|  |  |  |
| --- | --- | --- |
|  | **SRI KRISHNA INSTITUTIONS**  **(AN AUTONOMOUS INSTITUTION)**  **(APPROVED BY AICTE AND AFFILIATED TO ANNA UNIVERSITY CHENNAI)**  **ACCREDITED BY NAAC WITH “A” GRADE** |  |
|  |  |  |

ELECTRICAL CONTRACTS MANAGEMENT SYSTEM

COMPANY NAME:TECH I5

TEAM NUMBER:16

Submitted by

Rahul Glenn R

Sai Karthik M

Thulasi Ram S

Pradakshina TM

|  |  |
| --- | --- |
| Date | Reviewed by |
|  |  |

**TABLE OF CONTENTS**

1 About us & Contact us ………………… 3

2 Abstract ………………… 4

3 Objective ………………… 4

4 Existing Systems ………………… 5

5 Proposed System ………………… 5

6 Novelty ………………… 5

7 High level data documentation ………………… 6

8 Low level data documentation ………………… 10

8.1 Control flow diagram …………………… 10

8.2 Algorithm ..………………… 12

8.3 Pseudocode …………………. 16

9 Database Schema ………………… 21

10 Flow Chart ……………….. 23

11 Source Code ……………….. 27

12 Quality Assurance - TestCases ……………….. 135

13 Executable Screenshots ……………….. 146

14 Target Audience .……………….. 163

15 Future Enhancements ………………… 163

**About Us:**

Tech i5 Pvt Ltd. is a renowned Indian multi-national corporation providing software solutions to clients world-wide, head-quartered at Coimbatore. We have been partnering with 50+ companies to provide solutions as well as we have our own catalogue of products launched in market. Apart from these we study problems faced by common people and try to incorporate and work on solutions that would help them thrive. One such is the prototype in this project. We keep striving our way to build a better future.

**Contact Us:**

**Team-16**

**Tech i5**

* Rahul Glenn R

Mobile : 8667619406

* Sai Karthik

Mobile : 7010073121

* Thulasi Ram

Mobile : 7695941380

* Pradakshina

Mobile : 9629955154

**ABSTRACT:**

Digitalization is the current need of the society. Be it a small shop near our house, or a huge factory running business, if the data is digitalized, not only will it ease the effort involved periodically, to keep track of work performed, and employee benefits, but also help in co-ordination and faster delivery of products.

When we take in account the numerous small-scale industries in India today, most of them operate with data in papers and books which becomes unreliable over time.

As an example, in workshops which deal with motor rewinding, repairing, for a given motor spec, retrieving the turns and wire-gauge requirement written earlier in paper, is a tedious process. Not only that, in case of tenders, the work requires the owner to pay a security deposit, which would be paid back after commissioning of the repaired motor (max a year). Keeping track of this amount is another hectic job and often results in the owner losing track of this amount. Apart from this the employees working in the workshop should be tracked of their salary and leave status.

Combining the above-mentioned problems, we are coming up with a user-friendly Windows application, which solves the problems and is useful in co-ordinating various activities for the Owner as well as the employee when it comes to Workshops dealing with large scales electric motors/transformers re-winding.

**OBJECTIVE:**

• To develop a user-friendly application that helps workshops to store, retrieve and manipulate data

• To provide gateways for both admin and employees in small-scale industries.

• To provide the worker with an autonomous and secure environment.

• To help entrepreneurs of small-scale industries to keep track of their accounts and employee records.

• To innovate an application that assists every employer and entrepreneur owning workshops and into rewinding business.

**Existing Systems:**

In today’s scenario, most of the contractors/entrepreneurs are maintaining the rewinding details in books, where data retrieval is exhausting. Also, many a times, these contractors/entrepreneurs lose track of the security deposits that would have matured long back. Employee salary and leave tracking is also quite tiresome in these small-scale workshops. All these pain-points consume huge amount of time/work when manually maintained.

**Proposed System:**

With the proposed solution, we have figured out ways to streamline the employee leave and salary tracking. The proposed project will be featured with a sign up and login form through which the Admin will have a special key for his/her company and the employee would be identified using unique IDs. The admin could control the employee details. Primarily all the worker's days of work are calculated before generating their monthly salary. Interface to check the date of security deposit(amount paid after acquiring a tender for guarantee) maturing. Thus this software solves critical problems faced and helps maintain a constructive workplace and saves a lot of time and resources.

**Novelty:**

Small-scale industries around us don’t have a digitized way of handling their data and accounts even today. This would be a stand-alone easy to use software that could come in handy for such small shops.

**High Level Data Documentation:**

1. **Business Requirements:**

The following are the business requirements for this project.

|  |  |  |
| --- | --- | --- |
| s.no | requirement | status |
| 1 | Login and Sign Up | Implemented |
| 2 | Rewinding Details | Implemented |
| 3 | Work Details | Implemented |
| 4 | Employee Details | Implemented |
| 5 | Admin | Implemented |
| 6 | Employee Main Page | Implemented |

**2.Login & Sign Up:**

This module has two main functions.

* 1. Sign-up for new Employer.
  2. Login for existing Employer.

Sign-up:

The Employer details such as the Employer Name, D.O.B, Email-ID and Password are got from the employer.

Along with these details, the Activation Key(Product Key) which is to be purchased from the admin, in order to access this application is given.

Then the validation of the product key happens after pressing the validate button.

Once the Key gets validated, the employer can select the register option. If the details are appropriate, it gets stored in the DB and the employer is forwarded to the Employer MainFrame Page.

Login:

The Employer is required to input the Email ID and Password of his account.

Once after clicking the login Option, the Email ID and Password are verified by comparing those values in the DB. If valid the employer is forwarded to his MainFrame Page.

**3.Rewinding Details**

This module has four main functions.

1. Add Rewinding Details
2. View Rewinding Details
3. Update Rewinding Details
4. Delete Rewinding Details

Add Rewind:

The Employer can enter his required rewinding data in here (i.e) the HorsePower value and the rewinding data for that Horsepower motor.

This data is stored locally in the files first and then the path of the file containing details and HP value is stored in the DB.

View Rewind:

The Employer can view the stored rewinding details. The HP value are retrieved from the DB and dynamically added to the combobox present. On clicking the required HP value the file path corresponding to that HP value is got from the DB and then using that path the file is read and displayed on the text area present.

Update Rewind:

The Employer can update the wrongly written rewinding data using this option. The corresponding changes done are reflected in the respective file on clicking the update option.

Delete Rewind:

The Employer can also delete a specified rewinding data from the DB as well as the file corresponding to that specified path.

**4.Work Details:**

This module has four main functions.

1. Add Work Details
2. View Matured SD Work
3. Update Work Details
4. Work History

Add Work:

The Employer can add the work details of the granted tender work with the Security Deposit Value and the date of work using this option.

View Matured SD Work:

The Employer can view all the matured SD work details using the Date of Work given while adding the work details. Using this module the employer can keep track of his work details.

Update Work:

The Employer can also update the work details which was once added, if any of the work details was wrongly entered at that time.

Work History:

The Employer can view his entire work history using this option.

**5.Employee Details:**

This module consists of four main functions.

1. Add Employee
2. View Employee List
3. Delete Employee
4. Leave Entry

Add Employee:

Using this function the employer can add an employee to his organization by specifying the employee details.

View Employee List:

The Employer can also view the entire list of employees working in his organization using this function.

Delete Employee:

The Employer can remove an employee from his organization by specifying his empid and thereby deleting that employee record from the Database.

Leave Entry:

The Employer can mark the leave status of the employees by selecting the empid and later this entry helps in the payroll process during the distribution of their month end salary.

**6.Admin:**

The admin enters his login credentials (i.e) username and password, and clicks login. After that validation starts by comparing it with the record present in the Database. If valid, the Admin is forwarded to the Admin MainPage.

The Admin Main Page has two main functions.

1. Add Organization
2. Employer List

Add Organization:

The admin can add an organization in his application by entering the organization name in the textbox present there. After that, on clicking the generate button, an Universally Unique Identifier (UUID) is generated as the product key and this info is stored in the productinfo table in the Database.

Employer List:

The admin can also view the entire list of employers using this application including the activation date of their product.

**7.Employee Main Page:**

The Employee initially enters his Email ID and Password given by the Employer at the time of employee creation.

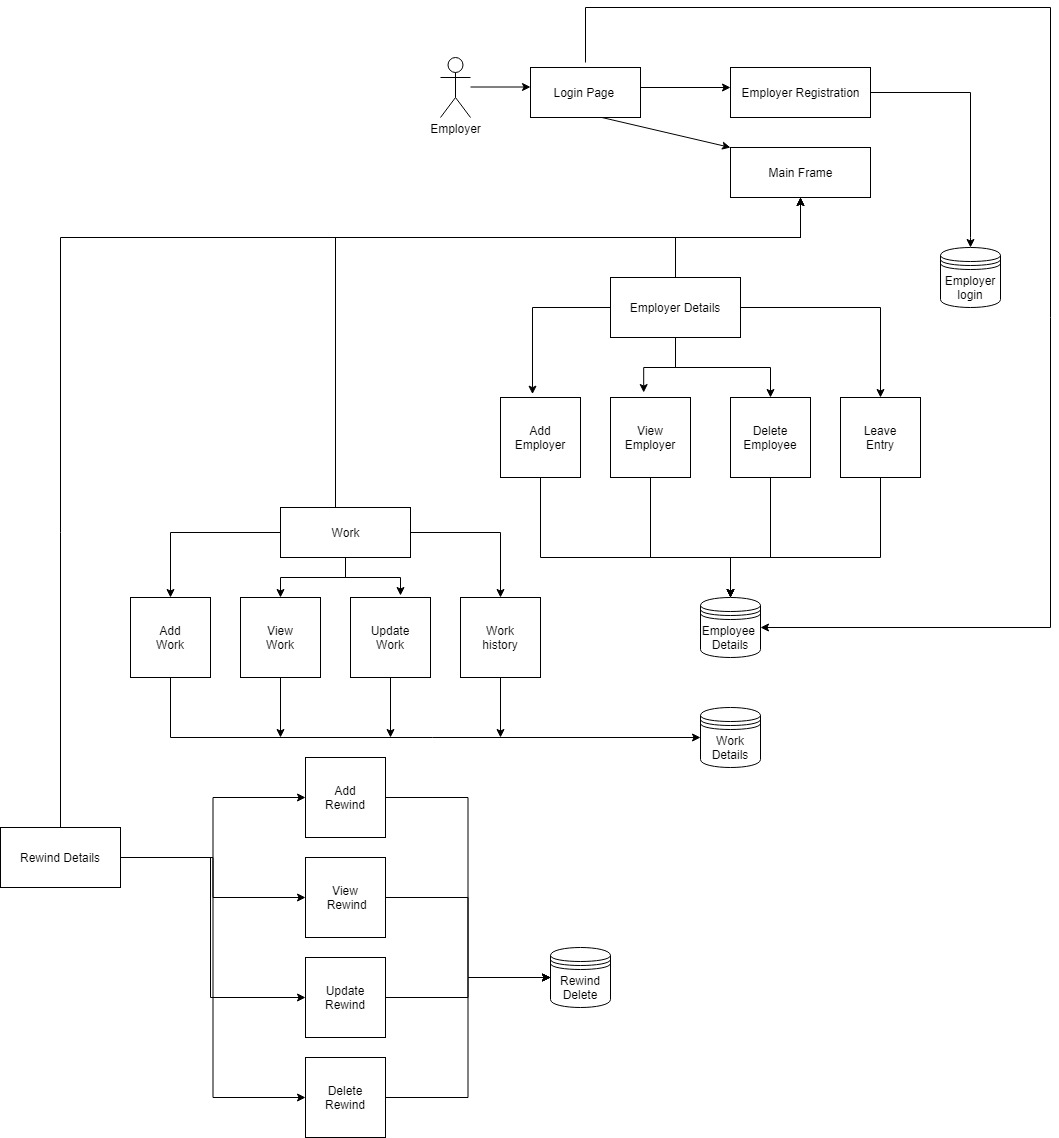
Being the first time, the employee is given a frame to change the password to his account. On changing the password, the employee is transferred to the same login page.

Now, on logging with his new password the employee is forwarded to the main page of his account, where he can see his profile along with his current month salary.

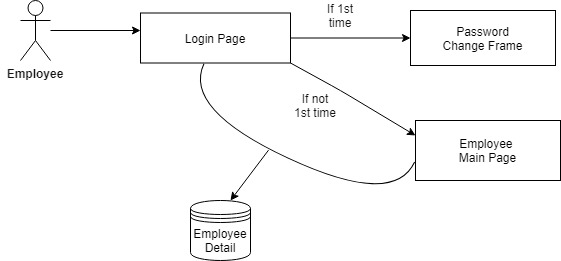
**Low Level Data Documentation:**

**Control Flow Diagram:**

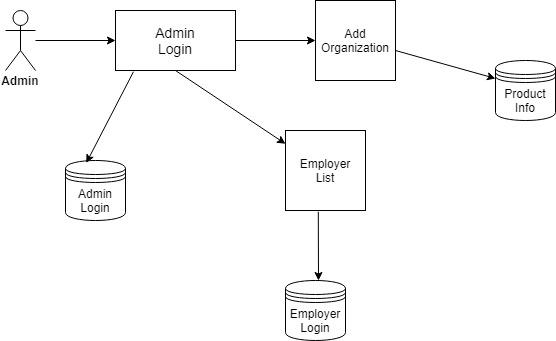
**Employer Side:**

****

**Employee Side:**

****

**Admin Side:**

****

**Algorithm:**

**Employer Sign Up:**

1. Start
2. Input Employer Details(Name, DOB, Email ID, Password)
3. Input the Product Key (Activation Key) which is got from the admin
4. Validate the product Key by clicking the validate button

If valid

Display Validation Successful

Else

Display Invalid Product Key

1. Click Register Button
2. If Details are appropriate

Go to Main Frame Page

1. Else

Display please fill all the details.

1. End

**Login:**

1. Start
2. Input Email ID and Password.
3. Check of it matches with any record in the DB.
4. If valid

Go to MainFrame Page

1. Else

Display Invalid Email Id or Password

1. End

**Rewinding Details:**

**Add Rewind:**

1. Start
2. Input HP value and Rewinding details.
3. Select Add Button.
4. Push the values into the Database
5. End

**View Rewind:**

1. Start
2. Select HP value from the combobox
3. Fetch the specified record from the DataBase.
4. Display those values in the textfield and textarea present.
5. End

**Update Rewind:**

1. Start
2. Select HP value from the combobox
3. Fetch the specified record from DB
4. Do the necessary changes in the details.
5. Update those changes in the Database.
6. End

**Delete Rewind:**

1. Start
2. Select HP value from the combobox
3. Display the specified record by fetching it from the DB.
4. If delete button is pressed,

Delete that record from the DB

1. End

**Work Details:**

**Add Work:**

1. Start
2. Input Work Details(Name,Date,Amount)
3. Push those values into the DB
4. End

**View Matured SD Work:**

1. Start
2. Select the Matured SD Work detail from the combobox
3. Fetch that record from the DB and display it.
4. If SD Collected Button is selected

Mark the collection status as collected

1. End

**Update Work:**

1. Start
2. Select the Work id needed from the combobox
3. Display that record by fetching it from the DB
4. Do the necessary changes and update it in the DB on clicking the update button
5. End

**Work History:**

1. Start
2. View the entire Work History
3. End

**Employee Details:**

**Add Employee:**

1. Start
2. Input Employee Details(Name,DOB, Address, Salary, Email Id and Password)
3. If data is appropriate

Insert those values in the DB

Else

Display Error Message

1. End

**View Employee List:**

1. Start
2. View the entire Employee List by fetching each records from the DB.
3. End

**Delete Employee:**

1. Start
2. Select the employee by selecting his empid from the combobox.
3. Fetch his record and display it
4. If delete button is pressed

Delete that record from the DB

1. End

**Leave Entry:**

1. Start
2. Select the empid from the combobox present
3. Display his details by fetching it from the DB
4. Input No. of Days
5. Based on the no.of days and his salary his current salary gets updated
6. End

**Admin:**

1. Start
2. Input username and Password
3. If validation Successful

Go to Admin Main Page

Else

Display Invalid Username or Password

1. If Add Organization
   1. Input Organization Name
   2. Click Generate Button
   3. Generate a UUID as Product Key

Else if Employer List

1. Display entire Employer List by fetching it from DB
2. End

**Employee Main Page:**

1. Start
2. Input Email ID and Password
3. If Login Credentials valid,

Go to Employee Main Page

Display Employee Profile

Else

Display Invalid Email ID or Password

1. End

**Pseudo Code:**

**Login & Employer Sign Up:**

Begin

If Login

Get Email ID and Password

If (input Email Id and Password matches a record in DB) then

Close the login page

Go to Main Frame Page

Else

print Invalid Email ID or Password

Else

Go to Sign Up Page

Input Employer Details

Input Activation Key

Click Validate

If (Activation Key==Key present in DB) then

Print Validation Successful

Else

Print Invalid Product Key

Click Register

End

**Rewinding Details:**

Begin

If Add Rewind then

Input Horsepower and Rewinding details

Store the details in DB

Else if View Rewind then

Select Required HP value from Combobox

Fetch the required Details from DB

Else if Update Rewind then

Select Required HP value from Combobox

Fetch the required Details from DB

Make the necessary changes

Update the Changes in DB

Else if Delete Rewind then

Select Required HP value from Combobox

Fetch the required Details from DB

Click delete button

Delete that record from DB

Else if Logout

Go to Login Page

End

**Work Details:**

Begin

If Add Work then

Input Name of Work, Date and Amount

Store the details in the DB

Else if View Work then

Select Required Work ID from Combobox

Fetch the required Details from DB

Else if Update Work then

Select Required work id from Combobox

Fetch the required Details from DB

Make the necessary changes

Update the Changes in DB

Else if Work History then

Print the entire work history in JTable

End

**Employee Details:**

Begin

If Add Employee then

Input Employee Details

Click Add

Store the Employee details in DB

Else if View Employee then

Print the employee list in JTable

Else if Delete Employee then

Select Required employee id from Combobox

Fetch the required Details from DB

Click Delete Button

Delete that record from DB

Else if Leave Entry then

Select Required employee id from Combobox

Fetch the required Details from DB

Input No. Of Days

Fetch Salary from DB

Change Cur\_Salary=Cur\_Salary-((Salary/30)\*No. of Days)

Update in DB

End

**Admin:**

Begin

Input Username and Password

If ((Username==Username in DB) && (Password==Password in DB))

Go to Admin Main Page

If Add Organization then

Input Organization Name

Click Generate

Generate an UUID

Print the UUID as Product Key

If Back Button Paget then

Go to Admin Main Page

Else if Employer List then

Print Employer list in JTable

Else

Print Invalid Username or Password

End

**Employee Main Page:**

Begin

Input Email ID and Password

If ((Email ID==Email ID in DB) && (Password==Password in DB))

Go to Employee Main Page

View the Employee Details in Main Page

Else

Print Invalid Email ID or Password

End

**Database Schema:**

****

**Table employerlogin used in:**

1. Storing employer info
2. Validating login info of employer

**Table employdet used in:**

1. Storing employee details
2. Validing login info of employee
3. Updation of leave status by employer
4. Storing the current moth salary details

**Table work\_details is used in:**

1. Storing tendered work details
2. Retrieving matured SD work details

**Table rewinddet is used in:**

1. Storing, Retrieving and Updating the rewinding data

**Table productinfo is used in:**

1. Storing product details at the time of purchase from the admin
2. Checking the Activation Status while Employer tries to use the product key.

**Table adminlogin is used in:**

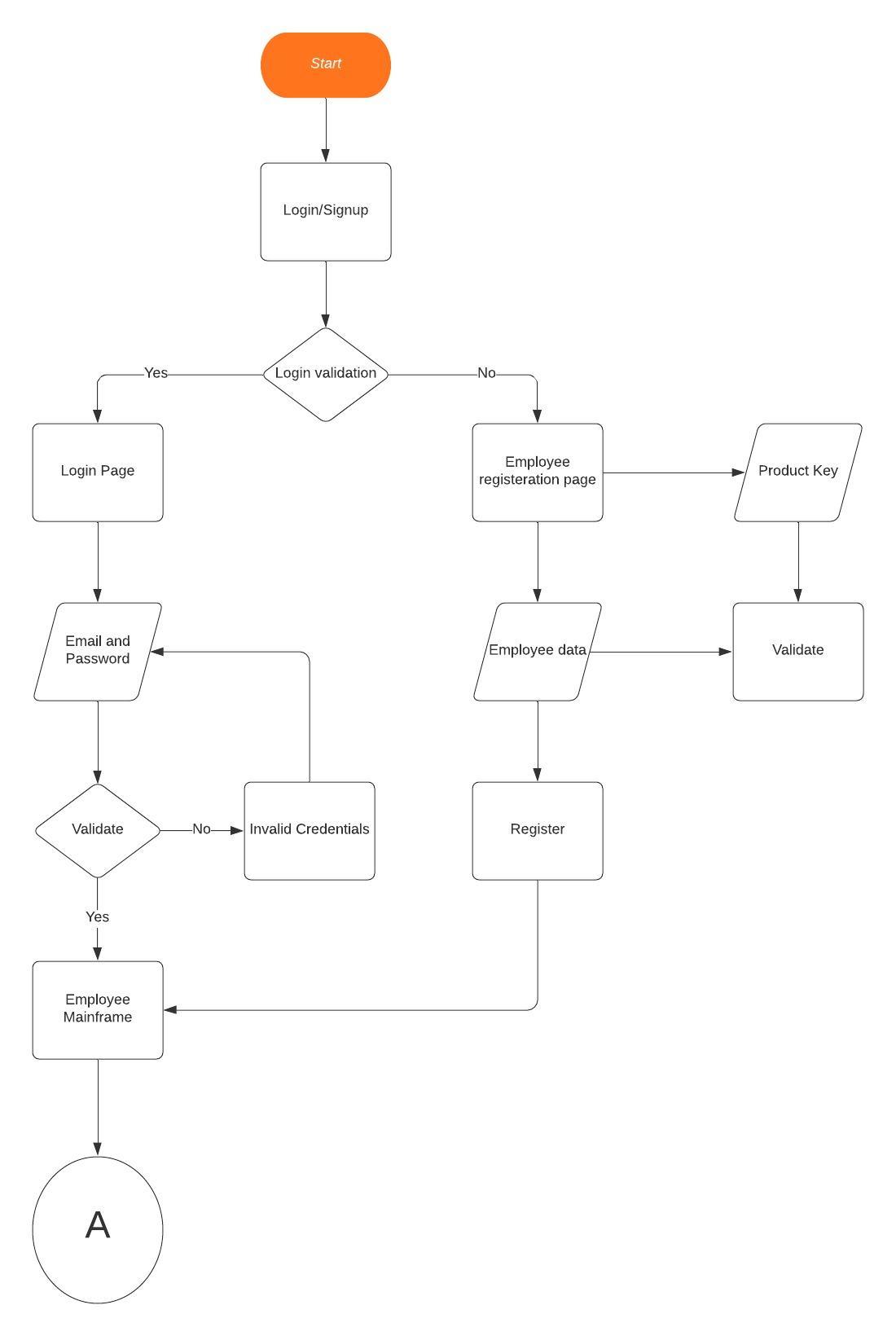
1. Used for Storing the login credentials of the admin
2. Used at the time of admin login validation.

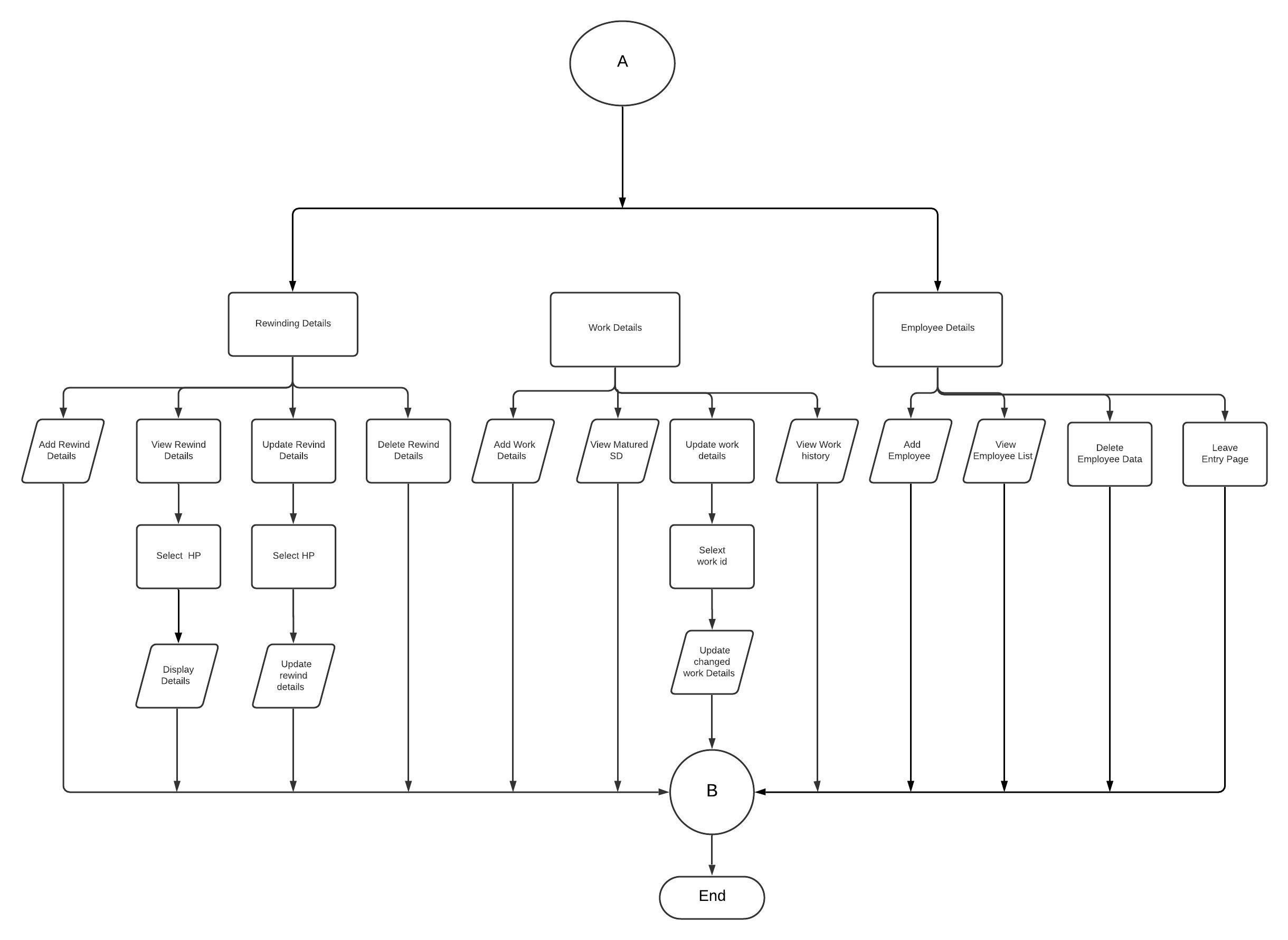
**Table logfile is used in:**

1. Used for Storing the log info such as Account created and logged in
2. Used to refresh the current salary of the employee at th start of the month.

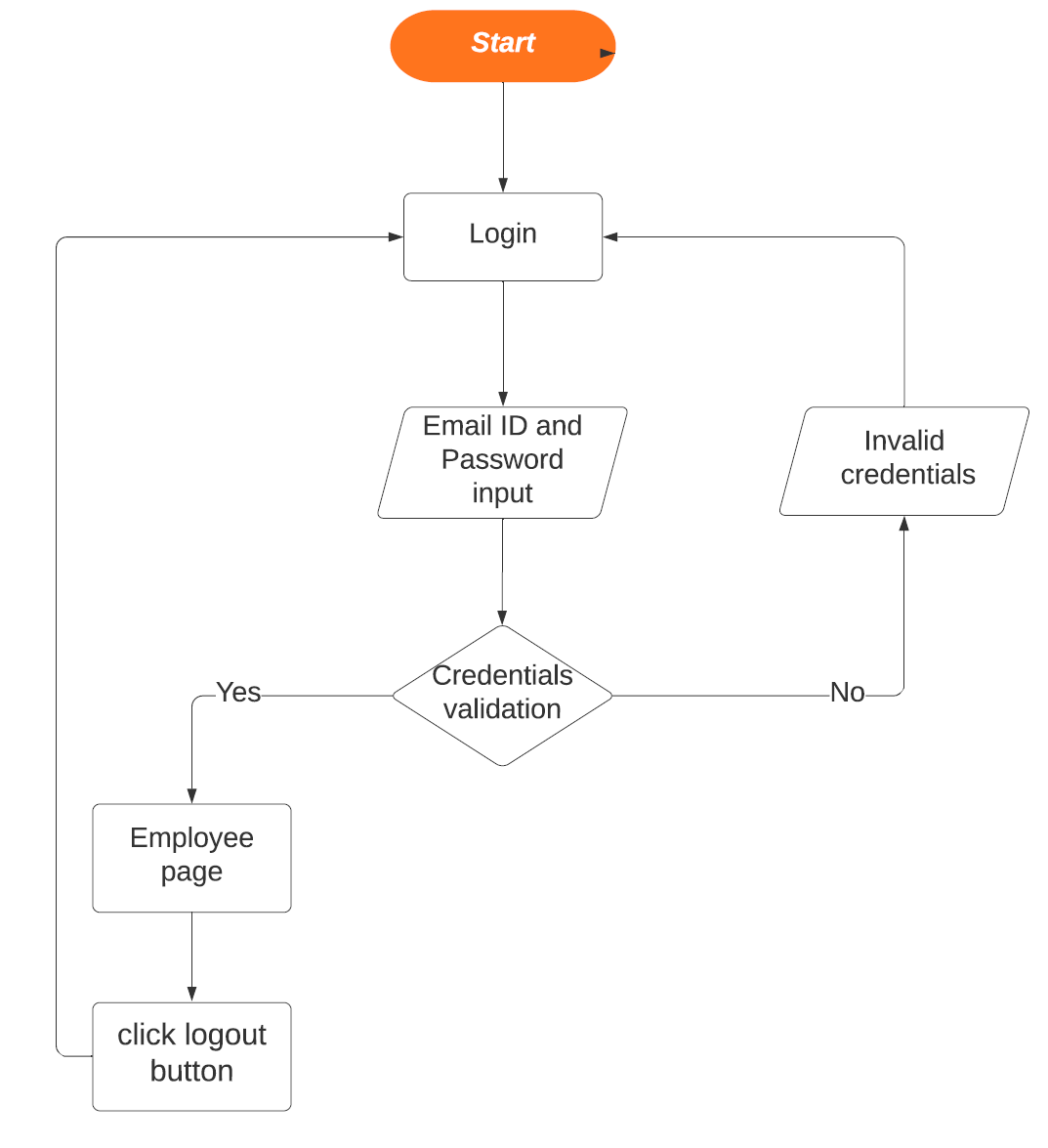
**Flow Charts:**

**Employer Program:**

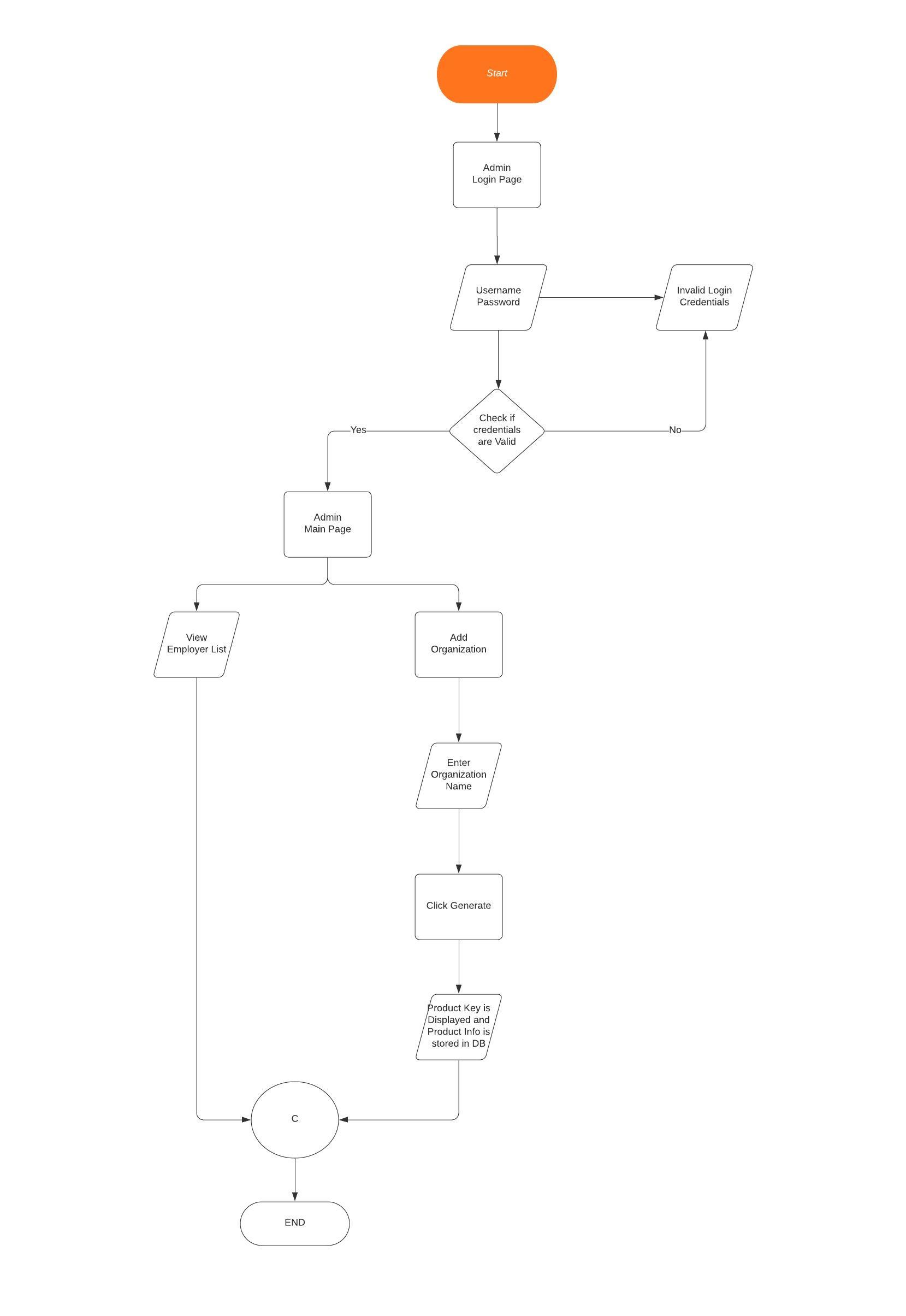
****

****

**Employee Program:**

****

**Admin Program:**



**Source Code:**

**1.Login Page:**

package Bootathon;

import Bootathon.uiworks.MyLabel;

import Bootathon.uiworks.MyTextField;

import Bootathon.uiworks.MyFrame;

import Bootathon.uiworks.MyButton;

import Bootathon.database.DBOperations;

import java.awt.Container;

import java.awt.HeadlessException;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.text.SimpleDateFormat;

import java.util.Date;

import javax.swing.ButtonGroup;

import javax.swing.JCheckBox;

import javax.swing.JOptionPane;

import javax.swing.JPasswordField;

import javax.swing.JRadioButton;

public class LoginPage {

LoginPage()

{

//creating frame

MyFrame log\_f = new MyFrame(1);

log\_f.setBounds(500,100,600,800);

//creating Login container

Container log\_c=log\_f.getContentPane();

log\_c.setLayout(null);

//login label creation

MyLabel log\_l = new MyLabel("Login",1);

log\_l.setBounds(40,10,200,50);

log\_c.add(log\_l);

//email label and textfield creation

MyLabel log\_l1 = new MyLabel("Email id");

log\_l1.setBounds(70,100,100,30);

log\_c.add(log\_l1);

MyTextField log\_t1 = new MyTextField();

log\_t1.setBounds(200,100,300,30);

log\_c.add(log\_t1);

//creating Panel2(Password)

MyLabel log\_l2=new MyLabel("Password");

log\_l2.setBounds(60,200,150,30);

JPasswordField log\_pass2=new JPasswordField(12);

log\_pass2.setBounds(200,200,300,30);

JCheckBox log\_cb2=new JCheckBox("Show Password");

log\_cb2.setBounds(200,250,150,30);

log\_cb2.setOpaque(false);

//password visibiity

log\_cb2.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

if(log\_cb2.isSelected())

log\_pass2.setEchoChar((char)0);

else

log\_pass2.setEchoChar('\*');

}

});

log\_c.add(log\_l2);

log\_c.add(log\_pass2);

log\_c.add(log\_cb2);

//creating radio button

JRadioButton log\_rb1 = new JRadioButton("<html><Font size = 4>Employee</Font></html>");

JRadioButton log\_rb2 = new JRadioButton("<html><Font size = 4>Employer</Font></html>");

log\_rb1.setBounds(70,320,150,30);

log\_rb2.setBounds(230,320,150,30);

log\_rb1.setOpaque(false);

log\_rb2.setOpaque(false);

log\_c.add(log\_rb1);

log\_c.add(log\_rb2);

ButtonGroup bg = new ButtonGroup();

bg.add(log\_rb2);

bg.add(log\_rb1);

//creating panel3(login & signup buttons)

MyButton log\_b1 = new MyButton("Login");

log\_b1.setBounds(50,400,150,30);

MyButton log\_b2=new MyButton("Sign Up");

log\_b2.setBounds(50,480,150,30);

log\_b2.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

log\_f.dispose();

new EmployerRegistration();

}

});

log\_c.add(log\_b2);

log\_c.add(log\_b1);

//login button action

log\_b1.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

if(!log\_rb1.isSelected() && !log\_rb2.isSelected())

{

JOptionPane.showMessageDialog(log\_f, "Please Select whether Employee or Employer!");

}

else if(log\_t1.getText().isEmpty() || String.valueOf(log\_pass2.getPassword()).isEmpty())

{

JOptionPane.showMessageDialog(log\_f, "Please enter Login Credentials!");

}

else{

if(log\_rb1.isSelected()) // employee radio button is selected

{

try{

Connection conn=DBOperations.getConn();

PreparedStatement st=conn.prepareStatement("Select \* from employdet where Username=? AND Password=?");

st.setString(1, log\_t1.getText());

st.setString(2, String.valueOf(log\_pass2.getPassword()));

ResultSet rs=st.executeQuery();

if(rs.next())

{

if(Integer.valueOf(rs.getString("PassChange"))==0)

{

log\_f.dispose();

new EmpPassChange(rs);

}

else{

log\_f.dispose();

new EmployeeMain(rs.getString("Empname"),rs.getInt("empid"),rs.getString("Address"),rs.getString("DOB"),rs.getString("Phone"),rs.getInt("Salary"),rs.getInt("CurSalary"));

}

}

else

{

JOptionPane.showMessageDialog(log\_f, "Invalid Username or Password!");

}

conn.close();

}

catch(HeadlessException ee)

{

System.out.println("Cannot insert the specified HTML header!"+ee);

}

catch(NumberFormatException ee)

{

System.out.println("The PassChange should be int"+ee);

}

catch(SQLException ee)

{

System.out.println("Cannot retrieve values from DB!!"+ee);

}

}

if(log\_rb2.isSelected()) // employer radio button is selected

{

try{

Connection conn=DBOperations.getConn();

PreparedStatement st=conn.prepareStatement("Select \* from employerlogin where Username=? AND Password=?");

st.setString(1, log\_t1.getText());

st.setString(2, String.valueOf(log\_pass2.getPassword()));

ResultSet rs=st.executeQuery();

if(rs.next())

{

st=conn.prepareStatement("select \* from logfile where emprid=? and logid=(select max(logid) from logfile where emprid=?)");

st.setInt(1, rs.getInt("emprid"));

st.setInt(2, rs.getInt("emprid"));

ResultSet rs2=st.executeQuery();

rs2.next();

String date=rs2.getString("Date");

st=conn.prepareStatement("insert into logfile values(?,0,?,?,?)");

st.setInt(1, rs.getInt("emprid"));

st.setString(2, rs.getString("Name"));

st.setString(3, new SimpleDateFormat("dd/MM/yyyy").format(new Date()));

st.setString(4, "Logged in!");

st.executeUpdate();

log\_f.dispose();

new MainFrame(rs.getInt("emprid"),rs.getString("Name"),date);

}

else

{

JOptionPane.showMessageDialog(log\_f, "Invalid Username or Password!");

}

conn.close();

}

catch(HeadlessException ee)

{

System.out.println("Cannot insert the specified HTML header!"+ee);

}

catch(SQLException ee)

{

System.out.println("Cannot retrieve values from DB!!"+ee);

}

}}

}

});

log\_f.setVisible(true);

}

public static void main(String[] args) {

new LoadFrame();

}

}

**2.Employer Registration Program:**

package Bootathon;

import Bootathon.database.DBOperations;

import Bootathon.uiworks.MyButton;

import Bootathon.uiworks.MyFrame;

import Bootathon.uiworks.MyLabel;

import Bootathon.uiworks.MyTextField;

import java.awt.Container;

import java.awt.HeadlessException;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.File;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.regex.Pattern;

import javax.swing.JOptionPane;

import javax.swing.JPasswordField;

public class EmployerRegistration {

EmployerRegistration()

{

//creating frame

MyFrame emprreg\_frame=new MyFrame(5);

emprreg\_frame.setBounds(500,100,600,800);

//Container creation

Container emprreg\_con = emprreg\_frame.getContentPane();

emprreg\_con.setLayout(null);

//employername Label creation

MyLabel emprreg\_ername = new MyLabel("Employer Name ");

emprreg\_ername.setBounds(50,40,200,50);

emprreg\_con.add(emprreg\_ername);

//organization label creation

MyLabel emprreg\_org = new MyLabel("Organization");

emprreg\_org.setBounds(50,200+40,200,50);

emprreg\_con.add(emprreg\_org);

//DOB label creation

MyLabel emprreg\_dob = new MyLabel("D.O.B ");

emprreg\_dob.setBounds(50,275+40,200,50);

emprreg\_con.add(emprreg\_dob);

//emailid label creation

MyLabel emprreg\_email= new MyLabel("Email ID");

emprreg\_email.setBounds(50,350+40,200,50);

emprreg\_con.add(emprreg\_email);

//password label creation

MyLabel emprreg\_pass= new MyLabel("Password");

emprreg\_pass.setBounds(50,425+40,200,50);

emprreg\_con.add(emprreg\_pass);

//retypepassword label creation

MyLabel emprreg\_repass= new MyLabel("RetypePassword");

emprreg\_repass.setBounds(50,500+40,200,50);

emprreg\_con.add(emprreg\_repass);

//Activation key label creation

MyLabel emprreg\_ak= new MyLabel("Activation key");

emprreg\_con.add(emprreg\_ak);

emprreg\_ak.setBounds(50,115,200,50);

//Textfield1 creation

MyTextField emprreg\_t1 = new MyTextField();

emprreg\_t1.setBounds(250,50,300,30);

emprreg\_con.add(emprreg\_t1);

//Textfield2 creation

MyTextField emprreg\_t2 = new MyTextField();

emprreg\_t2.setBounds(250,200+50,300,30);

emprreg\_t2.setEditable(false);

emprreg\_con.add(emprreg\_t2);

//Textfield3 creation

MyTextField emprreg\_t3 = new MyTextField();

emprreg\_t3.setBounds(250,275+50,300,30);

emprreg\_con.add(emprreg\_t3);

//Textfield4 creation

MyTextField emprreg\_t4 = new MyTextField();

emprreg\_t4.setBounds(250,350+50,300,30);

emprreg\_con.add(emprreg\_t4);

//Passwordfield t5 creation

JPasswordField emprreg\_t5 = new JPasswordField();

emprreg\_t5.setBounds(250,425+50,300,30);

emprreg\_con.add(emprreg\_t5);

//passwordfield t6 creation

JPasswordField emprreg\_t6 = new JPasswordField();

emprreg\_t6.setBounds(250,500+50,300,30);

emprreg\_con.add(emprreg\_t6);

//Textfield7 creation

MyTextField emprreg\_t7 = new MyTextField();

emprreg\_t7.setBounds(250,125,300,30);

emprreg\_con.add(emprreg\_t7);

MyLabel clickhere = new MyLabel("click here \u2191");

clickhere.setBounds(380,210,150,20);

emprreg\_con.add(clickhere);

// ActivationKey validation

MyButton b1 = new MyButton("Validate");

b1.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

if(emprreg\_t7.getText().isEmpty())

{

JOptionPane.showMessageDialog(emprreg\_frame, "Enter your Product Key!");

}

else

{

try

{

Connection conn=DBOperations.getConn();

PreparedStatement st=conn.prepareStatement("select \* from productinfo where ProductKey=?");

st.setString(1, emprreg\_t7.getText());

ResultSet rs=st.executeQuery();

if(rs.next() && rs.getInt("ActivationStatus")==0)

{

emprreg\_t2.setText(rs.getString("Organization"));

JOptionPane.showMessageDialog(emprreg\_frame, "Validation Succesful!");

}

else

{

JOptionPane.showMessageDialog(emprreg\_frame, "Invalid Product Key!");

}

conn.close();

}

catch(HeadlessException ee)

{

System.out.println("Cannot insert the specified HTML header!"+ee);

}

catch(SQLException ee)

{

System.out.println("Cannot retrieve the product info from DB!"+ee);

}

}}

});

b1.setBounds(380,180,110,20);

emprreg\_con.add(b1);

//Employer Registration

MyButton b = new MyButton("Register");

b.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

if(emprreg\_t2.getText().isEmpty())

{

JOptionPane.showMessageDialog(emprreg\_frame, "<html><font size=4>Please Validate your Product Key!","Validate",JOptionPane.ERROR\_MESSAGE);

}

else if(emprreg\_t1.getText().isEmpty() || emprreg\_t3.getText().isEmpty() || emprreg\_t4.getText().isEmpty() || String.valueOf(emprreg\_t5.getPassword()).isEmpty() || String.valueOf(emprreg\_t5.getPassword()).isEmpty())

{

JOptionPane.showMessageDialog(emprreg\_frame, "<html><font size=4>Please Fill all the details listed above!","Fill it",JOptionPane.ERROR\_MESSAGE);

}

else if(emprreg\_t1.getText().length()>20)

{

JOptionPane.showMessageDialog(emprreg\_frame, "<html><font size=4>The Employer Name can have a max of 20 Characters only!","Change it",JOptionPane.ERROR\_MESSAGE);

}

else if(emprreg\_t5.getText().length()>15 || emprreg\_t5.getText().length()<8)

{

JOptionPane.showMessageDialog(emprreg\_frame, "<html><font size=4>The Password should be of size min 8 characters and a max of 15 characters!","Change it",JOptionPane.ERROR\_MESSAGE);

}

else if(!Pattern.compile("^(3[01]|[12][0-9]|0[1-9])/(1[0-2]|0[1-9])/[0-9]{4}$").matcher(emprreg\_t3.getText()).matches())

{

JOptionPane.showMessageDialog(emprreg\_frame, "<html><font size=4>Invalid Date or Invalid Format! use (dd/mm/yyyy)!","Invalid",JOptionPane.ERROR\_MESSAGE);

}

else if(!Pattern.compile("^[\\w-\_\\.+]\*[\\w-\_\\.]\\@([\\w]+\\.)+[\\w]+[\\w]$").matcher(emprreg\_t4.getText()).matches())

{

JOptionPane.showMessageDialog(emprreg\_frame, "<html><font size=4>Invalid Email Address!","Invalid",JOptionPane.ERROR\_MESSAGE);

}

else if(!String.valueOf(emprreg\_t6.getPassword()).equals(String.valueOf(emprreg\_t5.getPassword())))

{

JOptionPane.showMessageDialog(emprreg\_frame, "<html><font size=4>The Password and Retype Password does not match!","re-type Error",JOptionPane.ERROR\_MESSAGE);

}

else{

try{

Connection conn=DBOperations.getConn();

PreparedStatement st=conn.prepareStatement("insert into employerlogin values(0,?,?,?,?,?,?,?)");

st.setString(1 , emprreg\_t1.getText());

st.setString(2, emprreg\_t2.getText());

st.setString(3, emprreg\_t3.getText());

st.setString(4, new SimpleDateFormat("dd/MM/yyyy").format(new Date()));

st.setString(5, emprreg\_t4.getText());

st.setString(6, String.valueOf(emprreg\_t5.getPassword()));

st.setString(7, emprreg\_t7.getText());

st.executeUpdate();

st=conn.prepareStatement("Select emprid from employerlogin where ProductKey=?");

st.setString(1, emprreg\_t7.getText());

ResultSet rs=st.executeQuery();

rs.next();

int id=rs.getInt("emprid");

st=conn.prepareStatement("insert into logfile values(?,0,?,?,?)");

st.setInt(1, id);

st.setString(2, emprreg\_t1.getText());

st.setString(3, new SimpleDateFormat("dd/MM/yyyy").format(new Date()));

st.setString(4, "Account Created!");

st.executeUpdate();

st=conn.prepareStatement("update productinfo set ActivationStatus=1 where ProductKey=?");

st.setString(1, emprreg\_t7.getText());

st.executeUpdate();

conn.close();

JOptionPane.showMessageDialog(emprreg\_frame,"<html><font size=4>Registration Successful","Successful",JOptionPane.INFORMATION\_MESSAGE);

//creating a new directory for the employer

new File("C:\\Electrical Data\\Empr"+String.valueOf(id)).mkdir();

emprreg\_frame.dispose();

new MainFrame(id,emprreg\_t1.getText(),"Nil");

}

catch(HeadlessException ee)

{

System.out.println("Cannot insert the specified HTML header!"+ee);

}

catch(SQLException ee)

{

System.out.println("Cannot insert values into DB!!"+ee);

}

}}

});

b.setBounds(30,620,150,30);

emprreg\_con.add(b);

MyButton bac = new MyButton("\u2190"+" BACK");

bac.setBounds(30,680,150,30);

emprreg\_con.add(bac);

bac.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

emprreg\_frame.dispose();

new LoginPage();

}

});

emprreg\_frame.setVisible(true);

}

public static void main(String[] args) {

new LoadFrame();

}

}

**3.Employer MainFrame:**

package Bootathon;

import Bootathon.database.DBOperations;

import Bootathon.uiworks.MyButton;

import Bootathon.uiworks.MyFrame;

import Bootathon.uiworks.MyLabel;

import Bootathon.uiworks.MyPanel;

import java.awt.Color;

import java.awt.Container;

import java.awt.FlowLayout;

import java.awt.GridLayout;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.SQLException;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Date;

import javax.swing.border.LineBorder;

import javax.swing.border.TitledBorder;

public class MainFrame {

static String EmpName,date;

static int id;

MainFrame(int empr,String name,String dat)

{

EmpName=name;

date=dat;

id=empr;

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

//if difference in month is encountered,the monthly salary is resetted

try{

if(sdf.parse(date).getMonth()!=new Date().getMonth())

{

Connection conn=DBOperations.getConn();

PreparedStatement st=conn.prepareStatement("update employdet set CurSalary=Salary");

st.executeUpdate();

conn.close();

}

}

catch(ParseException ee)

{

if(!date.equals("Nil"))

System.out.println("Invalid Date Format"+ee);

}

catch(SQLException ee)

{

System.out.println("Cannot update values into DB!!"+ee);

}

new MainFrame();

}

MainFrame()

{

// Frame creating

MyFrame mainFrame = new MyFrame(2);

mainFrame.setBounds(500,100,600,800);

//container creating

Container c = mainFrame.getContentPane();

c.setLayout(new GridLayout(5,1,30,30));

mainFrame.setVisible(false);

MyPanel panel0 = new MyPanel();

panel0.setLayout(null);

MyLabel paneLab = new MyLabel("Hi,"+EmpName+" ",1);

paneLab.setBounds(30,60,300,30);

c.add(panel0);

panel0.add(paneLab);

MyButton laa = new MyButton("Log out");

laa.setBounds(460,60,100,30);

laa.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

mainFrame.dispose();

new LoginPage();

}

});

panel0.add(laa);

//Rewinding details (panel 1)

MyPanel panel1 = new MyPanel();

panel1.setLayout(new FlowLayout(FlowLayout.LEFT,20,10));

TitledBorder rewindingDetails = new TitledBorder("<html><font size=6>Rewinding Details");

rewindingDetails.setBorder(new LineBorder(Color.black, 3));

panel1.setBorder(rewindingDetails);

//rewinding buttons

MyButton addRewind = new MyButton(" Add ");

addRewind.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

mainFrame.dispose();

new AddRewind(id);

}

});

MyButton viewRewind = new MyButton(" View ");

viewRewind.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

mainFrame.dispose();

new ViewRewind(id);

}

});

MyButton updateRewind = new MyButton(" Update ");

updateRewind.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

mainFrame.dispose();

new UpdateRewind(id);

}

});

MyButton deleteRewind = new MyButton(" Delete ");

deleteRewind.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

mainFrame.dispose();

new DeleteRewind(id);

}

});

//adding to panel

panel1.add(addRewind);

panel1.add(viewRewind);

panel1.add(updateRewind);

panel1.add(deleteRewind);

c.add(panel1);

//work details (panel 2)

MyPanel panel2 = new MyPanel();

//panel 2 border

TitledBorder workDetails = new TitledBorder("<html><font size=6>Work Details");

workDetails.setBorder(new LineBorder(Color.black, 3));

panel2.setBorder(workDetails);

//panel 2 layout

panel2.setLayout(new FlowLayout(FlowLayout.LEFT,20,10));

//panel 2 buttons

MyButton addWork = new MyButton("Add");

addWork.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

mainFrame.dispose();

new AddWork(id);

}

});

MyButton viewWork= new MyButton(" View ");

viewWork.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

mainFrame.dispose();

new ViewWork(id);

}

});

MyButton updateWork = new MyButton(" Update ");

updateWork.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

mainFrame.dispose();

new UpdateWork(id);

}

});

MyButton workhis = new MyButton(" Work history ");

workhis.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

mainFrame.dispose();

new WorkHistory(id);

}

});

//adding to panel 2

panel2.add(addWork);

panel2.add(viewWork);

panel2.add(updateWork);

panel2.add(workhis);

//adding panel to container

c.add(panel2);

//employee details (panel 3)

MyPanel panel3 = new MyPanel();

//panel 3 border

TitledBorder employeeDetails = new TitledBorder("<html><font size=6>Employee Details");

employeeDetails.setBorder(new LineBorder(Color.black, 3));

panel3.setBorder(employeeDetails);

//panel 3 layout

panel3.setLayout(new FlowLayout(FlowLayout.LEFT,20,10));

//panel 3 buttons

MyButton addEmployee = new MyButton(" Add ");

addEmployee.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

mainFrame.dispose();

new AddEmploy(id);

}

});

MyButton viewEmployee= new MyButton(" View ");

viewEmployee.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

mainFrame.dispose();

new ViewEmploy(id);

}

});

MyButton deleteEmployee = new MyButton(" Delete ");

deleteEmployee.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

mainFrame.dispose();

new DelEmploy(id);

}

});

MyButton leaveEntry = new MyButton(" Leave Entry ");

leaveEntry.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

mainFrame.dispose();

new LeaveEntry(id);

}

});

//adding to panel 3

panel3.add(addEmployee);

panel3.add(viewEmployee);

panel3.add(deleteEmployee);

panel3.add(leaveEntry);

//adding panel to container

c.add(panel3);

MyPanel last = new MyPanel();

last.setLayout(new FlowLayout(FlowLayout.LEFT,50,20) );

MyLabel paneLab1 = new MyLabel("Last login on " + date);

last.add(paneLab1);

c.add(last);

mainFrame.setVisible(true);

}

public static void main(String[] args) {

new LoadFrame();

}

}

**6.Add Rewind Program:**

package Bootathon;

import Bootathon.database.DBOperations;

import Bootathon.uiworks.MyButton;

import Bootathon.uiworks.MyFrame;

import Bootathon.uiworks.MyLabel;

import Bootathon.uiworks.MyTextArea;

import Bootathon.uiworks.MyTextField;

import java.awt.Container;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.BufferedWriter;

import java.io.FileWriter;

import java.io.IOException;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.SQLException;

import javax.swing.JOptionPane;

import javax.swing.JScrollPane;

public class AddRewind {

AddRewind(int id)

{

//Frame creating

MyFrame addrewind=new MyFrame(5);

addrewind.setBounds(500, 100, 600, 800);

//Container Creating

Container addrew\_c=addrewind.getContentPane();

addrew\_c.setLayout(null);

//add rewinding details label creation

MyLabel addrew\_ll= new MyLabel("Add Rewinding Details",1);

addrew\_ll.setBounds(30,30,350,50);

addrew\_c.add(addrew\_ll);

//HorsePower entry

MyLabel addrew\_label1 = new MyLabel("Enter HP ");

MyTextField enterhp = new MyTextField();

addrew\_label1.setBounds(50,130,100,30);

enterhp.setBounds(200,130,300,30);

addrew\_c.add(addrew\_label1);

addrew\_c.add(enterhp);

//Rewind details entry

MyLabel addrew\_l3 = new MyLabel("Enter the details");

addrew\_l3.setBounds(200, 230, 200, 30);

addrew\_c.add(addrew\_l3);

MyTextArea addrew\_ta = new MyTextArea();

// ta.setEditable(false);

JScrollPane sp = new JScrollPane(addrew\_ta,JScrollPane.VERTICAL\_SCROLLBAR\_AS\_NEEDED,JScrollPane.HORIZONTAL\_SCROLLBAR\_AS\_NEEDED);

sp.setBounds(30,280,550,400);

addrew\_c.add(sp);

//Submit and Back

MyButton back = new MyButton("\u2190"+" BACK");

back.setBounds(30,700,150,30);

back.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

addrewind.dispose();

new MainFrame();

}

});

MyButton submit = new MyButton("Submit "+"\u2192");

submit.setBounds(400, 230, 150, 30);

//the data is stored as files in local drive and path is stored in the database

submit.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

if(enterhp.getText().isEmpty() || addrew\_ta.getText().isEmpty())

{

JOptionPane.showMessageDialog(addrewind, "<html><font size=4>Please Fill all the details listed above!","Fill it",JOptionPane.ERROR\_MESSAGE);

}

else if(enterhp.getText().length()>10)

{

JOptionPane.showMessageDialog(addrewind, "The HP value can have a max of 10 characters only!");

}

else{

try{

String path="C:\\Electrical Data\\Empr"+String.valueOf(id)+"\\"+String.valueOf(id)+enterhp.getText()+".txt";

BufferedWriter buff=new BufferedWriter(new FileWriter(path));

buff.write(addrew\_ta.getText());

buff.close();

Connection conn = DBOperations.getConn();

PreparedStatement st=conn.prepareStatement("insert into rewinddet values(?,?,?)");

st.setInt(1, id);

st.setString(2, enterhp.getText());

st.setString(3, path.replace('\\','@'));

st.executeUpdate();

conn.close();

JOptionPane.showMessageDialog(addrewind,"<html><font size=4>Successfully Added.","Successful",JOptionPane.INFORMATION\_MESSAGE);

}

catch(SQLException ee)

{

System.out.println("Cannot insert values into rewinddet"+ee);

}

catch(IOException ee)

{

System.out.println("Cannot find the file in the Specified path"+ee);

}

}

}

});

addrew\_c.add(submit);

addrew\_c.add(back);

addrewind.setVisible(true);

}

public static void main(String[] args) {

new LoadFrame();

}

}

**5.View Rewind Program:**

package Bootathon;

import Bootathon.database.DBOperations;

import Bootathon.uiworks.MyButton;

import Bootathon.uiworks.MyComboBox;

import Bootathon.uiworks.MyFrame;

import Bootathon.uiworks.MyLabel;

import Bootathon.uiworks.MyTextArea;

import Bootathon.uiworks.MyTextField;

import java.awt.Container;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.ItemEvent;

import java.awt.event.ItemListener;

import java.io.BufferedReader;

import java.io.FileReader;

import java.io.IOException;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import javax.swing.JScrollPane;

public class ViewRewind {

ViewRewind(int id){

// TODO code application logic here

MyFrame viewrew\_f = new MyFrame(5);

viewrew\_f.setBounds(500,100,600,800);

//container creating

Container viewrew\_c = viewrew\_f.getContentPane();

viewrew\_c.setLayout(null);

MyLabel ll= new MyLabel("View details",1);

ll.setBounds(30,20,300,30);

viewrew\_c.add(ll);

MyComboBox cb = new MyComboBox();

cb.setEditable(false);

cb.addItem("Select HP");

//adding values to combobox

try{

Connection conn = DBOperations.getConn();

Statement st=conn.createStatement();

ResultSet rs=st.executeQuery("Select hp from rewinddet where emprid="+String.valueOf(id));

while(rs.next())

{

cb.addItem(rs.getString("hp"));

}

conn.close();

}

catch(SQLException ee)

{

System.out.println("Cannot retrieve values from rewinddet"+ee);

}

//HP selection

MyLabel viewrew\_l1 = new MyLabel("Select HP");

viewrew\_l1.setBounds(90,140,200,30);

cb.setBounds(250,140,300,30);

viewrew\_c.add(viewrew\_l1);

viewrew\_c.add(cb);

//displaying Horse Power

MyLabel viewrew\_l2 = new MyLabel("Horse Power ");

viewrew\_c.add(viewrew\_l2);

viewrew\_l2.setBounds(90,240,200,30);

MyTextField viewrew\_t = new MyTextField();

viewrew\_t.setBounds(250,240,300,30);

viewrew\_t.setEditable(false);

viewrew\_c.add(viewrew\_t);

//displaying rewinding details

MyLabel viewrew\_l3 = new MyLabel("Rewinding Details");

viewrew\_l3.setBounds(200,320,200,30);

MyTextArea ta = new MyTextArea();

ta.setEditable(false);

JScrollPane sp = new JScrollPane(ta,JScrollPane.VERTICAL\_SCROLLBAR\_AS\_NEEDED,JScrollPane.HORIZONTAL\_SCROLLBAR\_AS\_NEEDED);

sp.setBounds(80,370,470,300);

viewrew\_c.add(viewrew\_l3);

viewrew\_c.add(sp);

//back action

MyButton back = new MyButton("\u2190"+" BACK");

back.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

viewrew\_f.dispose();

new MainFrame();

}

});

viewrew\_c.add(back);

//fetches and displays the rewinding details from the local drive using the path in database once the item(HP) chosen

cb.addItemListener(new ItemListener() {

@Override

public void itemStateChanged(ItemEvent e) {

String item=(String)cb.getSelectedItem();

try{

Connection conn = DBOperations.getConn();

PreparedStatement st=conn.prepareStatement("Select \* from rewinddet where hp=? and emprid=?");

st.setString(1, item);

st.setInt(2, id);

ResultSet rs=st.executeQuery();

rs.next();

viewrew\_t.setText(rs.getString("hp"));

String path=rs.getString("details").replace('@', '\\');

BufferedReader buff=new BufferedReader(new FileReader(path));

int i;

String det="";

while((i=buff.read())!=-1)

{

det+=(char)i;

}

buff.close();

ta.setText(det);

conn.close();

}

catch(SQLException ee)

{

System.out.println("Cannot retrieve values from rewinddet"+ee);

}

catch(IOException ee)

{

System.out.println("Cannot find the file in the Specified path"+ee);

}

}

});

back.setBounds(30,700,150,30);

viewrew\_f.setVisible(true);

}

public static void main(String[] args) {

new LoadFrame();

}

}

**6.Update Rewind Program:**

package Bootathon;

import Bootathon.database.DBOperations;

import Bootathon.uiworks.MyButton;

import Bootathon.uiworks.MyComboBox;

import Bootathon.uiworks.MyFrame;

import Bootathon.uiworks.MyLabel;

import Bootathon.uiworks.MyTextArea;

import Bootathon.uiworks.MyTextField;

import java.awt.Container;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.ItemEvent;

import java.awt.event.ItemListener;

import java.io.BufferedReader;

import java.io.BufferedWriter;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import javax.swing.JOptionPane;

import javax.swing.JScrollPane;

/\*\*

\*

\* @author rahul

\*/

public class UpdateRewind {

UpdateRewind(int id){

// frame creating

MyFrame f = new MyFrame(4);

f.setBounds(500,100,600,800);

//container creating

Container c = f.getContentPane();

c.setLayout(null);

MyLabel ll= new MyLabel("Update details",1);

ll.setBounds(30,30,200,50);

c.add(ll);

MyComboBox cb = new MyComboBox();

cb.addItem("Select HP");

//adding values to combobox

try{

Connection conn = DBOperations.getConn();

Statement st=conn.createStatement();

ResultSet rs=st.executeQuery("Select hp from rewinddet where emprid="+String.valueOf(id));

while(rs.next())

{

cb.addItem(rs.getString("hp"));

}

conn.close();

}

catch(SQLException ee)

{

System.out.println("Cannot retrieve values from rewinddet"+ee);

}

//select hp combobox

MyLabel l1 = new MyLabel("Select HP");

l1.setBounds(90,140,200,30);

cb.setBounds(250,140,300,30);

cb.setEditable(false);

c.add(l1);

c.add(cb);

//horse power

MyLabel l2 = new MyLabel("Horse Power ");

c.add(l2);

l2.setBounds(90,240,200,30);

MyTextField t = new MyTextField();

t.setBounds(250,240,300,30);

t.setEditable(false);

c.add(t);

//rewind details textarea creation

MyLabel l3 = new MyLabel("Rewinding Details");

l3.setBounds(200,320,200,30);

MyTextArea ta = new MyTextArea();

JScrollPane sp = new JScrollPane(ta,JScrollPane.VERTICAL\_SCROLLBAR\_AS\_NEEDED,JScrollPane.HORIZONTAL\_SCROLLBAR\_AS\_NEEDED);

sp.setBounds(80,370,470,300);

c.add(l3);

c.add(sp);

//back action

MyButton back = new MyButton("\u2190"+" BACK");

back.addActionListener(new ActionListener(){

@Override

public void actionPerformed(ActionEvent e) {

f.dispose();

new MainFrame();

}

});

MyButton update = new MyButton("Update");

back.setBounds(370,700,150,30);

update.setBounds(400,320,150,30);

c.add(update);

c.add(back);

//the chosen hp,rewinding details displayed

cb.addItemListener(new ItemListener() {

@Override

public void itemStateChanged(ItemEvent e) {

String item=(String)cb.getSelectedItem();

try{

Connection conn = DBOperations.getConn();

PreparedStatement st=conn.prepareStatement("Select hp, details from rewinddet where hp=? and emprid=?");

st.setString(1, item);

st.setInt(2, id);

ResultSet rs=st.executeQuery();

rs.next();

t.setText(rs.getString("hp"));

String path=rs.getString("details").replace('@', '\\');

BufferedReader buff=new BufferedReader(new FileReader(path));

int i;

String det="";

while((i=buff.read())!=-1)

{

det+=(char)i;

}

buff.close();

ta.setText(det);

conn.close();

}

catch(SQLException ee)

{

System.out.println("Cannot retrieve values from rewinddet"+ee);

}

catch(IOException ee)

{

System.out.println("Cannot find the file in the Specified path"+ee);

}

}

});

//updated details overwritten

update.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

if(t.getText().isEmpty())

{

JOptionPane.showMessageDialog(f, "<html><font size=4>Please select which detail to update first!","Select",JOptionPane.ERROR\_MESSAGE);

}

else

{

int a=JOptionPane.showConfirmDialog(f,"<html><font size=4>Are you sure to Update?");

if(a==JOptionPane.YES\_OPTION){

try

{

String path="C:\\Electrical Data\\Empr"+String.valueOf(id)+"\\"+String.valueOf(id)+t.getText()+".txt";

BufferedWriter buff=new BufferedWriter(new FileWriter(path));

buff.write(ta.getText());

buff.close();

JOptionPane.showMessageDialog(f,"<html><font size=4>Successfully Updated.","Successful",JOptionPane.INFORMATION\_MESSAGE);

}

catch(IOException ee)

{

System.out.println("Cannot find the file in the Specified path"+ee);

}

}}}

});

f.setVisible(true);

}

public static void main(String[] args) {

new LoadFrame();

}

}

**7.Delete Rewind Program:**

package Bootathon;

import Bootathon.database.DBOperations;

import Bootathon.uiworks.MyButton;

import Bootathon.uiworks.MyComboBox;

import Bootathon.uiworks.MyFrame;

import Bootathon.uiworks.MyLabel;

import Bootathon.uiworks.MyTextArea;

import Bootathon.uiworks.MyTextField;

import java.awt.Container;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.ItemEvent;

import java.awt.event.ItemListener;

import java.io.BufferedReader;

import java.io.File;

import java.io.FileReader;

import java.io.IOException;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import javax.swing.JOptionPane;

import javax.swing.JScrollPane;

public class DeleteRewind {

DeleteRewind(int id)

{

// frame creating

MyFrame f = new MyFrame(4);

f.setBounds(500,100,600,800);

//container creating

Container c = f.getContentPane();

c.setLayout(null);

//title panel

MyLabel ll= new MyLabel("Delete details",1);

ll.setBounds(10,20,200,50);

c.add(ll);

//panel1 creating

MyComboBox cb = new MyComboBox();

cb.addItem("Select HP");

//adding values to combobox

try{

Connection conn = DBOperations.getConn();

Statement st=conn.createStatement();

ResultSet rs=st.executeQuery("Select hp from rewinddet where emprid="+String.valueOf(id));

while(rs.next())

{

cb.addItem(rs.getString("hp"));

}

conn.close();

}

catch(SQLException ee)

{

System.out.println("Cannot retrieve values from rewinddet"+ee);

}

MyLabel l1 = new MyLabel("Select HP ");

l1.setBounds(30,100,150,30);

cb.setBounds(230,100,300,30);

c.add(l1);

c.add(cb);

//panel2 creating

MyLabel l2 = new MyLabel("Horse Power ");

l2.setBounds(30,170,150,30);

c.add(l2);

MyTextField t = new MyTextField();

t.setEditable(false);

t.setBounds(230,170,300,30);

c.add(t);

//panel3 creating

MyLabel l3 = new MyLabel("Rewinding Details");

MyTextArea ta = new MyTextArea();

l3.setBounds(200,270,200,30);

ta.setEditable(false);

JScrollPane sp = new JScrollPane(ta,JScrollPane.VERTICAL\_SCROLLBAR\_AS\_NEEDED,JScrollPane.HORIZONTAL\_SCROLLBAR\_AS\_NEEDED);

sp.setBounds(70, 320, 500, 350);

c.add(sp);

c.add(l3);

MyButton back = new MyButton("\u2190"+" BACK");

back.setBounds(300,700,150,30);

back.addActionListener(new ActionListener(){

@Override

public void actionPerformed(ActionEvent e) {

f.dispose();

new MainFrame();

}

});

MyButton delete = new MyButton("Delete");

delete.setBounds(400,270,100,30);

c.add(delete);

c.add(back);

//the chosen hp,rewinding details displayed

cb.addItemListener(new ItemListener() {

@Override

public void itemStateChanged(ItemEvent e) {

String item=(String)cb.getSelectedItem();

try{

Connection conn = DBOperations.getConn();

PreparedStatement st=conn.prepareStatement("Select hp, details from rewinddet where hp=? and emprid=?");

st.setString(1, item);

st.setInt(2, id);

ResultSet rs=st.executeQuery();

rs.next();

t.setText(rs.getString("hp"));

t.setText(rs.getString("hp"));

String path=rs.getString("details").replace('@', '\\');

BufferedReader buff=new BufferedReader(new FileReader(path));

int i;

String det="";

while((i=buff.read())!=-1)

{

det+=(char)i;

}

buff.close();

ta.setText(det);

conn.close();

}

catch(SQLException ee)

{

System.out.println("Cannot retrieve values from rewinddet"+ee);

}

catch(IOException ee)

{

System.out.println("Cannot find the file in the Specified path"+ee);

}

}

});

//the rewinding details are deleted along with file as wellas the path in the database

delete.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

if(t.getText().isEmpty())

{

JOptionPane.showMessageDialog(f, "<html><font size=4>Please select which detail to delete first!","Select",JOptionPane.ERROR\_MESSAGE);

}

else

{

int a=JOptionPane.showConfirmDialog(f,"<html><font size=4>Are you sure to Delete?","Alert",JOptionPane.ERROR\_MESSAGE);

if(a==JOptionPane.YES\_OPTION){

try

{

Connection conn = DBOperations.getConn();

PreparedStatement st=conn.prepareStatement("delete from rewinddet where emprid=? and hp=?");

st.setInt(1, id);

st.setString(2, t.getText());

String path="C:\\Electrical Data\\Empr"+String.valueOf(id)+"\\"+String.valueOf(id)+t.getText()+".txt";

new File(path).delete();

st.executeUpdate();

conn.close();

JOptionPane.showMessageDialog(f,"<html><font size=4>Successfully Deleted.","Successful",JOptionPane.INFORMATION\_MESSAGE);

}

catch(SQLException ee)

{

System.out.println("Cannot delete the values from rewinddet"+ee);

}

f.dispose();

new DeleteRewind(id);

}}}

});

f.setVisible(true);

}

public static void main(String[] args) {

new LoadFrame();

}

}

**9.Add Work Program:**

package Bootathon;

import Bootathon.database.DBOperations;

import Bootathon.uiworks.MyButton;

import Bootathon.uiworks.MyFrame;

import Bootathon.uiworks.MyLabel;

import Bootathon.uiworks.MyTextField;

import java.awt.Container;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.SQLException;

import java.util.regex.Pattern;

import javax.swing.JOptionPane;

import javax.swing.JPasswordField;

public class AddEmploy {

AddEmploy(int id)

{

// frame creating

MyFrame addemp\_f = new MyFrame(2);

addemp\_f.setBounds(500,100,600,800);

//container creating

Container c = addemp\_f.getContentPane();

c.setLayout(null);

//label1

MyLabel l1 = new MyLabel("Add Employee",1);

l1.setBounds(30,10,300,50);

c.add(l1);

//employee name

MyLabel l2 = new MyLabel("Employee Name");

l2.setBounds(50,70,200,50);

MyTextField t = new MyTextField();

t.setBounds(250,80,200,30);

c.add(l2);

c.add(t);

//address

MyLabel l3 = new MyLabel("Address");

l3.setBounds(120,140,200,50);

MyTextField t1 = new MyTextField();

t1.setBounds(250,150,200,30);

c.add(l3);

c.add(t1);

//DOB

MyLabel l4 = new MyLabel("D.O.B");

l4.setBounds(140,210,200,50);

MyTextField t2 = new MyTextField();

t2.setBounds(250,220,200,30);

c.add(l4);

c.add(t2);

//phone no

MyLabel l5 = new MyLabel("Phone No");

l5.setBounds(105,280,200,50);

MyTextField t3 = new MyTextField();

t3.setBounds(250,290,200,30);

c.add(l5);

c.add(t3);

//salary

MyLabel l6 = new MyLabel("Salary");

l6.setBounds(135,350,200,50);

MyTextField t4 = new MyTextField();

t4.setBounds(250,360,200,30);

c.add(l6);

c.add(t4);

//email id

MyLabel l7 = new MyLabel("Email ID");

l7.setBounds(115,420,200,50);

MyTextField t5 = new MyTextField();

t5.setBounds(250,430,200,30);

c.add(l7);

c.add(t5);

//password

MyLabel l8 = new MyLabel("Password");

l8.setBounds(105,490,200,50);

JPasswordField t6 = new JPasswordField();

t6.setBounds(250,500,200,30);

c.add(l8);

c.add(t6);

//retype password

MyLabel l9 = new MyLabel("Re-type Password");

l9.setBounds(30,560,200,50);

JPasswordField t7 = new JPasswordField();

t7.setBounds(250,570,200,30);

c.add(l9);

c.add(t7);

MyButton enter = new MyButton("Enter");

enter.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

if(t.getText().isEmpty() || t1.getText().isEmpty() || t2.getText().isEmpty() || t3.getText().isEmpty() || t4.getText().isEmpty() || t5.getText().isEmpty() ||String.valueOf(t6.getPassword()).isEmpty() || String.valueOf(t7.getPassword()).isEmpty())

{

JOptionPane.showMessageDialog(addemp\_f, "<html><font size=4>Please Fill all the details listed above!","Fill it",JOptionPane.ERROR\_MESSAGE);

}

else if(t.getText().length()>20)

{

JOptionPane.showMessageDialog(addemp\_f, "<html><font size=4>The Employee Name can have a max of 20 Characters only!","Change it",JOptionPane.ERROR\_MESSAGE);

}

else if(t1.getText().length()>25)

{

JOptionPane.showMessageDialog(addemp\_f, "<html><font size=4>The Address can have a max of 25 Characters only!","Change it",JOptionPane.ERROR\_MESSAGE);

}

else if(t3.getText().length()>10)

{

JOptionPane.showMessageDialog(addemp\_f, "<html><font size=4>The Phone No. can have a max size of 10 only!","Change it",JOptionPane.ERROR\_MESSAGE);

}

else if(t6.getText().length()>15 || t6.getText().length()<8)

{

JOptionPane.showMessageDialog(addemp\_f, "<html><font size=4>The Password should be of size min 8 characters and a max of 15 characters!","Change it",JOptionPane.ERROR\_MESSAGE);

}

else if(!Pattern.matches("^[0-9]+$", t4.getText()))

{

JOptionPane.showMessageDialog(addemp\_f, "<html><font size=4>The Salary should be integer value!","Change it",JOptionPane.ERROR\_MESSAGE);

}

else if(Integer.valueOf(t4.getText())>100000)

{

JOptionPane.showMessageDialog(addemp\_f, "<html><font size=4>The Salary cannot be greater than 1 Lakh!","Change it",JOptionPane.ERROR\_MESSAGE);

}

else if(!Pattern.compile("^(3[01]|[12][0-9]|0[1-9])/(1[0-2]|0[1-9])/[0-9]{4}$").matcher(t2.getText()).matches())//date validation

{

JOptionPane.showMessageDialog(addemp\_f, "<html><font size=4>Invalid Date or Invalid Format! use (dd/mm/yyyy)!","Invalid!",JOptionPane.ERROR\_MESSAGE);

}

else if(!Pattern.compile("^[\\w-\_\\.+]\*[\\w-\_\\.]\\@([\\w]+\\.)+[\\w]+[\\w]$").matcher(t5.getText()).matches())//mail validation

{

JOptionPane.showMessageDialog(addemp\_f, "<html><font size=4>Invalid Email Address!","Invalid!",JOptionPane.ERROR\_MESSAGE);

}

else if(!String.valueOf(t6.getPassword()).equals(String.valueOf(t7.getPassword())))

{

JOptionPane.showMessageDialog(addemp\_f, "<html><font size=4>The Password and Retype Password does not match!","Re-type Error",JOptionPane.ERROR\_MESSAGE);

}

//inserting employee data to the database

else{

try{

Connection conn = DBOperations.getConn();

PreparedStatement st=conn.prepareStatement("insert into employdet values(?,0,?,?,?,?,?,?,?,?,0)");

st.setInt(1, id);

st.setString(2, t.getText());

st.setString(3, t1.getText());

st.setString(4, t2.getText());

st.setString(5, t3.getText());

st.setInt(6, Integer.valueOf(t4.getText()));

st.setInt(7, Integer.valueOf(t4.getText()));

st.setString(8, t5.getText());

st.setString(9, String.valueOf(t6.getPassword()));

st.executeUpdate();

conn.close();

JOptionPane.showMessageDialog(addemp\_f,"<html><font size=4>Successfully Added.","Successful",JOptionPane.INFORMATION\_MESSAGE);

}

catch(NumberFormatException ee)

{

System.out.println("Salary value should be an integer value"+ee);

}

catch(SQLException ee)

{

System.out.println("Couldnt insert values into employdet"+ee);

}

addemp\_f.dispose();

new AddEmploy(id);

}}

});

//back action

MyButton bac = new MyButton("\u2190"+" BACK");

bac.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

addemp\_f.dispose();

new MainFrame();

}

});

enter.setBounds(100,670,130,30);

bac.setBounds(310,670,130,30);

c.add(enter);

c.add(bac);

addemp\_f.setVisible(true);

}

public static void main(String[] args) {

new LoadFrame();

}

}

**9.View Work Program:**

package Bootathon;

import Bootathon.database.DBOperations;

import Bootathon.uiworks.MyButton;

import Bootathon.uiworks.MyComboBox;

import Bootathon.uiworks.MyFrame;

import Bootathon.uiworks.MyLabel;

import Bootathon.uiworks.MyTextArea;

import Bootathon.uiworks.MyTextField;

import java.awt.Container;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.ItemEvent;

import java.awt.event.ItemListener;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Calendar;

import java.util.Date;

import javax.swing.JScrollPane;

class ViewWork {

ViewWork(int id){

// frame creating

MyFrame viewwk\_f = new MyFrame(4);

viewwk\_f.setBounds(500,100,600,800);

//container creating

Container viewwk\_c = viewwk\_f.getContentPane();

viewwk\_c.setLayout(null);

MyLabel ll= new MyLabel("View details",1);

ll.setBounds(30,30,200,50);

viewwk\_c.add(ll);

MyComboBox cb = new MyComboBox();

cb.addItem("Select WorkID");

//Adding values to combobox with matured SD

try{

Connection conn = DBOperations.getConn();

Statement st=conn.createStatement();

ResultSet rs=st.executeQuery("Select Date, workid ,CollectionStatus from workdet where emprid="+String.valueOf(id));

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

Calendar cal=Calendar.getInstance();

while(rs.next())

{

cal.setTime(sdf.parse(rs.getString("Date")));

cal.add(Calendar.YEAR, 1);

if(cal.getTime().before(new Date()) && rs.getInt("CollectionStatus")==0)

cb.addItem(rs.getString("workid"));

}

conn.close();

}

catch(ParseException ee)

{

System.out.println("Invalid Date Format"+ee);

}

catch(SQLException ee)

{

System.out.println("Cannot retrieve values from DB!!"+ee);

}

MyLabel l1 = new MyLabel("Select Work ID ");

l1.setBounds(90,140,200,30);

cb.setBounds(250,140,300,30);

viewwk\_c.add(l1);

viewwk\_c.add(cb);

cb.setEditable(false);

MyLabel l2 = new MyLabel("Security Deposit Value ");

viewwk\_c.add(l2);

l2.setBounds(10,230,250,30);

MyTextField t = new MyTextField();

t.setEditable(false);

viewwk\_c.add(t);

t.setBounds(250,230,300,30);

MyLabel l3 = new MyLabel("Rewinding Details");

l3.setBounds(200,350,200,30);

viewwk\_c.add(l3);

MyTextArea ta = new MyTextArea();

ta.setEditable(false);

JScrollPane sp = new JScrollPane(ta,JScrollPane.VERTICAL\_SCROLLBAR\_AS\_NEEDED,JScrollPane.HORIZONTAL\_SCROLLBAR\_AS\_NEEDED);

sp.setBounds(80,400,450,300);

viewwk\_c.add(sp);

//back action

MyButton back = new MyButton("\u2190"+" BACK");

back.addActionListener(new ActionListener(){

@Override

public void actionPerformed(ActionEvent e) {

viewwk\_f.dispose();

new MainFrame();

}

});;

MyButton refund=new MyButton("SD Collected");

//changes the collection status of SD as 'Collected!'.

refund.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

try{

Connection conn = DBOperations.getConn();

PreparedStatement st=conn.prepareStatement("update workdet set CollectionStatus=1 where NameOfWork=? and emprid=?");

st.setString(1, ta.getText());

st.setInt(2, id);

st.executeUpdate();

conn.close();

}

catch(SQLException ee)

{

System.out.println("Cannot update values"+ee);

}

viewwk\_f.dispose();

new ViewWork(id);

}

});

//Displaying the selected item from combobox

cb.addItemListener(new ItemListener() {

@Override

public void itemStateChanged(ItemEvent e) {

String item=(String)cb.getSelectedItem();

try{

Connection conn = DBOperations.getConn();

PreparedStatement st=conn.prepareStatement("Select NameOfWork, Amount, workid from workdet where workid=? and emprid=?");

st.setString(1,item);

st.setInt(2, id);

ResultSet rs=st.executeQuery();

rs.next();

t.setText(rs.getString("Amount"));

ta.setText(rs.getString("NameOfWork"));

conn.close();

}

catch(SQLException ee)

{

System.out.println("Cannot retrieve values from workdet"+ee);

}

}

});

back.setBounds(340,710,150,30);

refund.setBounds(400,350,150,30);

viewwk\_c.add(refund);

viewwk\_c.add(back);

viewwk\_f.setVisible(true);

}

public static void main(String[] args) {

new LoadFrame();

}

}

**10.Work History Program:**

package Bootathon;

import Bootathon.database.DBOperations;

import Bootathon.uiworks.MyButton;

import Bootathon.uiworks.MyFrame;

import Bootathon.uiworks.MyLabel;

import Bootathon.uiworks.MyTable;

import java.awt.Container;

import javax.swing.JScrollPane;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import javax.swing.table.DefaultTableModel;

public class WorkHistory {

WorkHistory(int id)

{

//frame creation

MyFrame wh\_frame = new MyFrame(1);

wh\_frame.setBounds(100,200,1350,700);

//container creation

Container wh\_c = wh\_frame.getContentPane();

wh\_c.setLayout(null);

//work details label craetion

MyLabel label1 = new MyLabel("Work details",1);

label1.setBounds(30,5,200,50);

wh\_c.add(label1);

//table creation

MyTable table = new MyTable();

DefaultTableModel model = new DefaultTableModel();

model.addColumn(" Work Id ");

model.addColumn("Name of Work");

model.addColumn("Security deposit");

model.addColumn("Date");

model.addColumn("SD collection status");

table.setModel(model);

table.getColumnModel().getColumn(1).setPreferredWidth(350);

String coll\_status[]={"Pending","Collected"};

//data insertion to the table

try

{

Connection conn=DBOperations.getConn();

PreparedStatement st=conn.prepareStatement("select \* from workdet where emprid=?");

st.setInt(1, id);

ResultSet rs=st.executeQuery();

while(rs.next())

{

model.addRow(new Object[]{String.valueOf(rs.getInt("workid")),rs.getString("NameOfWork"),rs.getString("Amount"),rs.getString("Date"),coll\_status[rs.getInt("CollectionStatus")]});

}

conn.close();

}

catch(SQLException ee)

{

System.out.println("Cannot retrieve values from workdet"+ee);

}

table.setRowHeight(30);

JScrollPane scroll = new JScrollPane(table,JScrollPane.VERTICAL\_SCROLLBAR\_AS\_NEEDED,JScrollPane.HORIZONTAL\_SCROLLBAR\_AS\_NEEDED);

scroll.setBounds(20,70,1300,500);

wh\_c.add(scroll);

//back action

MyButton back = new MyButton("\u2190"+" BACK");

back.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

wh\_frame.dispose();

new MainFrame();

}

});

back.setBounds(30,600,150,30);

wh\_c.add(back);

wh\_frame.setVisible(true);

}

public static void main(String[] args) {

new LoadFrame();

}

}

**11.Add Employee Program:**

package Bootathon;

import Bootathon.database.DBOperations;

import Bootathon.uiworks.MyButton;

import Bootathon.uiworks.MyFrame;

import Bootathon.uiworks.MyLabel;

import Bootathon.uiworks.MyTextField;

import java.awt.Container;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.SQLException;

import java.util.regex.Pattern;

import javax.swing.JOptionPane;

import javax.swing.JPasswordField;

public class AddEmploy {

AddEmploy(int id)

{

// frame creating

MyFrame addemp\_f = new MyFrame(2);

addemp\_f.setBounds(500,100,600,800);

//container creating

Container c = addemp\_f.getContentPane();

c.setLayout(null);

//label1

MyLabel l1 = new MyLabel("Add Employee",1);

l1.setBounds(30,10,300,50);

c.add(l1);

//employee name

MyLabel l2 = new MyLabel("Employee Name");

l2.setBounds(50,70,200,50);

MyTextField t = new MyTextField();

t.setBounds(250,80,200,30);

c.add(l2);

c.add(t);

//address

MyLabel l3 = new MyLabel("Address");

l3.setBounds(120,140,200,50);

MyTextField t1 = new MyTextField();

t1.setBounds(250,150,200,30);

c.add(l3);

c.add(t1);

//DOB

MyLabel l4 = new MyLabel("D.O.B");

l4.setBounds(140,210,200,50);

MyTextField t2 = new MyTextField();

t2.setBounds(250,220,200,30);

c.add(l4);

c.add(t2);

//phone no

MyLabel l5 = new MyLabel("Phone No");

l5.setBounds(105,280,200,50);

MyTextField t3 = new MyTextField();

t3.setBounds(250,290,200,30);

c.add(l5);

c.add(t3);

//salary

MyLabel l6 = new MyLabel("Salary");

l6.setBounds(135,350,200,50);

MyTextField t4 = new MyTextField();

t4.setBounds(250,360,200,30);

c.add(l6);

c.add(t4);

//email id

MyLabel l7 = new MyLabel("Email ID");

l7.setBounds(115,420,200,50);

MyTextField t5 = new MyTextField();

t5.setBounds(250,430,200,30);

c.add(l7);

c.add(t5);

//password

MyLabel l8 = new MyLabel("Password");

l8.setBounds(105,490,200,50);

JPasswordField t6 = new JPasswordField();

t6.setBounds(250,500,200,30);

c.add(l8);

c.add(t6);

//retype password

MyLabel l9 = new MyLabel("Re-type Password");

l9.setBounds(30,560,200,50);

JPasswordField t7 = new JPasswordField();

t7.setBounds(250,570,200,30);

c.add(l9);

c.add(t7);

MyButton enter = new MyButton("Enter");

enter.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

if(t.getText().isEmpty() || t1.getText().isEmpty() || t2.getText().isEmpty() || t3.getText().isEmpty() || t4.getText().isEmpty() || t5.getText().isEmpty() ||String.valueOf(t6.getPassword()).isEmpty() || String.valueOf(t7.getPassword()).isEmpty())

{

JOptionPane.showMessageDialog(addemp\_f, "<html><font size=4>Please Fill all the details listed above!","Fill it",JOptionPane.ERROR\_MESSAGE);

}

else if(t.getText().length()>20)

{

JOptionPane.showMessageDialog(addemp\_f, "<html><font size=4>The Employee Name can have a max of 20 Characters only!","Change it",JOptionPane.ERROR\_MESSAGE);

}

else if(t1.getText().length()>25)

{

JOptionPane.showMessageDialog(addemp\_f, "<html><font size=4>The Address can have a max of 25 Characters only!","Change it",JOptionPane.ERROR\_MESSAGE);

}

else if(t3.getText().length()>10)

{

JOptionPane.showMessageDialog(addemp\_f, "<html><font size=4>The Phone No. can have a max size of 10 only!","Change it",JOptionPane.ERROR\_MESSAGE);

}

else if(t6.getText().length()>15 || t6.getText().length()<8)

{

JOptionPane.showMessageDialog(addemp\_f, "<html><font size=4>The Password should be of size min 8 characters and a max of 15 characters!","Change it",JOptionPane.ERROR\_MESSAGE);

}

else if(!Pattern.matches("^[0-9]+$", t4.getText()))

{

JOptionPane.showMessageDialog(addemp\_f, "<html><font size=4>The Salary should be integer value!","Change it",JOptionPane.ERROR\_MESSAGE);

}

else if(Integer.valueOf(t4.getText())>100000)

{

JOptionPane.showMessageDialog(addemp\_f, "<html><font size=4>The Salary cannot be greater than 1 Lakh!","Change it",JOptionPane.ERROR\_MESSAGE);

}

else if(!Pattern.compile("^(3[01]|[12][0-9]|0[1-9])/(1[0-2]|0[1-9])/[0-9]{4}$").matcher(t2.getText()).matches())//date validation

{

JOptionPane.showMessageDialog(addemp\_f, "<html><font size=4>Invalid Date or Invalid Format! use (dd/mm/yyyy)!","Invalid!",JOptionPane.ERROR\_MESSAGE);

}

else if(!Pattern.compile("^[\\w-\_\\.+]\*[\\w-\_\\.]\\@([\\w]+\\.)+[\\w]+[\\w]$").matcher(t5.getText()).matches())//mail validation

{

JOptionPane.showMessageDialog(addemp\_f, "<html><font size=4>Invalid Email Address!","Invalid!",JOptionPane.ERROR\_MESSAGE);

}

else if(!String.valueOf(t6.getPassword()).equals(String.valueOf(t7.getPassword())))

{

JOptionPane.showMessageDialog(addemp\_f, "<html><font size=4>The Password and Retype Password does not match!","Re-type Error",JOptionPane.ERROR\_MESSAGE);

}

//inserting employee data to the database

else{

try{

Connection conn = DBOperations.getConn();

PreparedStatement st=conn.prepareStatement("insert into employdet values(?,0,?,?,?,?,?,?,?,?,0)");

st.setInt(1, id);

st.setString(2, t.getText());

st.setString(3, t1.getText());

st.setString(4, t2.getText());

st.setString(5, t3.getText());

st.setInt(6, Integer.valueOf(t4.getText()));

st.setInt(7, Integer.valueOf(t4.getText()));

st.setString(8, t5.getText());

st.setString(9, String.valueOf(t6.getPassword()));

st.executeUpdate();

conn.close();

JOptionPane.showMessageDialog(addemp\_f,"<html><font size=4>Successfully Added.","Successful",JOptionPane.INFORMATION\_MESSAGE);

}

catch(NumberFormatException ee)

{

System.out.println("Salary value should be an integer value"+ee);

}

catch(SQLException ee)

{

System.out.println("Couldnt insert values into employdet"+ee);

}

addemp\_f.dispose();

new AddEmploy(id);

}}

});

//back action

MyButton bac = new MyButton("\u2190"+" BACK");

bac.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

addemp\_f.dispose();

new MainFrame();

}

});

enter.setBounds(100,670,130,30);

bac.setBounds(310,670,130,30);

c.add(enter);

c.add(bac);

addemp\_f.setVisible(true);

}

public static void main(String[] args) {

new LoadFrame();

}

}

**12.Leave Entry Program:**

package Bootathon;

import Bootathon.database.DBOperations;

import Bootathon.uiworks.MyButton;

import Bootathon.uiworks.MyComboBox;

import Bootathon.uiworks.MyFrame;

import Bootathon.uiworks.MyLabel;

import Bootathon.uiworks.MyTextField;

import java.awt.Container;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.ItemEvent;

import java.awt.event.ItemListener;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import java.util.regex.Pattern;

import javax.swing.JOptionPane;

public class LeaveEntry {

LeaveEntry(int id)

{

// frame creating

MyFrame f = new MyFrame(4);

f.setBounds(500,100,600,800);

//container creating

Container c = f.getContentPane();

c.setLayout(null);

//label 1

MyLabel l1 = new MyLabel("Leave Entry",1);

l1.setBounds(30,10,300,50);

c.add(l1);

//select employee

MyLabel l2 = new MyLabel("Select Employee");

l2.setBounds(50,100,300,50);

MyComboBox cb = new MyComboBox();

cb.addItem("Select EmpID");

//adding values to combobox

try{

Connection conn = DBOperations.getConn();

Statement st=conn.createStatement();

ResultSet rs=st.executeQuery("Select empid from employdet where emprid="+String.valueOf(id));

while(rs.next())

{

cb.addItem(rs.getString("empid"));

}

conn.close();

}

catch(SQLException ee)

{System.out.println("Cannot retrieve values from employdet"+ee);}

cb.setBounds(250,110,300,30);

cb.setEditable(false);

c.add(l2);

c.add(cb);

//employee id

MyLabel l3 = new MyLabel("Employee ID ");

l3.setBounds(50,170,300,50);

MyTextField t1 = new MyTextField();

t1.setBounds(250,180,300,30);

t1.setEditable(false);

c.add(l3);

c.add(t1);

//employee name

MyLabel l4 = new MyLabel("Employee name");

l4.setBounds(50,240,300,50);

MyTextField t2 = new MyTextField();

t2.setBounds(250,250,300,30);

t2.setEditable(false);

c.add(l4);

c.add(t2);

//dob

MyLabel l5 = new MyLabel(" No. of Days ");

l5.setBounds(50,310,300,50);

MyTextField t3 = new MyTextField();

t3.setBounds(250,320,300,30);

c.add(l5);

c.add(t3);

//buttons

MyButton enter = new MyButton("Enter");

//back action

MyButton bac = new MyButton("\u2190"+" BACK");

bac.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

f.dispose();

new MainFrame();

}

});

enter.setBounds(400,410,130,30);

bac.setBounds(400,480,130,30);

c.add(enter);

c.add(bac);

//selected employee details are displayed

cb.addItemListener(new ItemListener() {

@Override

public void itemStateChanged(ItemEvent e) {

try{

Connection conn = DBOperations.getConn();

PreparedStatement st=conn.prepareStatement("Select empid, EmpName from employdet where empid=? and emprid=?");

st.setInt(1, Integer.valueOf((String)cb.getSelectedItem()));

st.setInt(2, id);

ResultSet rs=st.executeQuery();

rs.next();

t1.setText(String.valueOf(rs.getInt("empid")));

t2.setText(rs.getString("EmpName"));

conn.close();

}

catch(NumberFormatException ee)

{

System.out.println("The selected item should be integer"+ee);

}

catch(SQLException ee)

{

System.out.println("Cannot retrieve values from employdet"+ee);

}

}

});

//Number of leave days entered,accordingly salary deducted

enter.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

if(t1.getText().isEmpty())

{

JOptionPane.showMessageDialog(f, "<html><font size=4>select an employee id first!","Select",JOptionPane.ERROR\_MESSAGE);

}

else if(t3.getText().isEmpty())

{

JOptionPane.showMessageDialog(f, "<html><font size=4>specify No. of Days!","Specify",JOptionPane.ERROR\_MESSAGE);

}

else if(!Pattern.matches("^[0-9]+$", t3.getText()))

{

JOptionPane.showMessageDialog(f, "<html><font size=4>No. of Days is an Integer Value!","Specify",JOptionPane.ERROR\_MESSAGE);

}

else if(Integer.valueOf(t3.getText())>30)

{

JOptionPane.showMessageDialog(f, "<html><font size=4>No. of Days cannot exceed 30!","Specify",JOptionPane.ERROR\_MESSAGE);

}

else

{

int a=JOptionPane.showConfirmDialog(f,"<html><font size=4>Are you sure?");

if(a==JOptionPane.YES\_OPTION){

try{

Connection conn = DBOperations.getConn();

PreparedStatement st=conn.prepareStatement("Select Salary, CurSalary from employdet where empid=? and emprid=?");

st.setInt(1, Integer.valueOf((String)cb.getSelectedItem()));

st.setInt(2, id);

ResultSet rs=st.executeQuery();

rs.next();

int cursal=Integer.valueOf(rs.getString("CurSalary")); //current salary

int sal=Integer.valueOf(rs.getString("Salary")); //monthly salary

int onedaysal=sal/30; //one day salary

PreparedStatement pst=conn.prepareStatement("update employdet set CurSalary=? where empid=? and emprid=?");

pst.setInt(1, cursal-(onedaysal\*Integer.valueOf(t3.getText())));

pst.setInt(2, Integer.valueOf((String)cb.getSelectedItem()));

pst.setInt(3, id);

pst.executeUpdate();

conn.close();

JOptionPane.showMessageDialog(f,"<html><font size=4>Successfully Marked!","Successful",JOptionPane.INFORMATION\_MESSAGE);

}

catch(NumberFormatException ee)

{

System.out.println("The selected item should be integer"+ee);

}

catch(SQLException ee)

{

System.out.println("Cannot retrieve values from employdet"+ee);

}

f.dispose();

new LeaveEntry(id);

}}}

});

f.setVisible(true);

}

public static void main(String[] args) {

new LoadFrame();

}

}

**13.Employee Main Page Program:**

package Bootathon;

import Bootathon.uiworks.MyButton;

import Bootathon.uiworks.MyFrame;

import Bootathon.uiworks.MyLabel;

import Bootathon.uiworks.MyTextField;

import java.awt.Container;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class EmployeeMain {

EmployeeMain(String name,int id,String add,String dob,String phone,int sal,int cur\_sal)

{

//creating components

MyFrame empmain\_f = new MyFrame(2);

Container empmain\_c = empmain\_f.getContentPane();

MyLabel Top = new MyLabel("Hi, "+name,1);Top.setBounds(20,20,300,40);

MyLabel NameL = new MyLabel("Name ");NameL.setBounds(20,110,150,30);

MyLabel IDL = new MyLabel("I.D No ");IDL.setBounds(20,180,150,30);

MyLabel AddL = new MyLabel("Address ");AddL.setBounds(20,240,150,30);

MyLabel DBL = new MyLabel("D.O.B ");DBL.setBounds(20,300,150,30);

MyLabel PhoneL = new MyLabel("Phone No ");PhoneL.setBounds(20,360,150,30);

MyLabel SalaryL = new MyLabel("Salary ");SalaryL.setBounds(20,420,150,30);

MyLabel CSalaryL = new MyLabel("Current Salary ");CSalaryL.setBounds(20,480,150,30);

//setting bounds of the components and the data of the employee

MyTextField NameT = new MyTextField(); NameT.setBounds(250,110,300,30); NameT.setEditable(false);

NameT.setText(" "+name);

MyTextField IDT = new MyTextField(); IDT.setBounds(250,180,300,30); IDT.setEditable(false);

IDT.setText(" "+String.valueOf(id));

MyTextField AddT = new MyTextField(); AddT.setBounds(250,240,300,30); AddT.setEditable(false);

AddT.setText(" "+add);

MyTextField DBT = new MyTextField(); DBT.setBounds(250,300,300,30); DBT.setEditable(false);

DBT.setText(" "+dob);

MyTextField PhoneT = new MyTextField(); PhoneT.setBounds(250,360,300,30); PhoneT.setEditable(false);

PhoneT.setText(" "+phone);

MyTextField SalaryT = new MyTextField(); SalaryT.setBounds(250,420,300,30); SalaryT.setEditable(false);

SalaryT.setText(" "+String.valueOf(sal));

MyTextField CSalaryT = new MyTextField(); CSalaryT.setBounds(250,480,300,30); CSalaryT.setEditable(false);

CSalaryT.setText(" "+String.valueOf(cur\_sal));

MyButton logout = new MyButton("Log Out");

logout.setBounds(400,30,120,30);

//logout action

logout.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

empmain\_f.dispose();

new LoginPage();

}

});

//adding components to the container

empmain\_c.add(Top);

empmain\_c.add(NameL);empmain\_c.add(IDL);empmain\_c.add(AddL);empmain\_c.add(DBL);empmain\_c.add(PhoneL);

empmain\_c.add(SalaryL);empmain\_c.add(CSalaryL);

empmain\_c.add(NameT);empmain\_c.add(IDT);empmain\_c.add(AddT);empmain\_c.add(DBT);empmain\_c.add(PhoneT);

empmain\_c.add(SalaryT);empmain\_c.add(CSalaryT);

empmain\_c.add(logout);

empmain\_c.setBounds(500,100,600,800);

empmain\_c.setLayout(null);

empmain\_f.setVisible(true);

}

public static void main(String[] args) {

new LoadFrame();

}

}

**14.Admin Add Organization Program:**

package Bootathon.Admin;

import Bootathon.database.DBOperations;

import Bootathon.uiworks.MyButton;

import Bootathon.uiworks.MyFrame;

import Bootathon.uiworks.MyLabel;

import Bootathon.uiworks.MyTextField;

import java.awt.Container;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.SQLException;

import java.util.UUID;

import javax.swing.JOptionPane;

public class AdminAddOrg {

AdminAddOrg()

{

//frame creation

MyFrame addorg\_frame = new MyFrame(2);

addorg\_frame.setBounds(500,200,500,700);

addorg\_frame.setDefaultCloseOperation(3);

Container c = addorg\_frame.getContentPane();

c.setLayout(null);

//add organization label creation

MyLabel addorg\_l1 = new MyLabel("Add organization",1);

addorg\_l1.setBounds(10,20,300,50);

c.add(addorg\_l1);

//organization name label and textfield creation

MyLabel orgname\_l2 = new MyLabel("Organization Name");

orgname\_l2.setBounds(50,130,200,30);

c.add(orgname\_l2);

MyTextField orgname\_t1 = new MyTextField();

orgname\_t1.setBounds(250,130,200,30);

c.add(orgname\_t1);

//activation key label and textfield creation

MyLabel ak\_l3 = new MyLabel("Activation key");

ak\_l3.setBounds(50,200,150,30);

c.add(ak\_l3);

MyTextField ak\_t2 = new MyTextField();

ak\_t2.setEditable(false);

ak\_t2.setBounds(250,200,200,30);

c.add(ak\_t2);

//generate button creation

MyButton gen = new MyButton("Generate");

//UUID generation

gen.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

if(orgname\_t1.getText().isEmpty())

{

JOptionPane.showMessageDialog(addorg\_frame, "Please Fill the Organization Name!");

}

else if(orgname\_t1.getText().length()>20)

{

JOptionPane.showMessageDialog(addorg\_frame, "The Organization Name can have a max of 20 characters!");

}

else

{

UUID key=UUID.randomUUID();

ak\_t2.setText(key.toString());

try{

Connection conn=DBOperations.getConn();

PreparedStatement st=conn.prepareStatement("insert into productinfo values(0,?,?,0)");

st.setString(1, orgname\_t1.getText());

st.setString(2, key.toString());

st.executeUpdate();

conn.close();

}

catch(SQLException ee)

{System.out.println("Cannot insert values into DB"+ee);}

JOptionPane.showMessageDialog(addorg\_frame,"Successfully Added.","Alert",JOptionPane.WARNING\_MESSAGE);

}}

});

gen.setBounds(70,300,150,30);

c.add(gen);

//back goes to the Admin Main frame

MyButton back = new MyButton("\u2190"+" BACK");

back.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

addorg\_frame.dispose();

new AdminMain();

}

});

back.setBounds(270,300,150,30);

c.add(back);

addorg\_frame.setVisible(true);

}

public static void main(String[] args) {

new AdminLogin();

}

}

**15.DB Operations Program:**

package Bootathon.database;

import java.sql.Connection;

import java.sql.DriverManager;

public class DBOperations {

private static Connection conn;

//connection object returned

public static Connection getConn() {

try{

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

conn = DriverManager.getConnection("jdbc:mysql://localhost/rewinddb", "testuser","tester");

}

catch(Exception ee)

{

System.out.println("Connection Error!"+ee);

}

return conn;

}

}

**16.MYFRAME Program:**

package Bootathon.uiworks;

import javax.swing.ImageIcon;

import javax.swing.JFrame;

import javax.swing.JLabel;

public class MyFrame extends JFrame{

public MyFrame(int a)

{

if(a==1)

{

setContentPane(new JLabel(new ImageIcon("C:/Users/rahul/Documents/NetBeansProjects/Bootcamp-Java/src/Bootathon/img/b1.jpg")));

}

else if(a==2)

{

setContentPane(new JLabel(new ImageIcon("C:/Users/rahul/Documents/NetBeansProjects/Bootcamp-Java/src/Bootathon/img/b2.jpg")));

}

else if(a==3)

{

setContentPane(new JLabel(new ImageIcon("C:/Users/rahul/Documents/NetBeansProjects/Bootcamp-Java/src/Bootathon/img/b3.jpg")));

}

else if(a==4)

{

setContentPane(new JLabel(new ImageIcon("C:/Users/rahul/Documents/NetBeansProjects/Bootcamp-Java/src/Bootathon/img/b4.jpg")));

}

else if(a==5)

{

setContentPane(new JLabel(new ImageIcon("C:/Users/rahul/Documents/NetBeansProjects/Bootcamp-Java/src/Bootathon/img/b5.jpg")));

}

setIconImage(new ImageIcon("C:/Users/rahul/Documents/NetBeansProjects/Bootcamp-Java/src/Bootathon/img/icon.jpg").getImage());

setTitle(" Tech i5");

setResizable(false);

setDefaultCloseOperation(3);

}

}

**17.MYLABEL Program:**

package Bootathon.uiworks;

import java.awt.Font;

import javax.swing.JLabel;

public class MyLabel extends JLabel {

public MyLabel(String text)

{

setFont(new Font("cambria",Font.BOLD,20));

setText(text);

}

public MyLabel(String text,int a)

{

setFont(new Font("cambria",Font.BOLD,30));

setText(text);

}

}

**18.MYBUTTON Program:**

package Bootathon.uiworks;

import java.awt.Color;

import java.awt.Font;

import java.awt.event.MouseEvent;

import java.awt.event.MouseListener;

import javax.swing.JButton;

public class MyButton extends JButton{

public MyButton(String text)

{

setBackground(new Color(33, 140, 0));

setText(text);

setBorderPainted(false);

setForeground(new Color(255,255,255));

setFont(new Font("arial",Font.PLAIN,20));

addMouseListener(new MouseListener() {

@Override

public void mouseClicked(MouseEvent e) {

}

@Override

public void mousePressed(MouseEvent e) {

}

@Override

public void mouseReleased(MouseEvent e) {

}

@Override

public void mouseEntered(MouseEvent e) {

setBackground(new Color(2, 179, 2));

}

@Override

public void mouseExited(MouseEvent e) {

setBackground(new Color(33, 140, 0));

}

});

}

}

**19.MYCOMBOBOX Program:**

package Bootathon.uiworks;

import java.awt.Font;

import javax.swing.JComboBox;

public class MyComboBox extends JComboBox{

public MyComboBox() {

setFont(new Font("arial",Font.PLAIN,18));

}

}

**20.MYPANEL Program:**

package Bootathon.uiworks;

import javax.swing.JPanel;

public class MyPanel extends JPanel{

public MyPanel() {

setOpaque(false);

}

}

**21.MYTABLE Program:**

package Bootathon.uiworks;

import java.awt.Font;

import javax.swing.JTable;

public class MyTable extends JTable{

public MyTable() {

setFont(new Font("arial",Font.PLAIN,18));

}

}

**22.MYTEXTAREA Program:**

package Bootathon.uiworks;

import java.awt.Font;

import javax.swing.JTextArea;

public class MyTextArea extends JTextArea{

public MyTextArea()

{

setFont(new Font("arial",Font.PLAIN,18));

}

}

**23.MYTEXTFIELD Program:**

package Bootathon.uiworks;

import java.awt.Font;

import javax.swing.JTextField;

public class MyTextField extends JTextField{

public MyTextField()

{

setFont(new Font("arial",Font.PLAIN,18));

}

}

**Quality Assurance – Test Cases:**

**User Test Case**

Testcase 1:

Add Employ

|  |  |  |  |
| --- | --- | --- | --- |
| MODULE | TEST CASE | STATUS | REMARKS |
| Add Employ | Employee Name =  Sai karthik | Accepts and then asks  for address | passed |
| Add Employ | Address = | Ask to fill , As it cannot  be empty | failed |
| Add Employ | D.O.B=33/12/2000 | Alerts to fill correctly as  a month can have a maximum of 31 days. | failed |
| Add Employ | Phone no = 9012390123 | Accepts and then ask for salary | passed |
| Add Employ | Salary = 40000 | Accepts and then ask for email id | passed |
| Add Employ | Email id = summa.com | Says “ Invalid mail id “ | Failed |
| Add Employ | Password = Summa123@ | Accepts and then waits  for Retype password | passed |
| Add Employ | Re-typed Password = Summa123@ | Accepts and after enter button is pressed goes to the next frame | passed |

Testcase 2:

|  |  |  |  |
| --- | --- | --- | --- |
| MODULE | TEST CASE | STATUS | REMARKS |
| Add Employ | Employee Name = | Ask to fill , As it cannot  be empty | failed |
| Add Employ | Address = xyz,abc. | Accepts and asks for date of birth | passed |
| Add Employ | D.O.B=20/12/2000 | Accepts and then ask for phone no | passed |
| Add Employ | Phone no = 9489498498 | Accepts and then ask for salary | passed |
| Add Employ | Salary = | Ask to fill , As it cannot  be empty | failed |
| Add Employ | Email id = summa@gmail.com | Accepts and ask for password | password |
| Add Employ | Password = as#ma143@ | Accepts and then waits  for Retype password | passed |
| Add Employ | Re-typed Password = as#ma143@ | Accepts and after enter button is pressed goes to the next frame | passed |

Testcase 3:

Add Rewind

|  |  |  |  |
| --- | --- | --- | --- |
| MODULE | TEST CASE | STATUS | REMARKS |
| Add Rewind | Enter hp =2.5 | Accepts and asks for  details | Passed |
| Add Rewind | Enter the details= | Ask to fill , As it cannot  be empty | failed |

Testcase 4:

|  |  |  |  |
| --- | --- | --- | --- |
| MODULE | TEST CASE | STATUS | REMARKS |
| Add Rewind | Enter hp =5.5hp | Accepts and asks for  details | Passed |
| Add Rewind | Enter the details=no of turn = 12 | Accepts and after submit button is clicked goes to next page | passed |

Testcase 5:

Add work

|  |  |  |  |
| --- | --- | --- | --- |
| MODULE | TEST CASE | STATUS | REMARKS |
| Add work | Name of work = winding | Accepts and asks for amount | passed |
| Add work | Amount = | Ask to fill ,As it cannot be empty | failed |
| Add work | Date = 22/45/2000 | Alerts to fill correctly as there are only 12 months | failed |

Testcase 6:

|  |  |  |  |
| --- | --- | --- | --- |
| MODULE | TEST CASE | STATUS | REMARKS |
| Add work | Name of work = | Ask to fill ,As it cannot be empty | failed |
| Add work | Amount = 2000 | Accepts and ask for date | passed |
| Add work | Date = 22/05/2000 | Accepts and after submit button is clicked goes to next frame | passed |

Testcase 7:

Delete employee

|  |  |  |  |
| --- | --- | --- | --- |
| MODULE | TEST CASE | STATUS | REMARKS |
| Delete employee | Select employee = raj | Accepts and goes to next page | passed |

Testcase 8:

|  |  |  |  |
| --- | --- | --- | --- |
| MODULE | TEST CASE | STATUS | REMARKS |
| Delete employee | Select employee = | Ask to fill ,As it cannot be empty | failed |

Testcase 9:

Delete Rewind

|  |  |  |  |
| --- | --- | --- | --- |
| MODULE | TEST CASE | STATUS | REMARKS |
| Delete Rewind | Select hp = 2.5hp | Accepts and goes to next page | passed |

Testcase 10:

|  |  |  |  |
| --- | --- | --- | --- |
| MODULE | TEST CASE | STATUS | REMARKS |
| Delete Rewind | Select hp= | Ask to fill ,As it cannot be empty | failed |

Testcase 11:

Employee password change

|  |  |  |  |
| --- | --- | --- | --- |
| MODULE | TEST CASE | STATUS | REMARKS |
| EmppassChange | Password = as#ma143@ | Accepts and then waits  for Retype password | passed |
| EmppassChange | Re-typed Password = as#ma143@ | Accepts and after enter button is pressed goes to the next frame | passed |

Testcase 12:

|  |  |  |  |
| --- | --- | --- | --- |
| MODULE | TEST CASE | STATUS | REMARKS |
| EmppassChange | Password = as#ma143@ | Accepts and then waits  for Retype password | passed |
| EmppassChange | Re-typed Password = as#12ma143@ | As password and retype password does not match alerts to fill it properly | failed |

Testcase 13:

Employer Registration

|  |  |  |  |
| --- | --- | --- | --- |
| MODULE | TEST CASE | STATUS | REMARKS |
| Employer Registration | Employer Name = | Ask to fill , As it cannot  be empty | failed |
| Add Employ | Activation key = 21312123-123-12-312-3-12-312-345 | Accepts and asks for date of birth | passed |
| Add Employ | D.O.B=20/12/2000 | Accepts and then ask for phone no | passed |
| Add Employ | Email id = summa@gmail.com | Accepts and ask for password | password |
| Add Employ | Password = as#ma143@ | Accepts and then waits  for Retype password | passed |
| Add Employ | Re-typed Password = as#ma143@ | Accepts and after register button is pressed goes to the next frame | passed |

Testcase 14:

|  |  |  |  |
| --- | --- | --- | --- |
| MODULE | TEST CASE | STATUS | REMARKS |
| Employer Registration | Employer Name =Sai karthik | Ask to fill , As it cannot  be empty | failed |
| Add Employ | Activation key = | Ask to fill , As it cannot  be empty | failed |
| Add Employ | D.O.B=30/12/2012 | Accepts and then ask for phone no | passed |
| Add Employ | Email id = summ2314a@gmail.com | Accepts and ask for password | password |
| Add Employ | Password = fdsaas#ma143@ | Accepts and then waits  for Retype password | passed |
| Add Employ | Re-typed Password = aasds#zscma143@ | As password and retype password does not match alerts to fill it properly | failed |

Testcase 15:

Leave Entry

|  |  |  |  |
| --- | --- | --- | --- |
| MODULE | TEST CASE | STATUS | REMARKS |
| Leave Entry | Select Employee = | Ask to select an employee id | Failed |
| Leave Entry | No the days = 5 | Ask are you Sure Accepts and after  Enter button is clicked goes to next page | passed |

Testcase 16:

|  |  |  |  |
| --- | --- | --- | --- |
| MODULE | TEST CASE | STATUS | REMARKS |
| Leave Entry | Select Employee = Sai Karthik | Accepts and asks for  No.of days | Passed |
| Leave Entry | No. of Days = | Ask to fill , As it cannot  be empty | Failed |

Testcase 17:

Load Frame

|  |  |  |  |
| --- | --- | --- | --- |
| MODULE | TEST CASE | STATUS | REMARKS |
| Login Page | Email id = prad.com | Prompts invalid Username or Password | Failed |
| Login Page | Password = 12345678asdfghsdfgvb  dfghjdfghjertyu | Prompts invalid Username or Password | failed |
| Login Page | Employee Employer =  Employee | Checks from Employee database | passed |

Testcase 18:

|  |  |  |  |
| --- | --- | --- | --- |
| MODULE | TEST CASE | STATUS | REMARKS |
| Login Page | Email id = prad@gmail.com | Accepts and checks password | Passed |
| Login Page | Password = pradythegreat | Logs into employee or employer profile | passed |
| Login Page | Employee Employer = | Ask to select any one option | failed |

Testcase 19:

Update Rewind

|  |  |  |  |
| --- | --- | --- | --- |
| MODULE | TEST CASE | STATUS | REMARKS |
| Update Rewind | Select Work ID= 23 | Accepts and checks date | Passed |
| Update Rewind | Date = 28/1/2020 | Accepts for Security Deposit value | passed |
| Update Rewind | Security Deposit Value = 120000 | Accepts the value | passed |

Testcase 20 :

|  |  |  |  |
| --- | --- | --- | --- |
| MODULE | TEST CASE | STATUS | REMARKS |
| Update Rewind | Select Work ID= | Accepts and checks date | failed |
| Update Rewind | Date = ab/100/2020 | Ask to type correct date format | failed |
| Update Rewind | Security Deposit Value=  500abcd | Ask to give only numbers | failed |

Testcase 21:

Admin login

|  |  |  |  |
| --- | --- | --- | --- |
| MODULE | TEST CASE | STATUS | REMARKS |
| Admin login | Username = Sai karthik | Accepts and ask for password | passed |
| Admin login | Password = 1233Safewr# | Accepts and goes to next frame | passed |

Testcase 22:

|  |  |  |  |
| --- | --- | --- | --- |
| MODULE | TEST CASE | STATUS | REMARKS |
| Admin login | Username = | Accepts and ask for password | passed |
| Admin login | Password = 1233Safewr# | Ask to fill , As it cannot  be empty | failed |

Testcase 23:

Admin add organization

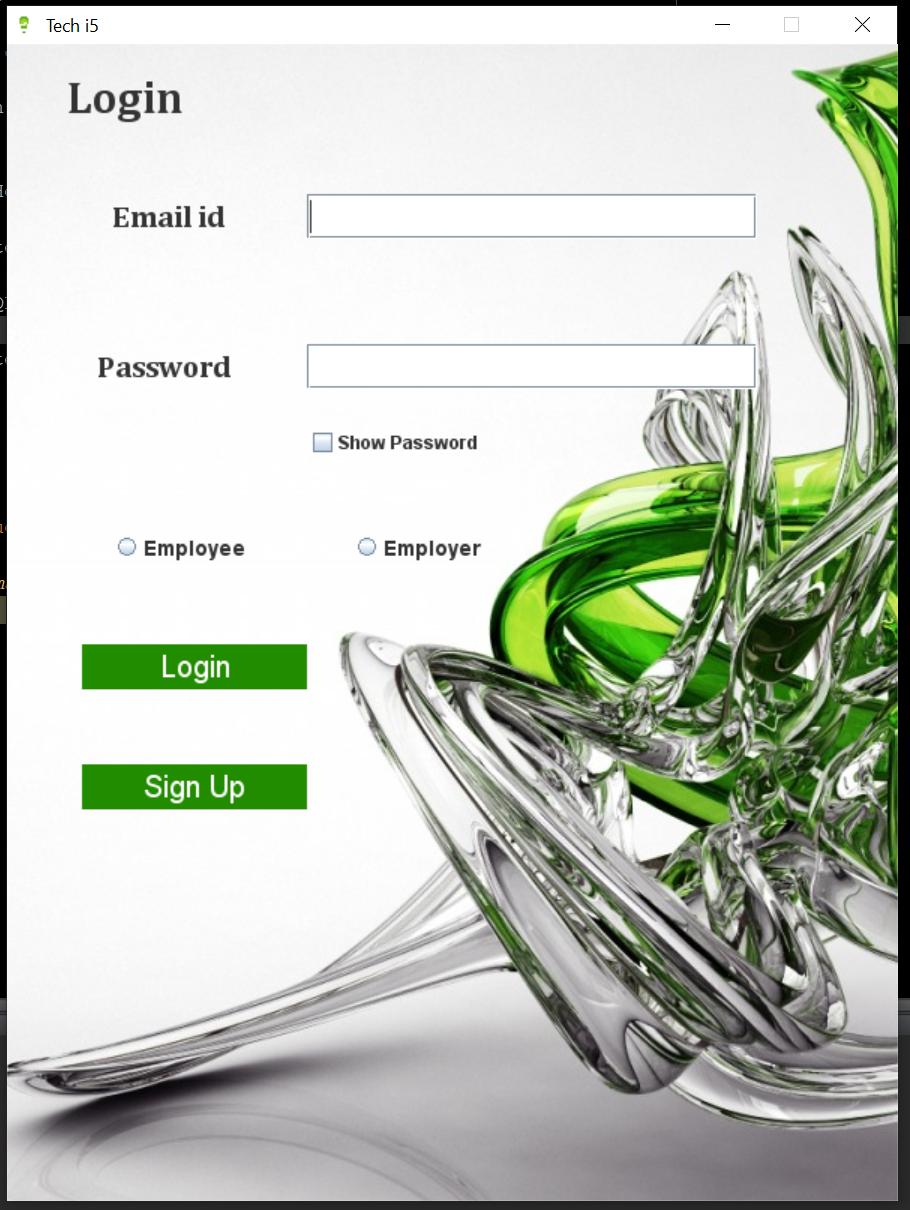
|  |  |  |  |
| --- | --- | --- | --- |
| MODULE | TEST CASE | STATUS | REMARKS |
| Admin add organization | Organization name =  Sun electricals | Accepts and goes to next frame after the generate button is clicked | passed |

Testcase 24:

|  |  |  |  |
| --- | --- | --- | --- |
| MODULE | TEST CASE | STATUS | REMARKS |
| Admin add organization | Organization name = | Ask to fill , As it cannot  be empty | failed |

**Executable ScreenShots:**

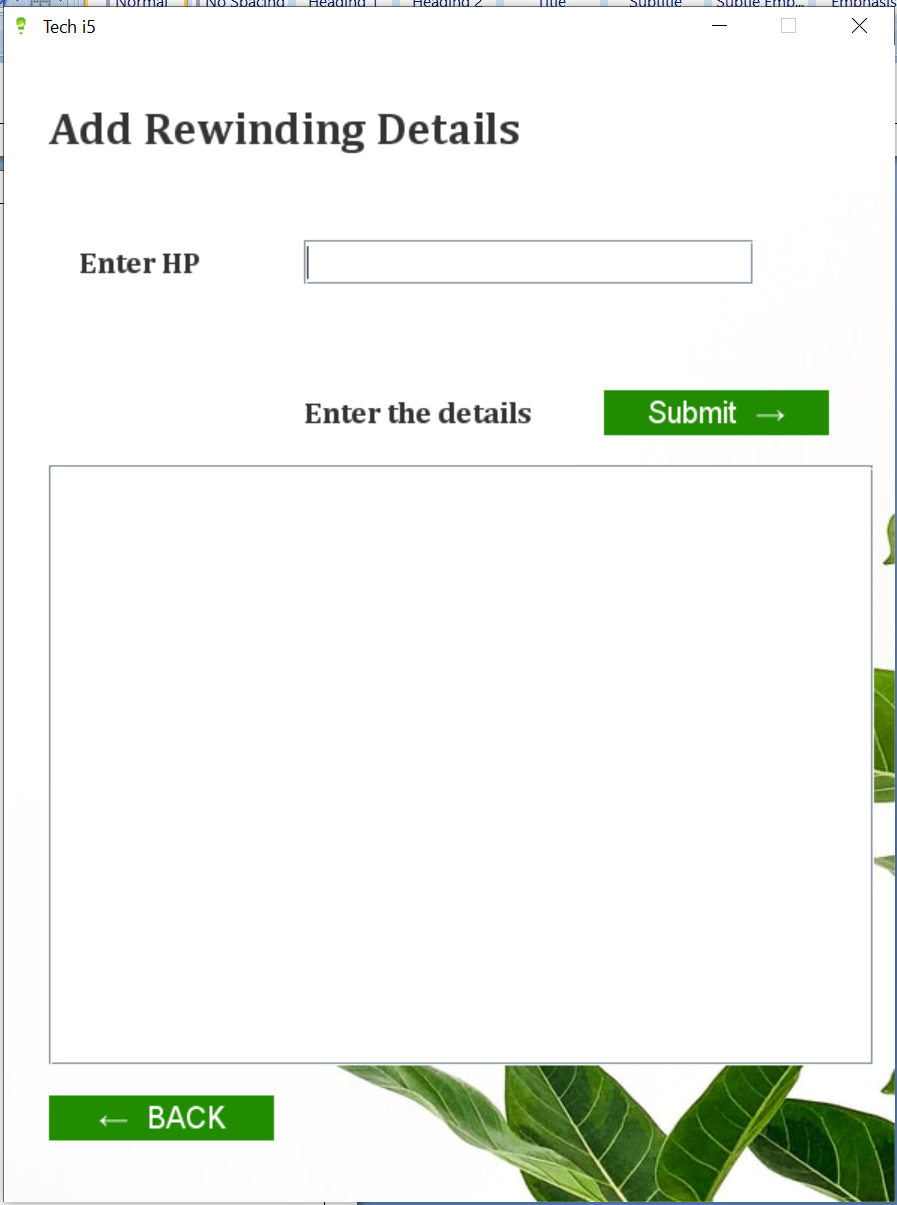
**Login Page**

****

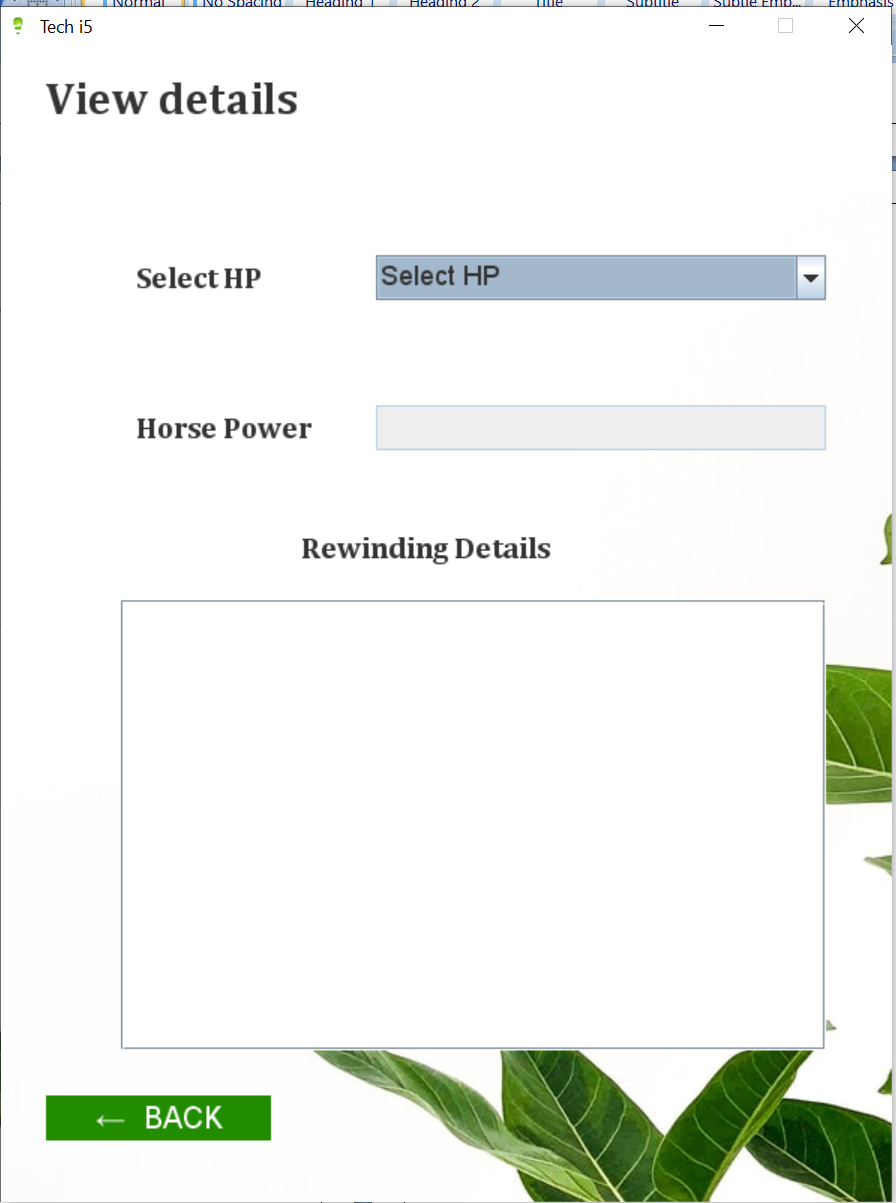
**Employer Main Page:**

****

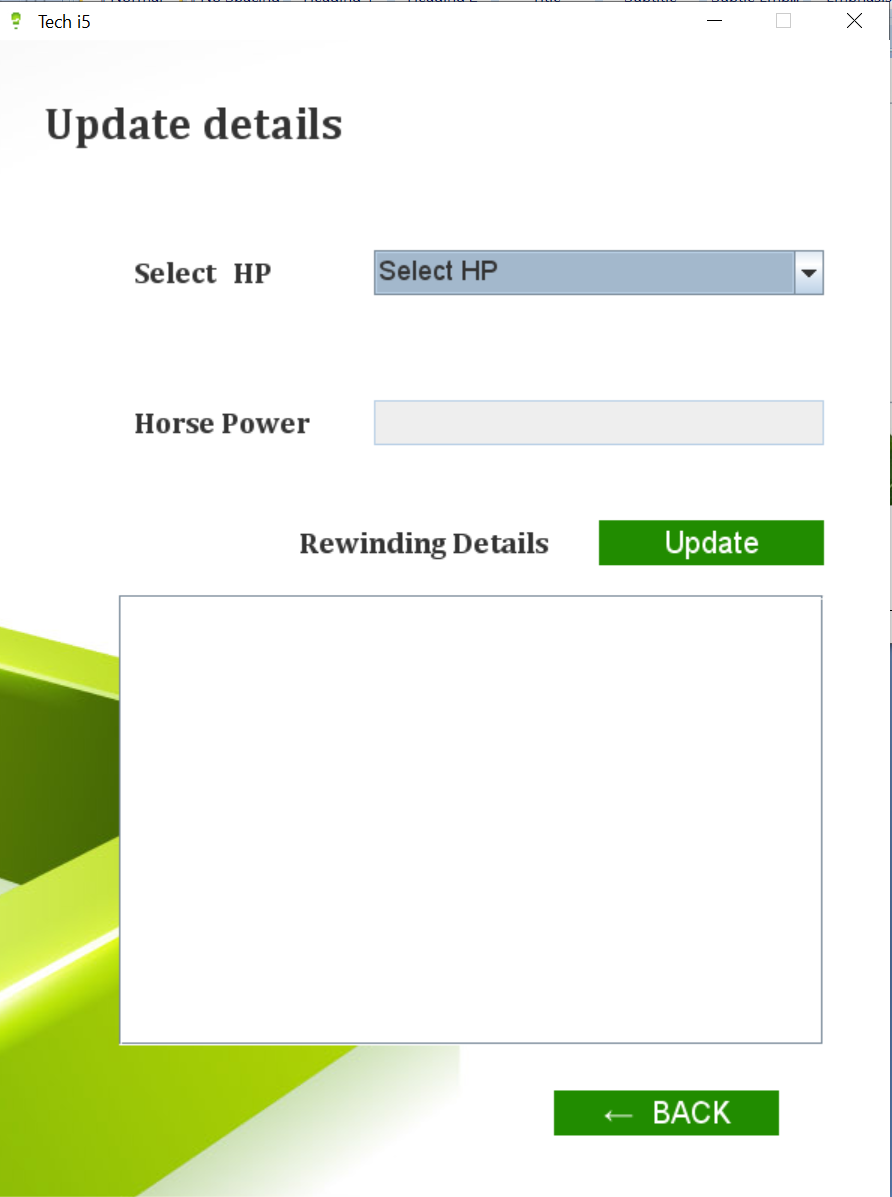
**Add Rewind:**

****

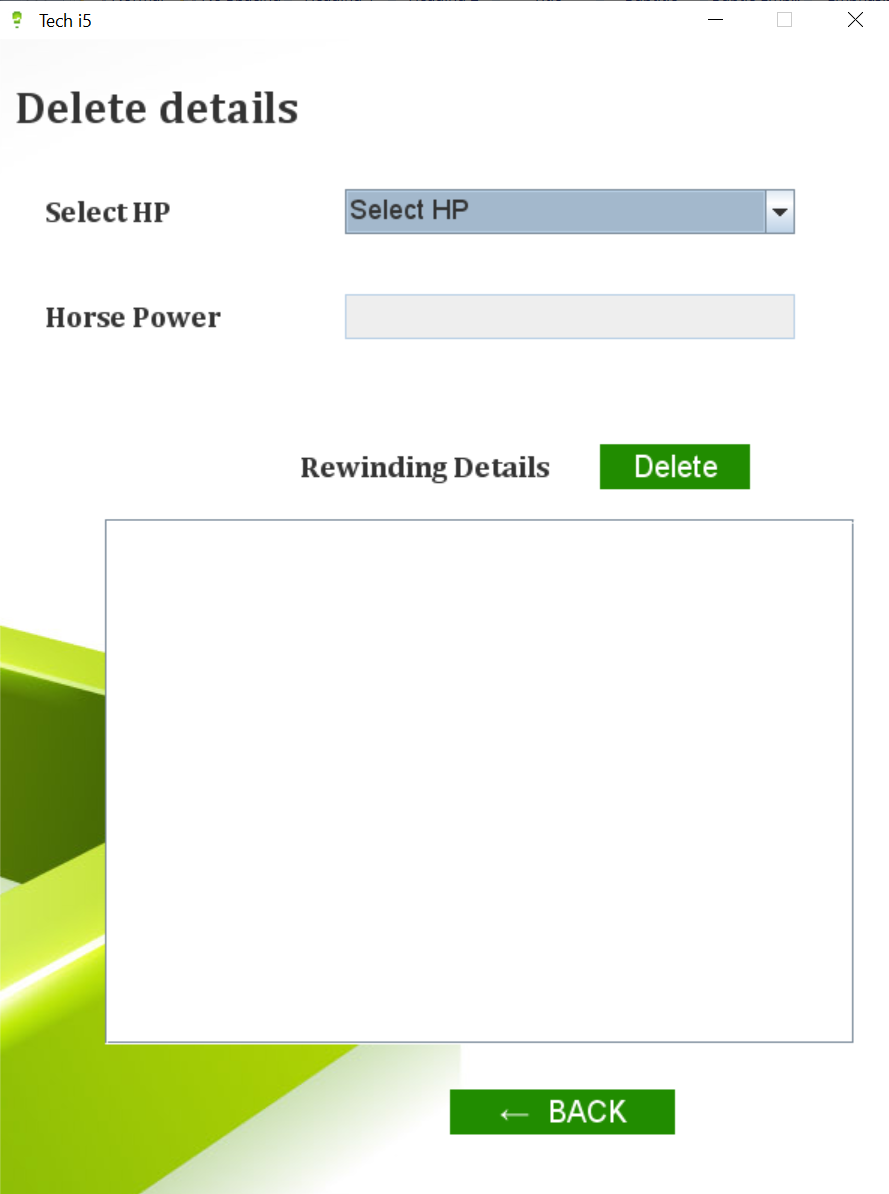
**View Rewind:**

****

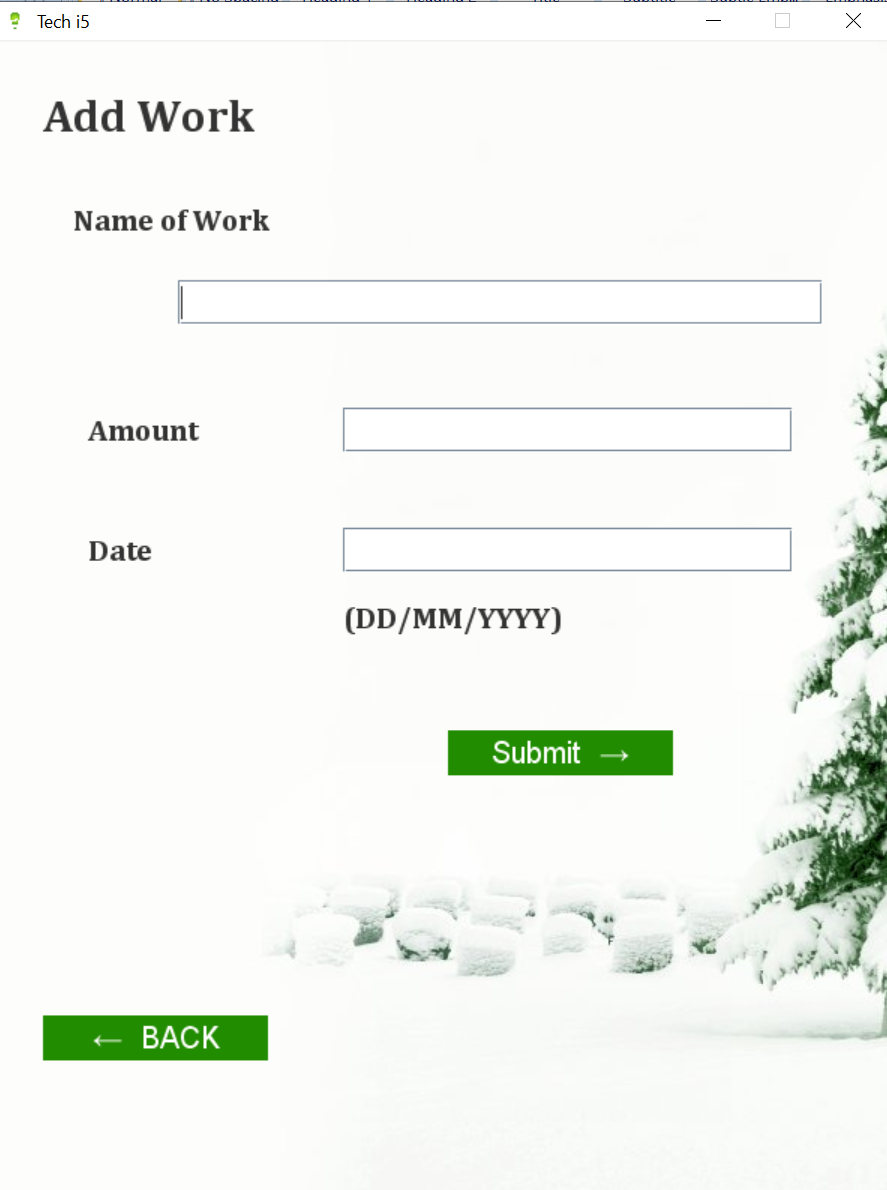
**Update Rewind:**

****

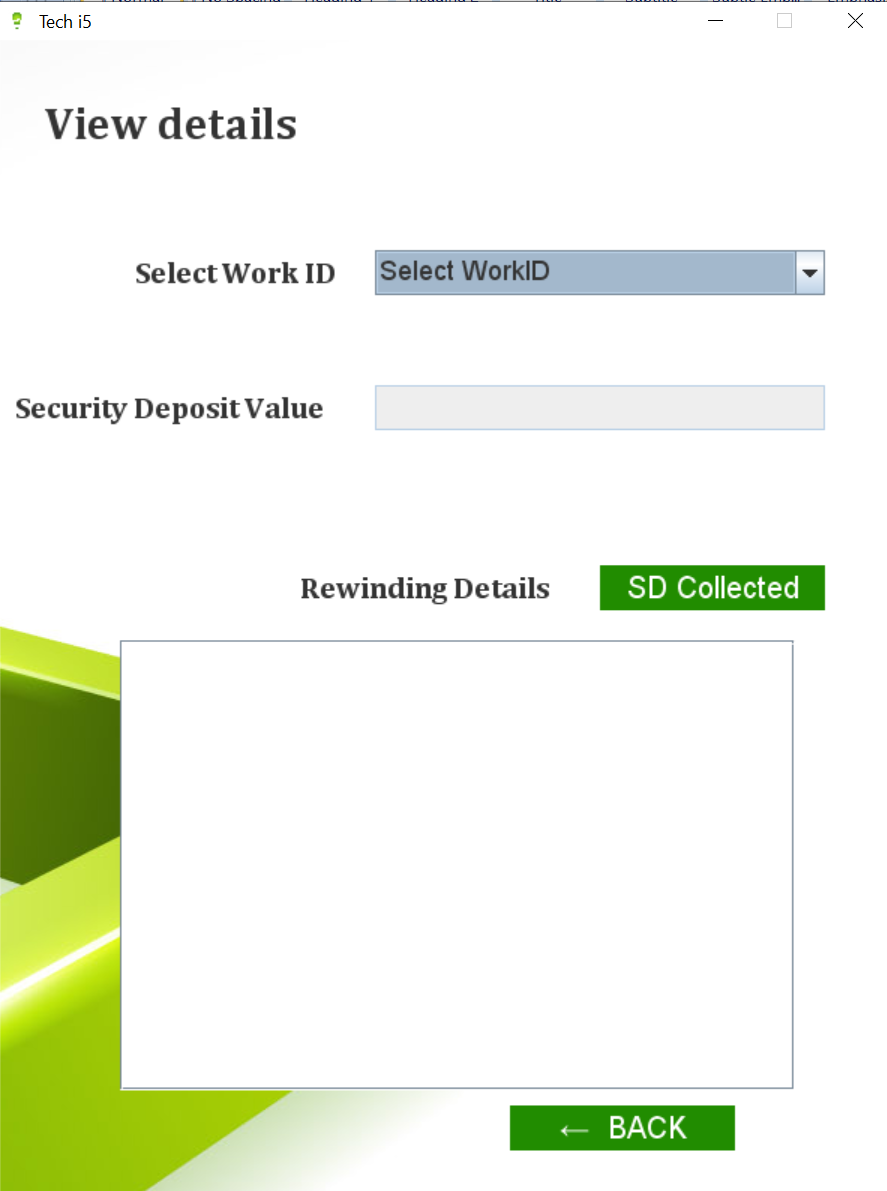
**Delete Rewind:**

****

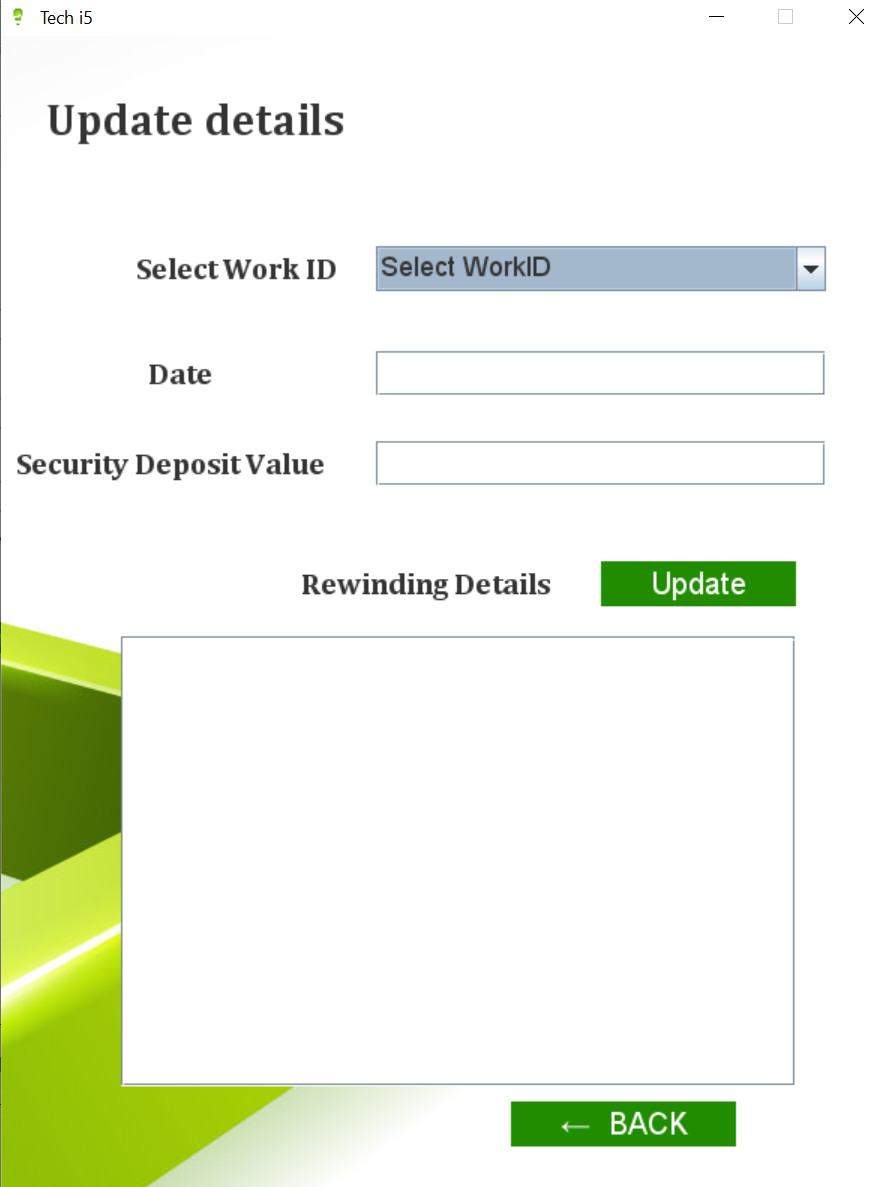
**Add Work:**

****

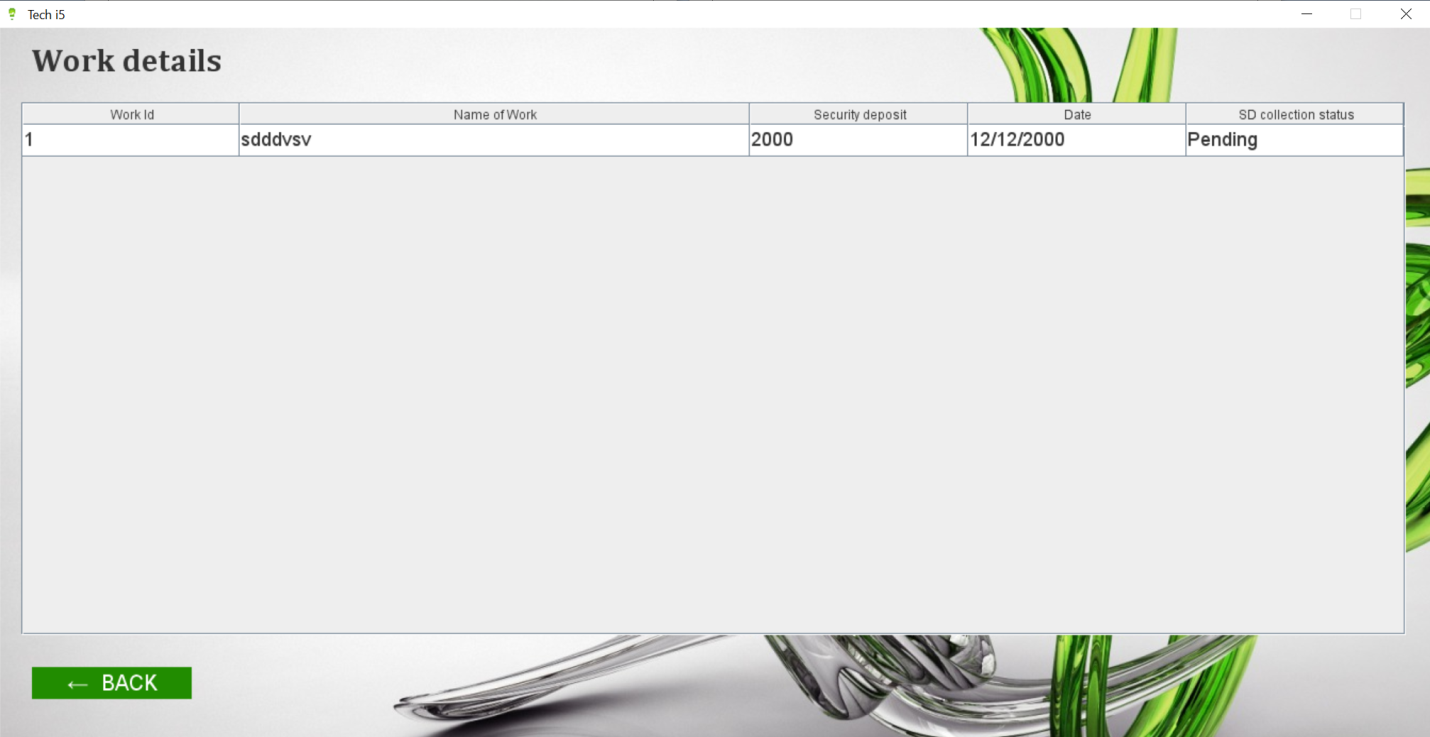
**View Work:**

****

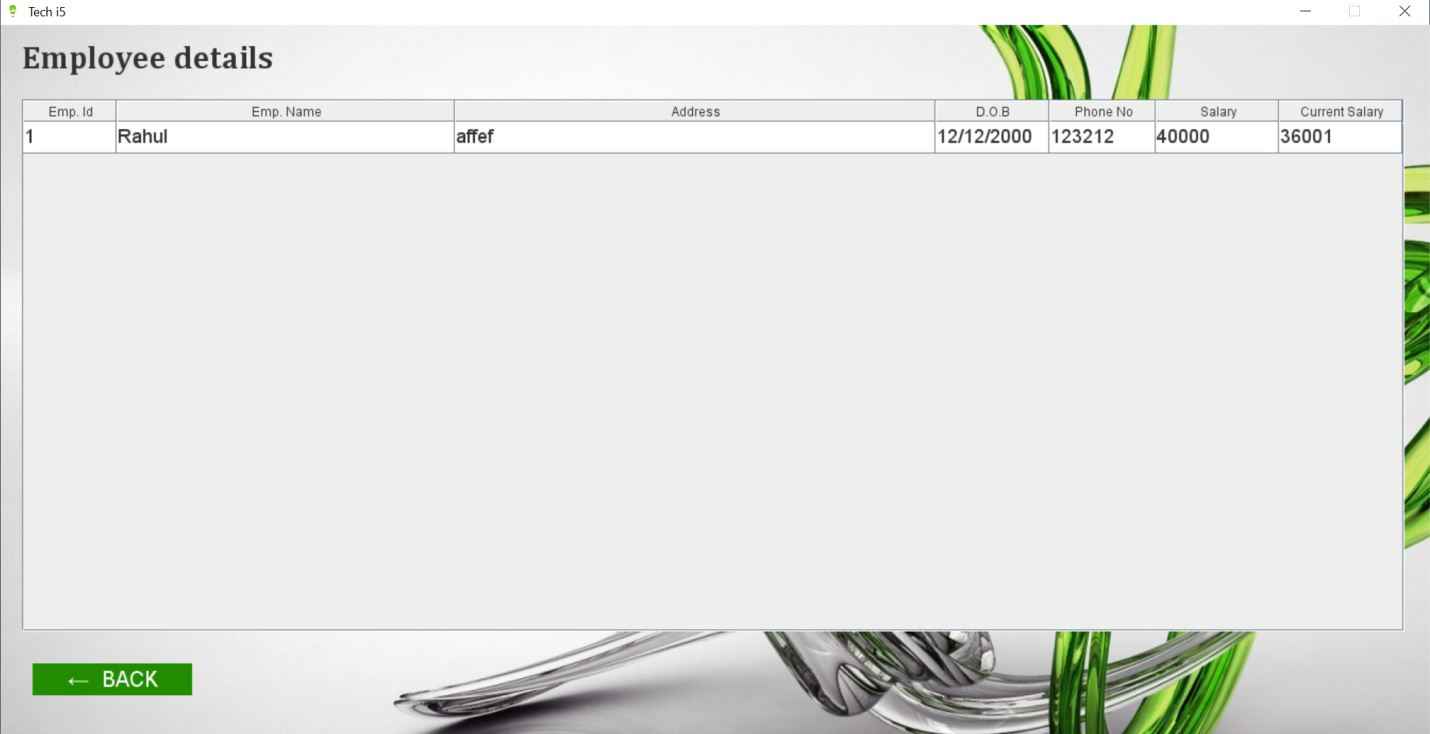
**Update Work:**

****

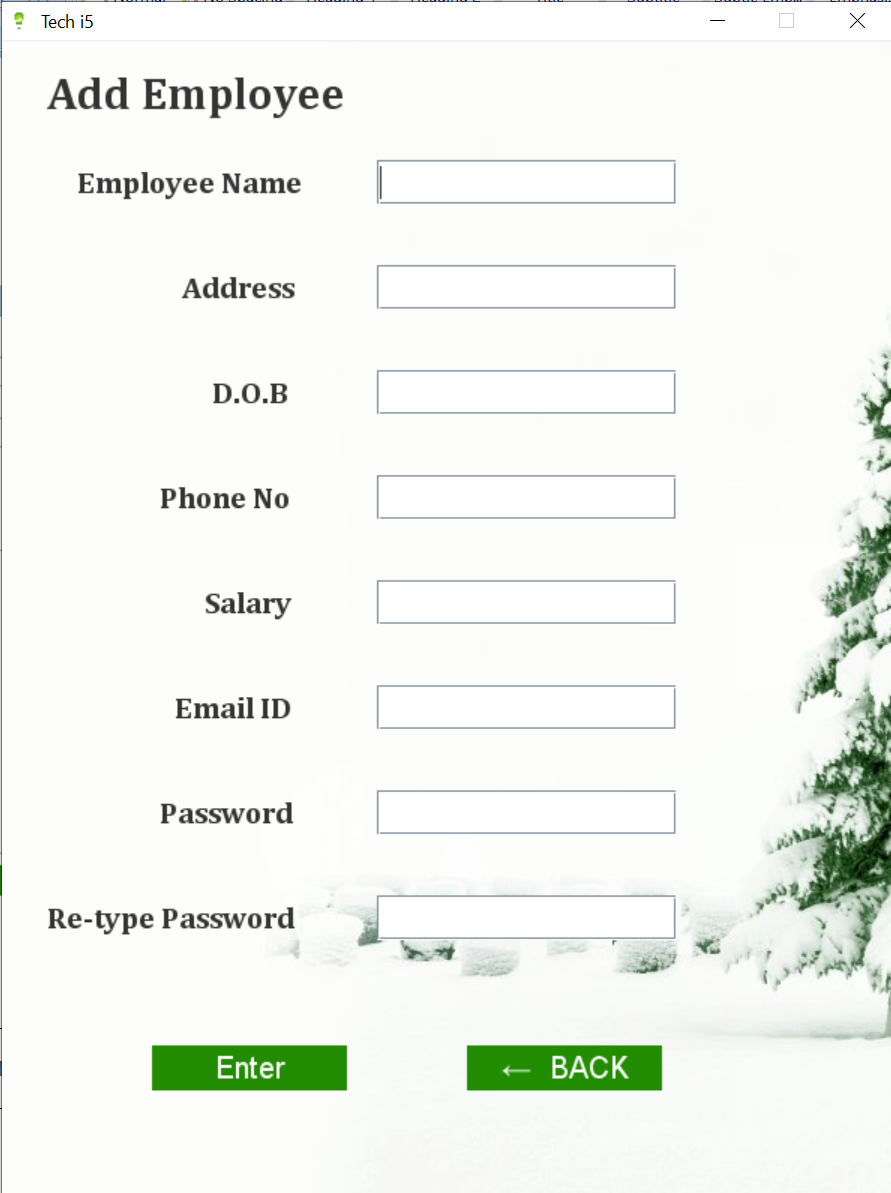
**Work History:**

****

**View Employee List:**

****

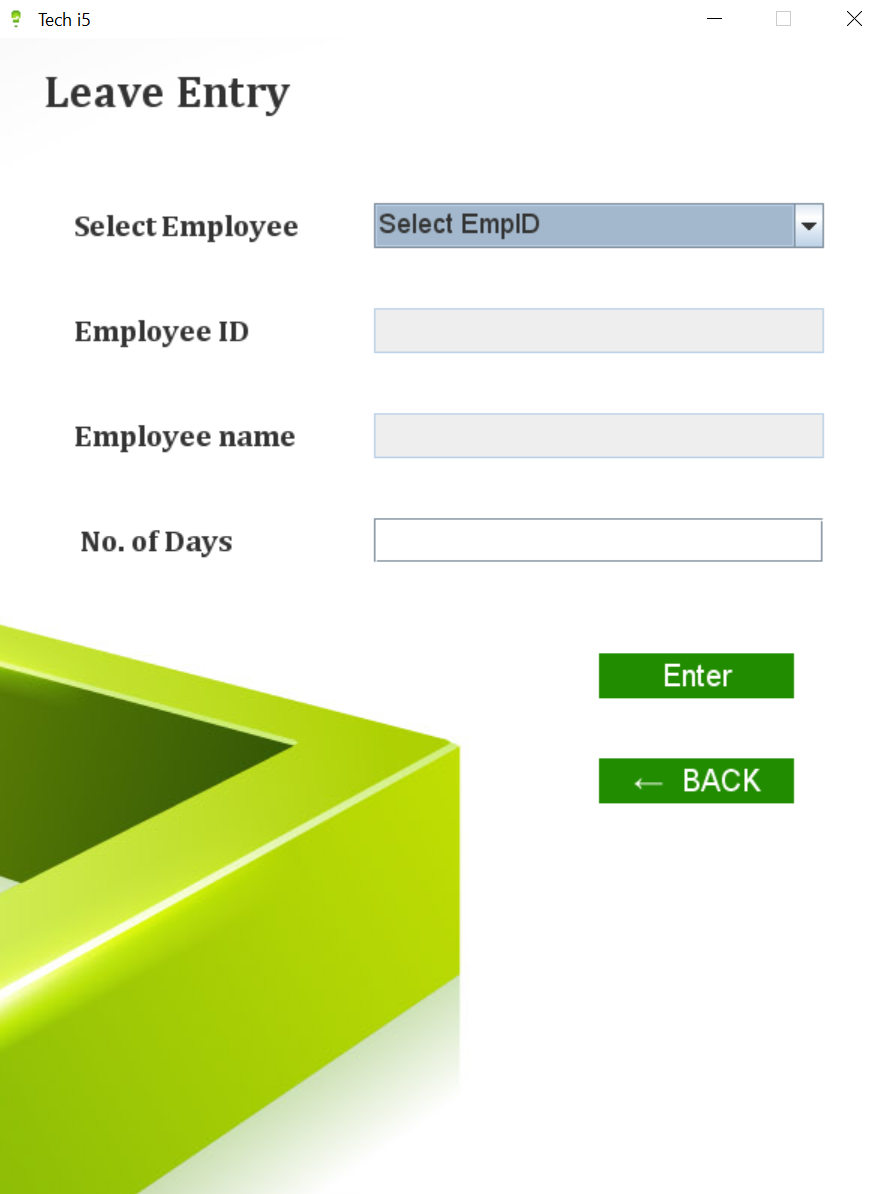
**Add Employee:**

****

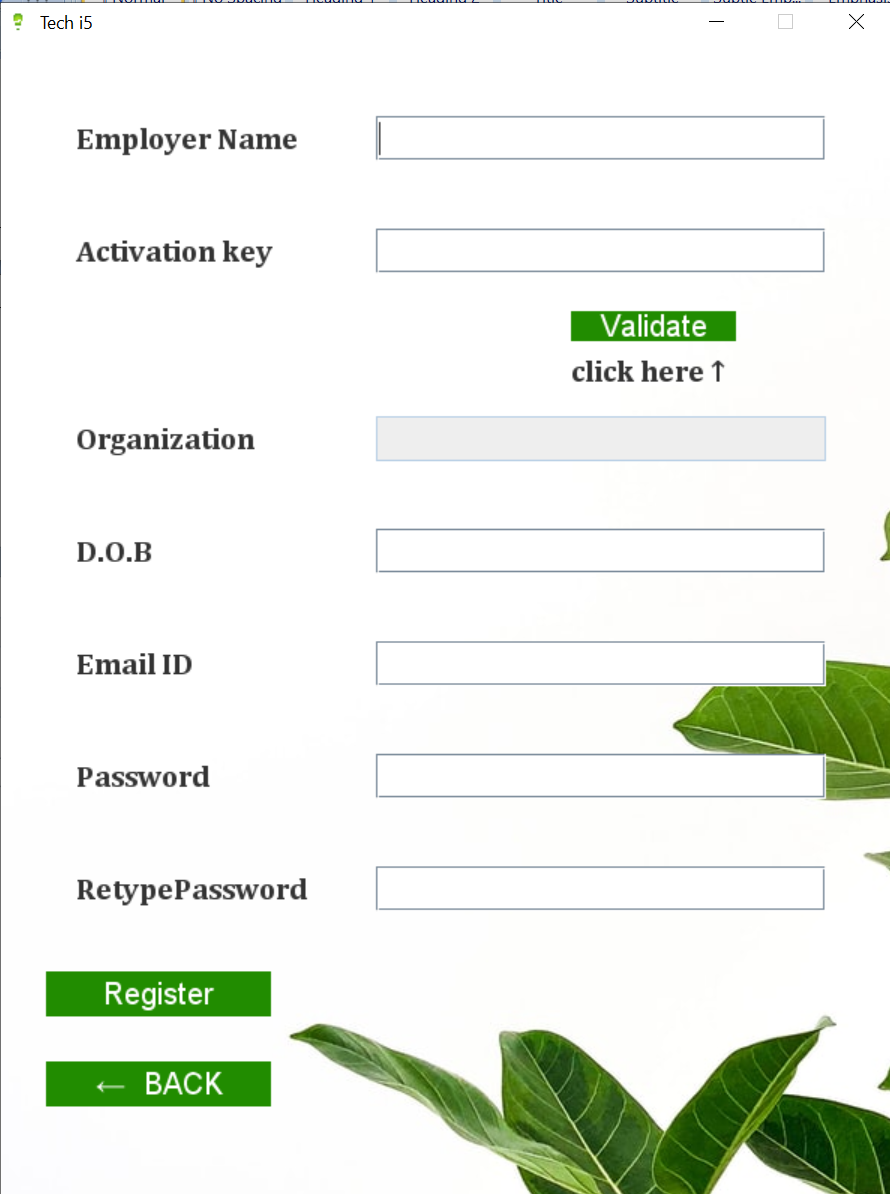
**Delete Employee:**

****

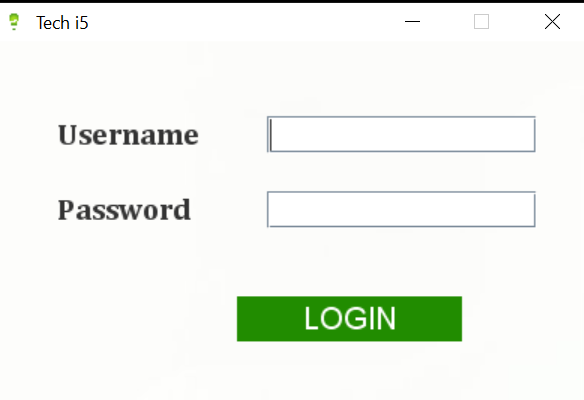
**Leave Entry:**

****

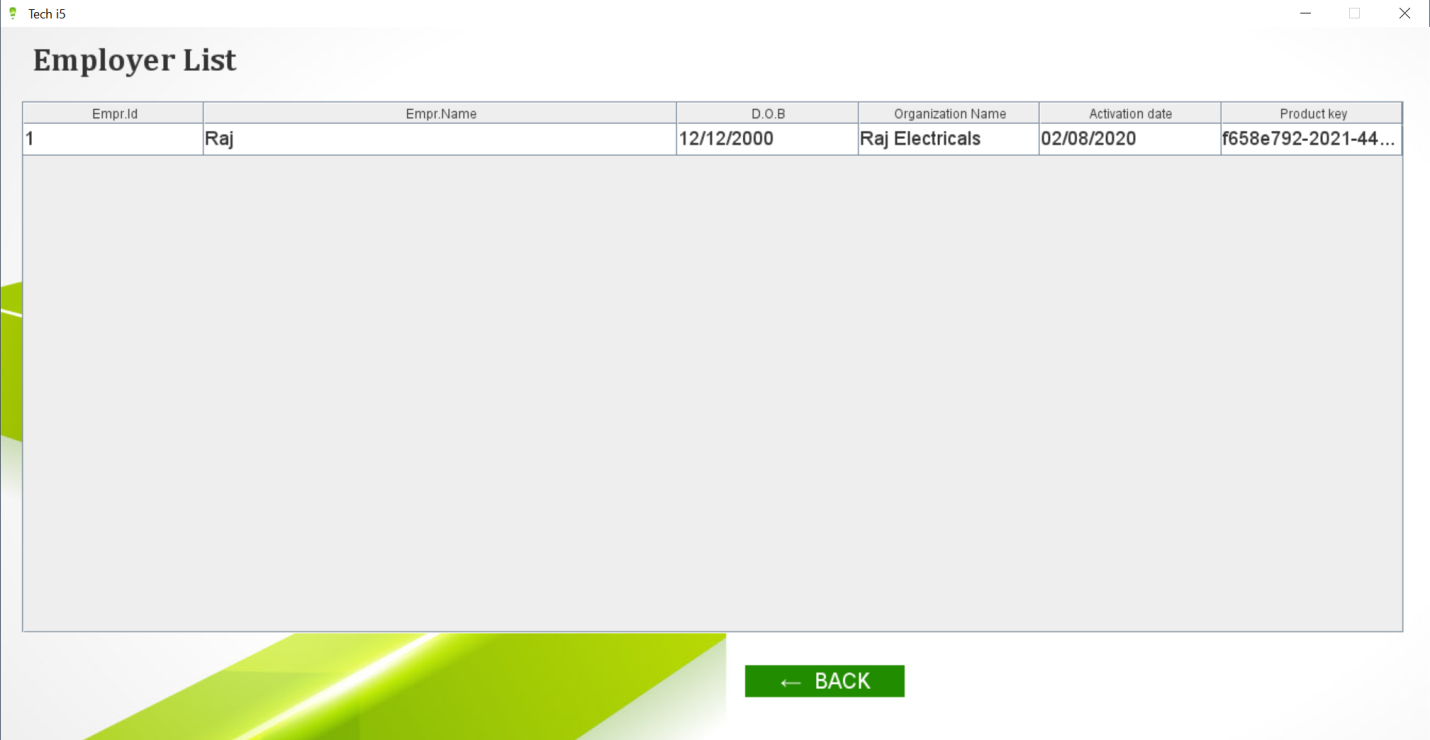
**Employer Registration:**

****

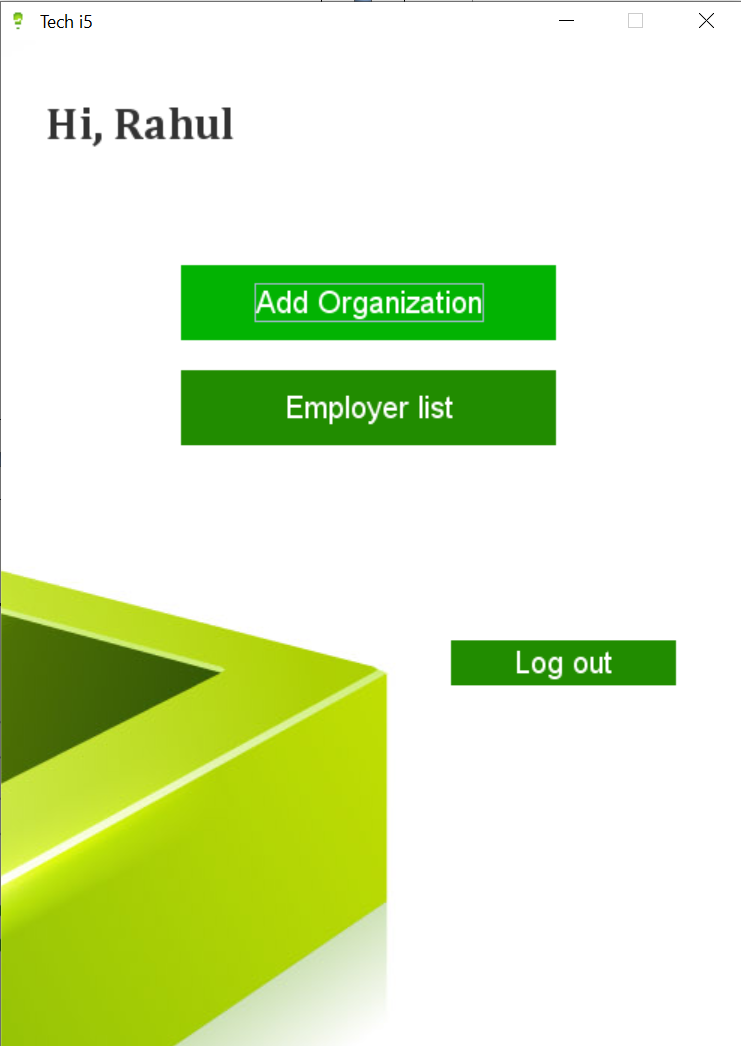
**Admin Login:**

****

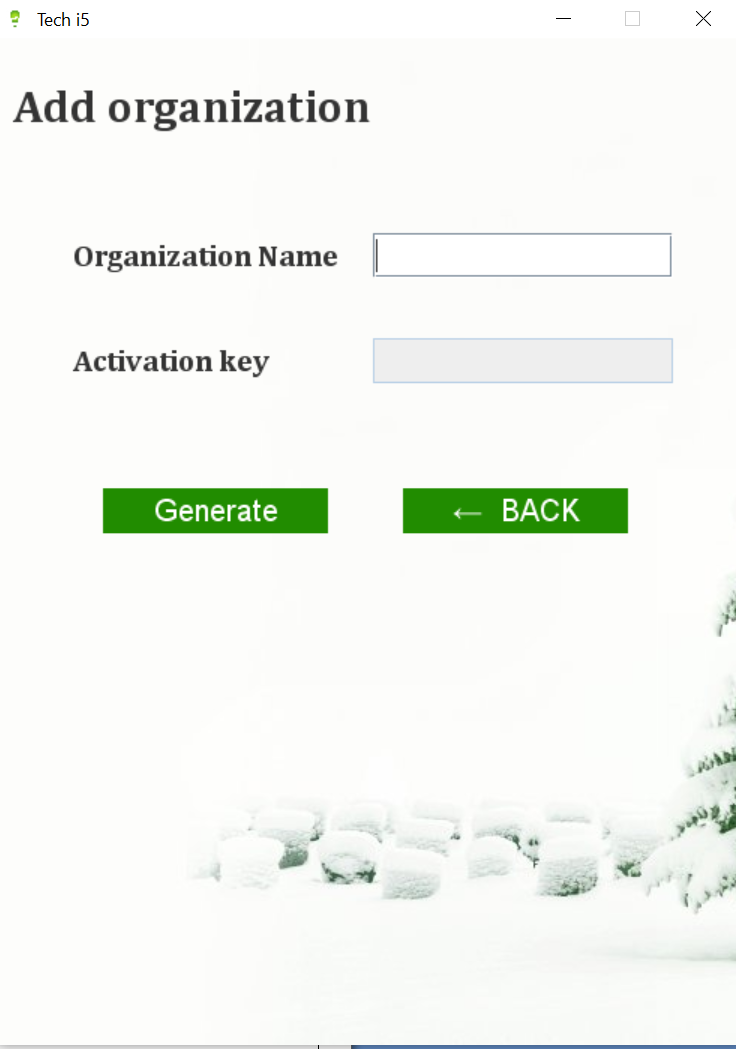
**Employer List:**

****

**Admin Main Page:**

****

**Add Organization:**

****

**Target Audience:**

The target audience for our current piece of project are primarily, small-scale electrical workshop owners, and people into re-winding/motor repairing business. This software will be expanded in future to support different types of small-scale industries so that digitization of data reaches every small industry.

**Future Enhancement:**

* + - Retaining employee data for 3 months from their date of leaving.
    - Instead of hitting the Database everytime, implementing Cache which would help in faster retrieval of data.
    - Expiration of product key after a year, post which the firm owner has to purchase a new product key.