PBL REPORT

ON

MEDICAL STORE MANAGEMENT SYSTEM

DBMS (KCS-551)

Submitted by-

Kushagra Singh (1900970130059)

Kuldeep Nirmal (1900970130058)

Mukul Nag (1900970130067)

Under the supervision of

Mr. Gagan Gupta

Department of Information Technology

Galgotias college of Engineering and Technology

Greater Noida

December 2021

ACKNOWLEDGEMENT

We want to give special thanks to our DBMS instructor Gagan Gupta for the timely advice and valuable guidance during designing and implementation of this project work.

We also want to express my sincere thanks and gratitude to Dr. Sanjeev Kumar Singh, Head of Department (HOD), for providing me with the facilities and for all the encouragement and support.

Finally, We express our sincere thanks to all staff members in the department of Information Technology branch for all the support and cooperation

Kushagra Singh (1900970130059)
Kuldeep Nirmal (1900970130058)
Mukul Nag (1900970130067)

Content

Introduction

MEDICAL STORE MANAGEMENT SYSTEM

It is a management system of Medical Store, to maintain and manage record of medicines, employees and transactions related information. It is a one stop intelligent solution of problems every medical store owners faces.
Moreover, its friendly GUI makes it more useful and time saving. Administrator can manage everything, and each employee will have there own account, thus, making each employee work private from each other. However, ADMIN can observe everything.

Employees will only have rights for adding records, and there attendance is monitored within the application. Employees can reset the password either by their unique User Name or by directly contacting the administrator of the store.

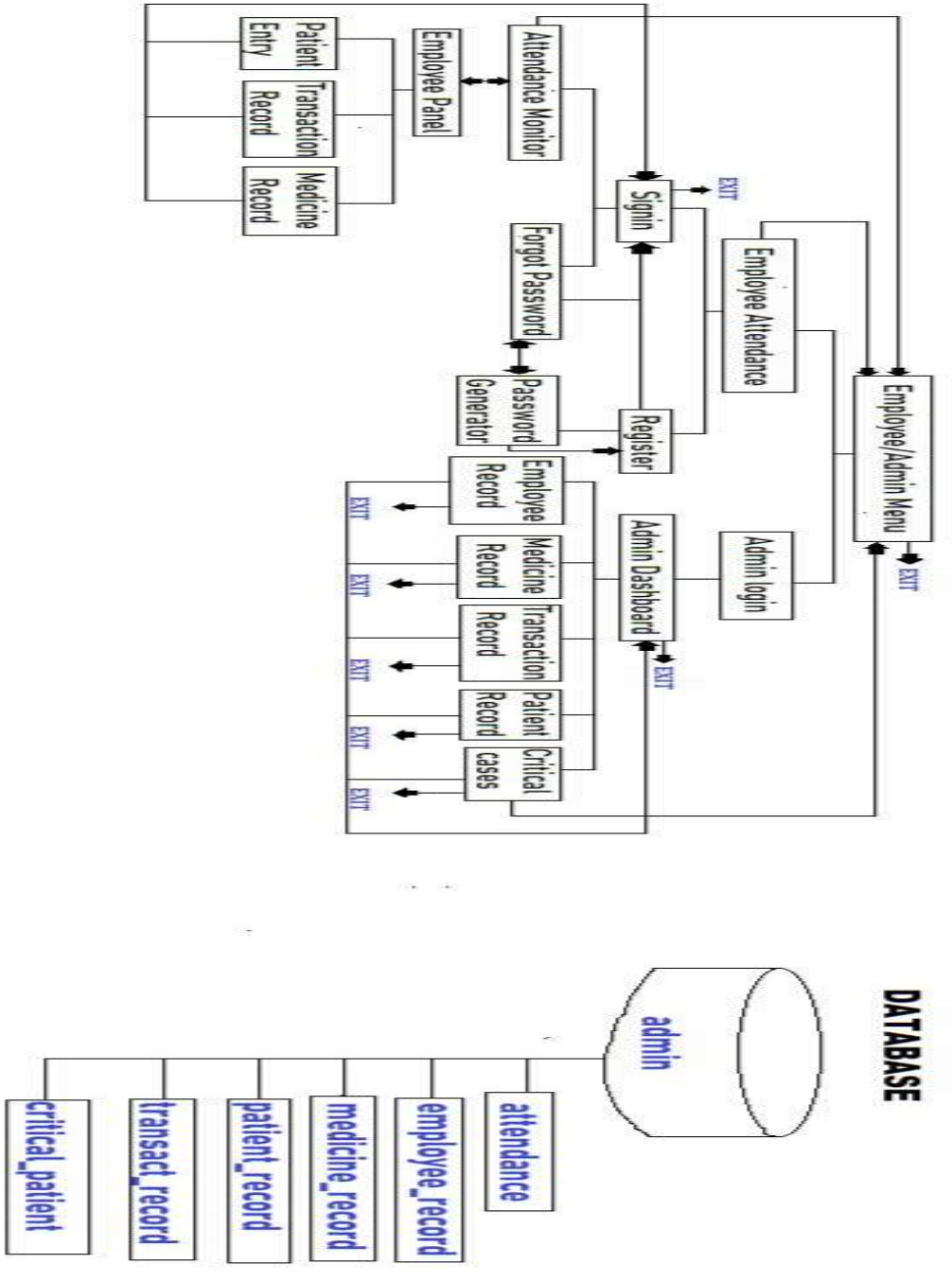
Administrator credentials are given below, they can not be reset by the admin itself, if in case lost or forgotten. Admin needs to contact the engineer for the recovery.
User Name: admin
Password: root
For this project, Java language and MySQL databases are used.

Features & Applications

1) It is very useful to monitor the attendance of each employee in your medical store.
2) Employees can manage their records with limited access as compared to admin.
3) The records saved are in the form of tables having rows and columns, which makes it more comfortable and easier to manage data.
4) Admin have some advance rights, other than employees, like managing attendance record, critical cases management system etc., which makes it more powerful software in the market.
5) Admin is provided with unique login credential, which can not be modified by anyone, which makes it more secure.
6) Each employee have its own login credential, making each employee private from each other.

Design

ER Diagram & Database Flow Chart



Database design

In this project, MySQL database management system is used, in which a database is created under which required tuples(tables) are stored.

In this application "Admin" database is used. It contains six tables as shown.

1) Database

```
MariaDB [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| admin              |
| information_schema |
| mysql              |
| performance_schema |
| phpmyadmin         |
| test               |
+--------------------+
6 rows in set (0.002 sec)

MariaDB [(none)]>
```

```
MariaDB [(none)]> use admin;
Database changed
MariaDB [admin]> show tables;
+--------------------+
| Tables_in_admin    |
+--------------------+
| attendence         |
| critical_patient   |
| employee_record    |
| medicine_record    |
| patient_record     |
| transact_record    |
+--------------------+
6 rows in set (0.001 sec)
```

2) Tables

Attendance Record

```
MariaDB [(none)]> use admin;
Database changed
MariaDB [admin]> desc attendence;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| Username   | varchar(20) | YES  |     | NULL    |       |
| Date       | date        | NO   | PRI | NULL    |       |
| Attendence | char(10)    | YES  |     | NULL    |       |
+------------+-------------+------+-----+---------+-------+
3 rows in set (0.031 sec)

MariaDB [admin]> insert into attendence values("kushagra","2020-10-14","P");
Query OK, 1 row affected (0.004 sec)

MariaDB [admin]> select * from attendence;
+----------+------------+------------+
| Username | Date       | Attendence |
+----------+------------+------------+
| kushagra | 2020-10-14 | P          |
+----------+------------+------------+
1 row in set (0.000 sec)
```

Employee Record

```
MariaDB [admin]> desc employee_record;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| UserName | varchar(10) | YES  | UNI | NULL    |       |
| Password | varchar(11) | YES  |     | NULL    |       |
| Name     | varchar(20) | YES  |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
3 rows in set (0.080 sec)

MariaDB [admin]> insert into employee_record values("harsh","6666","Harsh");
Query OK, 1 row affected (0.118 sec)

MariaDB [admin]> select * from employee_record;
+----------+----------+-------+
| UserName | Password | Name  |
+----------+----------+-------+
| harsh    | 6666     | Harsh |
+----------+----------+-------+
1 row in set (0.000 sec)
```

Medicine Record

```
MariaDB [admin]> desc medicine_record;
+----------+--------------+------+-----+---------+-------+
| Field    | Type         | Null | Key | Default | Extra |
+----------+--------------+------+-----+---------+-------+
| Name     | varchar(30)  | YES  | UNI | NULL    |       |
| MRP      | double(10,3) | YES  |     | NULL    |       |
| Discount | double(10,3) | YES  |     | NULL    |       |
+----------+--------------+------+-----+---------+-------+
3 rows in set (0.031 sec)

MariaDB [admin]> insert into medicine_record values("Paracetamol","65","0");
Query OK, 1 row affected (0.104 sec)

MariaDB [admin]> select * from medicine_record;
+-------------+--------+----------+
| Name        | MRP    | Discount |
+-------------+--------+----------+
| Paracetamol | 65.000 |    0.000 |
+-------------+--------+----------+
1 row in set (0.000 sec)
```

Patient Record

```
MariaDB [admin]> desc patient_record;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| PatientID | varchar(30) | YES  | UNI | NULL    |       |
| Name      | varchar(30) | YES  |     | NULL    |       |
| Medicine  | varchar(20) | YES  |     | NULL    |       |
| Cost      | double(10,3)| YES  |     | NULL    |       |
| AnyDebt   | double(10,2)| YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
5 rows in set (0.175 sec)

MariaDB [admin]> insert into patient_record values("PID01","Kushagra","Paracetamol","65","0");
Query OK, 1 row affected (0.068 sec)

MariaDB [admin]> select * from patient_record;
+-----------+---------+------------+--------+---------+
| PatientID | Name    | Medicine   | Cost   | AnyDebt |
+-----------+---------+------------+--------+---------+
| PID01     | Kushagra| Paracetamol| 65.000 |    0.00 |
+-----------+---------+------------+--------+---------+
1 row in set (0.000 sec)
```

Transaction Record

```
MariaDB [admin]> desc transact_record;
+---------------+-------------+------+-----+---------+-------+
| Field         | Type        | Null | Key | Default | Extra |
+---------------+-------------+------+-----+---------+-------+
| TransactionID | varchar(30) | YES  | UNI | NULL    |       |
| Cost          | double(10,3)| YES  |     | NULL    |       |
| Discount      | double(10,3)| YES  |     | NULL    |       |
| Date          | date        | YES  |     | NULL    |       |
+---------------+-------------+------+-----+---------+-------+
4 rows in set (0.055 sec)

MariaDB [admin]> insert into transact_record values("TID01","65","0","2020-11-15");
Query OK, 1 row affected (0.111 sec)

MariaDB [admin]> select * from transact_record;
+---------------+--------+----------+------------+
| TransactionID | Cost   | Discount | Date       |
+---------------+--------+----------+------------+
| TID01         | 65.000 |    0.000 | 2020-11-15 |
+---------------+--------+----------+------------+
1 row in set (0.000 sec)
```

Critical Patient Record

```
MariaDB [admin]> desc critical_patient;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| PatientID | varchar(30) | YES  | UNI | NULL    |       |
| Name      | varchar(30) | YES  |     | NULL    |       |
| Cost      | double(10,3)| YES  |     | NULL    |       |
| Date      | date        | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
4 rows in set (0.109 sec)

MariaDB [admin]> insert into critical_patient values("PID01","ABC","1000","2020-11-15");
Query OK, 1 row affected (0.082 sec)

MariaDB [admin]> select * from critical_patient;
+-----------+------+----------+------------+
| PatientID | Name | Cost     | Date       |
+-----------+------+----------+------------+
| PID01     | ABC  | 1000.000 | 2020-11-15 |
+-----------+------+----------+------------+
1 row in set (0.000 sec)
```

Implementation

The medical store management system is made using database (MySQL) and java application connectivity.

Following software are used for making the project:

Xampp database software for MySQL localhost server

In this project, MySQL have following credentials:

a) **User Name: root**
b) **** No password is set, so it needs to be empty.**
c) **Host: 127.0.0.1**
d) **Port: 3306**

MySQL J (Connector) for MySQL database and Java connectivity.

Java Development kit (JDK) needs to be installed for Java libraries.

NetBeans IDE.

rs2xml.jar file for editing queries into GUI table from SQL database.

Inno Setup Compiler for making executable (.exe) installer for making application independent of IDE.

Following are the hardware requirements for this project:

Minimum CPU: Platinum III or equivalent. Recommended CPU: intel core i3 with 6th generation or equivalent.

CPU speed of at least 1 Giga Hertz.

Minimum RAM: 256MB. Recommended RAM: 4GB.

Operating System: Windows XP. Recommended: Windows 8.1 or higher.

Low end graphics for visuals of at least 64MB. Disk Space: 2GB is recommended.

NetBeans IDE is used for designing GUI and Java coding. As MySQL queries are initiated in Java instructions, MySQL connector J is used which export the extracted SQL query from Java to MySQL database.

For this purpose, MySQL JDBC (Java Database Connectivity) driver is added in libraries of Java project. The folder containing .jar file of the connector.

The rs2xml.jar file which is added to the libraries of Java project, makes it possible to view any changes in table in GUI.

CODING

Panel Change

dispose(); function closes the old panel. Calling opening the targeted panel is in following format: <new_panel> <var> = new <new_panel>(); <new_panel>.setVisible(true);

For Example:

```
private void attendencebtnActionPerformed(java.awt.event.ActionEvent evt) {
    //  GO TO Attendence
    dispose();
    Attendence_register_login frame = new Attendence_register_login();
    frame.setVisible(true);
}
```

Libraries Used

a) To handle various exceptions "java.awt.HeadlessException" is used.

b) To Provides the connectivity SQL database "java.sql.*" is used.

c) To Provides connectivity to rs2xml library "net.proteanit.sql.*" is used.

d) To show dialog messages "javax.swing.JOptionPane" is used.

```
import java.awt.HeadlessException; //To handle various exceptions
import java.sql.*; //Provides the connectivity SQL database
import net.proteanit.sql.*; // Provides connectivity to rs2xml library
import javax.swing.JOptionPane; //To show dialog messages
```

To View Record in table

To display record:

try { } catch() {} function is used to handle errors.
First try function is executed, if any error occurs catch function handles the error.
Class.forName(); function loads the drivers.
DriverManager.getConnection() makes the connectivity to the target databases.
PreparedStatement function is used to prepared the statement (query) stored in some string variable.
executeQuery() functions executes the query in database.

<tableName>.setModel(DbUtils.resultSetToTableModel (<result>)) is used to display record to the table.

close() function closes the connection to the database.

```
// Display records
try{
    Class.forName("com.mysql.jdbc.Driver"); //Loads the driver
    try (Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/admin","root","")) {
        String sql="select * from medicine_record";
        PreparedStatement pstmt=con.prepareStatement(sql);
        ResultSet rs=pstmt.executeQuery();
        output.setModel(DbUtils.resultSetToTableModel(rs));
        con.close();
    }
}
catch(ClassNotFoundException | SQLException e){
    JOptionPane.showMessageDialog(null,"Error is: "+e.getMessage());
}
```

To add/modify/delete record

If()….else if()….else() functions are used to check if user has not made any invalid input to the textboxes. The last else() statement is used to execute if all conditions are satisfied. try{} and catch() {} functions are used to handle errors. Class.forName(); function loads the drivers. DriverManager.getConnection() makes the connectivity to the target databases.

PreparedStatement function is used to prepare the statement (query) stored in some string variable.

executeQuery() functions executes the query in database. JOptionPane.showMessageDialog(); function is used to display the successful message.

```
// Adding medicine record
            // Employee Record edit
        if(mednametxt.getText().trim().isEmpty())
        {
            JOptionPane.showMessageDialog(null,"Please enter the name of the medicine!");
        }
        else if(mrptxt.getText().trim().isEmpty())
        {
            JOptionPane.showMessageDialog(null,"Please enter MRP!");
        }
        else if(disctxt.getText().trim().isEmpty())
        {
            JOptionPane.showMessageDialog(null,"Please enter Discount! If NONE, enter 0.");
        }
        else{
            try{
            Class.forName("com.mysql.jdbc.Driver"); //Loads the driver
                try (Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/admin","root","")) {
                    String sql="insert into medicine_record values(?,?,?)";
                    PreparedStatement pstmt=con.prepareStatement(sql);
                    pstmt.setString(1,mednametxt.getText());
                    pstmt.setString(2,mrptxt.getText());
                    pstmt.setString(3,disctxt.getText());
                    pstmt.execute();
                    JOptionPane.showMessageDialog(null,"Medicine added successfulluy. Pleae refresh to see the changes!");
                    con.close();
                }
        }
        catch(HeadlessException | ClassNotFoundException | SQLException e){
            JOptionPane.showMessageDialog(null,"Error is: "+e.getMessage());
        }
        }
    }
```

```
// Change/Modifyusername
if(modifyusernametxt.getText().trim().isEmpty())
 {
     JOptionPane.showMessageDialog(null,"Please enter all fields!");
 }
else if(newusernametxt.getText().trim().isEmpty())
{
    JOptionPane.showMessageDialog(null,"Please enter all fields!");
}
 else{
     try{
     String olduser=modifyusernametxt.getText();
     String newuser=newusernametxt.getText();
     Class.forName("com.mysql.jdbc.Driver"); //Loads the driver
         try (Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/admin","root","")) {
             String sql="update employee_record set username='"+newuser+"' where username='"+olduser+"'";
             PreparedStatement pstmt=con.prepareStatement(sql);
             pstmt.execute();
             JOptionPane.showMessageDialog(null,"Employee updated successfulluy. Pleae refresh to see the changes!");
             con.close();
         }
 }
 catch(HeadlessException | ClassNotFoundException | SQLException e){
     JOptionPane.showMessageDialog(null,"Error is: "+e.getMessage());
 }
}
```

```
// Deletes the selected record
if(modifyusernametxt.getText().trim().isEmpty())
{
    JOptionPane.showMessageDialog(null,"Please enter UserName!");
}
else{
    try{
    String olduser=modifyusernametxt.getText();
    Class.forName("com.mysql.jdbc.Driver"); //Loads the driver
        try (Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/admin","root",""))
            String sql="delete from employee_record where username='"+olduser+"'";
            PreparedStatement pstmt=con.prepareStatement(sql);
            pstmt.execute();
            JOptionPane.showMessageDialog(null,"Employee deleted successfulluy.");
            con.close();
        }
}
catch(HeadlessException | ClassNotFoundException | SQLException e){
    JOptionPane.showMessageDialog(null,"Error is: "+e.getMessage());
}
```

Password Generator

private String list = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890";

This is the list of all characters from which Randoms are chosen by the program.

try{} and catch() {} functions are used to handle errors.

if()….else() statement is used to check for the condition if length provided is less than 100 or not.

The for() loop is used to get the random characters from the list the number of times, the length is provided.

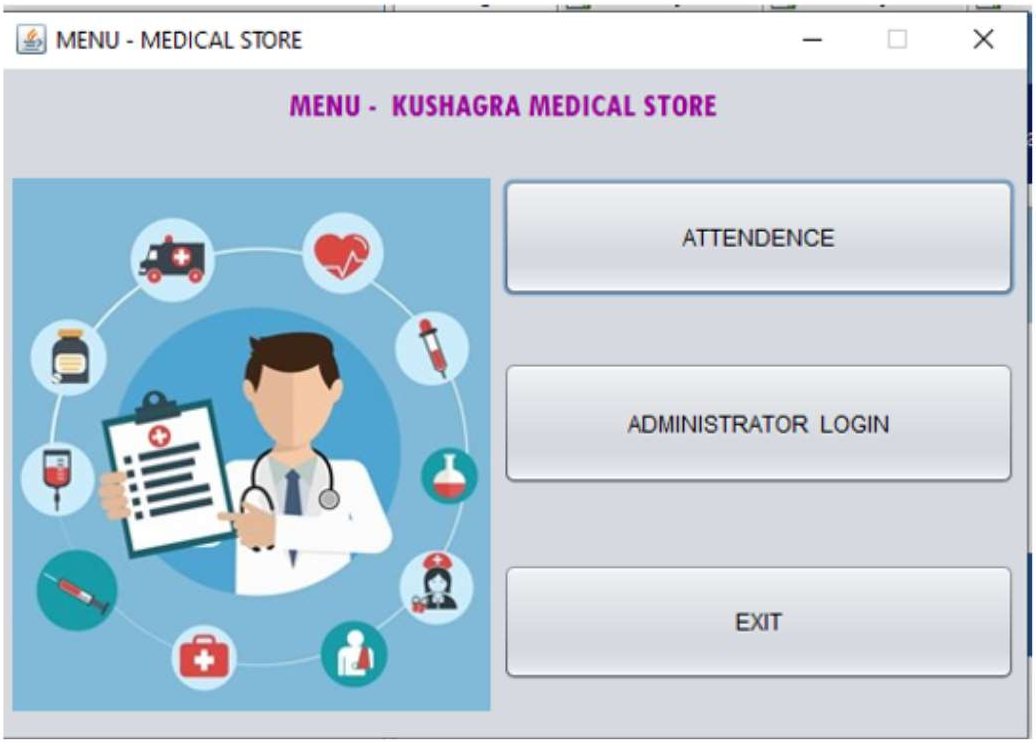
```
private String list = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890";

/**
 * This method is called from within the constructor to initialize the form.
 * WARNING: Do NOT modify this code. The content of this method is always
 * regenerated by the Form Editor.
 */
```

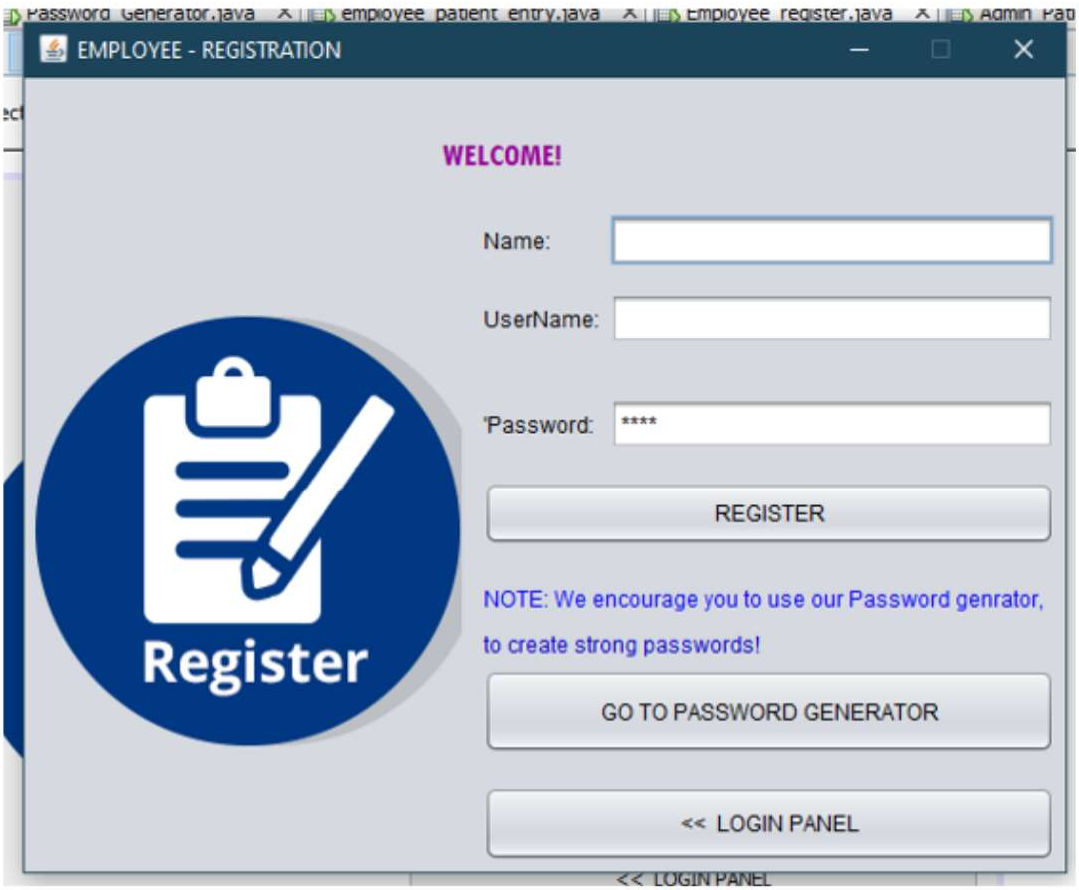
```
    // Password Generator:
    try{
        int length = Integer.parseInt(len.getText());
        if(length <= 100){
        StringBuilder str = new StringBuilder();
        for(int i = 0;i<length;i++){
            int num = getRandomNum();
            char ch = list.charAt(num);
            str.append(ch);
        }
        output.setText(str.toString());
    }
        else{
            JOptionPane.showMessageDialog(null,"Please enter length upto 100 only. You entered: "+length);
        }
    }
    catch(NumberFormatException e){
        JOptionPane.showMessageDialog(null,"Error: "+e);
    }
}
```

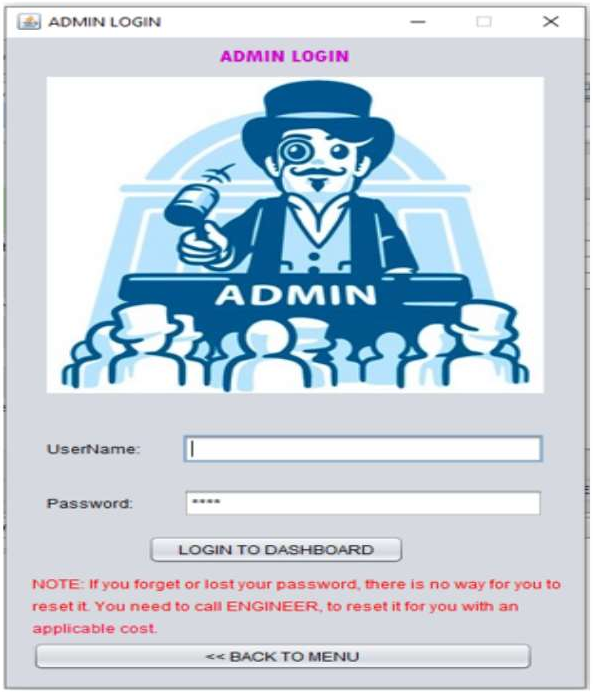
Snapshots & Result

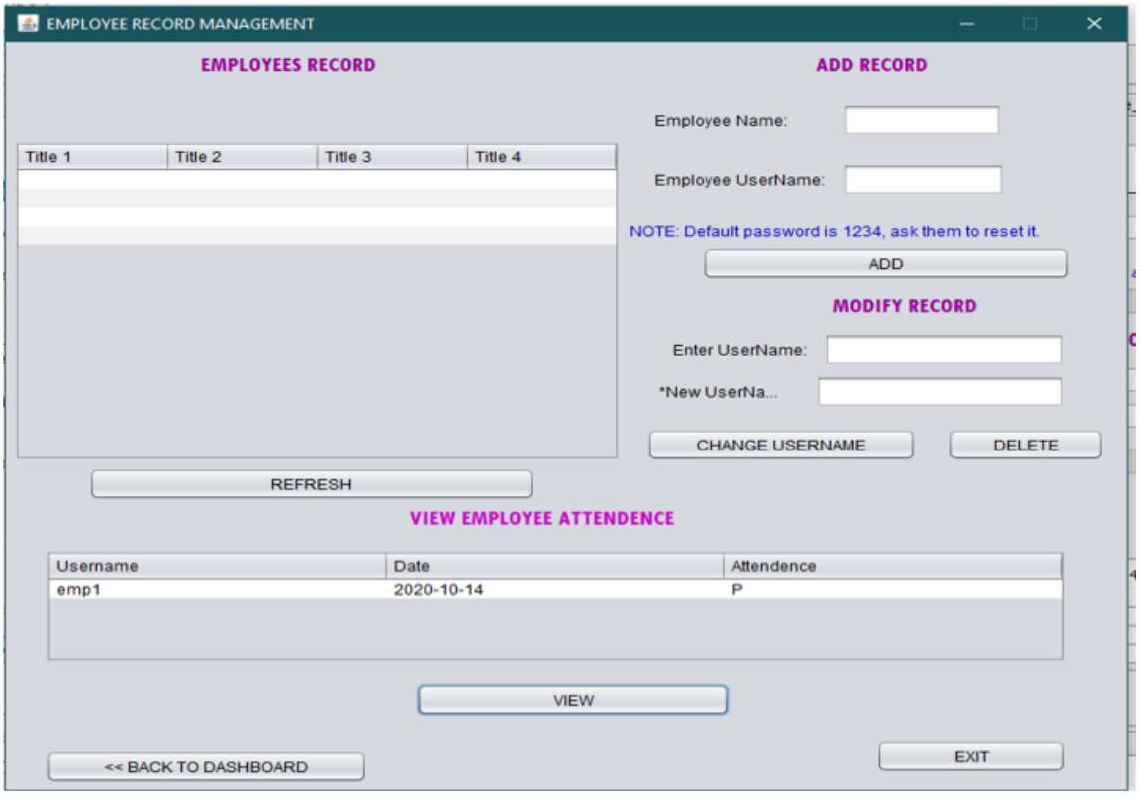
The application run successfully of which certain outputs are as:




EMPLOYEE - ATTENDENCE
EMPLOYEE - RGISTER / LOGIN
SIGNIN
REGISTER
<< MENU


EMPLOYEE - REGISTRATION
WELCOME!
Name:
UserName:
Password:
****
REGISTER
NOTE: We encourage you to use our Password genrator,
to create strong passwords!
GO TO PASSWORD GENERATOR
<< LOGIN PANEL
Register


ADMIN LOGIN
ADMIN LOGIN
ADMIN
UserName:
Password:
****
LOGIN TO DASHBOARD
NOTE: If you forget or lost your password, there is no way for you to reset it. You need to call ENGINEER, to reset it for you with an applicable cost.
<< BACK TO MENU


EMPLOYEE RECORD MANAGEMENT
EMPLOYEES RECORD
Title 1
Title 2
Title 3
Title 4
REFRESH
ADD RECORD
Employee Name:
Employee UserName:
NOTE: Default password is 1234, ask them to reset it.
ADD
MODIFY RECORD
Enter UserName:
*New UserNa...
CHANGE USERNAME
DELETE
VIEW EMPLOYEE ATTENDENCE
Username
Date
Attendence
emp1
2020-10-14
P
VIEW
<< BACK TO DASHBOARD
EXIT

Conclusion and Future Work

It is a basic version of the application. It can be used to store data in global database, so that database can be accessed from anywhere around the world. But to be in safe zone, private server is recommended.

References

1) **Official My SQL reference material:**
   https://dev.mysql.com/doc/refman/5.6/en/
2) **YouTube videos explaining the concepts:**
   Connectivity: https://www.youtube.com/watch?v=dO0iKzfXzl0
   Check Password:
   https://www.youtube.com/watch?v=R0slwlulmT8&t=906s
3) **https://www.google.com**