

**RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI**

**(AN AUTONOMOUS INSTITUTE)**

**A MINI PROJECT BY:**

**ARYAN SAI VENKAT M 230701040**

**ARUL RAJAN 230701035**

**IN PARTIAL FULFILLMENT OF THE AWARD OF THE DEGREE**

**OF**

**BACHELOR OF ENGINEERING IN**

**COMPUTER SCIENCE AND ENGINEERING**

**NOVEMBER 2024**

# BONAFIDE CERTIFICATE

Certified that this project EMPLOYEE ATTENDANCE MANAGEMENT SYSTEM is the bonafide work of **“ARYAN SAI VENKAT, ARUL RAJAN “** who carried out the project work under my supervision.

Submitted for the practical examination held on \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

SIGNATURE

Mrs.B.Deepa

Assistant Professor(SG), Computer Science and Engineering,

Rajalakshmi Engineering College (Autonomous), Thandalam, Chennai-602105.

INTERNAL EXAMINER EXTERNAL EXAMINER **ABSTRACT**

CONTENTS

1. INTRODUCTION
   1. Project Overview
   2. Objective
   3. Scope of the Application
2. SYSTEM SPECIFICATIONS

* 1. Hardware Specifications
  2. Software Specifications

1. SAMPLE CODE
2. SNAPSHOTS

4.1. Home Page

4.2. Edit Page

4.3. Login Page

4.4. Register Page

1. CONCLUSION

ABSTRACT

The **Employee Attendance Management System** (EAMS) is a software solution developed to automate and streamline the process of tracking and managing employee attendance within an organization. Built using Java for its cross-platform capabilities and MySQL for reliable data storage, this system provides a robust platform for recording attendance, generating reports, and analyzing attendance patterns.

The system enables administrators to manage employee records and log daily attendance efficiently. Each employee's attendance status (Present, Absent, or other attendance codes) is recorded in real time and stored in the MySQL database, ensuring accurate and up-to-date records. The system can calculate attendance percentages for each employee, which can be used in performance evaluations, payroll calculations, and compliance reporting. Additionally, the system includes features for leave management, allowing employees to apply for leave and view their attendance history, while HR personnel can review leave requests and track leave balances.

The system’s module calculates attendance, and performance records, and generates data for employees.

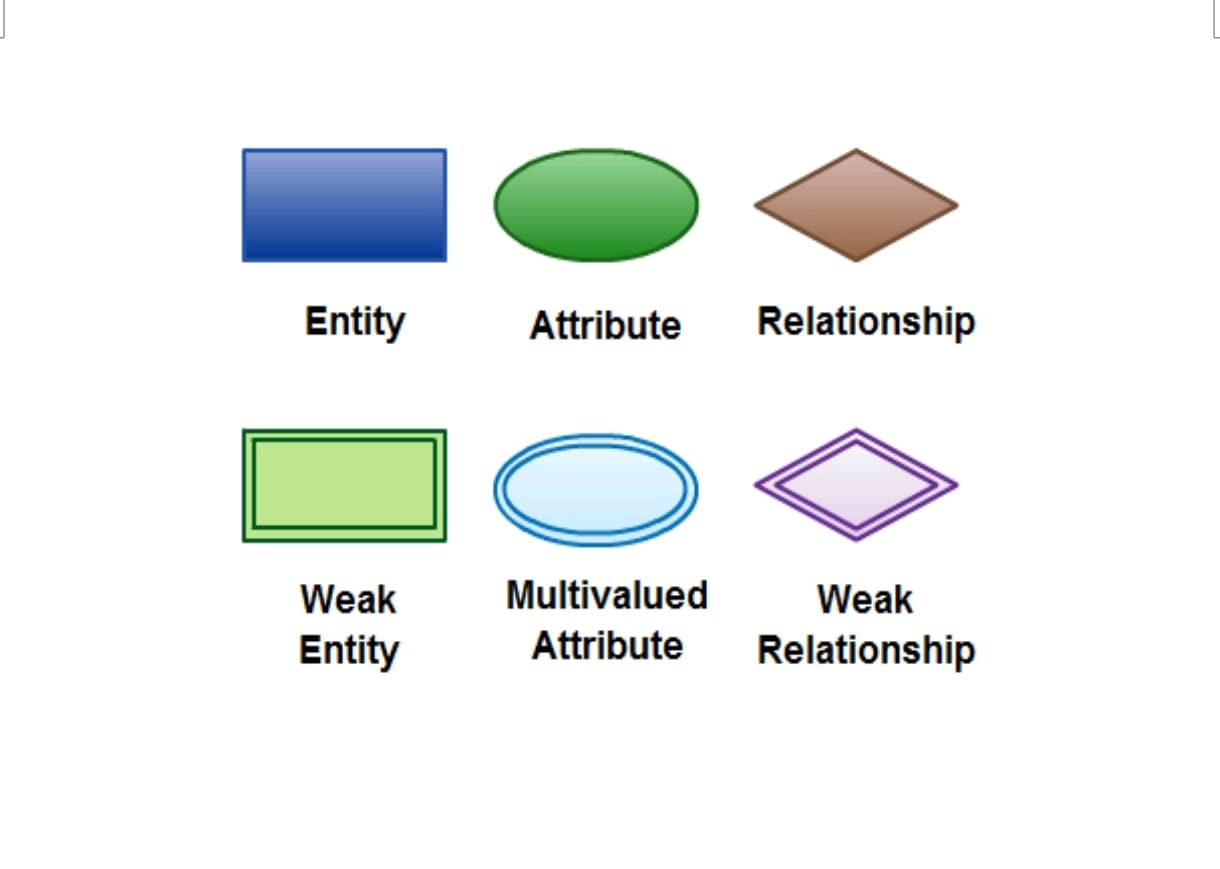
Introduction

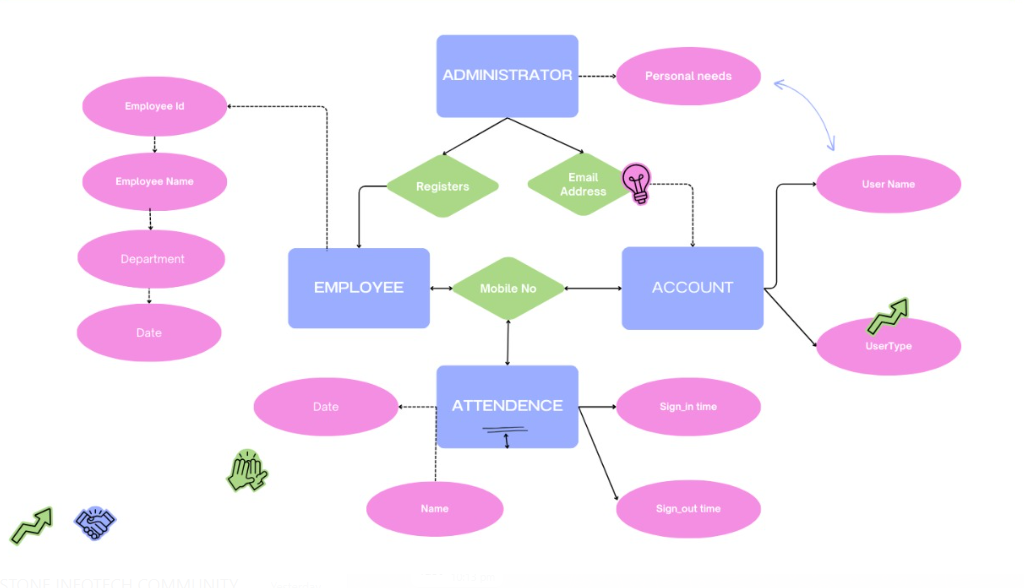
In today's competitive business environment, effective employee management is crucial for organizational success. One of the most essential aspects of managing a workforce is tracking employee attendance, which directly influences productivity, payroll, and overall operational efficiency. Traditionally, attendance tracking has relied on manual systems, which are time-consuming, error-prone, and challenging to manage on a large scale. To address these issues, we have developed a comprehensive Employee Attendance Management System using a full-stack approach.

The Employee Attendance Management System reduces the manual workload associated with attendance tracking, minimizes errors, and enhances data accuracy. By providing a centralized and automated approach to attendance management, the system supports efficient human resource operations and enables better decision-making regarding employee productivity and organizational policies.

ENTITY RELATIONAL DIAGRAM

An entity-relationship (ER) diagram is a specialized graphic that illustrates the interrelationships between entities in a database. ER diagrams often use symbols to represent three different types of information. Boxes are commonly used to represent entities. Diamonds are normally used to represent relationships and ovals are used to represent attributes. If the application is primarily a database application, the entity-relationship approach can be used effectively for modeling some parts of the problem. The main focus in ER modeling is the Data Items in the system and the relationship between them. It aims to create conceptual scheme for the Data from the user’s perspective. The model thus created is independent of any database model. The ER models are frequently represented as ER diagram. Here we present the ER diagram of the above mentioned project.





SCHEMA DIAGRAM

A database schema is the skeleton structure that represents the logical view of the entire database. A database schema defines its entities and the relationship among them. It contains a descriptive detail of the database, which can be depicted by means of schema diagrams. It defines how the data is organized and how the relations among them are associated. It formulates all the constraints that are to be applied on the data.

A database schema defines its entities and the relationship among them. It contains a descriptive detail of the database, which can be depicted by means of schema diagrams. It’s the database designers who design the schema to help programmers understand the database and make it useful.

A database schema can be divided broadly into two categories −

Physical Database Schema − This schema pertains to the actual storage of data and its form of storage like files, indices, etc. It defines how the data will be stored in a secondary storage.

Logical Database Schema − This schema defines all the logical constraints that need to be applied on the data stored.

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class EditEmployeeAttendancePage {

public EditEmployeeAttendancePage(JFrame homeFrame) {

// Create the Edit Employee Attendance Page frame

JFrame attendanceFrame = new JFrame("Edit Employee Attendance");

attendanceFrame.setSize(500, 300); // Adjust size to fit all components

attendanceFrame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

attendanceFrame.setLocationRelativeTo(null); // Center the window

attendanceFrame.setLayout(new GridLayout(6, 2)); // 6 rows, 2 columns grid layout

// Create input fields for employee attendance details

JTextField employeeNameField = new JTextField(20);

JTextField employeeIdField = new JTextField(20);

JTextField departmentField = new JTextField(20);

JTextField statusField = new JTextField(5); // "P" for Present, "A" for Absent

JTextField dateField = new JTextField(10); // Date in format yyyy-MM-dd

// Create labels for each field

JLabel nameLabel = new JLabel("Employee Name:");

JLabel idLabel = new JLabel("Employee ID:");

JLabel departmentLabel = new JLabel("Department:");

JLabel statusLabel = new JLabel("Status (P/A):");

JLabel dateLabel = new JLabel("Date (yyyy-mm-dd):");

// Create submit and back buttons

JButton submitButton = new JButton("Submit");

JButton backButton = new JButton("Back");

// Add components to the frame

attendanceFrame.add(nameLabel);

attendanceFrame.add(employeeNameField);

attendanceFrame.add(idLabel);

attendanceFrame.add(employeeIdField);

attendanceFrame.add(departmentLabel);

attendanceFrame.add(departmentField);

attendanceFrame.add(statusLabel);

attendanceFrame.add(statusField);

attendanceFrame.add(dateLabel);

attendanceFrame.add(dateField);

attendanceFrame.add(submitButton);

attendanceFrame.add(backButton);

// ActionListener for the Submit button

submitButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

// Get the input values from the fields

String employeeName = employeeNameField.getText().trim();

String employeeId = employeeIdField.getText().trim();

String department = departmentField.getText().trim();

String status = statusField.getText().trim().toUpperCase(); // Ensure 'P' or 'A'

String date = dateField.getText().trim();

// Validate the inputs

if (employeeName.isEmpty() || employeeId.isEmpty() || department.isEmpty() ||

status.isEmpty() || date.isEmpty() || !(status.equals("P") || status.equals("A"))) {

JOptionPane.showMessageDialog(attendanceFrame, "Please fill in all fields correctly.");

return;

}

// In a real app, here you'd save the attendance data (e.g., to a database)

// For now, just show a success message

JOptionPane.showMessageDialog(attendanceFrame, "Attendance for " + employeeName + " recorded successfully.");

}

});

// ActionListener for the Back button

backButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

// Close the current Edit Employee Attendance Page

attendanceFrame.dispose();

// Optionally, navigate back to EditPage or any other page you wish

homeFrame.setVisible(true); // Show the EditPage again

}

});

// Make the Edit Employee Attendance Page visible

attendanceFrame.setVisible(true);

}

}

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class EditPage {

public EditPage() {

// Create the Edit Page frame

JFrame editFrame = new JFrame("Edit Page");

editFrame.setSize(400, 300); // Increase size to fit all components

editFrame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

editFrame.setLocationRelativeTo(null); // Center the window

// Create a label to display a message

JLabel label = new JLabel("Choose an option", SwingConstants.CENTER);

label.setFont(new Font("Arial", Font.BOLD, 16)); // Set label font style

editFrame.add(label, BorderLayout.NORTH); // Add label to the top of the frame

// Create a panel for buttons

JPanel buttonPanel = new JPanel();

buttonPanel.setLayout(new FlowLayout()); // Set layout to FlowLayout for horizontal arrangement

// Create buttons

JButton registerButton = new JButton("Register");

JButton attendanceButton = new JButton("Attendance");

// Add buttons to the buttonPanel

buttonPanel.add(registerButton);

buttonPanel.add(attendanceButton);

// Add the buttonPanel to the center of the frame

editFrame.add(buttonPanel, BorderLayout.CENTER);

// Create a Back button and place it at the bottom

JButton backButton = new JButton("Back");

editFrame.add(backButton, BorderLayout.SOUTH);

// Action Listener for Register Button

registerButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

// Open RegisterPage when the Register button is clicked

new RegisterPage(); // Open RegisterPage

editFrame.dispose(); // Close the EditPage frame

}

});

// Action Listener for Attendance Button

attendanceButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

// Open EditEmployeeAttendancePage when the Attendance button is clicked

new EditEmployeeAttendancePage(editFrame); // Open EditEmployeeAttendancePage

editFrame.dispose(); // Close the EditPage frame

}

});

// Action Listener for Back Button

backButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

// Close the current EditPage frame

editFrame.dispose();

// Optionally, show the home screen (this assumes you have a HomePage class)

new HomePage(); // This is a placeholder for your home screen

}

});

// Make the frame visible

editFrame.setVisible(true);

}

public static void main(String[] args) {

// Start the EditPage when the program runs

new EditPage();

}

}

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class HomePage {

public HomePage() {

// Create the Home Page frame

JFrame homeFrame = new JFrame("Home Page");

homeFrame.setSize(500, 300); // Slightly larger frame for better spacing

homeFrame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

homeFrame.setLocationRelativeTo(null); // Center the window

// Create a label to display a welcome message

JLabel label = new JLabel("Welcome to the Home Page", SwingConstants.CENTER);

label.setFont(new Font("Arial", Font.BOLD, 18)); // Set label font style

homeFrame.add(label, BorderLayout.NORTH); // Add label to the top of the frame

// Create a panel for buttons

JPanel buttonPanel = new JPanel();

buttonPanel.setLayout(new FlowLayout()); // Set layout to FlowLayout for horizontal arrangement

// Create buttons

JButton homeButton = new JButton("Home");

JButton editButton = new JButton("Edit");

JButton infoButton = new JButton("Info");

JButton profileButton = new JButton("Profile");

// Add buttons to the buttonPanel

buttonPanel.add(homeButton);

buttonPanel.add(editButton);

buttonPanel.add(infoButton);

buttonPanel.add(profileButton);

// Add the buttonPanel to the top of the frame (instead of bottom)

homeFrame.add(buttonPanel, BorderLayout.NORTH); // Changed from SOUTH to NORTH

// Action Listener for Home Button

homeButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

JOptionPane.showMessageDialog(homeFrame, "You are already on the Home Page!");

}

});

// Action Listener for Edit Button

editButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

// Open the EditPage when Edit Button is clicked

new EditPage(); // Create an instance of EditPage

homeFrame.dispose(); // Close the HomePage

}

});

// Action Listener for Info Button

infoButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

// Open the InfoPage when the Info Button is clicked

new InfoPage(homeFrame); // Pass the homeFrame to InfoPage so we can return to it

homeFrame.setVisible(false); // Hide the HomePage

}

});

// Action Listener for Profile Button (Launch ProfilePage)

profileButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

// Open the ProfilePage when Profile Button is clicked

new ProfilePage(homeFrame); // Create an instance of ProfilePage

}

});

// Create a Logout button (to be positioned at bottom-right)

JButton logoutButton = new JButton("Logout");

// Panel to hold the logout button

JPanel logoutPanel = new JPanel();

logoutPanel.setLayout(new BorderLayout()); // Use BorderLayout to position the button

logoutPanel.add(logoutButton, BorderLayout.EAST); // Position the logout button on the right

// Action Listener for Logout Button

logoutButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

// Dispose of the HomePage frame

homeFrame.dispose();

// Open the LoginPage (you need to implement this)

new LoginPage(); // Assuming you have a LoginPage class

}

});

// Add the logoutPanel to the bottom of the frame

homeFrame.add(logoutPanel, BorderLayout.SOUTH);

// Make the frame visible

homeFrame.setVisible(true);

}

public static void main(String[] args) {

// Start the HomePage when the program runs

new HomePage();

}

}

import java.awt.\*;

import java.sql.\*;

import javax.swing.\*;

import javax.swing.table.DefaultTableModel;

public class InfoPage {

// MySQL database connection details

private static final String DB\_URL = "jdbc:mysql://localhost:3306/EmployeeAttendanceDB"; // Ensure this is correct

private static final String DB\_USERNAME = "root"; // Your DB username

private static final String DB\_PASSWORD = "root"; // Your DB password

public InfoPage(JFrame homeFrame) {

// Create the InfoPage frame with a smaller size

JFrame infoFrame = new JFrame("Employee Attendance Records");

infoFrame.setSize(600, 400); // Adjust the size

infoFrame.setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE); // Close the info window when it's closed

infoFrame.setLocationRelativeTo(null); // Center the window

// Create a label to display the title

JLabel infoLabel = new JLabel("Employee Attendance Records", SwingConstants.CENTER);

infoLabel.setFont(new Font("Arial", Font.BOLD, 16));

// Create a panel for the label

JPanel labelPanel = new JPanel();

labelPanel.setLayout(new BorderLayout());

labelPanel.add(infoLabel, BorderLayout.CENTER);

// Create a panel for the search bar

JPanel searchPanel = new JPanel();

JLabel searchLabel = new JLabel("Search by ID or Name: ");

JTextField searchField = new JTextField(15); // Adjust width of the search field

JButton searchButton = new JButton("Search");

// Add components to the search panel

searchPanel.setLayout(new FlowLayout(FlowLayout.LEFT, 10, 5));

searchPanel.add(searchLabel);

searchPanel.add(searchField);

searchPanel.add(searchButton);

// Create a table to display the attendance records

JTable attendanceTable = new JTable();

JScrollPane scrollPane = new JScrollPane(attendanceTable); // Scrollable table

// Create a DefaultTableModel for the JTable

DefaultTableModel tableModel = new DefaultTableModel(

new Object[]{"Employee ID", "Employee Name", "Department", "Date", "Status"}, 0); // Columns

attendanceTable.setModel(tableModel); // Set model to the table

attendanceTable.setPreferredScrollableViewportSize(new Dimension(550, 250)); // Set table size

// Fetch all attendance records initially

fetchAttendanceData(tableModel, "");

// Action Listener for the Search Button

searchButton.addActionListener(e -> {

String searchQuery = searchField.getText().trim(); // Get the search query from the text field

if (searchQuery.isEmpty()) {

JOptionPane.showMessageDialog(infoFrame, "Please enter Employee ID or Name to search.");

return;

}

// Fetch filtered attendance records based on the search query

fetchAttendanceData(tableModel, searchQuery);

});

// Action Listener for pressing Enter in the search field

searchField.addActionListener(e -> {

String searchQuery = searchField.getText().trim(); // Get the search query from the text field

if (!searchQuery.isEmpty()) {

fetchAttendanceData(tableModel, searchQuery);

}

});

// Create the Back button in the bottom-right corner

JButton backButton = new JButton("Back");

backButton.setPreferredSize(new Dimension(80, 30)); // Set size of the button

backButton.addActionListener(e -> {

infoFrame.dispose(); // Close the current InfoPage window

homeFrame.setVisible(true); // Make the HomePage visible again

});

// Create a panel for the Back button and set its layout to BorderLayout

JPanel backButtonPanel = new JPanel();

backButtonPanel.setLayout(new BorderLayout());

backButtonPanel.add(backButton, BorderLayout.EAST); // Align the button to the right

// Layout of the frame

infoFrame.setLayout(new BorderLayout());

infoFrame.add(labelPanel, BorderLayout.NORTH); // Add title panel

infoFrame.add(searchPanel, BorderLayout.NORTH); // Add search bar panel

infoFrame.add(scrollPane, BorderLayout.CENTER); // Add scrollable table

infoFrame.add(backButtonPanel, BorderLayout.SOUTH); // Add Back button panel

// Make the frame visible

infoFrame.setVisible(true);

}

// Method to fetch attendance data from the database based on a search query

private void fetchAttendanceData(DefaultTableModel tableModel, String searchQuery) {

// SQL query to get the attendance records from the Employee table

String query = "SELECT employee\_id, employee\_name, employee\_department, date, status FROM Employee WHERE " +

"(employee\_id LIKE ? OR employee\_name LIKE ?)";

// Clear the existing data in the table

tableModel.setRowCount(0);

try (Connection connection = DriverManager.getConnection(DB\_URL, DB\_USERNAME, DB\_PASSWORD);

PreparedStatement statement = connection.prepareStatement(query)) {

// Use "%" wildcard for SQL LIKE query

statement.setString(1, "%" + searchQuery + "%");

statement.setString(2, "%" + searchQuery + "%");

ResultSet resultSet = statement.executeQuery();

// Process the result set and add records to the table model

while (resultSet.next()) {

int employeeId = resultSet.getInt("employee\_id");

String employeeName = resultSet.getString("employee\_name");

String department = resultSet.getString("employee\_department");

Date date = resultSet.getDate("date");

String status = resultSet.getString("status");

// Add a new row to the table model

tableModel.addRow(new Object[]{employeeId, employeeName, department, date, status});

}

} catch (SQLException ex) {

ex.printStackTrace();

JOptionPane.showMessageDialog(null, "Error while fetching data from the database: " + ex.getMessage());

}

}

public static void main(String[] args) {

// Create the home frame to pass to the InfoPage

JFrame homeFrame = new JFrame("Home Page");

homeFrame.setSize(500, 300);

homeFrame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

// Make the home frame visible

homeFrame.setLocationRelativeTo(null); // Center the window

homeFrame.setVisible(true);

// Open the InfoPage when the program runs

new InfoPage(homeFrame);

homeFrame.setVisible(false); // Initially hide the home frame

}

}

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class LoginPage {

public LoginPage() {

// Create the Login frame

JFrame loginFrame = new JFrame("Login Page");

loginFrame.setSize(400, 200);

loginFrame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

loginFrame.setLocationRelativeTo(null); // Center the window

// Create a panel for login form

JPanel panel = new JPanel();

panel.setLayout(new GridLayout(3, 2));

// Add labels and text fields for login

panel.add(new JLabel("Username:"));

JTextField usernameField = new JTextField();

panel.add(usernameField);

panel.add(new JLabel("Password:"));

JPasswordField passwordField = new JPasswordField();

panel.add(passwordField);

// Login button

JButton loginButton = new JButton("Login");

panel.add(loginButton);

// Add the panel to the frame

loginFrame.add(panel, BorderLayout.CENTER);

// Action Listener for Login Button

loginButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

// Simulate successful login (you can replace this with actual login logic)

JOptionPane.showMessageDialog(loginFrame, "Login successful!");

// Dispose of the login frame and open HomePage

loginFrame.dispose();

new HomePage(); // Open HomePage after successful login

}

});

// Make the frame visible

loginFrame.setVisible(true);

}

public static void main(String[] args) {

// Start the LoginPage when the program runs

new LoginPage();

}

}

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

public class MySQLConnection {

public static void main(String[] args) {

// Database credentials

String url = "jdbc:mysql://localhost:3306/EmployeeAttendanceDB";

String user = "root"; // Change to your MySQL username

String password = "root"; // Change to your MySQL password

// Try to connect to the database

try {

// Load and register MySQL JDBC driver

Class.forName("com.mysql.cj.jdbc.Driver");

// Establish the connection

Connection connection = DriverManager.getConnection(url, user, password);

// If connection is successful, print success message

System.out.println("Connected to the database successfully!");

// Don't forget to close the connection

connection.close();

} catch (SQLException | ClassNotFoundException e) {

// Handle exceptions

System.err.println("Error connecting to the database: " + e.getMessage());

}

}

}

import java.awt.\*;

import java.sql.\*;

import javax.swing.\*;

public class ProfilePage {

// MySQL database connection details

private static final String DB\_URL = "jdbc:mysql://localhost:3306/EmployeeAttendanceDB"; // Ensure this is correct

private static final String DB\_USERNAME = "root"; // Your DB username

private static final String DB\_PASSWORD = "root"; // Your DB password

public ProfilePage(JFrame homeFrame) {

// Create the ProfilePage frame

JFrame profileFrame = new JFrame("Profile Page");

profileFrame.setSize(500, 400); // Adjust the size to fit content

profileFrame.setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE); // Close on window close

profileFrame.setLocationRelativeTo(null); // Center the window

// Create a label to display the title

JLabel profileLabel = new JLabel("Enter Employee ID or Name", SwingConstants.CENTER);

profileLabel.setFont(new Font("Arial", Font.BOLD, 18));

// Create a panel for input (employee ID or name)

JPanel inputPanel = new JPanel();

inputPanel.setLayout(new FlowLayout());

JLabel employeeIdLabel = new JLabel("Employee ID:");

JTextField employeeIdField = new JTextField(10); // Input field for employee ID or name

JButton loadButton = new JButton("Load Profile");

// Add components to the input panel

inputPanel.add(employeeIdLabel);

inputPanel.add(employeeIdField);

inputPanel.add(loadButton);

// Create the Back button at the bottom-right

JButton backButton = new JButton("Back");

backButton.setPreferredSize(new Dimension(80, 30)); // Set size of the button

backButton.addActionListener(e -> {

profileFrame.dispose(); // Close the profile page

homeFrame.setVisible(true); // Show the home frame again

});

// Create a panel for the Back button and set its layout to BorderLayout

JPanel backButtonPanel = new JPanel();

backButtonPanel.setLayout(new BorderLayout());

backButtonPanel.add(backButton, BorderLayout.EAST); // Align the button to the right

// Layout the components

profileFrame.setLayout(new BorderLayout());

profileFrame.add(profileLabel, BorderLayout.NORTH);

profileFrame.add(inputPanel, BorderLayout.CENTER);

profileFrame.add(backButtonPanel, BorderLayout.SOUTH); // Add Back button panel to the bottom

// Add action listener to load profile when button is clicked

loadButton.addActionListener(e -> {

try {

int employeeId = Integer.parseInt(employeeIdField.getText().trim()); // Get the employee ID

// Open the profile page when employee ID is entered

new ProfilePageDetails(profileFrame, employeeId); // Show profile details based on employee ID

} catch (NumberFormatException ex) {

JOptionPane.showMessageDialog(null, "Please enter a valid Employee ID.");

}

});

// Make the profile frame visible

profileFrame.setVisible(true);

homeFrame.setVisible(false); // Hide the home frame when the profile page is shown

}

public static void main(String[] args) {

// Simulate home frame

JFrame homeFrame = new JFrame("Home Page");

homeFrame.setSize(500, 300);

homeFrame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

// Open the ProfilePage when the program runs

new ProfilePage(homeFrame);

}

}

class ProfilePageDetails {

private static final String DB\_URL = "jdbc:mysql://localhost:3306/EmployeeAttendanceDB";

private static final String DB\_USERNAME = "root";

private static final String DB\_PASSWORD = "root";

public ProfilePageDetails(JFrame profileFrame, int employeeId) {

// Create the ProfilePage frame

JFrame profileDetailFrame = new JFrame("Profile Details");

profileDetailFrame.setSize(500, 400);

profileDetailFrame.setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);

profileDetailFrame.setLocationRelativeTo(null);

JLabel profileLabel = new JLabel("User Profile", SwingConstants.CENTER);

profileLabel.setFont(new Font("Arial", Font.BOLD, 18));

JPanel profilePanel = new JPanel();

profilePanel.setLayout(new GridLayout(9, 2, 10, 10));

JLabel[] profileLabels = new JLabel[9];

String[] labelNames = {

"First Name:", "Last Name:", "Email:", "Phone Number:", "Sex:",

"Qualification:", "Date of Birth:", "Date of Joining:"

};

// Add labels for profile details

for (int i = 0; i < labelNames.length; i++) {

profilePanel.add(new JLabel(labelNames[i]));

profileLabels[i] = new JLabel(); // Initialize the JLabel for displaying data

profilePanel.add(profileLabels[i]);

}

// Fetch user data based on employeeId

fetchUserData(profileLabels, employeeId);

// Create the Back button at the bottom-right

JButton backButton = new JButton("Back");

backButton.setPreferredSize(new Dimension(80, 30));

backButton.addActionListener(e -> {

profileDetailFrame.dispose(); // Close the profile detail page

profileFrame.setVisible(true); // Show the previous frame (home frame)

});

JPanel backButtonPanel = new JPanel();

backButtonPanel.setLayout(new BorderLayout());

backButtonPanel.add(backButton, BorderLayout.EAST);

profileDetailFrame.setLayout(new BorderLayout());

profileDetailFrame.add(profileLabel, BorderLayout.NORTH);

profileDetailFrame.add(profilePanel, BorderLayout.CENTER);

profileDetailFrame.add(backButtonPanel, BorderLayout.SOUTH);

profileDetailFrame.setVisible(true);

}

private void fetchUserData(JLabel[] profileLabels, int employeeId) {

String query = "SELECT first\_name, last\_name, email, phone\_number, sex, qualification, dob, date\_of\_joining "

+ "FROM Register WHERE employee\_id = ?";

try (Connection connection = DriverManager.getConnection(DB\_URL, DB\_USERNAME, DB\_PASSWORD);

PreparedStatement statement = connection.prepareStatement(query)) {

statement.setInt(1, employeeId);

ResultSet resultSet = statement.executeQuery();

if (resultSet.next()) {

// Populate the profile fields with the retrieved data

profileLabels[0].setText(resultSet.getString("first\_name"));

profileLabels[1].setText(resultSet.getString("last\_name"));

profileLabels[2].setText(resultSet.getString("email"));

profileLabels[3].setText(resultSet.getString("phone\_number"));

profileLabels[4].setText(resultSet.getString("sex"));

profileLabels[5].setText(resultSet.getString("qualification"));

profileLabels[6].setText(resultSet.getString("dob"));

profileLabels[7].setText(resultSet.getString("date\_of\_joining"));

} else {

JOptionPane.showMessageDialog(null, "Employee not found.");

}

} catch (SQLException ex) {

ex.printStackTrace();

JOptionPane.showMessageDialog(null, "Error while fetching user data: " + ex.getMessage());

}

}

}

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

import java.sql.\*;

import java.text.SimpleDateFormat;

public class RegisterPage {

// MySQL Database connection parameters

private static final String DB\_URL = "jdbc:mysql://localhost:3306/EmployeeAttendanceDB";

private static final String DB\_USERNAME = "root"; // Replace with your MySQL username

private static final String DB\_PASSWORD = "root"; // Replace with your MySQL password

public RegisterPage() {

// Create the RegisterPage frame

JFrame registerFrame = new JFrame("Register Employee");

registerFrame.setSize(400, 400);

registerFrame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

registerFrame.setLocationRelativeTo(null); // Center the window

// Create a label for the form

JLabel label = new JLabel("Register Employee", SwingConstants.CENTER);

label.setFont(new Font("Arial", Font.BOLD, 16));

registerFrame.add(label, BorderLayout.NORTH);

// Create input fields and labels (for simplicity, I use text fields here)

JPanel panel = new JPanel(new GridLayout(9, 2)); // Grid layout for form fields

JTextField empIdField = new JTextField(20);

JTextField firstNameField = new JTextField(20);

JTextField lastNameField = new JTextField(20);

JTextField emailField = new JTextField(20);

JTextField phoneField = new JTextField(20);

JTextField sexField = new JTextField(20); // Gender field

JTextField qualificationField = new JTextField(20); // Qualification field

JTextField dobField = new JTextField(20); // Date of birth (yyyy-mm-dd)

JTextField dojField = new JTextField(20); // Date of joining (yyyy-mm-dd)

panel.add(new JLabel("Employee ID:"));

panel.add(empIdField); // Emp ID is auto-incremented, so no need for input here

panel.add(new JLabel("First Name:"));

panel.add(firstNameField);

panel.add(new JLabel("Last Name:"));

panel.add(lastNameField);

panel.add(new JLabel("Email:"));

panel.add(emailField);

panel.add(new JLabel("Phone Number:"));

panel.add(phoneField);

panel.add(new JLabel("Sex:"));

panel.add(sexField);

panel.add(new JLabel("Qualification:"));

panel.add(qualificationField);

panel.add(new JLabel("Date of Birth (yyyy-mm-dd):"));

panel.add(dobField);

panel.add(new JLabel("Date of Joining (yyyy-mm-dd):"));

panel.add(dojField);

// Create a submit button

JButton submitButton = new JButton("Register");

// Create an Exit button

JButton exitButton = new JButton("Exit");

// Add action listener to the Exit button

exitButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

registerFrame.dispose(); // Close RegisterPage

new EditPage(); // Open EditPage again

}

});

// Add action listener to the Register button

submitButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

// Get the data entered by the user

String employeeId = empIdField.getText().trim(); // Employee ID is auto-incremented, we can leave it empty

String firstName = firstNameField.getText().trim();

String lastName = lastNameField.getText().trim();

String email = emailField.getText().trim();

String phone = phoneField.getText().trim();

String sex = sexField.getText().trim();

String qualification = qualificationField.getText().trim();

String dob = dobField.getText().trim(); // Date of birth (yyyy-mm-dd)

String doj = dojField.getText().trim(); // Date of joining (yyyy-mm-dd)

// Validate inputs (basic validation)

if (firstName.isEmpty() || lastName.isEmpty() || email.isEmpty() || phone.isEmpty() ||

sex.isEmpty() || qualification.isEmpty() || dob.isEmpty() || doj.isEmpty()) {

JOptionPane.showMessageDialog(registerFrame, "All fields are required.");

return;

}

// Format dates (ensure the date format is correct)

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");

java.sql.Date sqlDob = null;

java.sql.Date sqlDoj = null;

try {

sqlDob = new java.sql.Date(sdf.parse(dob).getTime());

sqlDoj = new java.sql.Date(sdf.parse(doj).getTime());

} catch (Exception ex) {

JOptionPane.showMessageDialog(registerFrame, "Invalid date format. Use yyyy-mm-dd.");

return;

}

// Database connection and insertion

try (Connection conn = DriverManager.getConnection(DB\_URL, DB\_USERNAME, DB\_PASSWORD)) {

// SQL query to insert data into the register table

String sql = "INSERT INTO Register (first\_name, last\_name, email, phone\_number, sex, qualification, dob, date\_of\_joining) " +

"VALUES (?, ?, ?, ?, ?, ?, ?, ?)";

try (PreparedStatement stmt = conn.prepareStatement(sql)) {

// Set the parameters for the SQL query

stmt.setString(1, firstName);

stmt.setString(2, lastName);

stmt.setString(3, email);

stmt.setString(4, phone);

stmt.setString(5, sex);

stmt.setString(6, qualification);

stmt.setDate(7, sqlDob);

stmt.setDate(8, sqlDoj);

// Execute the query

int rowsInserted = stmt.executeUpdate();

// Check if the insertion was successful

if (rowsInserted > 0) {

JOptionPane.showMessageDialog(registerFrame, "Employee registered successfully.");

registerFrame.dispose(); // Close RegisterPage

new EditPage(); // Open EditPage after successful registration

} else {

JOptionPane.showMessageDialog(registerFrame, "Failed to register employee.");

}

} catch (SQLException ex) {

ex.printStackTrace();

JOptionPane.showMessageDialog(registerFrame, "SQL Error: " + ex.getMessage());

}

} catch (SQLException ex) {

ex.printStackTrace();

JOptionPane.showMessageDialog(registerFrame, "Database Error: " + ex.getMessage());

}

}

});

// Add everything to the frame

JPanel buttonPanel = new JPanel(new FlowLayout());

buttonPanel.add(submitButton);

buttonPanel.add(exitButton);

registerFrame.add(panel, BorderLayout.CENTER);

registerFrame.add(buttonPanel, BorderLayout.SOUTH); // Add the buttons at the bottom

// Show the RegisterPage frame

registerFrame.setVisible(true);

}

public static void main(String[] args) {

// Start the RegisterPage when the program runs

new RegisterPage();

}

}

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class SignupPage {

public SignupPage(JFrame loginFrame) { // Accept login page frame to close it when back is pressed

JFrame frame = new JFrame("Sign Up Page");

frame.setSize(400, 300);

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.setLocationRelativeTo(null);

frame.setLayout(new BorderLayout()); // Use BorderLayout for the frame layout

// Create a panel for the form inputs (username, password, and confirm password)

JPanel formPanel = new JPanel(new GridLayout(3, 2)); // 3x2 grid layout for form fields

JTextField usernameField = new JTextField(20);

JPasswordField passwordField = new JPasswordField(20);

JPasswordField confirmPasswordField = new JPasswordField(20);

formPanel.add(new JLabel("Username:"));

formPanel.add(usernameField);

formPanel.add(new JLabel("Password:"));

formPanel.add(passwordField);

formPanel.add(new JLabel("Confirm Password:"));

formPanel.add(confirmPasswordField);

// Add formPanel to the center of the frame

frame.add(formPanel, BorderLayout.CENTER);

// Create buttons panel for Sign Up and Back buttons

JPanel buttonPanel = new JPanel();

JButton signupButton = new JButton("Sign Up");

JButton backButton = new JButton("Back");

// Add buttons to buttonPanel

buttonPanel.add(signupButton);

buttonPanel.add(backButton);

// Add the buttonPanel to the bottom of the frame

frame.add(buttonPanel, BorderLayout.SOUTH);

// Action Listener for Sign Up Button

signupButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

String username = usernameField.getText();

String password = new String(passwordField.getPassword());

String confirmPassword = new String(confirmPasswordField.getPassword());

if (!password.equals(confirmPassword)) {

JOptionPane.showMessageDialog(frame, "Passwords do not match!");

return;

}

JOptionPane.showMessageDialog(frame, "Sign Up Successful!");

frame.dispose(); // Close the signup page

new LoginPage(); // Open the login page

}

});

// Action Listener for Back Button

backButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

frame.dispose(); // Close the signup page

loginFrame.setVisible(true); // Show the login page again

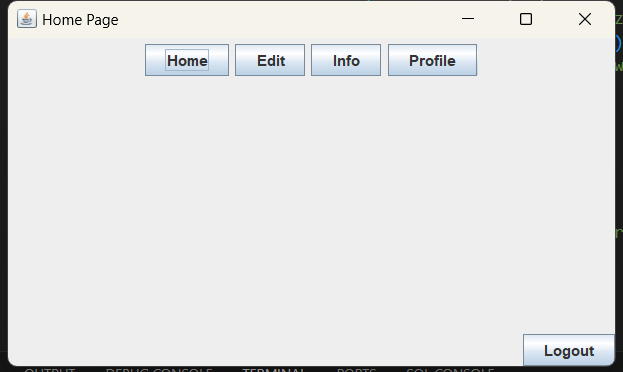
}

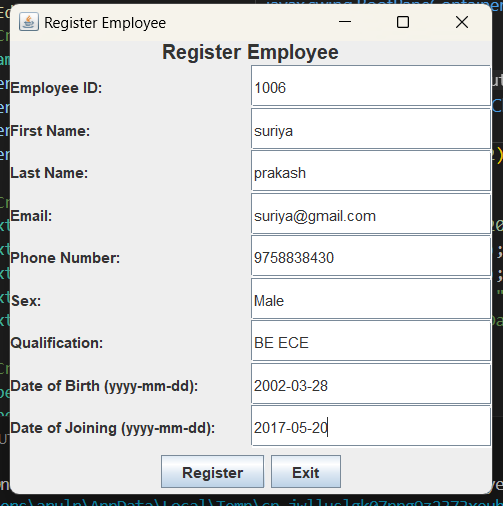
});

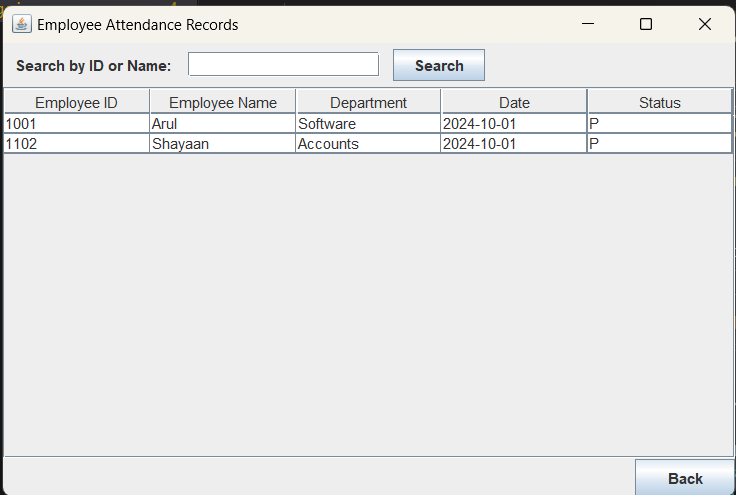
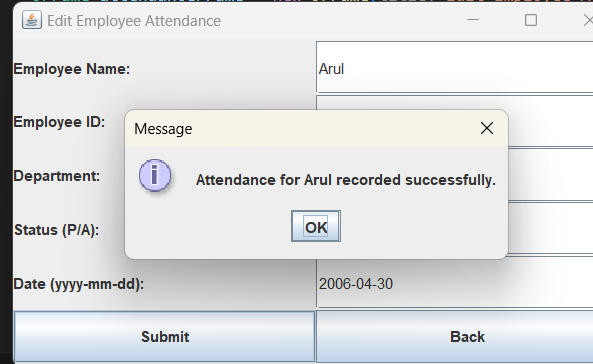
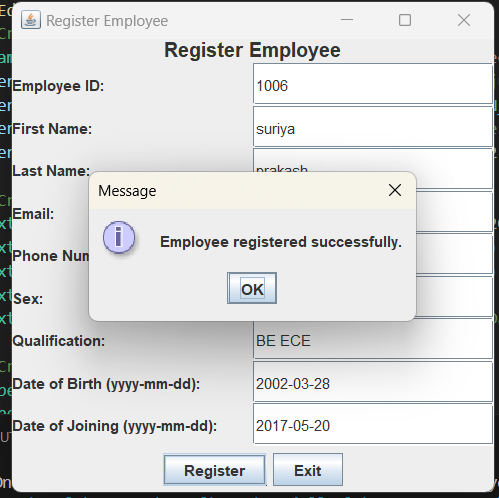
frame.setVisible(true);

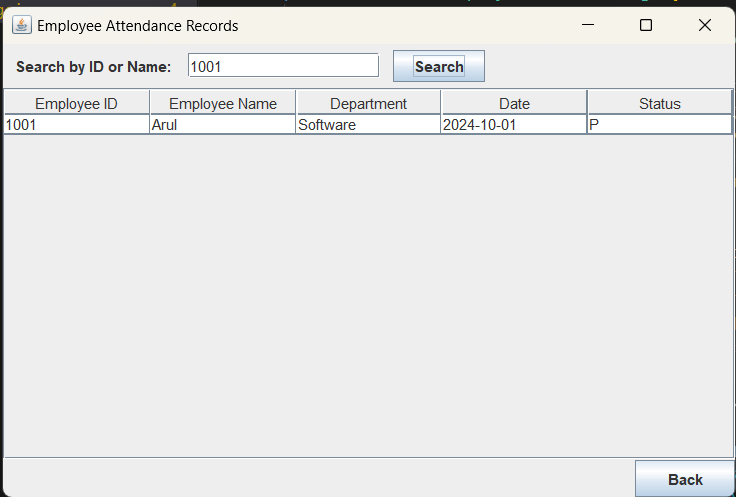
}

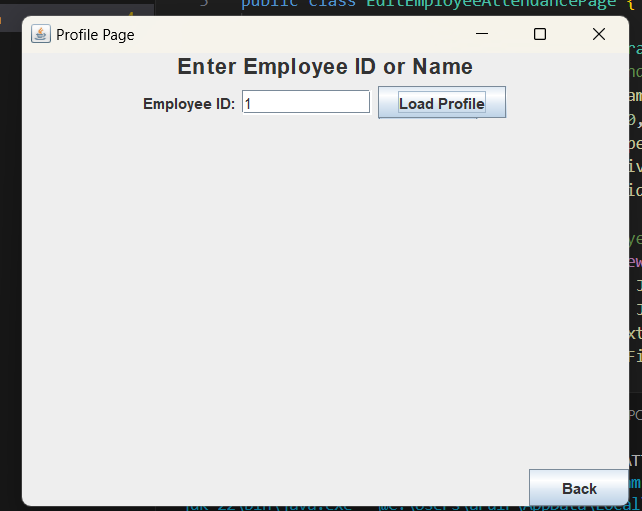
}

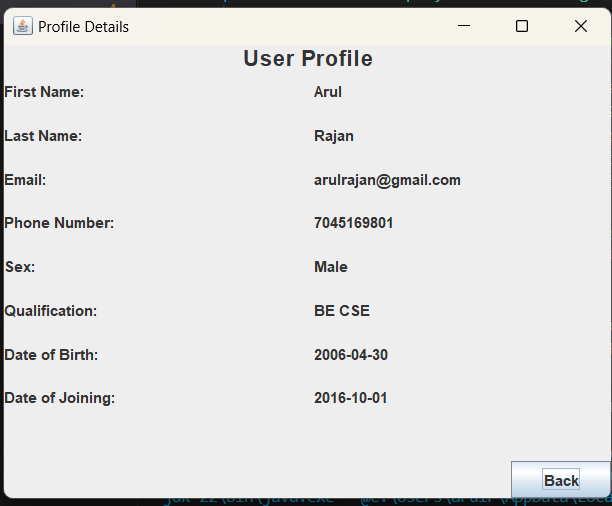












CONCLUSION

The Employee Attendance Management System (EAMS) successfully achieves its objective of automating and simplifying the attendance tracking process for organizations. By implementing this system with Java and MySQL, it provides an efficient, reliable, and secure way to manage employee attendance records, calculate attendance percentages, and generate detailed attendance reports. This system reduces the need for manual entry, minimizes errors, and saves time for HR personnel, leading to a more streamlined and accurate attendance management process.

Through the centralized database, EAMS ensures that employee attendance data is easily accessible and manageable, improving data accuracy and enabling HR and management to make informed decisions. The system’s user-friendly interface and role-based access controls enhance its usability and security, ensuring that only authorized users can access and modify sensitive information.