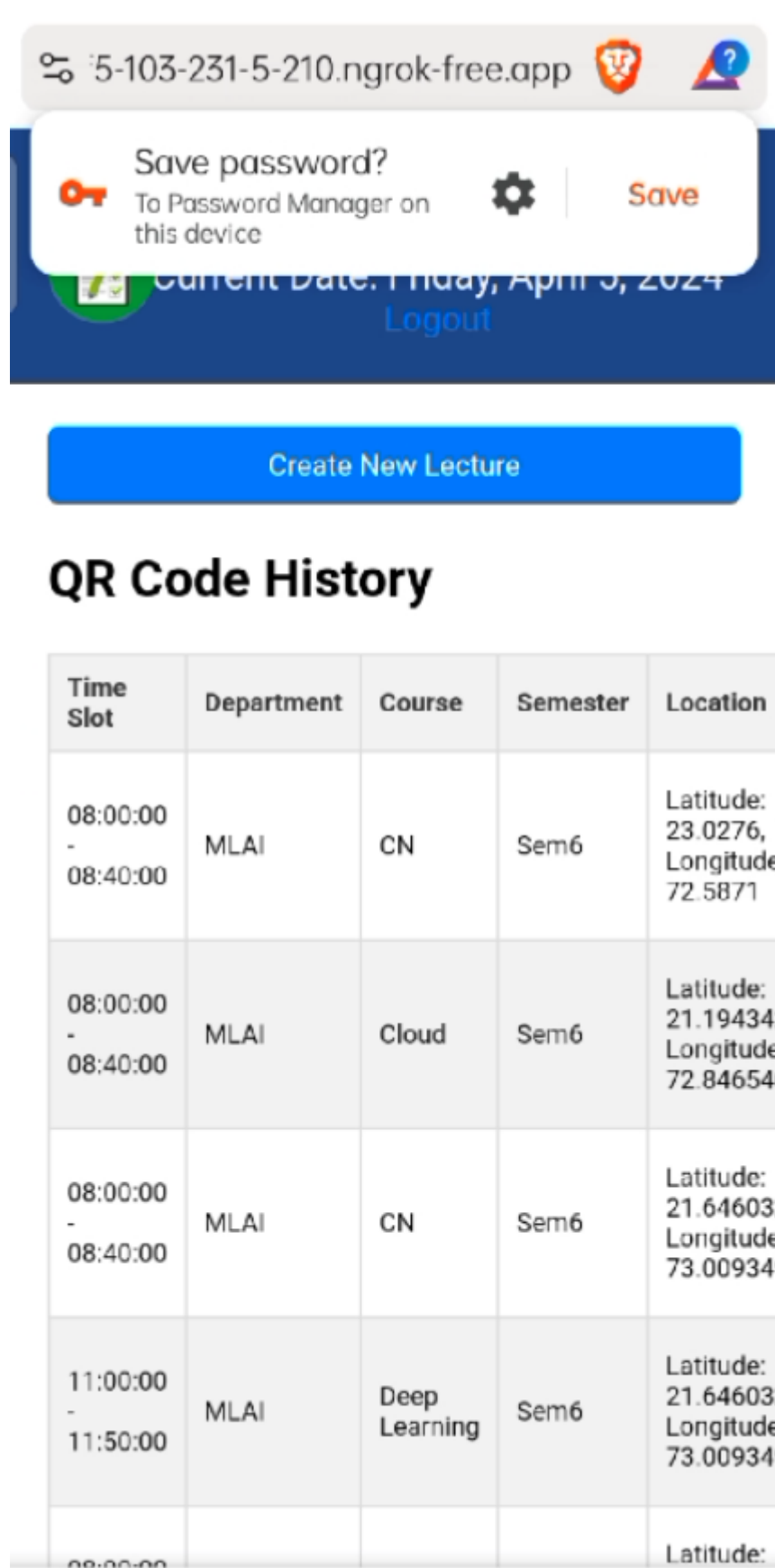





Fig.16: Screenshot 14

 5-103-231-5-210.ngrok-free.app

Welcome, L2

Logout

Generate QR Code

Time Slot:

08:00:00 - 08:40:00

Department:

MLAI

Course:

Cloud

Semester:

Sem6

Location:

Latitude: 21.4974943, Longitude: 73.0096101

Latitude:

21.4974943

Longitude:

73.0096101

Generate QR Code

Fig.17: Screenshot 15



Fig.18: Screenshot 16

CHAPTER 5

CONCLUSION AND FUTURE WORK

CONCLUSION:

The implementation of an attendance management system utilizing QR code technology and location-based services presents a significant advancement in streamlining attendance tracking processes for educational institutions and organizations. Through the integration of frontend technologies such as HTML, CSS, and JavaScript, coupled with backend frameworks like Python with Flask or Django, the system offers a user-friendly interface for teachers, students, and administrators to manage attendance efficiently.

QR codes serve as a convenient and reliable method for marking attendance, while location-based services enhance accuracy and accountability by verifying student presence at designated class locations. Despite the challenges and limitations, such as privacy concerns and dependency on technology, the system demonstrates promising potential in enhancing productivity, reducing administrative burden, and optimizing resource allocation.

FUTURE SCOPE:

1. Enhanced Security Measures:

- Implementing advanced security measures such as encryption and digital signatures to safeguard QR codes and user data against tampering and unauthorized access.

2. Integration of Biometric Authentication:

- Exploring the integration of biometric authentication methods, such as fingerprint or facial recognition, to further enhance security and prevent attendance fraud.

3. Mobile Application Development:

- Developing a mobile application version of the attendance management system to provide users with greater flexibility and accessibility, allowing them to mark attendance on-the-go.

4. Machine Learning for Predictive Analytics:

- Leveraging machine learning algorithms to analyze attendance data and predict patterns or trends, enabling proactive interventions to improve student engagement and academic performance.

5. Integration with Learning Management Systems (LMS):

- Integrating the attendance management system with existing learning management systems to synchronize student information and streamline administrative processes seamlessly.

6. Expansion to Other Use Cases:

- Extending the functionality of the system beyond attendance management to support other use cases, such as event registration, access control, and resource allocation in diverse organizational settings.

REFERENCES

- <https://docs.djangoproject.com/en/5.0/>
- <https://html.com/>
- <https://developer.mozilla.org/en-US/docs/Web/CSS>