



MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

**Mini Project Report
of
Database Systems Lab (CSE 2262)**

**Attendance management
system using Face
Recognition**

**SUBMITTED
BY**

Kavish Shah - 220962398 - Roll No. 66

**Department of Computer Science and
Engineering
Manipal Institute of Technology, Manipal.
April 2023**



MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

**Manipal
19/04/2024**

CERTIFICATE

This is to certify that the project titled attendance management system is a record of the bonafide work done by **Kavish Shah(Reg.No.220962398)** and **Kabir Bhalodi(Reg.No.220962424)** submitted in partial fulfilment of the requirements for the award of the Degree of Bachelor of Technology (B.Tech.) in COMPUTER SCIENCE & ENGINEERING of Manipal Institute of Technology, Manipal, Karnataka, (A Constituent Institute of Manipal Academy of Higher Education), during the academic year 2022-2023.

Name and Signature of Examiners:

1. Dr. Roopalakshmi R, Associate Professor, CSE Dept.
2. Prof. Tanuja Shailesh, Assistant Professor, CSE Dept.

TABLE OF CONTENTS

CHAPTER 1: ABSTRACT & INTRODUCTION

CHAPTER 2: PROBLEM STATEMENT &
OBJECTIVES

CHAPTER 3: DATABASE SCHEMA

CHAPTER 4: DDL COMMANDS

CHAPTER 5: SQL QUERIES

CHATER 6: UI DESIGN

CHAPTER 7: CONCLUSION

Abstract and Introduction

This report presents a comprehensive overview of an Attendance System developed using face recognition technology and integrated with a SQL database for efficient management of attendance records. The system employs OpenCV and face_recognition libraries for facial recognition, tkinter for the user interface, and SQ for database management.

Attendance management in educational institutions is traditionally a labor-intensive and error-prone task, often reliant on manual processes. The emergence of face recognition technology offers a promising solution to this challenge by automating the attendance marking process. This project aims to harness the power of face recognition in conjunction with a robust database management system to create an efficient and accurate Attendance System. By seamlessly integrating facial recognition capabilities with a user-friendly interface and a reliable database back end, the system seeks to enhance efficiency, accuracy, and convenience in managing attendance records.

Problem Statement

Manual attendance management processes in educational institutions are fraught with inefficiencies, inaccuracies, and administrative burdens. The primary challenges include:

- Time-Consuming Process
- Error-Prone
- Lack of Real-Time Updates
- Data Accessibility

To address these challenges, the Attendance System developed in this project utilizes face recognition technology to automate attendance marking, providing real-time updates and seamless integration with a SQ database. By doing so, the system aims to improve the efficiency, accuracy, and accessibility of attendance management processes in educational institutions.

Data Requirements

- **Student Information:** The system should store comprehensive details of students, including their name, roll number, class, and any other relevant information.
- **Attendance Records:** The system must maintain a database of attendance records, including date, time, student ID, and attendance status (present/absent).

- **Face Templates:** For each student, the system needs to store facial templates generated from their images during enrollment. These templates will be used for face recognition during attendance marking.

Functional Requirements

- **Face Detection and Recognition:** Utilize Python libraries (e.g., OpenCV, face_recognition) to detect and recognize faces captured by the webcam.
- **Database Integration:** Utilize SQL to establish and manage the back end database for storing student information and attendance records.
- **Attendance Marking:** Automatically mark students as present or absent based on the recognition results and update the attendance database accordingly.
- **User Interface:** Develop a user-friendly interface to interact with the system, allowing administrators to view attendance reports and manage student information.

Database Schema

The database system comprises multiple tables designed to store information related to students, their attendance records, and subject-wise attendance. The tables and their relationships are as follows:

1. **students_info table:**

The students_info table serves as the primary repository for student information, storing details such as registration number, name, section, semester, and branch.

Attributes:

- **reg_no:** An integer representing the registration number of the student. It serves as the primary key for this table, ensuring each student has a unique identifier.
- **name:** A varchar field storing the name of the student.
- **section:** A character field indicating the section in which the student is enrolled (e.g., A, B, C).
- **semester:** An integer field representing the current semester of the student.
- **branch:** A varchar field specifying the branch of study for the student (e.g., AIML, CSE).

2. **Subject-wise tables(i.e DBS,OS,Maths tables):**

The database consists of a separate table to keep a track of the attendance of the students in each subject, marking them as present/absent for every required date. Here, for

consideration three subjects are taken : DBS,OS and Maths.

Each of the tables independently are used to mark the attendance of the students enrolled in that subject.

Attributes:

- **reg_no:** An integer field serving as the primary key and foreign key referencing the reg_no field in the students_info table. It establishes a relationship between students and their enrollment in individual subjects.
- In addition ,the table is altered and a new column denoting the date is added every time to record attendance for that particular date. These columns take values either 0 or 1 ,denoting absent and present respectively.

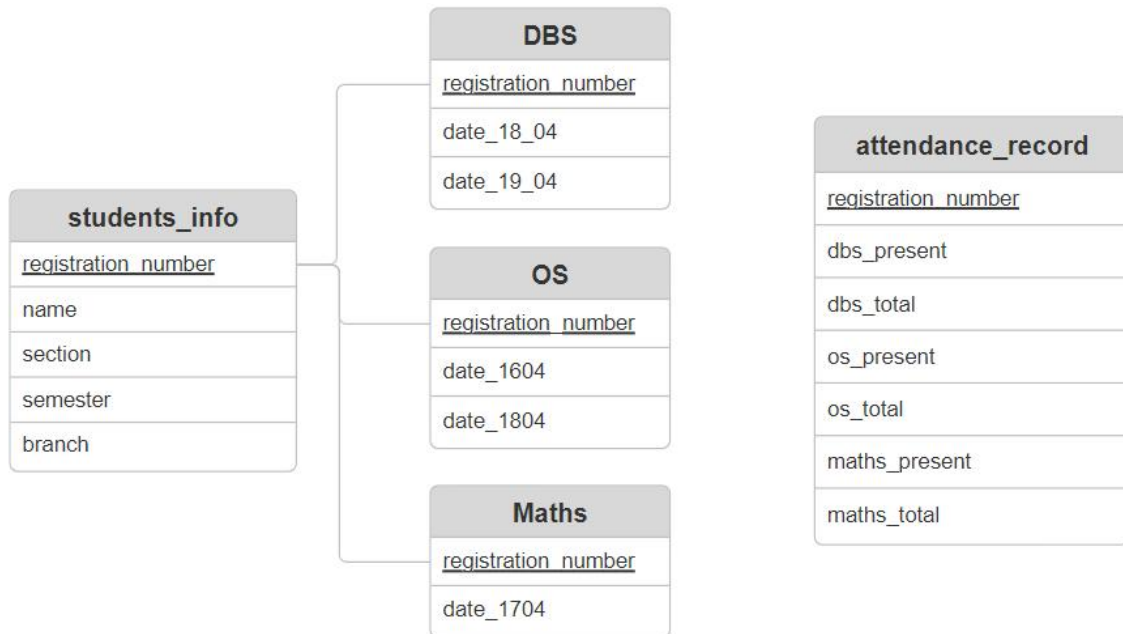
3.attendance_record table :

This table maintains records of student attendance across different subjects to get an analysis of the attendance percentage.

Attributes:

- **reg_no:** An integer field serving as the primary key and foreign key referencing the reg_no field in the students_info table. It establishes a relationship between student attendance records and student information.
- **dbb_present, maths_present, os_present:** Integer fields representing the number of times a student is present in the respective subject.

- **dbb_total, maths_total, os_total:** Integer fields representing the total number of classes conducted for each subject.



Relationships:

- The **students_info** table serves as the parent table, with primary keys referenced as foreign keys in the **dbb**, **maths**, and **os** tables.
- This relationship establishes the enrollment of students in specific subjects.
- The **attendance_record** table is linked to the **students_info** table through the reg_no field, ensuring that attendance records are associated with individual students.

Integrity Constraints:

- **Primary Key Constraint:** Each table has a primary key constraint defined on the reg_no field, ensuring the uniqueness of student identifiers within each table.
- **Foreign Key Constraint:** Foreign key constraints are defined on the reg_no fields in the dbs, maths, os, and attendance_record tables, referencing the reg_no field in the students_info table. This ensures referential integrity, maintaining consistency between related tables.
- **Check constraint :** The check constraint is used to ensure that only 0 or 1 can be input while marking the attendance for each subject.
- **Default Constraint :** It is used to set default of total classes attended/conducted to 0.

DDL Commands

The DDL (Data Definition Language) commands used to create the tables with necessary integrity constraints are as follows:

1. `CREATE TABLE students_info (
 reg_no INTEGER PRIMARY KEY,
 name VARCHAR(20),
 section CHAR(1),
 semester INTEGER,
 branch VARCHAR(30)`

);

```
2.CREATE TABLE dbs (  
    reg_no INTEGER PRIMARY KEY,  
    FOREIGN KEY (reg_no) REFERENCES  
students_info(reg_no)  
);
```

```
CREATE TABLE maths (  
    reg_no INTEGER PRIMARY KEY,  
    FOREIGN KEY (reg_no) REFERENCES  
students_info(reg_no)  
);
```

```
CREATE TABLE os (  
    reg_no INTEGER PRIMARY KEY,  
    FOREIGN KEY (reg_no) REFERENCES  
students_info(reg_no)  
);
```

```
3.CREATE TABLE attendance_record (  
    reg_no INTEGER PRIMARY KEY,  
    dbs_present INTEGER DEFAULT 0,  
    dbs_total INTEGER DEFAULT 0,  
    maths_present INTEGER DEFAULT 0,  
    maths_total INTEGER DEFAULT 0,  
    os_present INTEGER DEFAULT 0,  
    os_total INTEGER DEFAULT 0,  
    FOREIGN KEY (reg_no) REFERENCES  
students_info(reg_no)  
);
```

Triggers

- Triggers are employed in the database to update attendance records whenever changes are made to student enrollment in specific subjects.
- The database uses 3 triggers : update_dbs_record, update_os_record and update_maths_record like:

```
CREATE TRIGGER update_dbs_record
    BEFORE UPDATE ON dbs
    BEGIN
        UPDATE attendance_record
        SET   dbs_present = dbs_present + 1
        WHERE reg_no = NEW.reg_no;
    END;
```

- Each of these 3 triggers are executed before an update operation is performed on the table of that particular subject.
- For instance ,when a new record is inserted into the dbs table (I.e marking a student present for a particular class), then dbs_present field in the attendance_record table is incremented by 1 for the corresponding student.
- It ensures that whenever a student's enrollment in the dbs subject is updated, their attendance record in the attendance_record table is automatically adjusted to reflect the change.

SQL Queries

Various SQL queries are used in the project to give an in-depth and easy way of accessing the attendance of various students for various subjects.

1. UPDATE {subject}

SET {date} = 1 WHERE reg_no = ?

This query updates the attendance record for a specific subject and date by setting the value to 1 (present) for the given registration number (reg_no).

The {subject} and {date} placeholders are replaced with the actual subject name and date column name, respectively.

2. ALTER TABLE {subject}

ADD COLUMN {date} INTEGER DEFAULT 0 CHECK ({date} IN (0, 1));

This query adds a new column ({date}) to the specified {subject} table. The new column is an integer type, with a default value of 0, representing the student's attendance status for the given date. The CHECK constraint ensures that the column values are either 0 (absent) or 1 (present).

3. UPDATE attendance_record

SET {subj_tot} = {subj_tot} + 1;

This query increments the total attendance count for a particular subject in subj_tot in the attendance_record table. It is used to keep track of the total number of attendance sessions for each subject.

4. SELECT * FROM {subject};

This query retrieves all records from the specified {subject} table, including the registration numbers, names, and attendance records for each date.

5. SELECT

```
students_info.reg_no,  
students_info.name,  
attendance_record.dbs_present AS present,  
attendance_record.dbs_total AS total,  
(CAST(attendance_record.dbs_present AS FLOAT)  
/attendance_record.dbs_total)* 100 AS percentage  
FROM students_info JOIN attendance_record  
ON students_info.reg_no = attendance_record.reg_no;
```

This query retrieves information about students' attendance in the "DBS" subject. It selects the registration number, name, present count, total count, and **percentage** of attendance for each student. Similar query is implemented for all the subjects.

6. SELECT

```
students_info.reg_no,  
students_info.name,  
attendance_record.dbs_present AS present,  
attendance_record.dbs_total AS total,  
(CAST(attendance_record.dbs_present AS FLOAT)  
/attendance_record.dbs_total)* 100 AS percentage  
FROM students_info JOIN attendance_record  
ON students_info.reg_no = attendance_record.reg_no;  
where percentage<75%;
```

This query is used to display the attendance of students with less than 75% for a particular subject.

7. SELECT * FROM students_info;

This query retrieves all records from the students_info table, which contains information about the students, such as their registration numbers and names.

8.SELECT

(CAST({subj}_present AS FLOAT) / {subj}_total) * 100 AS {subj},

FROM attendance_record WHERE reg_no = ?;

This query retrieves the percentage attendance for each subject ({subj}) for a specific student identified by the reg_no.

The percentage is calculated by casting the present count to a float, dividing it by the total count, and multiplying by 100.

The results are returned as a row with columns for each subject's percentage attendance.

UI Design

The screenshot displays the user interface of an Attendance Management System. It features several input fields and buttons for managing student attendance records.

At the top, there is a "Date:" field with the value "1904". Below it is a "Subject:" field with the value "Maths".

There are two buttons: "Capture Attendance" and "Stop Capturing".

Below these buttons is a label "Students data :" followed by a "Display:" button.

Further down, there is a field "Enter subject for attendance record:" followed by a "Display Attendance" button.

Below that is a field "Enter subject to view net attendance(with %):" with the value "DBS", followed by a "Display Net Attendance" button and a "< 75%" label.

At the bottom, there is a field "Enter student registration number to view attendance:" followed by a "Display Student Attendance" button.

Below the input fields is a table displaying student data:

	reg_no	name	section	semester	branch
0	1234	Kavish	B	4	AIML
1	1546	RyanGosling	C	4	CSE
2	2348	RyanReynolds	B	6	CSE
3	4832	DwayneJohnson	A	5	CSE
4	5678	Kabir	B	4	AIML
5	7852	VinDiesel	A	4	AIML
6	8556	PaulWalker	A	5	AIML

The User Interface (UI) in the Attendance Management System is designed to provide a user-friendly interaction platform for managing attendance records of students.

The UI is developed using the Tkinter library in Python, offering various functionalities such as capturing attendance, displaying student data, and viewing attendance statistics.

UI Components:

1. Date Entry:

Purpose: Allows the user to enter the date for which attendance needs to be captured.

Implementation: Utilizes a text entry field (Entry) to accept the date input from the user.

2. **Subject Entry:**

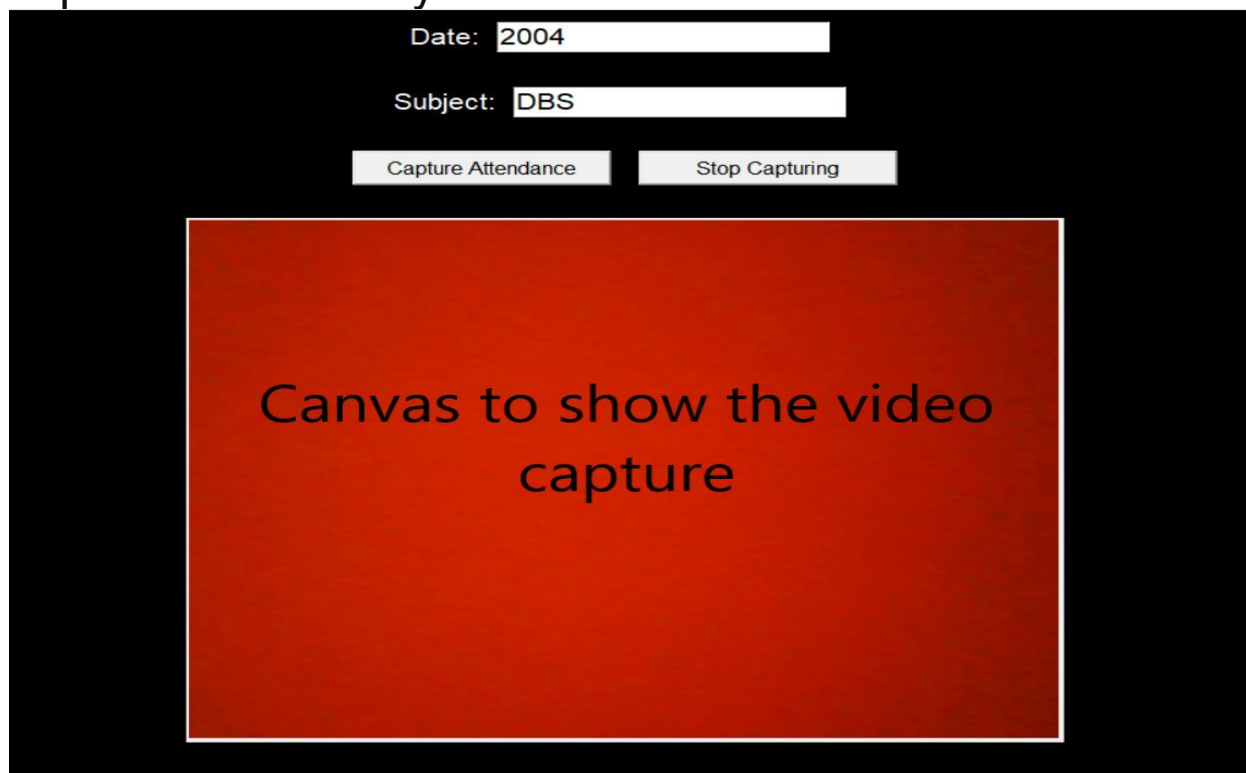
Purpose: Enables the user to specify the subject for which attendance is being recorded.

Implementation: Utilizes another text entry field (Entry) for the user to input the subject name.

3. **Capture Attendance Button:**

Purpose: Triggers the process of capturing attendance for the specified date and subject.

Implementation: Implemented as a button (Button) with a corresponding command to execute the attendance capture functionality.



The screenshot shows a web form with a black background. At the top, there is a label "Date:" followed by a text input field containing "2004". Below this is a label "Subject:" followed by a text input field containing "DBS". Underneath the subject field are two buttons: "Capture Attendance" and "Stop Capturing". In the center of the form is a large red rectangle with a white border, containing the text "Canvas to show the video capture" in black.

4. **Stop Capturing Button:**

Purpose: Stops the process of capturing attendance.

Implementation: Implemented as a button (Button) to halt the attendance capture process.

5. Canvas:

Purpose: Displays the live video feed for capturing student attendance through facial recognition.

Implementation: Utilizes a canvas (Canvas) to display the video feed, which updates in real-time.

6. Display Buttons:

Purpose: Allows the user to view various types of attendance-related information, such as individual student attendance, subject-wise attendance, and net attendance.

Implementation: Implemented as buttons (Button) with corresponding commands to display the requested information.

7. Text Widgets:

Purpose: Displays the retrieved attendance data in a tabular format for easy viewing by the user.

Implementation: Utilizes text widgets (Text) to display the attendance records retrieved from the database.

Features:

1. Attendance Capture: Allows real-time capture of student attendance using facial recognition technology.
2. Data Display: Provides options to display various attendance-related data, including individual student attendance, subject-wise attendance, and net attendance.
3. Ease of Use: Offers a simple and intuitive interface for users to interact with, facilitating easy navigation and operation.

CONCLUSION

In conclusion, the Attendance Management System project offers a comprehensive solution for tracking and managing student attendance, seamlessly integrating image processing, face recognition, and database management technologies. The utilization of SQL as the database management system ensures efficient data storage, retrieval, and manipulation, providing a reliable foundation for storing attendance records.

The SQL queries implemented in the project enable dynamic data management, allowing for the creation, updating, and querying of attendance records with ease. Through SQL, the system can perform operations such as adding new attendance columns for each date, updating attendance status for individual students, and calculating net attendance percentages across different subjects.

Educational institutions can leverage the SQL capabilities of the system to generate comprehensive reports, track student attendance trends, and identify areas for improvement.

Overall, the integration of SQL into the Attendance Management System project underscores its robustness and scalability, offering educational institutions a powerful tool for streamlining attendance management processes and promoting student engagement and accountability.