LADOKE AKINTOLA UNIVERSITY TECHNOLOGY

SUBMITTED TO
FACULTY OF COMPUTING AND INFORMATICS

DEPARTMENT OF COMPUTER SCIENCE

COURSE TITLE:
Software Engineering

COURSE CODE:
CSC 403

SUPERVISOR:
PROF ISMAILA

JANUARY 2026.

Group 18 Project Topic:
QR-Based Student Attendance Management System

BY:

S/N	NAME	MATRIC NO
1.	Olayiwola Abdussomad Damilola	2022004308
2.	Awoyemi Jedidiah Gbemileke	2022002368
3.	Adisa Mojolaoluwa Marvelous	2022007385
4.	Wahab Aminat Adejoke	2022006479
5.	Adetoyebi Botiwuoluwa Eniola	2022008571
6.	Rabiu Aisha Adeola	2022003194
7.	Gazal Yusuff Olaleye	2022001704
8.	Apata Daniel Oluwaseyi	2022005385
9.	Azeez Aishat Eniola	2022009931
10.	Onaolapo Abdullateef Olalekan	2022010699

Table of Contents Page

DEDICATION

This CSC 403 report is dedicated to the Almighty God, to all the lecturers in the department of Computer Science, the Faculty of Computing and Informatics and to Ladoke Akintola University of Technology (LAUTECH) Citadel of Learning.

ACKNOWLEDGEMENT

Our undivided gratitude goes to the Almighty God, the Creator of heaven and earth and the giver of wisdom, understanding, ideas, time, and strength, all of which have collectively made the successful completion of this project possible.

We also extend our profound appreciation to our able supervisor, Prof. Ismaila, for his invaluable advice, guidance, and support throughout this program. His unwavering dedication and expertise have been a source of inspiration and motivation to us.

Furthermore, we are grateful to the faculty members and staff who contributed in one way or another to the smooth execution of this program. Their assistance, encouragement, and contributions have been instrumental to its success.

Lastly, we express our heartfelt thanks to our colleagues and teammates for their cooperation, teamwork, and commitment, which made the entire experience fulfilling and enriching.

CHAPTER 1

INTRODUCTION

The Student Attendance Tracker is an automated solution designed to replace manual roll-calling in educational environments. By utilizing QR Code technology, the system ensures a fast, paperless, and verifiable method of recording presence. The application is built using Python, leveraging the Gradio framework for the user interface and SQLite for robust data storage.

The primary objective of this project is to provide:

1. A secure registration and login system for both students and teachers.
2. A dynamic QR code generation tool for instructors.
3. A computer-vision-based scanner for students to mark attendance via image upload.
4. Real-time data visualization of attendance records for administrative oversight.

CHAPTER TWO

METHODOLOGY

The development follows a modular approach, separating the system into environment configuration, database management, backend logic, and frontend interface.

2.1 Environment and Dependency Setup

The first part of the code prepares the virtual environment (specifically designed for Google Colab).

!apt-get install -y libzbar0 > /dev/null 2>&1

- **Explanation**: This command installs `libzbar0`, a system-level library in Linux required for the `pyzbar` Python package to decode QR codes and barcodes.
-

```
# !pip install gradio qrcode pillow pyzbar opencv-python-headless -q
```

- **Explanation**: Installs the Python packages: `gradio` (UI), `qrcode` (creating codes), `pillow` (image handling), `pyzbar` (decoding), and `opencv-python-headless` (processing images without a GUI).

2.2 Database Architecture

The system uses an embedded SQLite database to persist data.

Python
```
def setup_database():
    conn = sqlite3.connect("attendance.db", check_same_thread=False)
    cursor = conn.cursor()
```

- **`sqlite3.connect`**: Creates or opens the database file `attendance.db`.
- **`check_same_thread=False`**: Allows multiple Gradio threads to access the database simultaneously without crashing.

```
   cursor.execute("""CREATE TABLE IF NOT EXISTS users (...)""")
    cursor.execute("""CREATE TABLE IF NOT EXISTS attendance
(...)""")
```

- **`CREATE TABLE IF NOT EXISTS`**: Ensures that the `users` table (for accounts) and `attendance` table (for logs) are created only if they don't already exist.

2.3 Authentication Logic

These functions manage user access and state.

Python
```
current_user = {"username": None, "role": None}
```

- **Explanation**: A global dictionary used as a session tracker to remember who is currently logged in and what their permissions (role) are.

Python
```
def register_user(username, password, role):
    ...
    cursor.execute("INSERT INTO users VALUES (?, ?, ?)", (username,
password, role))
```

- **Explanation**: Takes input from the UI and saves a new user to the database. It uses parameterized queries (`?`) to prevent SQL injection attacks.

Python
```
def login_user(username, password):
    cursor.execute("SELECT role FROM users WHERE username=? AND
password=?", (username, password))
    result = cursor.fetchone()
```

- **Explanation**: Searches the database for a matching username/password pair. If found, it fetches the `role` and updates the `current_user` global state.

2.4 QR Code Logic (Core Functionality)

This section handles the generation and decoding of attendance tokens.

Python
```
def generate_qr_code():
    if current_user["role"] != "Teacher": return None, "Error..."
    data = f"ATTENDANCE_{datetime.now().strftime('%Y%m%d_%H%M%S')}"
```

- **Explanation**: First, it checks if the user is a teacher. Then, it generates a unique string combining the word "ATTENDANCE" with a precise timestamp to ensure each session code is unique.

Python
```
def scan_qr_code(image):
    img = np.array(image)
    img = cv2.cvtColor(img, cv2.COLOR_RGB2BGR)
    decoded_objects = decode(img)
```

- **Explanation**: Converts the uploaded image into a NumPy array, changes the color space to BGR (which `pyzbar` prefers), and then uses `decode()` to find the hidden text inside the QR code.

2.5 User Interface (Gradio Blocks)

The frontend is built using a Tabbed layout for organization.

Python
```
with gr.Blocks(theme=gr.themes.Soft(), title="Student Attendance
System") as app:
    with gr.Tabs():
        with gr.Tab("📝 Register"): ...
        with gr.Tab("🔐 Login"): ...
```

- **`gr.Blocks`**: The main container for the UI.
- **`gr.Tabs`**: Creates the navigational menu at the top.
- **`gr.Button.click`**: Connects UI buttons to the Python functions defined earlier (e.g., clicking "Register" triggers `register_user`).

CHAPTER THREE

RESULTS AND EVALUATION

When the application is executed, it generates a public URL via Gradio's sharing feature.

- **Teacher Perspective**: The teacher generates a QR code, which appears on the screen as a PNG image. They can also view a formatted table showing all student names and timestamps.
- **Student Perspective**: The student logs in, uploads the teacher's QR code, and receives a confirmation message: "✅ Attendance recorded". They can also see a personal history of their past scans.

The database successfully handles multiple entries, preventing duplicate usernames and ensuring data integrity between sessions.

CHAPTER FOUR

CONCLUSION AND UML DIAGRAM ANALYSIS

The Student Attendance Tracker project effectively demonstrates the integration of computer vision and database management to solve a common administrative problem. By using Python's extensive library ecosystem, the system provides a high degree of automation with minimal code complexity.

The **Student Attendance Tracker** demonstrates how off-the-shelf Python libraries can be combined to solve real-world administrative problems.

4.1 The Importance of UML Diagrams

To ensure the system is well-architected and scalable, **Unified Modeling Language (UML)** diagrams play a vital role:

- **Use Case Diagram**: This helps define the roles of the "Teacher" and "Student." It clarifies that while both can "Login," only the Teacher can "Generate QR" and "View All Logs," while the Student can "Scan QR".
- **Sequence Diagram**: This is critical for visualizing the attendance marking process. It shows the flow of data from the Student (uploading an image) to the Backend (decoding via OpenCV/pyzbar), then to the Database (verifying and saving), and finally returning a success message to the Student.
- **Class Diagram**: This diagram models the data structures, such as the `User` and `AttendanceRecord` entities, and how they relate to the database management functions.

Future Enhancements

- **Geofencing**: Integrating GPS data to ensure students are in the classroom when scanning.
- **Encryption**: Hashing passwords in the database for better security.
- **Export**: Adding a feature to download attendance logs as CSV/Excel files.