**EMPLOYEE MANAGEMENT SYSTEM**

**A PROJECT REPORT**

**for**

**Mini Project (KCA353)**

**Session (2023-24)**

**Submitted by**

**MANSI PANWAR**

**(University Roll No. 2200290140087)**

**Submitted in partial fulfillment of the**

**Requirements for the Degree of**

**MASTER OF COMPUTER APPLICATION**

**Under the Supervision of:**

**Dr. AMIT KUMAR**

(ASSISTANT PROFESSOR)



**Submitted to:**

**Department Of Computer Applications**

**KIET GROUP OF INSTITUTIONS, GHAZIABAD**

**UTTAR PRADESH-201206**

**(JANUARY,2024)**

**CERTFICATE**

Certified that **Mansi Panwar 220029014060087** has carried out the project work having “**Employee Management System**” (**Mini Project-KCA353**) for **Master of Computer Application** from Dr. A.P.J. Abdul Kalam Technical University (AKTU**)** (formerly UPTU), Lucknow under my supervision. The project report embodies original work, and studies are carried out by the student herself and the contents of the project report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

**Date:**

**Mansi Panwar (2200290140087)**

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Date:

**Dr. Amit Kumar Dr. Arun Tripathi**

**Assistant Professor Head**

**Department of Computer Applications Department of Computer Applications**

**KIET Group of Institutions, Ghaziabad KIET Group of Institutions, Ghaziabad**

**EMPLOYEE MANAGEMENT SYSTEM**

**MANSI PANWAR**

**ABSTRACT**

Human resource difficulties face all businesses, large and small. Because every organization has different staff management needs, we create custom employee management solutions that are tailored to your needs. This is intended to aid strategic planning and guarantee that your firm has the appropriate degree of human resources to meet your long-term objectives. This approach will help you to better manage your resources in the long run.

Employee management system is an application-based system, having two applications developed, one for employers to manage employee details and another for employees to mark their attendance. Every organization whether government or private uses an information system to store data of their staff. However, in India it is found that many small-scale industries use pen and paper to keep a record. However, there are many advanced technology systems available that can do this work but they all are costly for these low-level industries. This paper discusses making a system for solving problems for them at a cheaper cost. This system will mark attendance of each employee and calculate the salary of them at the end of month. It also calculates overtime and total working hours of each employee. As in small scale each company has their own holidays preference and variable week off for employees, so all this power is given to the employer to manage holidays and week days of each employee separately. It saves lots of time and has no error in pay calculation hence preventing clashes between HR Team and employees. So that both employer and employee can focus on their work to develop their company

**ACKNOWLEDGEMENT**

Success in life is never attained single-handedly. My deepest gratitude goes to my project supervisor, **Dr. Amit Kumar** for his guidance, help, and encouragement throughout my project work. Their enlightening ideas, comments, and suggestions.

Words are not enough to express my gratitude to **Dr. Arun Kumar Tripathi**, Professor and Head, Department of Computer Applications, for his insightful comments and administrative help on various occasions.

Fortunately, I have many understanding friends, who have helped me a lot on many critical conditions.

Finally, my sincere thanks go to my family members and all those who have directly and indirectly provided me with moral support and other kind of help. Without their support, completion of this work would not have been possible in time. They keep my life filled with enjoyment and happiness.

**Mansi Panwar­**

|  |  |  |
| --- | --- | --- |
| **S. No.** | **TABLE OF CONTENTS**  **Topic** | **Page No.** |
| 1 | Declaration | 2 |
| 2 | Abstract | 3 |
| 3 | Acknowledgement | 4 |
| 4 | Chapter-1: Introduction | 7 |
| 5 | Chapter-2: Objective  2.1 Efficiency Enhancement  2.2 Data Accuracy and Security  2.3 Employee Empowerment  2.4 Performance Insights  2.5 Cost Optimization  2.6 Foster a Good Business Environment  2.7 Expected Accuracy and Delicacy | 8 |
| 6 | Chapter-3: Methodology  3.1 NETBEANS  3.2 JAVA  3.3 SWING  3.4 SQL | 9 |
| 7  8 | Chapter-4: Feasibility Study  4.1-Technical Feasibility  4.2-Operational Feasibility  4.3-Economic Feasibility  Chapter-5: Source Code | 10  12-31 |
| 9 | Chapter-6: Implementation and Results  6.1 Login Frame  6.2 Main Dashboard  6.3 Add Employee  6.4 Remove Employee  6.5 View and Update Employee | 32-36 |
| 10 | Chapter-7: ER Diagram | 37 |
| 11 | Chapter-8: Data Flow Diagram | 38 |
| 12 | Chapter-9: Testing | 39 |
| 13  14  15 | Chapter-10: Conclusion  Chapter-11: Future Scope  11.1 Artificial Intelligence Integration  11.2 Employee Experience Enhancement  11.3 Remote Work Support  11.4 Real-Time Performance Analytics  11.5 Blockchain for Security and Transparency  Chapter 12-References | 40  41-42  43 |

|  |  |  |
| --- | --- | --- |
|  |  |  |

**TABLE OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **Figure No.** | **Caption of the Figure** | **Page No.** |
| 1 | Login frame | 11 |
| 2 | Main Dashboard | 12 |
| 3 | Add employee | 12 |
| 4 | Remove employee | 13 |
| 5 | View and Update employee | 14 |

## CHAPTER 1

## INTRODUCTION

Employee management system is an application based system, having two applications developed, one for employers to manage employee details and another for employees to mark their attendance. Every organization whether government or private uses an information system to store data of their staff. However, in India it is found that many small scale industries use pen and paper to keep a record. However, there are many advanced technology systems available that can do this work but they all are costly for these low level industries. This paper discusses making a system for solving problems for them at a cheaper cost. This system will mark attendance of each employee and calculate the salary of them at the end of month. It also calculates overtime and total working hours of each employee. As in small scale each company has their own holidays preference and variable week off for employees, so all this power is given to the employer to manage holidays and week days of each employee separately. It saves lots of time and has no error in pay calculation hence preventing clashes between HR Team and employees. So that both employer and employee can focus on their work to develop their company.

An employee management system is technology designed to streamline core HR services and improve workforce productivity. It accomplishes these goals largely by automating labor-intensive, administrative tasks and using analytics to drive business decisions.

## CHAPTER 2

## OBJECTIVE

The objective of this work is to give a complete approach to personnel information management. This will be accomplished by developing and deploying an HR management system that will result in a significant shift in the way employee data is managed. The purpose of an employee management system is to help improve workforce productivity, identify ways to engage and retain talent, and alleviate administrative burdens for HR professionals. Achieving greater efficiency through the use of technology can also help control costs and minimize compliance risks.

This system's objectives include the following:

**2.1** Efficiency Enhancement

**2.2** Data Accuracy and Security

**2.3** Employee Empowerment

**2.4** Performance Insights

**2.5** Cost Optimization

**2.6** Foster a Good Business Environment

**2.7** Expected Accuracy and Delicacy

**2.8** Assists with pursuing Better Business Choices and Employee Satisfaction

**2.9** Continuous Analysis and Report Management

## CHAPTER 3

## METHODOLOGY

The methodology to complete this project is as follows:

1. I explored net beans, concepts of swings and applets.
2. For further and a deeper understanding, I even referred to some articles, books, journals, websites and news articles.

Below are the important concepts on which the work has been done and with the support of these I was able to work on my project.

**3.1 NET BEANS-** NetBeans is a Java-based integrated development environment (IDE). NetBeans enables the creation of applications using a set of modular software components known as modules. NetBeans is compatible with Windows, Mac OS X, Linux, and Solaris. It also allows other programming languages to be extended. In addition to Java programming, Third-party developers can expand NetBeans-based applications, including the NetBeans IDE.

**3.2 JAVA-** High-level, Object-Oriented programming language which help programmers to run their applications efficiently. JAVA is the programming language which comes into our minds when we talk about android application. By using JAVA as a programming language, programmer can develop any type of android application easily. JAVA also provides many libraries which also helps in making efficient android application. Swing is a Java GUI widget toolkit. It's part of Oracle's Java Foundation Classes (JFC), which provides an API for creating graphical-user- interfaces for Java programmes.

**3.3 SWING-** Swing is a Java GUI widget toolkit. It's part of Oracle's Java Foundation Classes (JFC), which provides an API for creating-graphical-user-interfaces for Java programmes. Swing was created to give a more advanced collection of graphical user interface components than the previous Abstract Window Toolkit (AWT). Swing offers a pluggable look and feel that allows applications to have a look & feel that is unconnected to the underlying platform, as well as a look & feel that emulates the look & feel of numerous platforms.

**3.4 SQL-** SQL (Structured Query Language) is a computer language that is used to manage data in a relational database management system (RDBMS) or for stream processing in a relational data stream management system (RDSMS). It's especially beneficial for dealing with structured data, or data that has relationships between entities and variables.

## CHAPTER 4

## FEASIBILITY STUDY

In order to do a feasibility study, we must consider the following:

* 1. **Technical Feasibility**

The availability of hardware & Software necessary for the creation of the system, as-well- as the compatibility and maturity of the technology planned to be used, and the availability of the requisite technical staff to create the system, are all factors to consider.

* 1. **Operational Feasibility**

Problems that may develop during operations are the focus of operation feasibility. There are two parts to this problem to consider:

* + What are the chances that the solution provided will not be used or will not work?
  + What is the inclination of-the management and end users towards the solution?
  1. **Economic Feasibility**

The concept of economic feasibility is determining whether or not the potential benefit of fixing difficulties is worth-while. Because member needs &alternative solutions haven’t been specified at this point, it is difficult to estimate the cost at this level.

## CHAPTER 5

## SOURCE CODE

## SPLASH.JAVA

## package employee.management.system;

## import javax.swing.\*;

## import java.awt.\*;

## import java.awt.event.\*;

## public class Splash extends JFrame implements ActionListener {

## 

## Splash() {

## 

## getContentPane().setBackground(Color.WHITE);

## setLayout(null);

## 

## JLabel heading = new JLabel("EMPLOYEE MANAGEMENT SYSTEM");

## heading.setBounds(80, 30, 1200, 60);

## heading.setFont(new Font("serif", Font.PLAIN, 60));

## heading.setForeground(Color.RED);

## add(heading);

## 

## ImageIcon i1 = new ImageIcon(ClassLoader.getSystemResource("icons/front.jpg"));

## Image i2 = i1.getImage().getScaledInstance(1100, 700, Image.SCALE\_DEFAULT);

## ImageIcon i3 = new ImageIcon(i2);

## JLabel image = new JLabel(i3);

## image.setBounds(50, 100, 1050, 500);

## add(image);

## 

## JButton clickhere = new JButton("CLICK HERE TO CONTINUE");

## clickhere.setBounds(400, 400, 300, 70);

## clickhere.setBackground(Color.BLACK);

## clickhere.setForeground(Color.WHITE);

## clickhere.addActionListener(this);

## image.add(clickhere);

## 

## 

## setSize(1170, 650);

## setLocation(100, 30);

## setVisible(true);

## 

## while(true) {

## heading.setVisible(false);

## try {

## Thread.sleep(500);

## } catch (Exception e){

## 

## }

## 

## heading.setVisible(true);

## try {

## Thread.sleep(500);

## } catch (Exception e){

## 

## }

## }

## }

## 

## public void actionPerformed(ActionEvent ae) {

## setVisible(false);

## new Login();

## }

## 

## public static void main(String args[]) {

## new Splash();

## }

## }

## ADDEMPLOYEE.JAVA

## package employee.management.system;

## import java.awt.\*;

## import javax.swing.\*;

## import com.toedter.calendar.JDateChooser;

## import java.util.\*;

## import java.awt.event.\*;

## public class AddEmployee extends JFrame implements ActionListener{

## 

## Random ran = new Random();

## int number = ran.nextInt(999999);

## 

## JTextField tfname, tffname, tfaddress, tfphone, tfaadhar, tfemail, tfsalary, tfdesignation;

## JDateChooser dcdob;

## JComboBox cbeducation;

## JLabel lblempId;

## JButton add, back;

## 

## AddEmployee() {

## getContentPane().setBackground(Color.WHITE);

## setLayout(null);

## 

## JLabel heading = new JLabel("Add Employee Detail");

## heading.setBounds(320, 30, 500, 50);

## heading.setFont(new Font("SAN\_SERIF", Font.BOLD, 25));

## add(heading);

## 

## JLabel labelname = new JLabel("Name");

## labelname.setBounds(50, 150, 150, 30);

## labelname.setFont(new Font("serif", Font.PLAIN, 20));

## add(labelname);

## 

## tfname = new JTextField();

## tfname.setBounds(200, 150, 150, 30);

## add(tfname);

## 

## JLabel labelfname = new JLabel("Father's Name");

## labelfname.setBounds(400, 150, 150, 30);

## labelfname.setFont(new Font("serif", Font.PLAIN, 20));

## add(labelfname);

## 

## tffname = new JTextField();

## tffname.setBounds(600, 150, 150, 30);

## add(tffname);

## 

## JLabel labeldob = new JLabel("Date of Birth");

## labeldob.setBounds(50, 200, 150, 30);

## labeldob.setFont(new Font("serif", Font.PLAIN, 20));

## add(labeldob);

## 

## dcdob = new JDateChooser();

## dcdob.setBounds(200, 200, 150, 30);

## add(dcdob);

## 

## JLabel labelsalary = new JLabel("Salary");

## labelsalary.setBounds(400, 200, 150, 30);

## labelsalary.setFont(new Font("serif", Font.PLAIN, 20));

## add(labelsalary);

## 

## tfsalary = new JTextField();

## tfsalary.setBounds(600, 200, 150, 30);

## add(tfsalary);

## 

## JLabel labeladdress = new JLabel("Address");

## labeladdress.setBounds(50, 250, 150, 30);

## labeladdress.setFont(new Font("serif", Font.PLAIN, 20));

## add(labeladdress);

## 

## tfaddress = new JTextField();

## tfaddress.setBounds(200, 250, 150, 30);

## add(tfaddress);

## 

## JLabel labelphone = new JLabel("Phone");

## labelphone.setBounds(400, 250, 150, 30);

## labelphone.setFont(new Font("serif", Font.PLAIN, 20));

## add(labelphone);

## 

## tfphone = new JTextField();

## tfphone.setBounds(600, 250, 150, 30);

## add(tfphone);

## 

## JLabel labelemail = new JLabel("Email");

## labelemail.setBounds(50, 300, 150, 30);

## labelemail.setFont(new Font("serif", Font.PLAIN, 20));

## add(labelemail);

## 

## tfemail = new JTextField();

## tfemail.setBounds(200, 300, 150, 30);

## add(tfemail);

## 

## JLabel labeleducation = new JLabel("Higest Education");

## labeleducation.setBounds(400, 300, 150, 30);

## labeleducation.setFont(new Font("serif", Font.PLAIN, 20));

## add(labeleducation);

## 

## String courses[] = {"BBA", "BCA", "BA", "BSC", "B.COM", "BTech", "MBA", "MCA", "MA", "MTech", "MSC", "PHD"};

## cbeducation = new JComboBox(courses);

## cbeducation.setBackground(Color.WHITE);

## cbeducation.setBounds(600, 300, 150, 30);

## add(cbeducation);

## 

## JLabel labeldesignation = new JLabel("Designation");

## labeldesignation.setBounds(50, 350, 150, 30);

## labeldesignation.setFont(new Font("serif", Font.PLAIN, 20));

## add(labeldesignation);

## 

## tfdesignation = new JTextField();

## tfdesignation.setBounds(200, 350, 150, 30);

## add(tfdesignation);

## 

## JLabel labelaadhar = new JLabel("Aadhar Number");

## labelaadhar.setBounds(400, 350, 150, 30);

## labelaadhar.setFont(new Font("serif", Font.PLAIN, 20));

## add(labelaadhar);

## 

## tfaadhar = new JTextField();

## tfaadhar.setBounds(600, 350, 150, 30);

## add(tfaadhar);

## 

## JLabel labelempId = new JLabel("Employee id");

## labelempId.setBounds(50, 400, 150, 30);

## labelempId.setFont(new Font("serif", Font.PLAIN, 20));

## add(labelempId);

## 

## lblempId = new JLabel("" + number);

## lblempId.setBounds(200, 400, 150, 30);

## lblempId.setFont(new Font("serif", Font.PLAIN, 20));

## add(lblempId);

## 

## add = new JButton("Add Details");

## add.setBounds(250, 550, 150, 40);

## add.addActionListener(this);

## add.setBackground(Color.BLACK);

## add.setForeground(Color.WHITE);

## add(add);

## 

## back = new JButton("Back");

## back.setBounds(450, 550, 150, 40);

## back.addActionListener(this);

## back.setBackground(Color.BLACK);

## back.setForeground(Color.WHITE);

## add(back);

## 

## setSize(900, 700);

## setLocation(300, 50);

## setVisible(true);

## }

## 

## public void actionPerformed(ActionEvent ae) {

## if (ae.getSource() == add) {

## String name = tfname.getText();

## String fname = tffname.getText();

## String dob = ((JTextField) dcdob.getDateEditor().getUiComponent()).getText();

## String salary = tfsalary.getText();

## String address = tfaddress.getText();

## String phone = tfphone.getText();

## String email = tfemail.getText();

## String education = (String) cbeducation.getSelectedItem();

## String designation = tfdesignation.getText();

## String aadhar = tfaadhar.getText();

## String empId = lblempId.getText();

## 

## try {

## Conn conn = new Conn();

## String query = "insert into employee values('"+name+"', '"+fname+"', '"+dob+"', '"+salary+"', '"+address+"', '"+phone+"', '"+email+"', '"+education+"', '"+designation+"', '"+aadhar+"', '"+empId+"')";

## conn.s.executeUpdate(query);

## JOptionPane.showMessageDialog(null, "Details added successfully");

## setVisible(false);

## new Home();

## } catch (Exception e) {

## e.printStackTrace();

## }

## } else {

## setVisible(false);

## new Home();

## }

## }

## public static void main(String[] args) {

## new AddEmployee();

## }

## }

## HOME.JAVA

package employee.management.system;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class Home extends JFrame implements ActionListener{

JButton view, add, update, remove;

Home() {

setLayout(null);

ImageIcon i1 = new ImageIcon(ClassLoader.getSystemResource("icons/home.jpg"));

Image i2 = i1.getImage().getScaledInstance(1120, 630, Image.SCALE\_DEFAULT);

ImageIcon i3 = new ImageIcon(i2);

JLabel image = new JLabel(i3);

image.setBounds(0, 0, 1050, 630);

add(image);

JLabel heading = new JLabel("Employee Management System");

heading.setBounds(620, 20, 400, 40);

heading.setFont(new Font("Raleway", Font.BOLD, 25));

image.add(heading);

add = new JButton("Add Employee");

add.setBounds(650, 80, 150, 40);

add.addActionListener(this);

image.add(add);

view = new JButton("View Employees");

view.setBounds(820, 80, 150, 40);

view.addActionListener(this);

image.add(view);

update = new JButton("Update Employee");

update.setBounds(650, 140, 150, 40);

update.addActionListener(this);

image.add(update);

remove = new JButton("Remove Employee");

remove.setBounds(820, 140, 150, 40);

remove.addActionListener(this);

image.add(remove);

setSize(1120, 630);

setLocation(100, 30);

setVisible(true);

}

public void actionPerformed(ActionEvent ae) {

if (ae.getSource() == add) {

setVisible(false);

new AddEmployee();

} else if (ae.getSource() == view) {

setVisible(false);

new ViewEmployee();

} else if (ae.getSource() == update) {

setVisible(false);

new ViewEmployee();

} else {

setVisible(false);

new RemoveEmployee();

}

}

public static void main(String[] args) {

new Home();

}

}

## 

## LOGIN.JAVA

## package employee.management.system;

## 

## import javax. swing.\*;

## import java.awt.\*;

## import java.awt.event.\*;

## import java.sql.\*;

## public class Login extends JFrame implements ActionListener{

## 

## JTextField tfusername, tfpassword;

## 

## Login() {

## 

## getContentPane().setBackground(Color.WHITE);

## setLayout(null);

## 

## JLabel lblusername = new JLabel("Username");

## lblusername.setBounds(40, 20, 100, 30);

## add(lblusername);

## 

## tfusername = new JTextField();

## tfusername.setBounds(150, 20, 150, 30);

## add(tfusername);

## 

## JLabel lblpassword = new JLabel("Password");

## lblpassword.setBounds(40, 70, 100, 30);

## add(lblpassword);

## 

## tfpassword = new JTextField();

## tfpassword.setBounds(150, 70, 150, 30);

## add(tfpassword);

## 

## JButton login = new JButton("LOGIN");

## login.setBounds(150, 140, 150, 30);

## login.setBackground(Color.BLACK);

## login.setForeground(Color.WHITE);

## login.addActionListener(this);

## add(login);

## 

## ImageIcon i1 = new ImageIcon(ClassLoader.getSystemResource("icons/second.jpg"));

## Image i2 = i1.getImage().getScaledInstance(200, 200, Image.SCALE\_DEFAULT);

## ImageIcon i3 = new ImageIcon(i2);

## JLabel image = new JLabel(i3);

## image.setBounds(350, 0, 200, 200);

## add(image);

## 

## setSize(600, 300);

## setLocation(450, 200);

## setVisible(true);

## }

## 

## public void actionPerformed(ActionEvent ae) {

## try {

## String username = tfusername.getText();

## String password = tfpassword.getText();

## 

## Conn c = new Conn();

## String query = "select \* from login where username = '"+username+"' and password = '"+password+"'";

## 

## ResultSet rs = c.s.executeQuery(query);

## if (rs.next()) {

## setVisible(false);

## new Home();

## } else {

## JOptionPane.showMessageDialog(null, "Invalid username or password");

## setVisible(false);

## }

## } catch (Exception e) {

## e.printStackTrace();

## }

## }

## 

## public static void main(String[] args) {

## new Login();

## }

## }

## CONN.JAVA

## package employee.management.system;

## import java.sql.\*;

## public class Conn {

## 

## Connection c;

## Statement s;

## public Conn () {

## try {

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

## c = DriverManager.getConnection("jdbc:mysql:///employeemanagementsystem", "root", "PHW#84#jeor");

## s = c.createStatement();

## } catch (Exception e) {

## e.printStackTrace();

## }

## }

## }

## UPDATEEMPLOYEE.JAVA

## package employee.management.system;

## import java.awt.\*;

## import javax.swing.\*;

## import java.awt.event.\*;

## import java.sql.\*;

## public class UpdateEmployee extends JFrame implements ActionListener{

## 

## JTextField tfeducation, tffname, tfaddress, tfphone, tfaadhar, tfemail, tfsalary, tfdesignation;

## JLabel lblempId;

## JButton add, back;

## String empId;

## 

## UpdateEmployee(String empId) {

## this.empId = empId;

## getContentPane().setBackground(Color.WHITE);

## setLayout(null);

## 

## JLabel heading = new JLabel("Update Employee Detail");

## heading.setBounds(320, 30, 500, 50);

## heading.setFont(new Font("SAN\_SERIF", Font.BOLD, 25));

## add(heading);

## 

## JLabel labelname = new JLabel("Name");

## labelname.setBounds(50, 150, 150, 30);

## labelname.setFont(new Font("serif", Font.PLAIN, 20));

## add(labelname);

## 

## JLabel lblname = new JLabel();

## lblname.setBounds(200, 150, 150, 30);

## add(lblname);

## 

## JLabel labelfname = new JLabel("Father's Name");

## labelfname.setBounds(400, 150, 150, 30);

## labelfname.setFont(new Font("serif", Font.PLAIN, 20));

## add(labelfname);

## 

## tffname = new JTextField();

## tffname.setBounds(600, 150, 150, 30);

## add(tffname);

## 

## JLabel labeldob = new JLabel("Date of Birth");

## labeldob.setBounds(50, 200, 150, 30);

## labeldob.setFont(new Font("serif", Font.PLAIN, 20));

## add(labeldob);

## 

## JLabel lbldob = new JLabel();

## lbldob.setBounds(200, 200, 150, 30);

## add(lbldob);

## 

## JLabel labelsalary = new JLabel("Salary");

## labelsalary.setBounds(400, 200, 150, 30);

## labelsalary.setFont(new Font("serif", Font.PLAIN, 20));

## add(labelsalary);

## 

## tfsalary = new JTextField();

## tfsalary.setBounds(600, 200, 150, 30);

## add(tfsalary);

## 

## JLabel labeladdress = new JLabel("Address");

## labeladdress.setBounds(50, 250, 150, 30);

## labeladdress.setFont(new Font("serif", Font.PLAIN, 20));

## add(labeladdress);

## 

## tfaddress = new JTextField();

## tfaddress.setBounds(200, 250, 150, 30);

## add(tfaddress);

## 

## JLabel labelphone = new JLabel("Phone");

## labelphone.setBounds(400, 250, 150, 30);

## labelphone.setFont(new Font("serif", Font.PLAIN, 20));

## add(labelphone);

## 

## tfphone = new JTextField();

## tfphone.setBounds(600, 250, 150, 30);

## add(tfphone);

## 

## JLabel labelemail = new JLabel("Email");

## labelemail.setBounds(50, 300, 150, 30);

## labelemail.setFont(new Font("serif", Font.PLAIN, 20));

## add(labelemail);

## 

## tfemail = new JTextField();

## tfemail.setBounds(200, 300, 150, 30);

## add(tfemail);

## 

## JLabel labeleducation = new JLabel("Higest Education");

## labeleducation.setBounds(400, 300, 150, 30);

## labeleducation.setFont(new Font("serif", Font.PLAIN, 20));

## add(labeleducation);

## 

## tfeducation = new JTextField();

## tfeducation.setBounds(600, 300, 150, 30);

## add(tfeducation);

## 

## JLabel labeldesignation = new JLabel("Designation");

## labeldesignation.setBounds(50, 350, 150, 30);

## labeldesignation.setFont(new Font("serif", Font.PLAIN, 20));

## add(labeldesignation);

## 

## tfdesignation = new JTextField();

## tfdesignation.setBounds(200, 350, 150, 30);

## add(tfdesignation);

## 

## JLabel labelaadhar = new JLabel("Aadhar Number");

## labelaadhar.setBounds(400, 350, 150, 30);

## labelaadhar.setFont(new Font("serif", Font.PLAIN, 20));

## add(labelaadhar);

## 

## JLabel lblaadhar = new JLabel();

## lblaadhar.setBounds(600, 350, 150, 30);

## add(lblaadhar);

## 

## JLabel labelempId = new JLabel("Employee id");

## labelempId.setBounds(50, 400, 150, 30);

## labelempId.setFont(new Font("serif", Font.PLAIN, 20));

## add(labelempId);

## 

## lblempId = new JLabel();

## lblempId.setBounds(200, 400, 150, 30);

## lblempId.setFont(new Font("serif", Font.PLAIN, 20));

## add(lblempId);

## 

## try {

## Conn c = new Conn();

## String query = "select \* from employee where empId = '"+empId+"'";

## ResultSet rs = c.s.executeQuery(query);

## while(rs.next()) {

## lblname.setText(rs.getString("name"));

## tffname.setText(rs.getString("fname"));

## lbldob.setText(rs.getString("dob"));

## tfaddress.setText(rs.getString("address"));

## tfsalary.setText(rs.getString("salary"));

## tfphone.setText(rs.getString("phone"));

## tfemail.setText(rs.getString("email"));

## tfeducation.setText(rs.getString("education"));

## lblaadhar.setText(rs.getString("aadhar"));

## lblempId.setText(rs.getString("empId"));

## tfdesignation.setText(rs.getString("designation"));

## 

## }

## } catch (Exception e) {

## e.printStackTrace();

## }

## 

## add = new JButton("Update Details");

## add.setBounds(250, 550, 150, 40);

## add.addActionListener(this);

## add.setBackground(Color.BLACK);

## add.setForeground(Color.WHITE);

## add(add);

## 

## back = new JButton("Back");

## back.setBounds(450, 550, 150, 40);

## back.addActionListener(this);

## back.setBackground(Color.BLACK);

## back.setForeground(Color.WHITE);

## add(back);

## 

## setSize(900, 700);

## setLocation(300, 50);

## setVisible(true);

## }

## 

## public void actionPerformed(ActionEvent ae) {

## if (ae.getSource() == add) {

## String fname = tffname.getText();

## String salary = tfsalary.getText();

## String address = tfaddress.getText();

## String phone = tfphone.getText();

## String email = tfemail.getText();

## String education = tfeducation.getText();

## String designation = tfdesignation.getText();

## 

## try {

## Conn conn = new Conn();

## String query = "update employee set fname = '"+fname+"', salary = '"+salary+"', address = '"+address+"', phone = '"+phone+"', email = '"+email+"', education = '"+education+"', designation = '"+designation+"' where empId = '"+empId+"'";

## conn.s.executeUpdate(query);

## JOptionPane.showMessageDialog(null, "Details updated successfully");

## setVisible(false);

## new Home();

## } catch (Exception e) {

## e.printStackTrace();

## }

## } else {

## setVisible(false);

## new Home();

## }

## }

## public static void main(String[] args) {

## new UpdateEmployee("");

## }

## }

## VIEWEMPLOYEE.JAVA

## package employee.management.system;

## import javax.swing.\*;

## import java.awt.\*;

## import java.sql.\*;

## import net.proteanit.sql.DbUtils;

## import java.awt.event.\*;

## public class ViewEmployee extends JFrame implements ActionListener{

## JTable table;

## Choice cemployeeId;

## JButton search, print, update, back;

## 

## ViewEmployee() {

## 

## getContentPane().setBackground(Color.WHITE);

## setLayout(null);

## 

## JLabel searchlbl = new JLabel("Search by Employee Id");

## searchlbl.setBounds(20, 20, 150, 20);

## add(searchlbl);

## 

## cemployeeId = new Choice();

## cemployeeId.setBounds(180, 20, 150, 20);

## add(cemployeeId);

## 

## try {

## Conn c = new Conn();

## ResultSet rs = c.s.executeQuery("select \* from employee");

## while(rs.next()) {

## cemployeeId.add(rs.getString("empId"));

## }

## } catch (Exception e) {

## e.printStackTrace();

## }

## 

## table = new JTable();

## 

## try {

## Conn c = new Conn();

## ResultSet rs = c.s.executeQuery("select \* from employee");

## table.setModel(DbUtils.resultSetToTableModel(rs));

## } catch (Exception e) {

## e.printStackTrace();

## }

## 

## JScrollPane jsp = new JScrollPane(table);

## jsp.setBounds(0, 100, 900, 600);

## add(jsp);

## 

## search = new JButton("Search");

## search.setBounds(20, 70, 80, 20);

## search.addActionListener(this);

## add(search);

## 

## print = new JButton("Print");

## print.setBounds(120, 70, 80, 20);

## print.addActionListener(this);

## add(print);

## 

## update = new JButton("Update");

## update.setBounds(220, 70, 80, 20);

## update.addActionListener(this);

## add(update);

## 

## back = new JButton("Back");

## back.setBounds(320, 70, 80, 20);

## back.addActionListener(this);

## add(back);

## 

## setSize(900, 700);

## setLocation(300, 100);

## setVisible(true);

## }

## 

## public void actionPerformed(ActionEvent ae) {

## if (ae.getSource() == search) {

## String query = "select \* from employee where empId = '"+cemployeeId.getSelectedItem()+"'";

## try {

## Conn c = new Conn();

## ResultSet rs = c.s.executeQuery(query);

## table.setModel(DbUtils.resultSetToTableModel(rs));

## } catch (Exception e) {

## e.printStackTrace();

## }

## } else if (ae.getSource() == print) {

## try {

## table.print();

## } catch (Exception e) {

## e.printStackTrace();

## }

## } else if (ae.getSource() == update) {

## setVisible(false);

## new UpdateEmployee(cemployeeId.getSelectedItem());

## } else {

## setVisible(false);

## new Home();

## }

## }

## public static void main(String[] args) {

## new ViewEmployee();

## }

## }

## CHAPTER 6

## IMPLEMENTATION AND RESULT

Following are the screens of the Employee Management System where you can see all the features of this system in use and you can also see the GUI of the system:

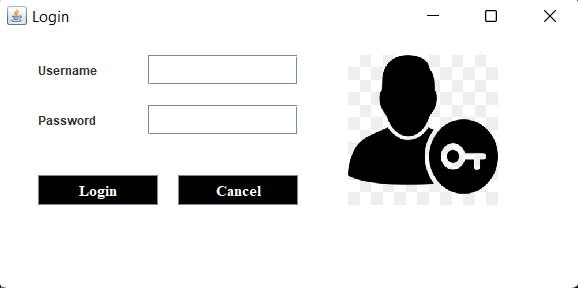
* 1. **Login frame** – This is the login frame of this system where user have to enter the required credentials to have access for the main dashboard.

Fig. 1

* 1. **Main Dashboard** – After login in, user is directed to the main dashboard of this system where user can perform various operations like adding an employee, deleting an employee

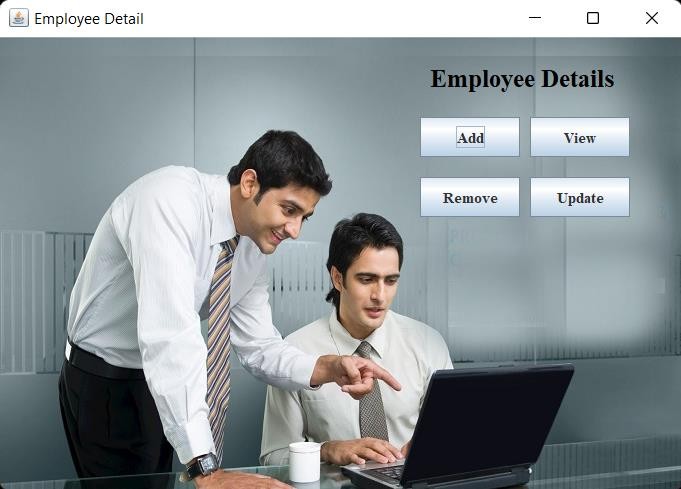


Fig. 2

* 1. **Add employee** – Here user have to enter all the required credentials to add a new employee to the system.

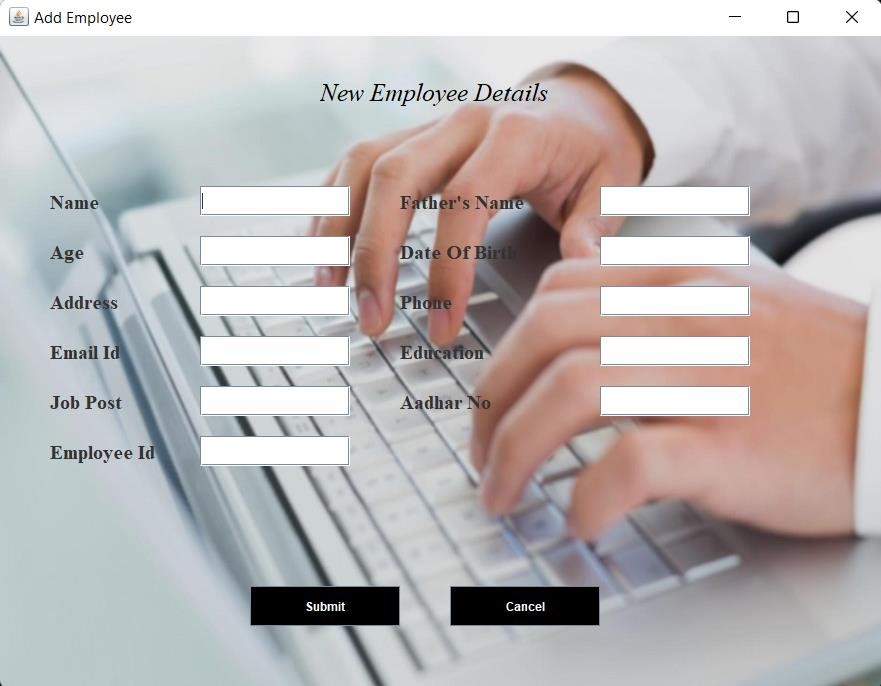


Fig. 3

* 1. **Remove employee** – User has to enter the employee id in order to delete his information from the system.



Fig. 4

5.5 **View and update employee** – In order to view and update employee information, the user have to enter employee ID



Fig. 5

## CHAPTER 6

## ER DIAGRAM

## 

## CHAPTER 7

## DATA FLOW DIAGRAM

## 

## CHAPTER 8

## TESTING

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test ID** | **Test- Purpose** | **Test-Condition** | **Expected-Output** | **Output** | **Remark** |
| TC1 | Check Username & Password | If user details are not correct, display error message | Grant access to main dashboard. | Access granted to main dashboard | Test successful |
| TC2 | To add new user to the system | If user already exists, error message should be displayed. | New user should be added. | New user added successfully | Test Successful |
| TC3 | To view existing employee information | If employee exists, then information should be displayed, else error message should be displayed. | Employee information should be displayed. | Employee information displayed. | Test Successful |
| TC4 | To remove an employee | If employee exists, then employee should be removed else error message should be displayed. | Employee should be removed. | Employee removed successfully. | Test Successful |
| TC5 | Update employee information | If employee exists, then information should be updated. | Employee information should be updated. | Employee information updated successfully | Test Successful |

## 

## CHAPTER 9

## CONCLUSION

In conclusion, the Employee Management System (EMS) stands as a pivotal solution for organizations seeking to navigate the complexities of workforce management in a rapidly evolving business landscape. The system's multifaceted capabilities, ranging demonstrate its integral role in fostering efficiency, transparency, and collaboration within an organization.

As technology continues to advance, the future of EMS holds promise in addressing emerging challenges and meeting evolving organizational needs. The integration of artificial intelligence, enhanced remote work support, real-time analytics, and advanced security measures ensures that the EMS remains a dynamic and adaptive tool for managing the modern workforce.

Moreover, the commitment to improving the employee experience, promoting ongoing learning, and supporting overall wellbeing reflects a forward-looking approach that recognizes the significance of human-centric management practices. The customization and scalability of EMS further underline its versatility, making it applicable across diverse industries and organizational structures.

In essence, the Employee Management System emerges not merely as a software solution but as a strategic enabler for businesses aiming to cultivate a productive, engaged, and resilient workforce. By streamlining processes, harnessing data-driven insights, and aligning with future trends, the EMS contributes to the creation of a workplace that is not only efficient but also conducive to the growth and development of both individuals and the organization as a whole.

## 

## CHAPTER 10

## FUTURE SCOPE

The future scope of an Employee Management System (EMS) is dynamic and continually evolving as technological advancements and organizational needs progress. Several trends and potential enhancements can be anticipated in the future scope of an EMS:

* 1. **Artificial Intelligence (AI) Integration:** Integration of AI for predictive analytics, allowing the system to forecast employee performance, identify potential attrition risks, and recommend personalized development plans.
  2. **Employee Experience Enhancement:** Focus on enhancing the overall employee experience by incorporating features such as wellness programs, employee engagement surveys, and feedback mechanisms to foster a positive work environment.
  3. **Remote Work Support:** Continued adaptation to the rise of remote work by providing tools and features that facilitate virtual collaboration, performance monitoring in a remote setting, and flexible scheduling options.
  4. **Real-Time Performance Analytics:** Implementation of real-time performance analytics and dashboards, enabling managers to make data-driven decisions on the spot and providing employees with immediate feedback on their performance.
  5. **Blockchain for Security and Transparency:** Utilization of blockchain technology to enhance data security and ensure transparency in sensitive processes such as payroll management, ensuring that employee records are tamper-proof and secure.
  6. **Mobile and cross-Platform Accessibility:** To accommodate the growing trend of mobile and flexible work arrangements, EMS can offer mobile-friendly interfaces and cross-platform compatibility, enabling employees to access essential functions from anywhere, anytime.
  7. **Predictive Analytics for HR Planning:** By leveraging predictive analytics, EMS can assist HR departments in forecasting workforce trends, identifying talent gaps, and making strategic decisions related to recruitment, retention, and succession planning.
  8. **Blockchain for Transparent HR Transactions:** Integration of blockchain technology can enhance transparency and security in HR transactions such as payroll management, employee records, and identity verification.
  9. **Social Collaboration and Knowledge Sharing:** Future EMS may incorporate social collaboration features and knowledge sharing platforms to facilitate communication, teamwork, and information exchange among employees within the organization.
  10. **Continuous Feedback and Performance Management:** Moving away from traditional annual performance reviews, EMS can support continuous feedback mechanisms and agile performance management frameworks, fostering ongoing dialogue and development between managers and employees.

## CHAPTER 11

## REFERENCES

* 1. Renae Broderick, John W. Boudreau, “Human resource management, information technology, and the competitive edge”, Academy of Management Executive, 1992 Vol. 6 No. 2
  2. Julie Bulmash, “Human Resource Management and Technology”, Chapter 3.
  3. Ian Sommerville, “Software Engineering”, 9th Edition, Addison-Wesley, 2011.
  4. Avison, D. and Fitzgerald, G. (2003). Information systems Development Methodologies, Techniques and Tools.3rd Edition. McGraw-Hill Education Limited Bershire.
  5. Juan Manuel Munoz Palacio, Information systems development methodologies for Data-driven Decision Support Systems, 2010.
  6. Deitel, PJ & Deitel, HM, 2008, Internet & World Wide Web How To Program, Dorling Kindersley, India.