Project on Java Programming: Attendance systems

1 Introduction

Facial recognition technology has revolutionized attendance systems, replacing traditional methods like roll calls, signatures, or swipe cards with a more efficient approach. This technology uses cameras to capture and analyze facial features to identify individuals accurately. Companies like Google and Amazon are leading examples of its application, with Google implementing facial recognition for security at its Washington campus and Amazon offering these services through AWS for HR management.

For this project, we aim to develop an attendance system that integrates with MySQL database, requiring Java programming skills and basic database management as well as GUI design (`Swing library`). This project bridges real-world application with practical, cutting-edge technology.

Basic Requirement: The basic version of the system operates without facial recognition. Attendance records are managed through manual signatures, designed with an interactive GUI to allow employees to sign in and view attendance details. [database+GUI]

Advanced: The advanced requirement involves integrating facial recognition technology, which allows the system to automatically capture photos of employees and compare them with stored images in the local database to validate attendance. To achieve this, you can use the "OpenCV" library, which provides tools for image processing and face detection. Specifically, you can use OpenCV's pre-trained Haar cascades to detect faces from the camera feed. Once a face is detected, the system can extract the facial region and compare it with stored images using simple methods like histogram comparison or pixel matching, or even more advanced algorithms like "deep learning." If a match is found, the system will mark the employee's attendance automatically. [database+GUI+opencv]

Here's a breakdown of the project tasks for a team of 2-4 developers:

This project is the most challenge one! You are going to use MySQL and OpenCV.

- **Submission Requirements:** Please compile all files into a ZIP folder named `JAVA_project_VR.zip` and include a detailed report explaining the project.
- **My advice on Role Distribution:**
  - **Database Management:** One developer should focus on integrating and managing the MySQL database. – Database Interaction, see CM5
  - **User Interface:** One developer should handle the development of user interface panels. –GUI Design, see CM6
  - **Facial Recognition:** One developer should specialize in implementing facial recognition features using OpenCV. – How to integrate outside library "OpenCV"
- You can use other tools like SQLite, VSCode if you prefer. But I do not prepare a tutorial for integrating these tools.
- Java Swing Components: Use JFrame, JPanel, JButton, JTextField, JDialog, etc., to build the GUI.
- Database Interactions: Use JDBC to connect to the database, execute SQL queries, and update the database.

2 Requirements Analysis

The attendance system is designed to perform three core functions: attendance recognition, priority and maintaining employee records. We will detail these functionalities as follows:

- The attendance recognition:
  - The official working hours start at 10:00 AM. Employees who check in before 10:00 AM are marked as "On Time", those checking in after 10:00 AM are marked as "Late", and employees who fail to check in are recorded as "Absent". Record the daily attendance of each employee into Table 3.

  a) Basic version: Develop a user-friendly GUI that allows employees to check in by entering their employee ID. A validation button will confirm the ID and log the check-in.

  Tips:

  * Interface Design: Use Swing to create the GUI. The main interface could include a `JTextField` for the employee to enter their employee ID, a `Button` labeled "Sign In" below the text field.
  * Event Listener for sign in: Attach an `ActionListener` to the "Sign in" button. When the button is clicked, the listener retrieves the employee ID from the text field.
  * Record sign-in information. Validate the employee ID. If it's correct, record the sign-in time and employee ID. This chould be stored in the Table 3.

  b) Advanced version with face recognition: Integrate a camera to capture an employee's facial data. The system will compare the captured data against stored records in the company's database. If a match is found, it logs the check-in time and displays a confirmation message for successful attendance. New employees are registered by capturing their facial image with the camera using the "Capture" button. This process generates and stores the facial data file necessary for attendance.

  Tips:

  * OpenCV integration: Install OpenCV and configure it with Java. Ensure that the OpenCV's `opencv-xxx.jar` is added to your project's build path and that the native library location is specified.
  * Capture Facial Image: Use OpenCV to interact with the webcam and capture real-time images. Implement a GUI with Java Swing including a "Capture" button to trigger the camera feed.
  * Facial recognition setup: Preprocess and store facial images in a database as a registration process for new employees. Use OpenCV's face detection and recognition algorithms to compare the live captured image with the stored images.

- Priority:
  - The system recognizes two user roles: `admin` and `employee`. Only `admin` users have the ability to register new employees or delete existing records. A dedicated table for registering system administrators can be designed, as shown in Table 4.
  - All users, regardless of role, can access the attendance recognition system.

- Maintaining Employee Records:
  - Employee records are managed using two tables:

    a) `Employee_Information`: This table contains detailed information about each employee, as shown in Table 2. Key fields include `id`, `name`, and `code`. The `code` field stores a unique identifier for each employee's facial features, typically a UUID. For a basic version of the system, the `code` is optional and employees can check in using their ID.

b) `t_lock_in_record`: Documented in Table 3, this table records the daily check-in times alongside the IDs of the employees who checked in.

3 Dataset Analysis

This section describes the data management strategies and database structures used in our facial recognition attendance system. As the attendance system accumulates data over time, efficient data management becomes crucial. We employ a MySQL database to facilitate the addition, deletion, updating, and querying of large data volumes. We create a database named `db_time_attendance` as in Table 1. Inside of `db_time_attendance`, we have four tables:

1. *Employee Information*: Details about each employee, see Table 2. The employee data includes: 1) An `name` to identify the employee; 2) A `unique employee ID` that auto-increments with each new registration; 3) A `photo` file named `feature_code.png`, where 'feature code' is a UUID generated by the system for unique identification.
2. *Attendance Records*: Logs of when employees clock in and out, see Table 3
3. *Administrator Users*: Information on system administrators, see Table 4.
4. *Work Schedule*: Specifies the standard working hours, see Table 5.

Table Name	Illustration	Table Name	Illustration
`t_emp`	Employee Information	`t_lock_in_record`	Attendance Records
`t_admin`	Admin Users Table	`t_work_time`	Work Schedule

Table 1: Database: `db_time_attendance`

Field Name	Data Type	Field Size	Primary Key	Description
id	INT	DEFAULT	YES	Employee ID
name	VARCHAR	20		Employee Name
code	CHAR	36		UUID

Table 2: `t_emp`: Employee Information Table

Field Name	Data Type	Field Size	Primary Key	Description
id	INT	DEFAULT	YES	Employee ID
check_in_time	DATETIME	DEFAULT		Check in Time

Table 3: `t_lock_in_record`: Attendance Records Table

As illustrated in Table 2 and Table 3, employee information and attendance records are organized into two distinct tables. The first table stores basic details such as the employee's name, ID number, and feature code. The second table logs each employee's clock-in times. By separating these data, we enhance both clarity and efficiency, accommodating multiple attendance records for each employee.

4 GUI design

For your attendance system project, here are specific suggestions for the main window design and interactions for administrators and employees:

Field Name	Data Type	Field Size	Primary Key	Description
id	INT	DEFAULT	YES	Admin ID
username	VARCHAR	20		Admin Username
password	VARCHAR	20		Admin Password

Table 4: `t_admin`: Admin User Table

Field Name	Data Type	Field Size	Primary Key	Description
start	TIME	DEFAULT		Check-in Time

Table 5: `t_work_time`: Work Schedule Table

- Main Window Design: The main window should offer a straightforward user interface that allows users to choose whether to log in as an employee or as an administrator:
  - `Employee_Login_Button`: When clicked, a dialog box pops up prompting the employee to enter their ID and record their attendance.
  - `Administrator_Login_Button`: When clicked, it transitions to an administrator login page, where username and password are required for authentication.
- Employee Interface design: When an employee clicks the login button:
  - `Dialog_Box`: employees are asked to enter their ID.
  - `Sign_Button`: After entering their ID, employees click this button to log their sign-in time, which the sytem records in the `t_work_time` table.
  - (optinal) `View_Attendance_Record_Button`: Allows employees to view all their attendance records.
- Administrator interface design: After administrators verify their identity, they should have access to the following functionalities.
  - User Management: add, delete, or edit employee information
  - View all attendance records
  - System settings: change system-related settings, such as the effective time window for attendance (Table 5).
  - Generate reports (optional): create reports on employee attendance, such as monthly attendance summaries.
  - Security (optional): Ensure encryption for administrator logins to keep passwords secure and not transmitted or stored in plaintext.