**Project on**

**Java Swing using JDBC**

**and Microsoft SQL**

Developed By:

Chaitanya Saurya

S201171400339

**Project on**

**Java Swing using JDBC**

**and Microsoft SQL**

Batch Code: S201171400339

Start Date of Project: 28/06/20

End Date of Project: 22/07/20

Name of the coordinator: Lopamudra Ma’am

Name of the developer: Chaitanya Saurya

**Certificate**

This is to certify that this report titled Java Swing using JDBC and Microsoft SQL embodies the original work done by Chaitanya Saurya, in partial fulfilment of his course requirement at NIIT.

Coordinator:

**Acknowledgment**

I cannot express enough thanks to my mentor for their continued support and encouragement: Lopamudra Ma’am. I offer my sincere appreciation for the learning opportunities provided by my mentor. Thanks to my parents as well.

**Abstract**

This project gives you a brief understanding of different types of GUI used in Java Swing. How to send and retrieve data from SQL server using SQL driver. This procedure comes under JDBC (Java Database Connectivity). How to execute different SQL Queries.

**Configuration**

Operating System used: Microsoft Windows 10

Software used: 1. IntelliJ IDE

2.Microsoft SQL Server Management

Studio

**Table of Contents:**

1. Chapter 1: Introduction
2. Java Swing
3. Java Database Connectivity (JDBC)
4. Microsoft SQL
5. Chapter 2: Project Analysis
6. Database/Tables in SQL used in the project
7. JDBC Components used in the project
8. Java Swing Components used in the project
9. Chapter 3: Implementation Manual

**Chapter 1**

**Introduction**

**Java Swing**

Java Swing is a part of Java Foundation Classes (JFC) which was designed for enabling large-scale enterprise development of Java applications. Java Swing is a set of APIs that provides graphical user interface (GUI) for Java programs. Java Swing is also known as Java GUI widget toolkit.

Java Swing or Swing was developed based on earlier APIs called Abstract Windows Toolkit (AWT).  Swing provides richer and more sophisticated GUI components than AWT. The GUI components are ranging from a simple label to complex tree and table. Besides emulating look and feel of various platforms, Swing also provides the pluggable look and feel to allow look and feel of Java programs independent from the underlying platform.

Swing Architecture

Swing is platform independent and enhanced MVC (Model –View – Controller) framework for Java application. Here are the most important features in Swing architecture.

* **Pluggable look and feel**:  Swing support several looks and feels and currently supports Windows, UNIX, Motif, and native Java metal look and feel. Swing allows users to switch look and feel at runtime without restarting the application. By doing this, users can make their own choice to choose which look and feel is the best for them instantly.
* **Lightweight components**: All swing components are lightweight except some top-level containers.  Lightweight means component renders or paints itself using drawing primitives of the *Graphics*object instead of relying on the host operating system (OS). As the result, the application presentation is rendered faster and consumed less memory than previous Java GUI applications.
* **Simplified MVC**: Swing uses simplified model-view-architecture (MVC) as the core design behind each of its components called model-delegate. Based on this architecture, each swing component contains a model and a UI delegate. A UI delegate wraps a view and a controller in MVC architecture as the picture below.  UI delegate is responsible for painting screen and handling GUI events. Model is in charge of maintaining information or states of the component.

Swing packages

Here is the short description of each package in Java Swing:

| **Packages** | **Description** |
| --- | --- |
| javax.accessibility | Supports technology which help people with disabilities adapt to the Java applications including audible text reader and screen magnification. |
| javax.swing | Contains core swing components, including most of the model interfaces and helper classes. |
| javax.swing.border | Contains classes for different kinds of borders wrapping around components. |
| javax.swing.colorchooser | Provides colour dialog chooser to allow users to choose a wide range of colours using a visual interface. |
| javax.swing.event | Defines listeners and events which swing components use to communicate asynchronously between objects of classes. |
| javax.swing.filechooser | Contains classes to support file chooser dialog to allow the user to choose files and directories. |
| javax.swing.plaf | PLAF stands for the pluggable look and feel. It contains a wide range of classes and interfaces to support pluggable look and feel. |
| javax.swing.table | Contains classes to support data manipulation in table or spreadsheet format. |
| javax.swing.text | Contains classes and interfaces that are supporting well-known design known as document / view. |
| javax.swing.text.html | Supports reading and formatting HTML text through the editor. |
| javax.swing.text.html.parser | Contains classes and interfaces for HTML parsing. |
| javax.swing.text.rtf | Supports reading and formatting Rich Text Format (RTF) text through the editor. |
| javax.swing.tree | Contains classes and interfaces to define models and views for a hierarchical tree component. |
| javax.swing.undo | Contains classes and interfaces supporting undo-able functionality. |
|  |  |

**Java Database Connectivity (JDBC)**

JDBC stands for **J**ava **D**ata**b**ase **C**onnectivity, which is a standard Java API for database-independent connectivity between the Java programming language and a wide range of databases.

The JDBC library includes APIs for each of the tasks mentioned below that are commonly associated with database usage.

* Making a connection to a database.
* Creating SQL or MySQL statements.
* Executing SQL or MySQL queries in the database.
* Viewing & Modifying the resulting records.

Fundamentally, JDBC is a specification that provides a complete set of interfaces that allows for portable access to an underlying database. Java can be used to write different types of executables, such as −

* Java Applications
* Java Applets
* Java Servlets
* Java Server Pages (JSPs)
* Enterprise JavaBeans (EJBs).

All of these different executables are able to use a JDBC driver to access a database, and take advantage of the stored data.

JDBC provides the same capabilities as ODBC, allowing Java programs to contain database-independent code.

## Pre-Requisite

Before moving further, you need to have a good understanding of the following two subjects −

* [Core JAVA Programming](https://www.tutorialspoint.com/java/index.htm)
* [SQL or MySQL Database](https://www.tutorialspoint.com/mysql/index.htm)

JDBC Architecture

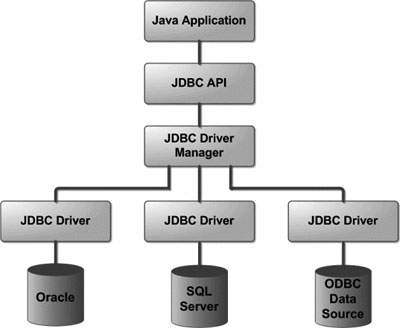
The JDBC API supports both two-tier and three-tier processing models for database access but in general, JDBC Architecture consists of two layers −

* **JDBC API:** This provides the application-to-JDBC Manager connection.
* **JDBC Driver API:** This supports the JDBC Manager-to-Driver Connection.

The JDBC API uses a driver manager and database-specific drivers to provide transparent connectivity to heterogeneous databases.

The JDBC driver manager ensures that the correct driver is used to access each data source. The driver manager is capable of supporting multiple concurrent drivers connected to multiple heterogeneous databases.

Following is the architectural diagram, which shows the location of the driver manager with respect to the JDBC drivers and the Java application −



**Common JDBC Components**

The JDBC API provides the following interfaces and classes −

* **DriverManager:** This class manages a list of database drivers. Matches connection requests from the java application with the proper database driver using communication sub protocol. The first driver that recognizes a certain subprotocol under JDBC will be used to establish a database Connection.
* **Driver:** This interface handles the communications with the database server. You will interact directly with Driver objects very rarely. Instead, you use DriverManager objects, which manages objects of this type. It also abstracts the details associated with working with Driver objects.
* **Connection:** This interface with all methods for contacting a database. The connection object represents communication context, i.e., all communication with database is through connection object only.
* **Statement:** You use objects created from this interface to submit the SQL statements to the database. Some derived interfaces accept parameters in addition to executing stored procedures.
* **ResultSet:** These objects hold data retrieved from a database after you execute an SQL query using Statement objects. It acts as an iterator to allow you to move through its data.
* **SQLException:** This class handles any errors that occur in a database application.

The JDBC 4.0 Packages

The java.sql and javax.sql are the primary packages for JDBC 4.0. This is the latest JDBC version at the time of writing this tutorial. It offers the main classes for interacting with your data sources.

The new features in these packages include changes in the following areas −

* Automatic database driver loading.
* Exception handling improvements.
* Enhanced BLOB/CLOB functionality.
* Connection and statement interface enhancements.
* National character set support.
* SQL ROWID access.
* SQL 2003 XML data type support.
* Annotations.

**Microsoft SQL**

MS SQL Server is a relational database management system (RDBMS) developed by Microsoft. This product is built for the basic function of storing retrieving data as required by other applications. It can be run either on the same computer or on another across a network. This tutorial explains some basic and advanced concepts of SQL Server such as how to create and restore data, create login and backup, assign permissions, etc.

Key Components and Services of SQL Server

**Database Engine:**This component handle storage, Rapid transaction Processing, and Securing Data.

**SQL Server:**This service starts, stops, pauses, and continues an instance of Microsoft SQL Server. Executable name is sqlservr.exe.

**SQL Server Agent:**It performs the role of Task Scheduler. It can be triggered by any event or as per demand. Executable name is sqlagent.exe.

**SQL Server Browser:**This listens to the incoming request and connects to the desired SQL server instance. Executable name is sqlbrowser.exe.

**SQL Server Full-Text Search:**This lets user running full-text queries against Character data in SQL Tables.Executable name is fdlauncher.exe.

**SQL Server VSS Writer:**This allows backup and restoration of data files when the SQL server is not running.Executable name is sqlwriter.exe.

**SQL Server Analysis Services (SSAS):**Provide Data analysis, Data mining and Machine Learning capabilities. SQL server is integrated with R and Python language for advanced analytics. Executable name is msmdsrv.exe.

**SQL Server Reporting Services (SSRS):**Provides reporting features and decision-making capabilities. It includes integration with Hadoop. Executable name is ReportingServicesService.exe

**SQL Server Integration Services (SSIS):**Provided Extract-Transform and Load capabilities of the different type of data from one source to another. It can be view as converting raw information into useful information. Executable name is MsDtsSrvr.exe

SQL Server Instances

SQL Server allows you to run multiple services at a go, with each service having separate logins, ports, databases, etc. These are divided into two:

* Primary instances
* Named instances.

There are two ways through which we may access the primary instance. First, we can use the server name. Secondly, we can use its IP address. Named instances are accessed by appending a backslash and instance name.

For example, to connect to an instance named xyx on the local server, you should use 127.0.0.1\xyz. From SQL Server 2005 and above, you are allowed to run up to 50 instances simultaneously on a server.

Note that even though you can have multiple instances on the same server, only one of them must be the default instance while the rest must be named instances. One can run all the instances concurrently, and each instance runs independent of the other instances.

Importance of SQL Server Instances

The following are the advantages of SQL Server instances:

**1. For installation of different versions on one machine**

You can have different versions of SQL Server on a single machine. Each installation works independently from the other installations.

**2. For cost reduction**

Instances can help us reduce the costs of operating SQL Server, especially in purchasing the SQL Server license. You can get different services from different instances, hence no need for purchasing one license for all services.

**3. For maintenance of development, production and test environments separately**

This is the main benefit of having many SQL Server instances on a single machine. You can use different instances for development, production and test purposes.

**4. For reducing temporary database problems**

When you have all services running on a single SQL Server instance, there are high chances of having problems with the problems, especially problems that keep on recurring. When such services are run on different instances, you can avoid having such problems.

**5. For separating security privileges**

When different services are running on different SQL Server instances, you can focus on securing the instance running the most sensitive service.

**6. For maintaining a standby server**

**Chapter 2**

**Project Analysis**

**Database in SQL**

I created a database in using Microsoft SQL with the name Swing demo.

**Tables Created**

For storing various and different types of data, I created 5 tables. The following tables with their Columns are listed below:

1.**EMP Table:**



first\_name is of varchar datatype.

last\_name is of varchar datatype.

User\_name is of varchar datatype and is unique.

Password is of varchar datatype.

Email\_id is of varchar datatype.

Mobile\_number is of varchar datatype.

Gender is of varchar datatype.

Status is of varchar datatype.

Skills is of varchar datatype.

Jobs is of varchar datatype.

2.**HR Table:**

****

first\_name is of varchar datatype.

last\_name is of varchar datatype.

User\_name is of varchar datatype and is unique.

Password is of varchar datatype.

Email\_id is of varchar datatype.

Mobile\_number is of varchar datatype.

Gender is of varchar datatype.

3.**PM Table:**

****

first\_name is of varchar datatype.

last\_name is of varchar datatype.

User\_name is of varchar datatype and is unique.

Password is of varchar datatype.

Email\_id is of varchar datatype.

Mobile\_number is of varchar datatype.

Gender is of varchar datatype.

**4.Jobs Table:**

****

JobId is of int datatype and is a primary key.

JobTitle is of varchar datatype.

Location is of varchar datatype.

KeySkill is of varchar datatype.

Salary is of is of varchar datatype.

Status is of varchar datatype.

5.**Skills Table:**



User\_name is of varchar datatype.

Skill is of varchar datatype.

StatusOfSkill is of varchar datatype.

**JDBC Components**

The following JDBC components used in the project:

1.**Register the Driver Clas**s- The **forName()** method of Class is used to register the driver class. This method is used to dynamically load the driver class.

2**.getConnection()**- This method of DriverManager class is used to establish connection with the database.

3. **createStatement()-** The createStatement() method of Connection interface is used to create statement. The object of statement is responsible to execute queries with the database.

4.**PreparedStatement**- The PreparedStatement interface is a subinterface of Statement. It is used to execute parameterized query.

5. **executeQuery()-** The executeQuery() method of Statement interface is used to execute queries to the database. This method returns the object of ResultSet that can be used to get all the records of a table.

6. **executeUpdate():** This method is used to execute statements such as insert, update, delete. It returns an integer value representing the number of rows affected.

7. **getMetaData()-** The getMetaData() method of Connection interface returns the object of DatabaseMetaData.

8. **close()-** By closing connection object statement and ResultSet will be closed automatically. The close() method of Connection interface is used to close the connection.

**Java Swing Components**

The following Swing Components used in this project are:

1.**JLabel**- The object of JLabel class is a component for placing text in a container. It is used to display a single line of read only text. The text can be changed by an application but a user cannot edit it directly. It inherits JComponent class.

2.**JButton**- The JButton class is used to create a labeled button that has platform independent implementation. The application result in some action when the button is pushed. It inherits AbstractButton class.

3.**JPanel**- The JPanel is a simplest container class. It provides space in which an application can attach any other component. It inherits the JComponents class.

4.**JFrame**- The javax.swing.JFrame class is a type of container which inherits the java.awt.Frame class. JFrame works like the main window where components like labels, buttons, textfields are added to create a GUI.

5.**JTextField**- The object of a JTextField class is a text component that allows the editing of a single line text. It inherits JTextComponent class.

6.**JPasswordField**- The object of a JPasswordField class is a text component specialized for password entry. It allows the editing of a single line of text. It inherits JTextField class.

7.**JRadioButton**- The JRadioButton class is used to create a radio button. It is used to choose one option from multiple options. It is widely used in exam systems or quiz.

8.**JComboBox**- The object of Choice class is used to show popup menu of choices. Choice selected by user is shown on the top of a menu. It inherits JComponent class.

9.**JTable**- The JTable class is used to display data in tabular form. It is composed of rows and columns.

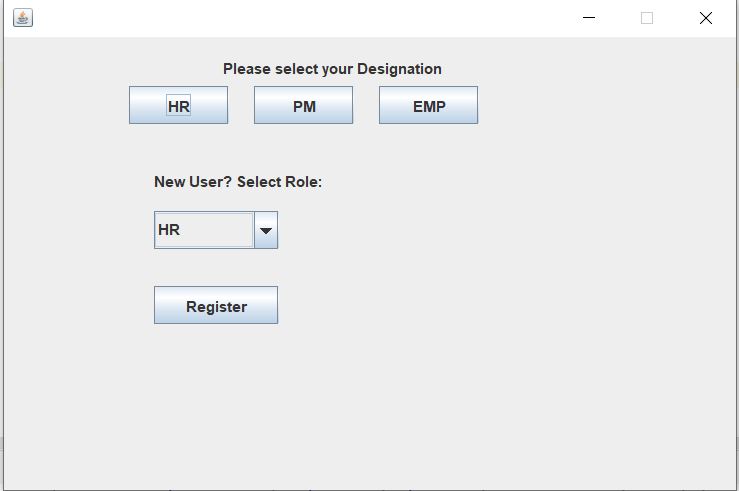
10.**JOptionPane**- The JOptionPane class is used to provide standard dialog boxes such as message dialog box, confirm dialog box and input dialog box. These dialog boxes are used to display information or get input from the user. The JOptionPane class inherits JComponent class.

m

**Chapter 3:**

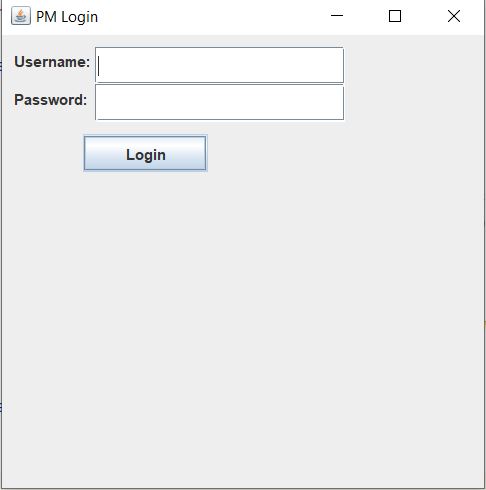
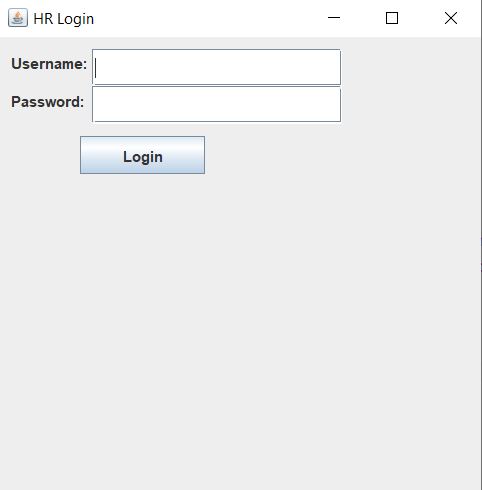
**Implementation Manual**

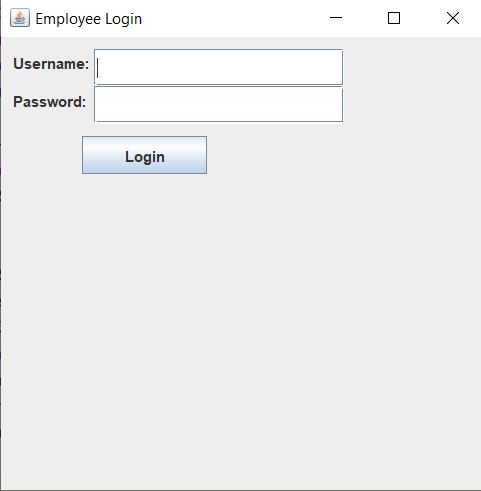
1.**Pre-Login Frame**:



The application starts with a Pre-Login frame in which you to select your designation that you are a Human Resource Manager or a Project Manager or an Employee and then it will take you to the login page where you have to enter your credentials. If you are new to the company then you have to select the role that the company has assigned you and click on register which will take you to the register Page.

2.**Login Frame**-

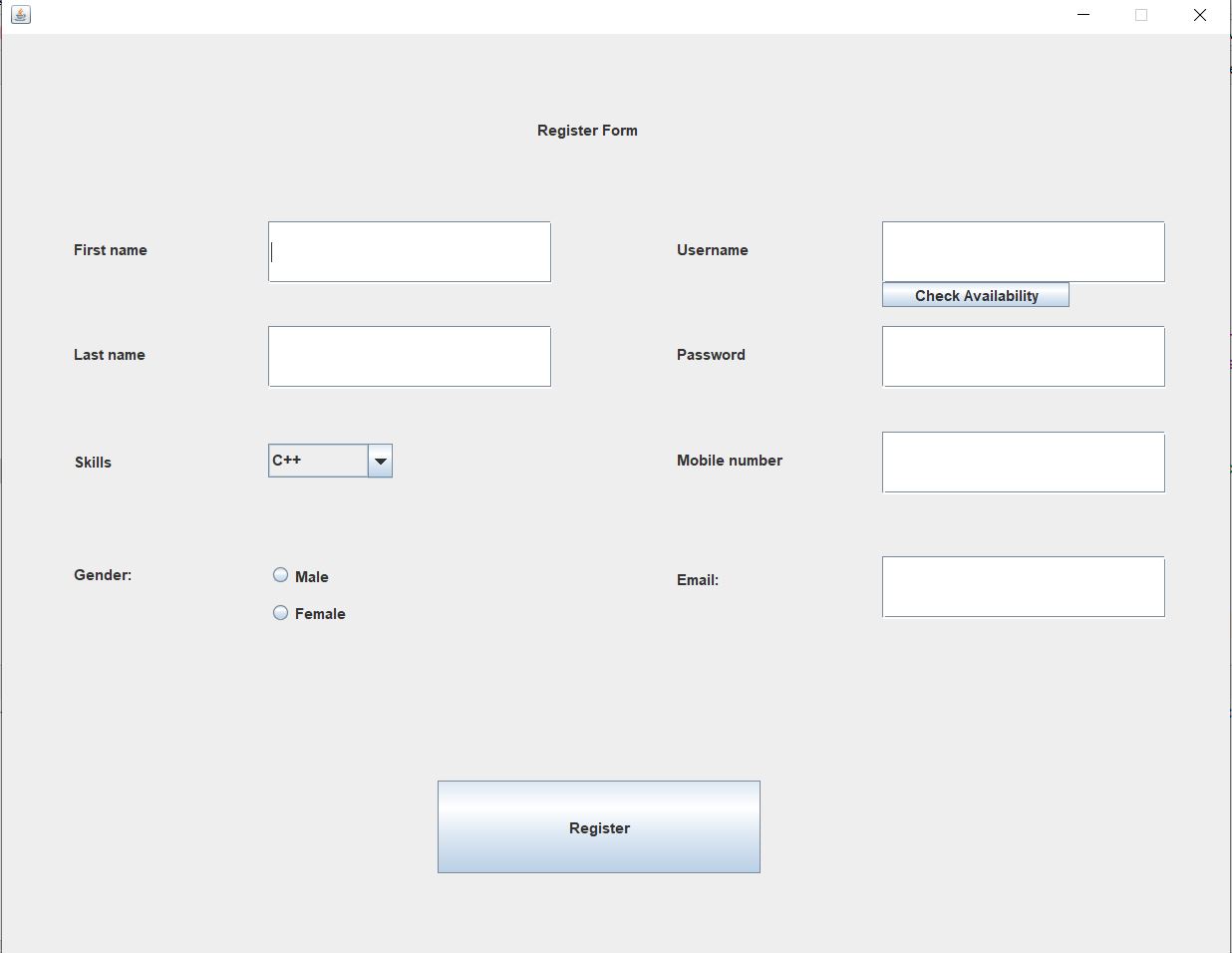




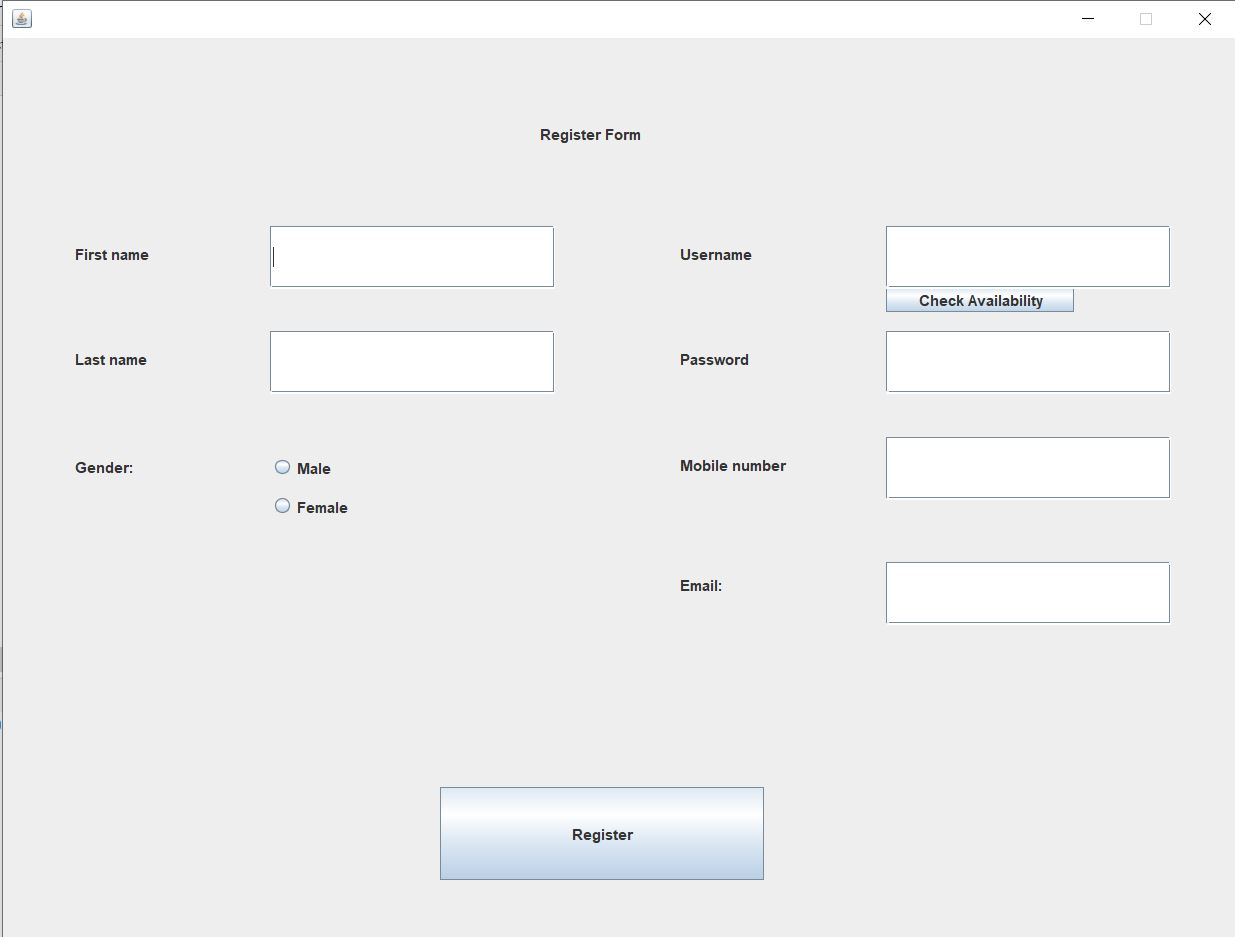
Here is the login frame that will open when you select any designation in pre-login frame respectively. In this login frame you have to enter your username and password and click on login button.

3.**Register Frame**-

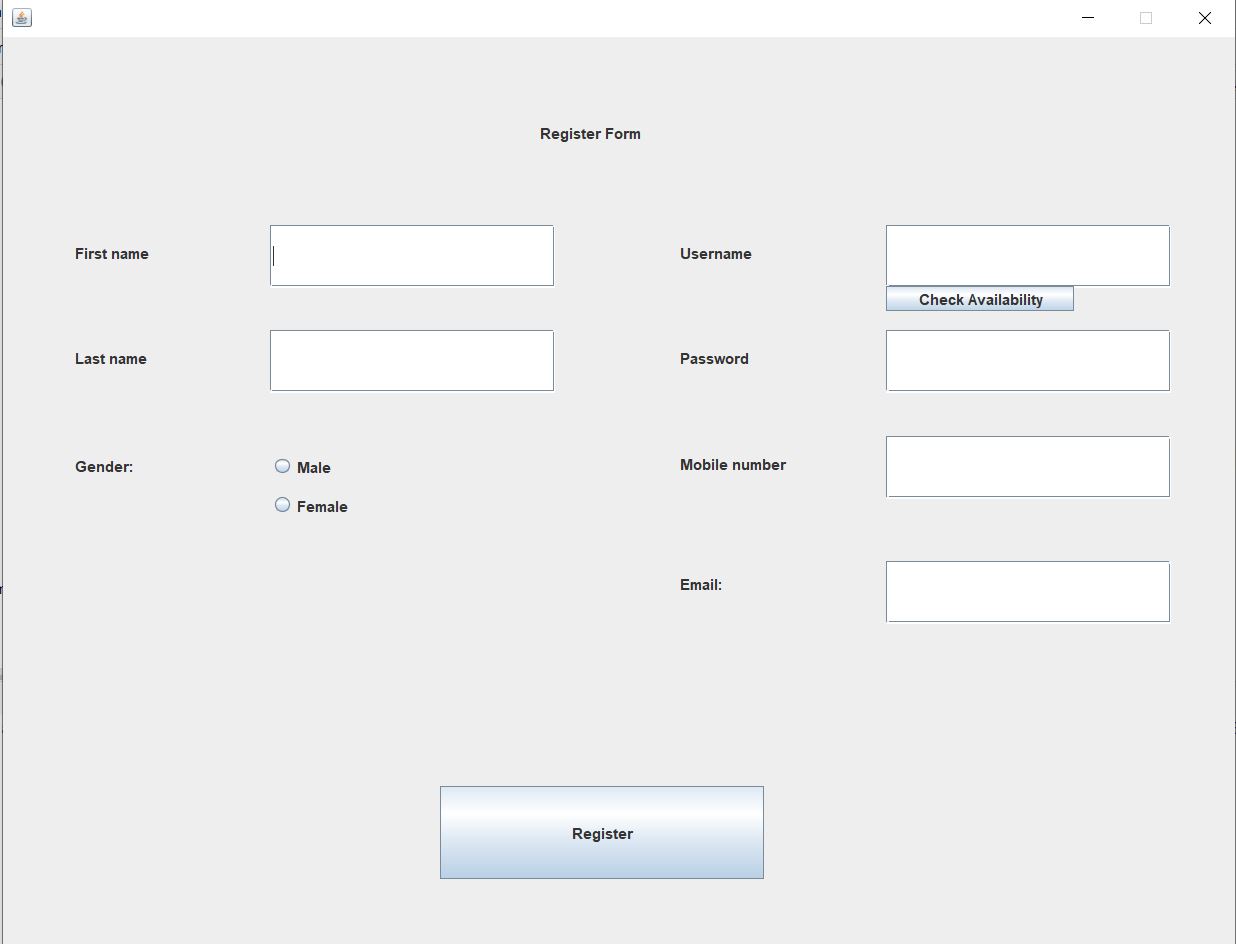
i)



ii)



iii)



The register frame will open when you select role in new user and click on register.

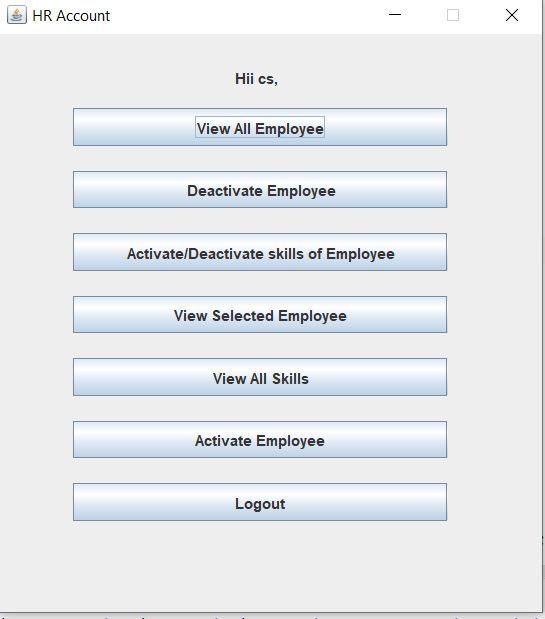
If you select EMP role then first frame will open up.

If you select HR role then second frame will open up.

If you select HR role then third frame will open up.

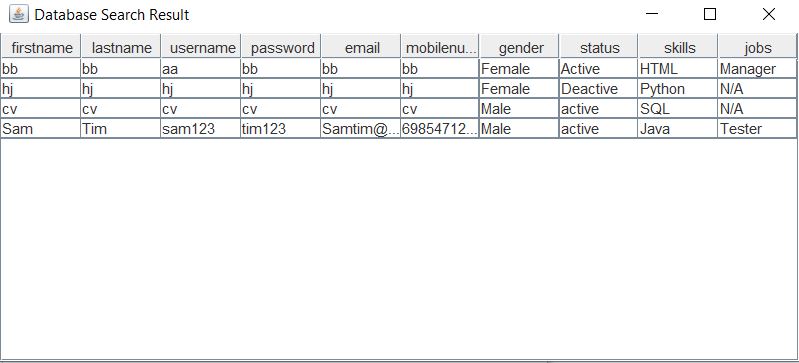
In all Register frame to register you have to fill all the fields and in username you have to type a username and click on check Availability. If the username is available then a pop-up window will come with a message that the username is available otherwise error message will pop-up and you have to enter the username again. At last, you click on register and that a confirmation message will pop-up that you are successfully registered in the database.

4.**HR Account Frame**-



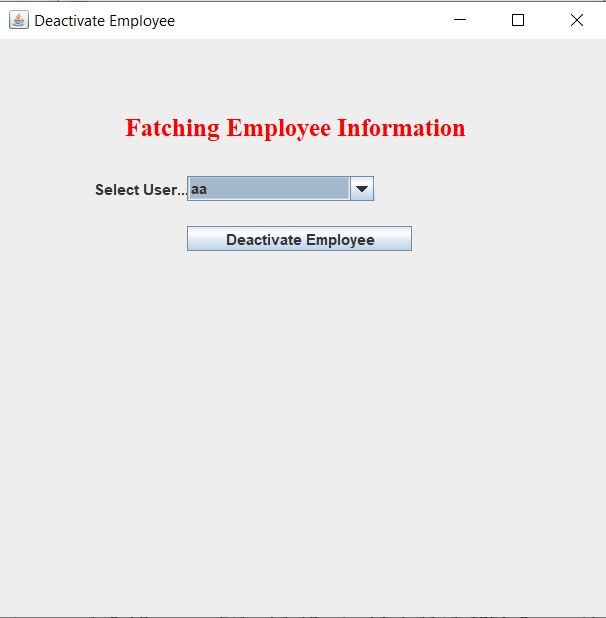
If you login as HR, you will see this interface with various buttons. The role of all buttons is explained below-

* 1. **View All Employee**: On clicking this button a Table will be displayed in which you can see the list of all Employees as shown below-



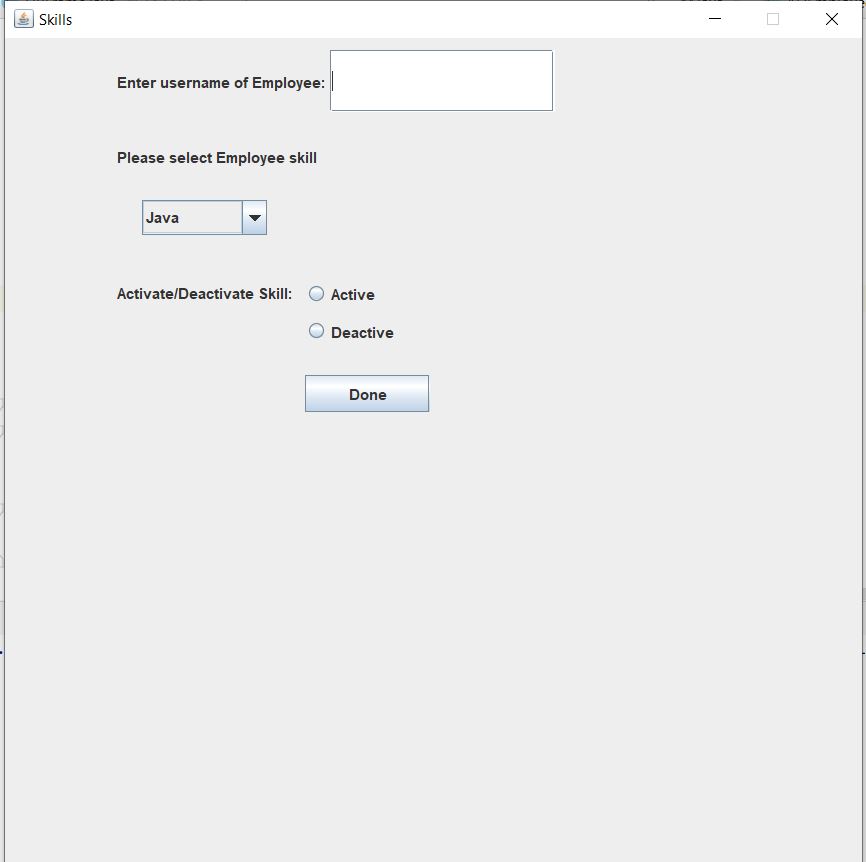
The table fetches the EMP table data from the SQL database.

ii)**Deactivate Employee**: On clicking this button another frame will pop-up which has a combo box in which all the employee username will be displayed and you just have to select the username which you have to deactivate and click on deactivate button as shown below-



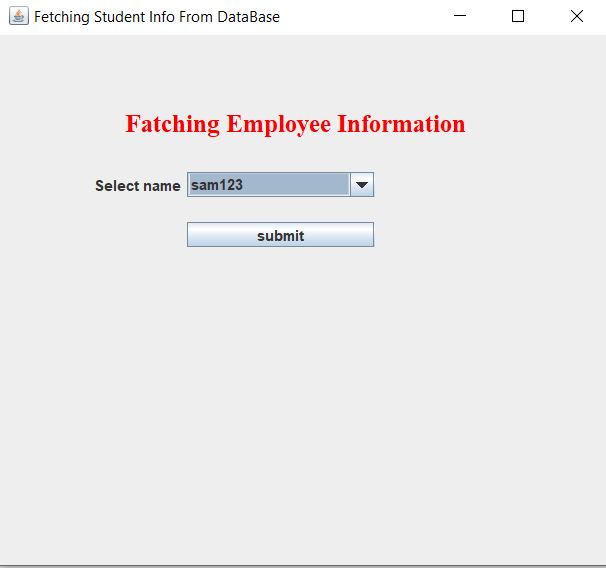
The table fetches the EMP table data from the SQL database.

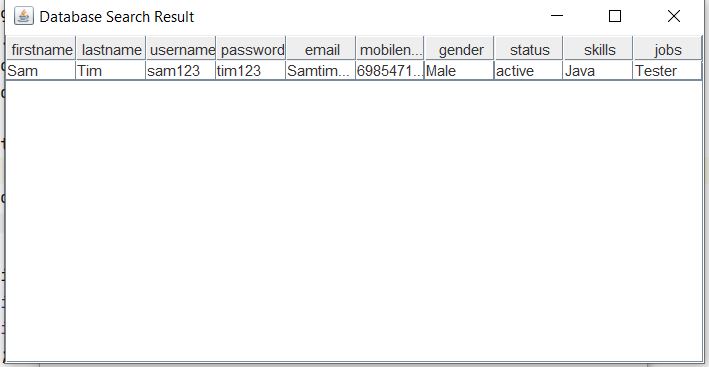
iii)Activate/Deactivate Skill of Employee: On clicking this button another frame will pop-up where you will be asked to enter user-name and Skill of the employee and whether to activate the skill or deactivate the skill and finally click on done button.



This frame makes changes to the Skill Table in the database.

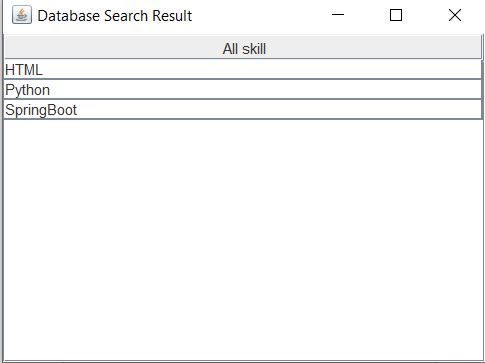
iv) View selected Employee: On clicking this button another frame will pop-up in which you can select an employee from the list of employees in the combo box and then can view the particular employee records in a Table.





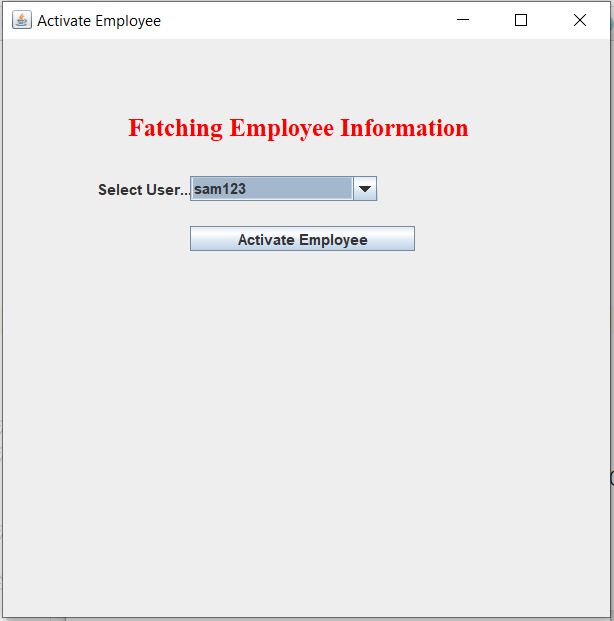
This frame fetches data from EMP table in the database.

v)View All Skills: On clicking on this button you will be able to see the list of all Skills which are there in the database.



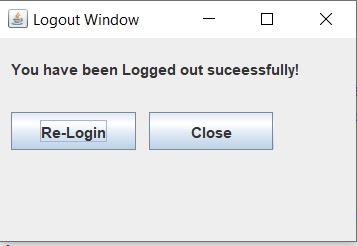
The frame fetches data from the Skill Table in the database.

vi)Activate Employee: Just like deactivate you can activate an employee also by clicking on this button.



The frame fetches data from the EMP table in the database.

vii)Logout: On clicking on this button you logout from your account and a frame pops up asking you to re-login or close the window.



5.**PM Account Frame**-

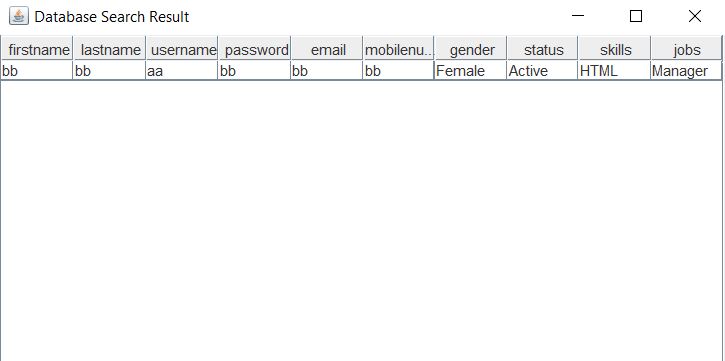


If you login as PM, you will see this interface with various buttons. The role of all buttons is explained below-

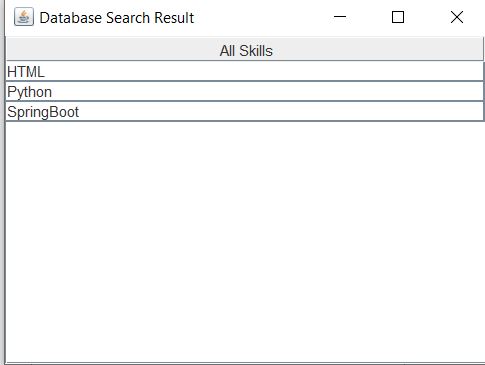
1. View Employee of Particular List: On clicking this button another frame will pop-up in which you have to select a particular skill and the list of employee who has that particular skill will be displayed in a Table.



The frame fetches data from the skill and EMP table in the database.

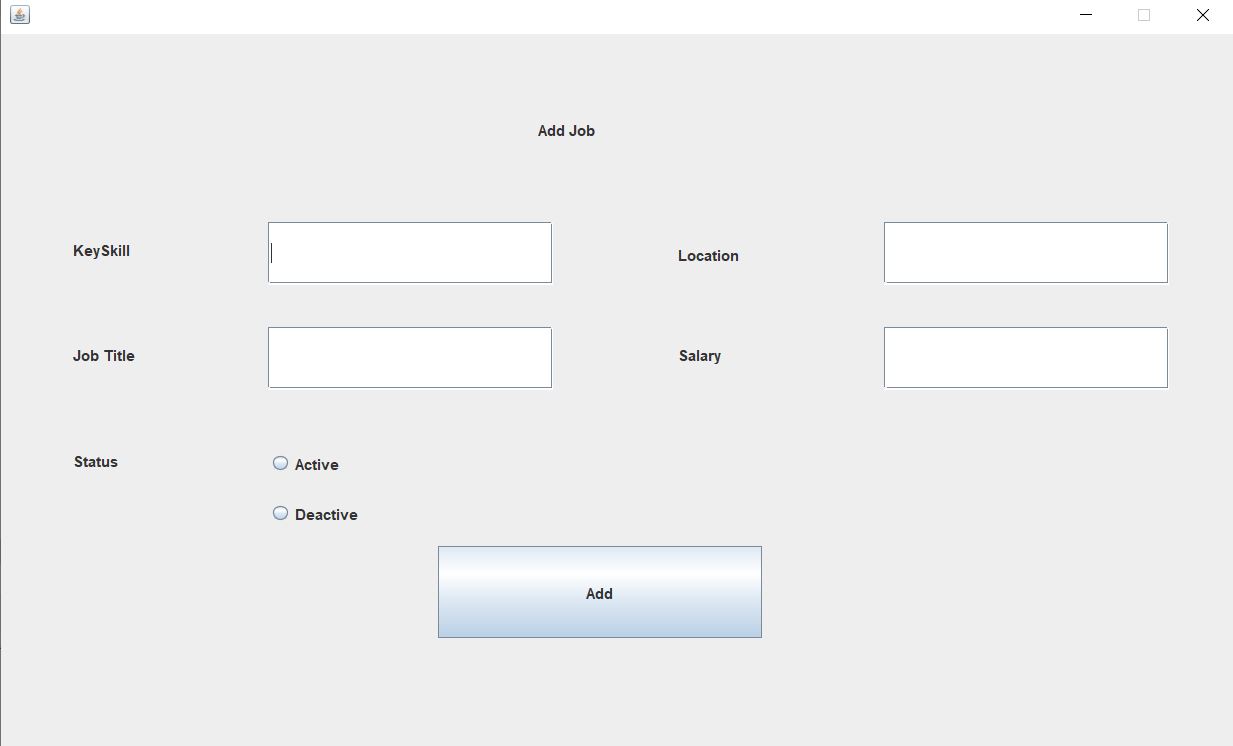


1. View All Skills: On clicking on this button you will be able to see the list of all Skills which are there in the database.



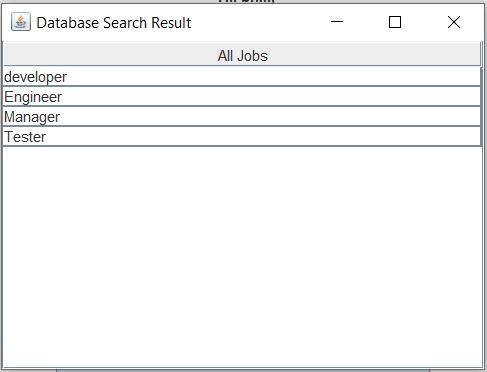
The frame fetches data from the Skill Table in the database.

1. Add Job: On clicking this button a frame will pop-up in which you can add a job to the database by filling all the required fields. After that the employee will be able to apply for the job.



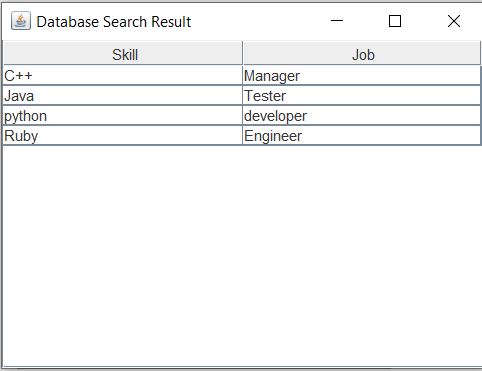
All values of this frame are inserted in the Job Table in the database.

1. View All job: On clicking this button a table will be displayed in which you can see all the jobs which are there in the database.



This frame fetches data from Job Table in the database.

1. View Skill-Wise Job: On clicking this button a table will be displayed in which you can see the job and the skill required for the job.



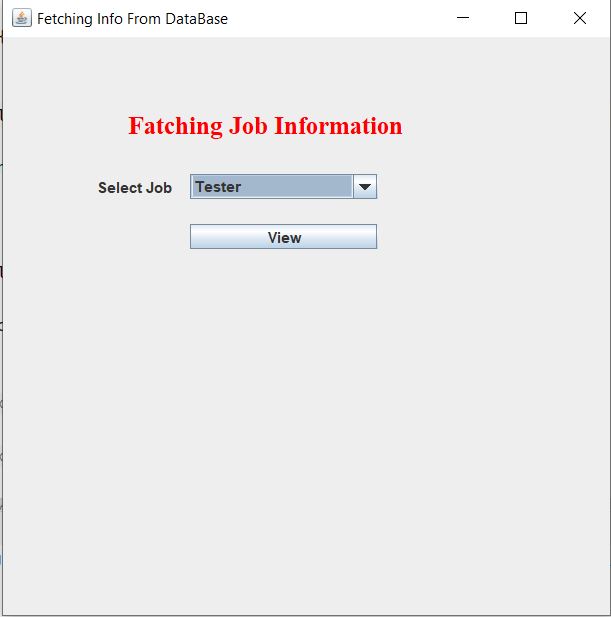
This frame fetches data from Job Table in the database.

1. Deactivate Job: On clicking on this button another frame will pop-up in which you have to select the job you want to deactivate from the list in the combo box and click on deactivate button.



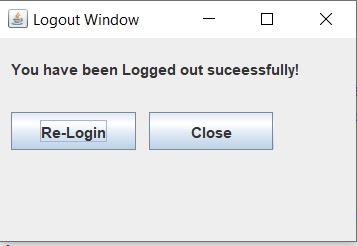
This frame will update the status of the job in the Job Table.

1. View Employee List who applied for job: On clicking on this button another frame will pop-up in which you have to select the job in which you want to see which employee applied from the list of jobs in the combo box. On clicking a Table will pop-up displaying the list of employees who applied for particular job.

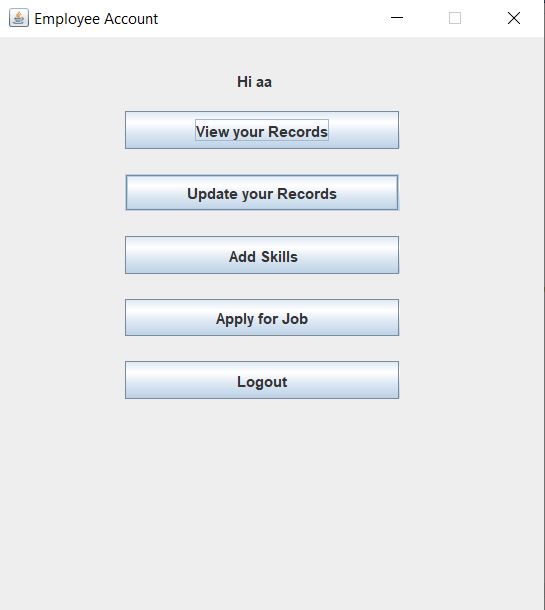


The table fetched data from the Job Table.

1. Logout: On clicking on this button you logout from your account and a frame pops up asking you to re-login or close the window.

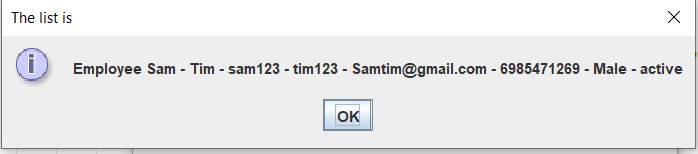


6.**EMP Account Frame**-



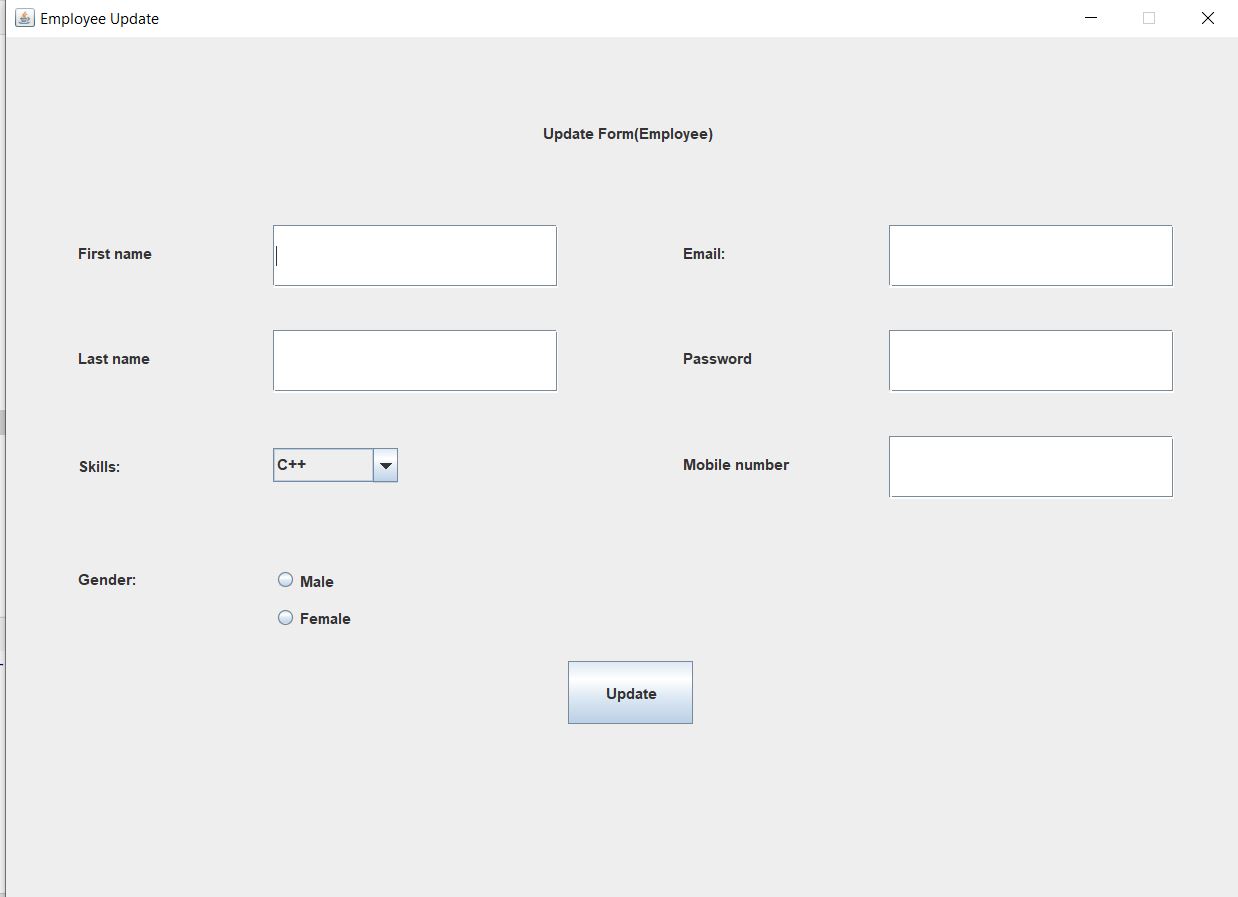
If you login as EMP, you will see this interface with various buttons. The role of all buttons is explained below-

1. View Your Records: On clicking on this button you will be able to view all your records.



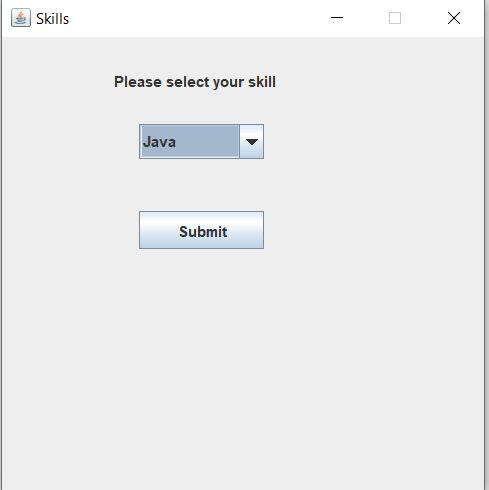
This frame fetches data from the EMP Table in the database.

1. Update your Records: On clicking all this button you will be able to update your records in the database. But username can be only chosen once and cannot be updated.



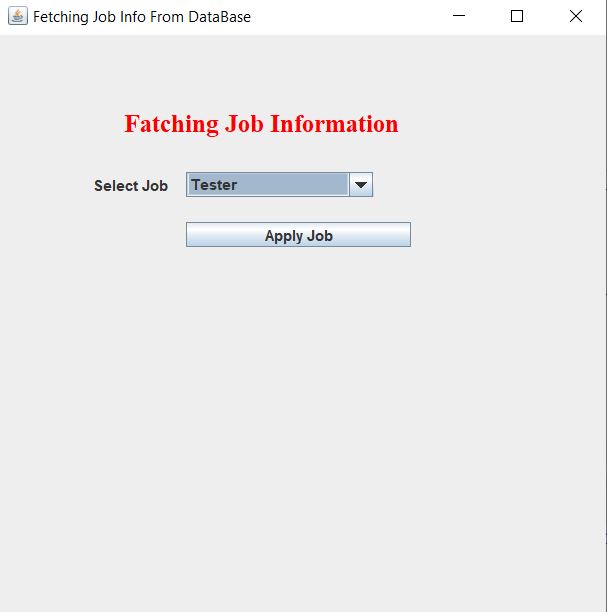
The frame updates the EMP table in the database.

1. Add Skill: On clicking on this button you will able to add skill to your account in the database. You have to just click on the skill present in the combo box and click on Submit button.

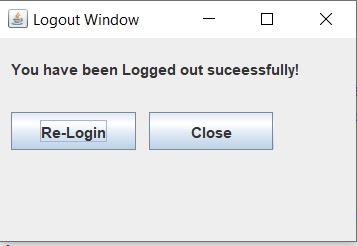


The frame inserts data in the Skill Table in the database.

1. Apply for Job: On clicking on this button another frame will pop-up in which you will see the list of all active jobs that you can apply which is set by Project Manager(PM) account.



1. Logout: On clicking on this button you logout from your account and a frame pops up asking you to re-login or close the window.



**Thank You**