Doctohome

by Saravana Kumar B

TABLE OF CONTENTS

CHAPTERS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Java is a general-purpose language that is object oriented and is designed to have few implementation dependencies. It is a Write-Once-Run-Anywhere language that is platform independent meaning Java programs can be written on one OS and run on a different one. This is possible because of Java bytecode that can run on any JVM (Java Virtual Machine).

MySQL is an open-source relational database management system developed by MySQL AB, MySQL can be connected to Java using MySQL connector and can be used in Java using the JDBC. This project has been done using JAVA and MySQL. This project focuses on implementing a Graphical User Interface application on Java and connecting it to a backend database that allows for dynamic change of data on the Graphical User Interface.

1.1 PROBLEM DEFINITION

- Since Hospital is associated with the lives of common people and their day-to-day routines so I decided to work on this project.
- The manual handling of the record is time consuming and highly prone to error. The purpose of this project is to automate or make online, the process of day-to-day activities like Admission of New Patient, Discharge of Patient, Assign a Doctor, and finally compute the bill etc.
- I have tried to design the software in such a way that user may not have any difficulty in using this package & further expansion is possible without much effort. Even though I cannot claim that this work to be entirely exhaustive, the main purpose of my exercise is perform each Hospital's activity in computerized way rather than manually which is time consuming.
- I am confident that this software package can be readily used by non-programming personal avoiding human handled chance of error.

1.2 COURSE OBJECTIVES

- This mini project has been basically programmed using JAVA. From this course, we will have a short research on the concept that is used for creation of a Graphical User Interface (GUI). i.e., Swing. Swing is a library of JAVA that provides many tools which is used for creation of a GUI.
- We can also have a glance about MySQL, which is the database that is being used in this project. MySQL plays a very important role in storage management in this project.
- We have also learnt how to connect JAVA to MySQL using a tool called JDBC. In order to perform operations on the MySQL table via JAVA, JDBC is required.

1.3 EXPECTED OUTCOME

- Recording information about the Patients that come.
- Generating bills.
- Recording information related to diagnosis given to Patients.
- Keeping record of the Immunization provided to children/patients.
- Keeping information about various diseases and medicines available to cure them.

1.4 HARDWARE AND SOFTWARE REQUIREMENTS

The requirements of the project are as follows:

Hardware Requirements:

- **Operating System** – Windows 7, 8, 10
- **RAM** – 1 GB or more
- **Processor Chip** – Intel core i3, i5, i7

Software Requirements:

- **Language** – JAVA Swing
- **Database** – MySQL
- **Application** – NetBeans IDE, MySQL Workbench, MySQL Command Client

CHAPTER 2

OBJECT ORIENTED CONCEPTS

Object-oriented programming System (OOPs) is a programming paradigm based on the concept of “objects” that contain data and methods. The primary purpose of object-oriented programming is to increase the flexibility and maintainability of programs. Object oriented programming brings together data and its behaviour(methods) in a single location makes it easier to understand how a program works.

2.1 CLASS

The class is a group of similar entities. It is only a logical component and not the physical entity.

Example:

```
Public class MyClass

{

int x=5;

}
```

2.2 OBJECT

An object can be defined as an instance of a class, and there can be multiple instances of a class in a program. An Object contains both the data and the function, which operates on the data.

Example:

```
public class MyClass

{

int x=5;

public static void main (String[] args)

{

MyClass Obj = new MyClass(); System.out.println(obj.x);

}

}
```

2.3 INHERITANCE:

Inheritance is an OOPS concept in which one object acquires the properties and behaviour of the parent object. It’s creating a parent-child relationship between two classes. It offers robust and natural mechanism for organizing and structure of any software.

Example:

```
class Employee

{

float salary=40000;

}

class Programmer extends Employee
```

```
{
int bonus=10000;
public static void main(String args[]){ Programmer p=new Programmer();
System.out.println("Programmer salary is:"+p.salary); System.out.println("Bonus of
Programmer is:"+p.bonus);
}
}
```

The different types of inheritance are:

- **Single Inheritance:** This type of inheritance allows a single subclass to inherit the properties of a parent class.

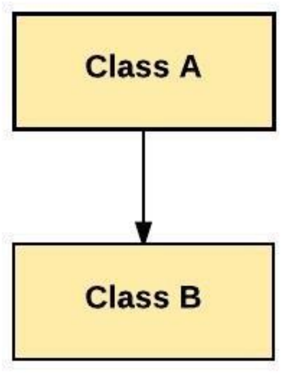


Fig 2.3.1 Single Inheritance

- **Multilevel Inheritance:** This type of inheritance allows a chain of inheritance where each subclass inherits the properties of the parent class and the grandparent class.

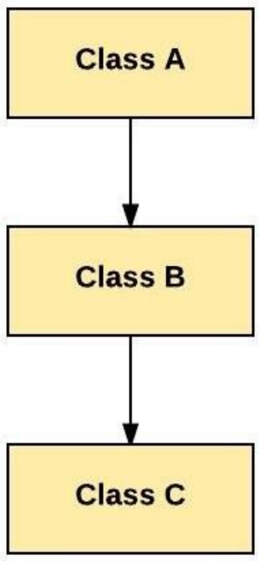


Fig 2.3.2 Multilevel Inheritance

- **Hierarchical Inheritance:** In this type of inheritance several subclasses inherit the properties of a single parent class.

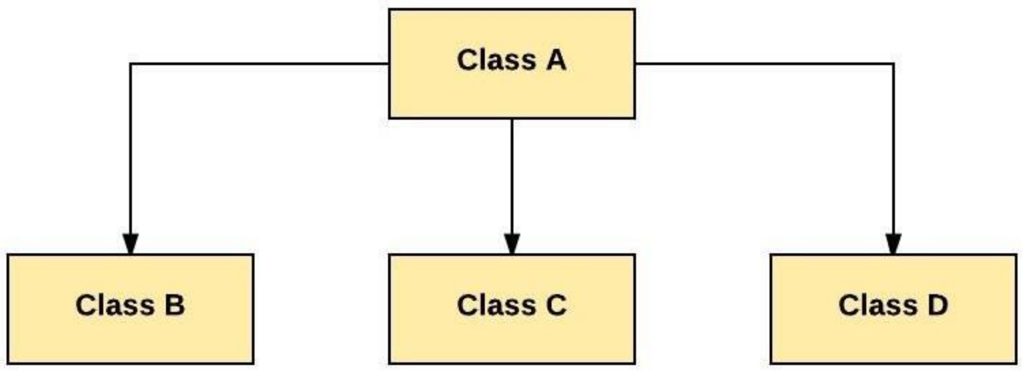
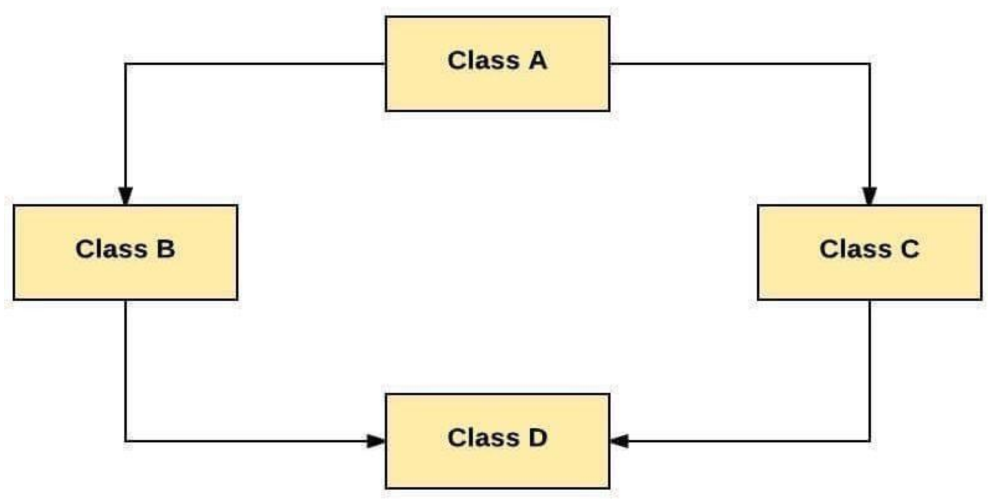


Fig 2.3.3 Hierarchical Inheritance

- **Hybrid inheritance:** This type of inheritance consists of different types of inheritances put together.



2.3.4 Hybrid Inheritance

2.4 POLYMORPHISM:

The word polymorphism means having many forms. In simple words, we can define polymorphism as the ability of a message to be displayed in more than one form.

There are two types of polymorphism:

- **Compile-time polymorphism:** The type of polymorphism which his achieved during operations such as method overloading is called compile-time polymorphism.

Method Overloading:

When there are multiple functions with same name but different parameters then these functions are said to be overloaded. Normally, functions can be overloaded either by change in number of arguments or by change in type of arguments.

Example:

```
class Multiply {
static int Mul(int a, int b)
{
return a * b;
}
static double Mul(double a, double b)
{
return a * b;
}
}
```

```
class Main {

public static void main(String[] args)

{

System.out.println(Multiply.Mul(6, 8));

System.out.println(Multiply.Mul(6.4, 7.8));

}

}
```

- **Run-time polymorphism:** It is also known as Dynamic Method Dispatch. It is a process in which a function call to the overridden method is resolved at Runtime. This type of polymorphism is achieved during method overriding.

Example:

```
class Parent {

void Print()

{

System.out.println("Parent Class");

}

}

class childclass1 extends Parent { void Print()

{

System.out.println("childclass1");
}

}

class chldclass2 extends Parent { void Print()

{

System.out.println("childclass2");

}

}

Class Main {

public static void main(String[] args)

{

Parent p;

p = new childclass1(); p.Print();

p = new childclass2(); p.Print();

}

}
```

2.5 ABSTRACT CLASS:

Abstraction is a process of hiding the implementation details and showing only functionality to the user. it shows only essential things to the user and hides the internal details. It can be achieved using two components: Abstract class and Interfaces.

Abstract class is a restricted class that cannot be used to create objects. In order to access an abstract class, it must be inherited from another class. It can have abstract and non-abstract methods. It needs to be extended. It cannot be instantiated.

Example:

```
abstract class MyClass{

abstract void print();

}


class Main extends Class{ void print(){

System.out.println("Main class");

}

public static void main(String args[]){ MyClass obj = new Main(); obj.run(); }

}
```

2.6 MULTITHREADING:

A thread is a light-weight smallest part of a process that can run concurrently with the other parts(other threads) of the same process.

Java has a feature known as multithreading that allows concurrent execution of different parts of a program, this allows for maximum CPU utilization.

Two methods to implement multithreading are: -

- By extending the Thread class
- By implementing the Runnable interface

The working of a thread is as follows:

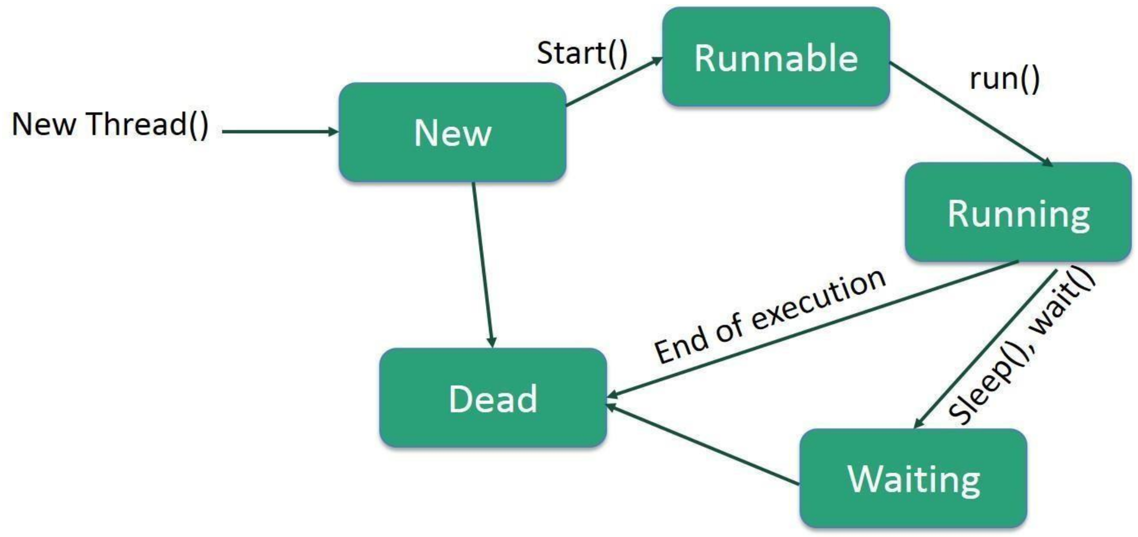


Fig 2.6.1 Thread working

2.7 I/O FUNCTIONS:

Java has various I/O streams included with its I/O package the three main streams are as follows: -

- **System.in:** System.in is used to read in data from any standard input device. Generally done using a java scanner.

- **System.out**: System.out is used to show the result of a particular executed program on a standard output device.
  There are three System.out functions: -
  - print() Prints data passed as an argument onto the console screen.
  - println() Does the same function as print but moves cursor to the next line.
  - System.err: The standard error stream used to output the data that is thrown as an error by the program on a standard output device.

2.8 JAVA PACKAGES:

A java package is a group of similar types of classes, interfaces and sub-packages. Package in java can be categorized in two forms, built-in package and user-defined package. There are many built-in packages such as java, lang, awt, javax, swing, net, io, util, sql etc. User-defned package is a package which is created by the user. It contains various methods, variables, etc. which have been added by the user for his purpose.

Example:

```
Package mypack; public class Main{
```

public static void main(String[] args) { System.out.println(“Welcome!”); }}

To compile and run a JAVA package, the steps are as follows:

Compile: javac filename.java;

Run: java Mainclassname;

There are two types of package:

- **Built-in Packages**
- **User-Defined Packages**

2.9 EXCPETION HANDLING:

An exception is an even that disrupts the flow of a program and is thrown at runtime.

The key words used to handle exceptions in java are as follows: -

- **try**

  This keyword is used to specify the code where an exception block is typed out.

- **catch**

  This keyword is used to handle the exceptions and precedes the try block.

- **throw**

  This keyword is used to throw an exception.

- **throws**

  It specifies that an exception may occur in a method.

- **finally**

  This keyword is used to execute a block irrespective of whether an exception is handled.

CHAPTER 3

DESIGN

3.1 DESIGN GOALS

The design goals of this project are as follows:

- By demonstrating OOP concepts like polymorphism, inheritance with proper programming structure.
- Implementation of JAVA features like Java Swing, JDBC, I/O manipulation and Exception Handling.
- Creation of a GUI application which performs all the operations and provides valid results.

3.2 ALGORITHM/PSEUDOCODE

```
STEP 1: Start
STEP 2: Open Registration Page
STEP 3: Allocate Memory For New Patient Information
STEP 4: Enter Information
STEP 5: Create Unique Patient Id
STEP 6: Create Username And Password
STEP 7: Create Entry To The Database
STEP 8: Allocate Doctor For The Patient
STEP 9: Tests And Reports Entry
```

STEP 10: Initiate Advance Fees

STEP 11: Allocation Of Bed And Room For Patient

STEP 12: Patient Health Progress

STEP 13: Patient Discharge

STEP 14: Bill Generation

STEP 15: End

CHAPTER 4

IMPLEMENTATION

Implementation is the stage of the project when the theoretical design is turned out into a working system. Thus it can be considered to be the most critical stage in achieving a successful new system and in giving the user, confidence that the new system will work and be effective.

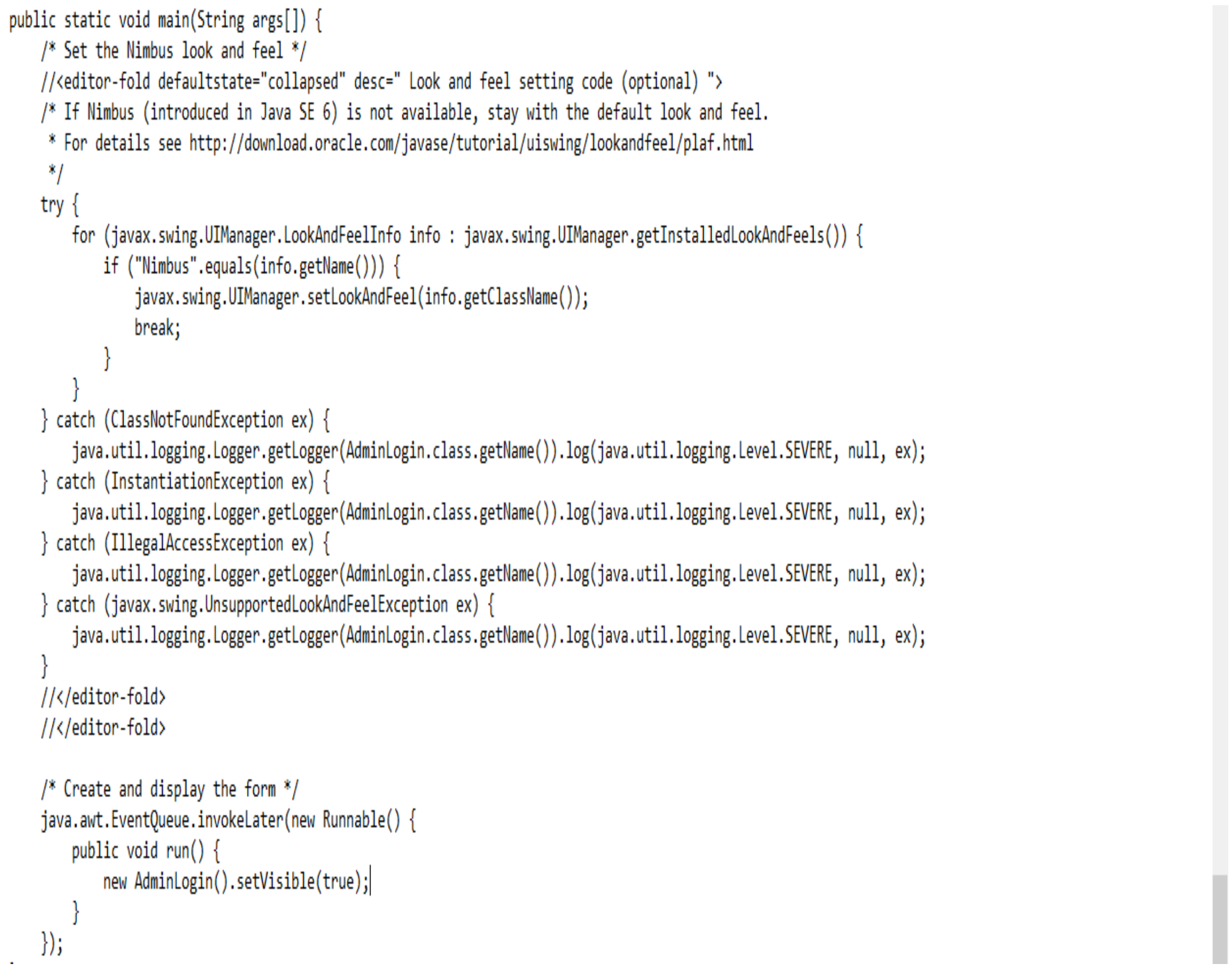
The implementation stage involves careful planning, investigation of the existing system and it's constraints on implementation, designing of methods to achieve changeover and evaluation of changeover methods.

The entire project mainly consists of 7 modules, which are

- ❖ Admin module
- ❖ User module (patient)
- ❖ Doctor module
- ❖ Nurse module
- ❖ Pharmacist module
- ❖ Laboratorist module
- ❖ Accountant module

4.1 Admin module:

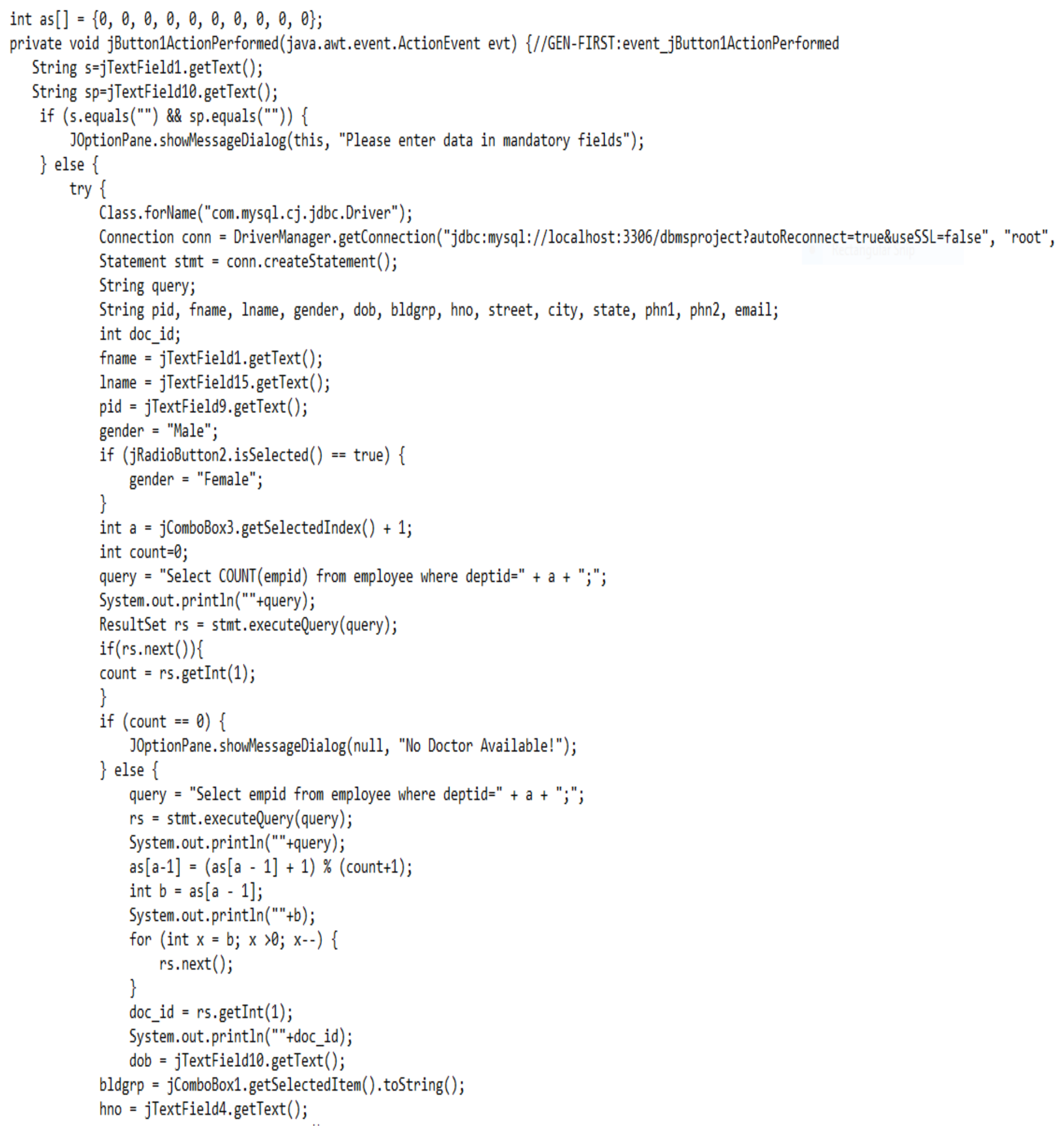
- Manage department of hospitals,
- User,
- Doctor,
- Nurse,
- Pharmacist,
- Laboratorist
- Accounts.
- Etc…,



```
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
    /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
     * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
     */
    try {
        for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
            if ("Nimbus".equals(info.getName())) {
                javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
    } catch (ClassNotFoundException ex) {
        java.util.logging.Logger.getLogger(AdminLogin.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    } catch (InstantiationException ex) {
        java.util.logging.Logger.getLogger(AdminLogin.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    } catch (IllegalAccessException ex) {
        java.util.logging.Logger.getLogger(AdminLogin.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {
        java.util.logging.Logger.getLogger(AdminLogin.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    }
    //</editor-fold>
    //</editor-fold>

    /* Create and display the form */
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new AdminLogin().setVisible(true);
        }
    });
```

4.2 User module(patient):

- View prescription details
- View medication from doctor
- View doctor list
- View admit history. like bed, ward
- Manage own profile

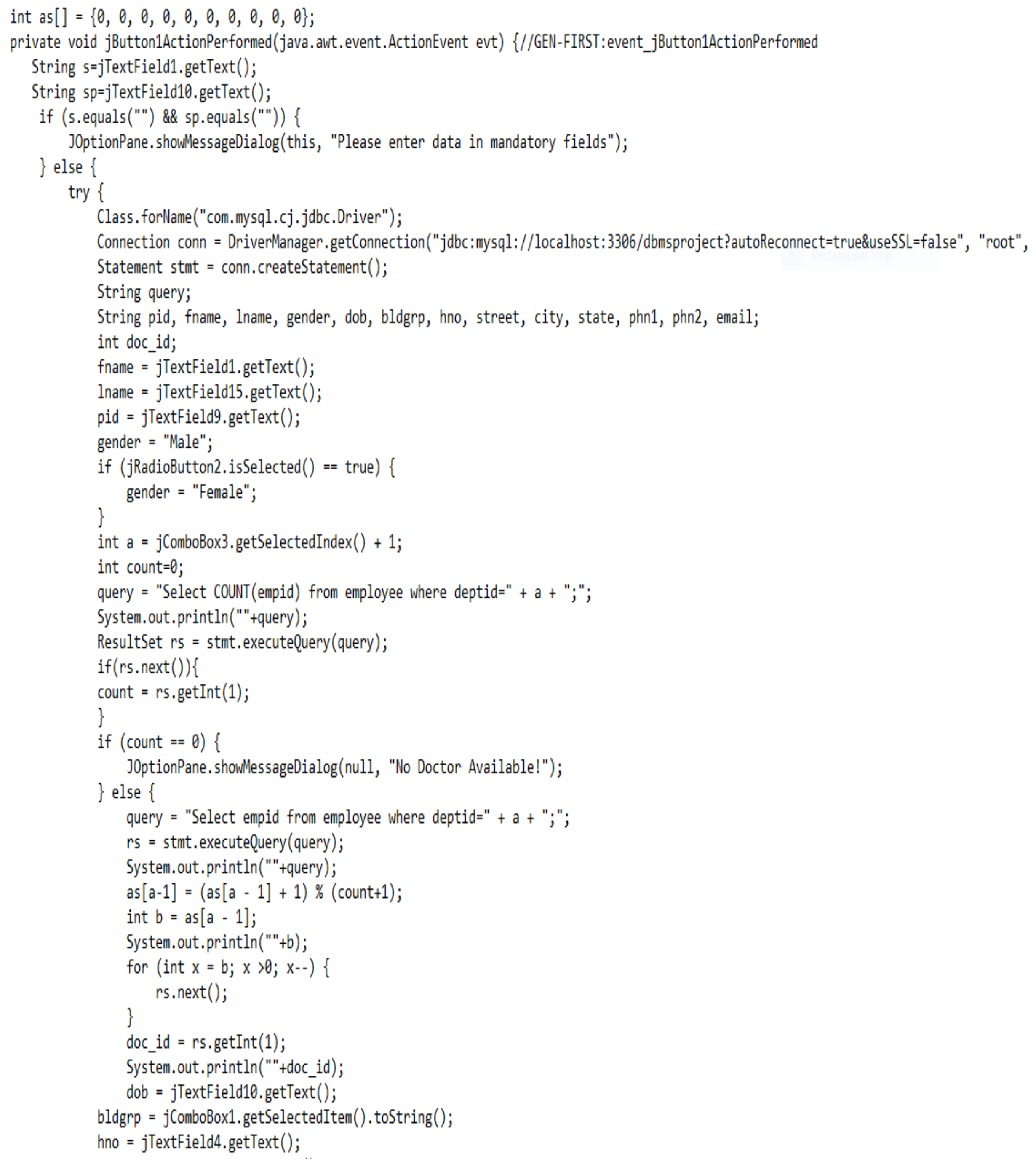


```
int as[] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0};
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_jButton1ActionPerformed
   String s=jTextField1.getText();
   String sp=jTextField10.getText();
    if (s.equals("") && sp.equals("")) {
        JOptionPane.showMessageDialog(this, "Please enter data in mandatory fields");
    } else {
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/dbmsproject?autoReconnect=true&useSSL=false", "root",
            Statement stmt = conn.createStatement();
            String query;
            String pid, fname, lname, gender, dob, bldgrp, hno, street, city, state, phn1, phn2, email;
            int doc_id;
            fname = jTextField1.getText();
            lname = jTextField15.getText();
            pid = jTextField9.getText();
            gender = "Male";
            if (jRadioButton2.isSelected() == true) {
                gender = "Female";
            }
            int a = jComboBox3.getSelectedIndex() + 1;
            int count=0;
            query = "Select COUNT(empid) from employee where deptid=" + a + ";";
            System.out.println(""+query);
            ResultSet rs = stmt.executeQuery(query);
            if(rs.next()){
            count = rs.getInt(1);
            }
            if (count == 0) {
                JOptionPane.showMessageDialog(null, "No Doctor Available!");
            } else {
                query = "Select empid from employee where deptid=" + a + ";";
                rs = stmt.executeQuery(query);
                System.out.println(""+query);
                as[a-1] = (as[a - 1] + 1) % (count+1);
                int b = as[a - 1];
                System.out.println(""+b);
                for (int x = b; x >0; x--) {
                    rs.next();
                }
                doc_id = rs.getInt(1);
                System.out.println(""+doc_id);
                dob = jTextField10.getText();
            bldgrp = jComboBox1.getSelectedItem().toString();
            hno = jTextField4.getText();
```

```
                JOptionPane.showMessageDialog(null, "Please Enter valid data!");
            }
        }
// TODO add your handling code here:
    }//GEN-LAST:event_jButton1ActionPerformed

    private void jComboBox2ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_jComboBox2ActionPerformed
        String str;
        str = jComboBox2.getSelectedItem().toString();

        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/dbmsproject", "root", "tiger");
            Statement stmt = conn.createStatement();
            String query;
            query = "Select min(rid) from room where roomtype=\"" + str + "\" and isfree=0;";
            System.out.println(""+query);
            ResultSet rs = stmt.executeQuery(query);
            int n=0,m=0;
            if (!rs.next()) {
                JOptionPane.showMessageDialog(this, "No Room Available!");
            }
            else{
                  n = rs.getInt(1);
                  jTextField16.setText("" + n);
            }

            query = "Select min(nid) from nurse_assigned where countpatient=(Select min(countpatient) from nurse_assigned);";
           // query = "Select nid from nurse_assigned where countpatient=(Select countpatient from nurse_assigned);";
            ResultSet rs1 = stmt.executeQuery(query);
            if (!rs1.next()) {
                JOptionPane.showMessageDialog(this, "No Nurse Available!");
            }
            else{
                  n = rs1.getInt(1);
                  if(n!=0)
                  jTextField17.setText("" + n);
                  else
                      JOptionPane.showMessageDialog(null,"No Nurse Available!");
            }

             System.out.println("MIUUH");
            rs.close();
            rs1.close();
            stmt.close();
            conn.close();
        } catch (Exception e) {
                e.printStackTrace();
            //JOptionPane.showMessageDialog(null, "Error in connectivity");
            System.out.println("bug4"+e);
```

4.3 Doctor module:

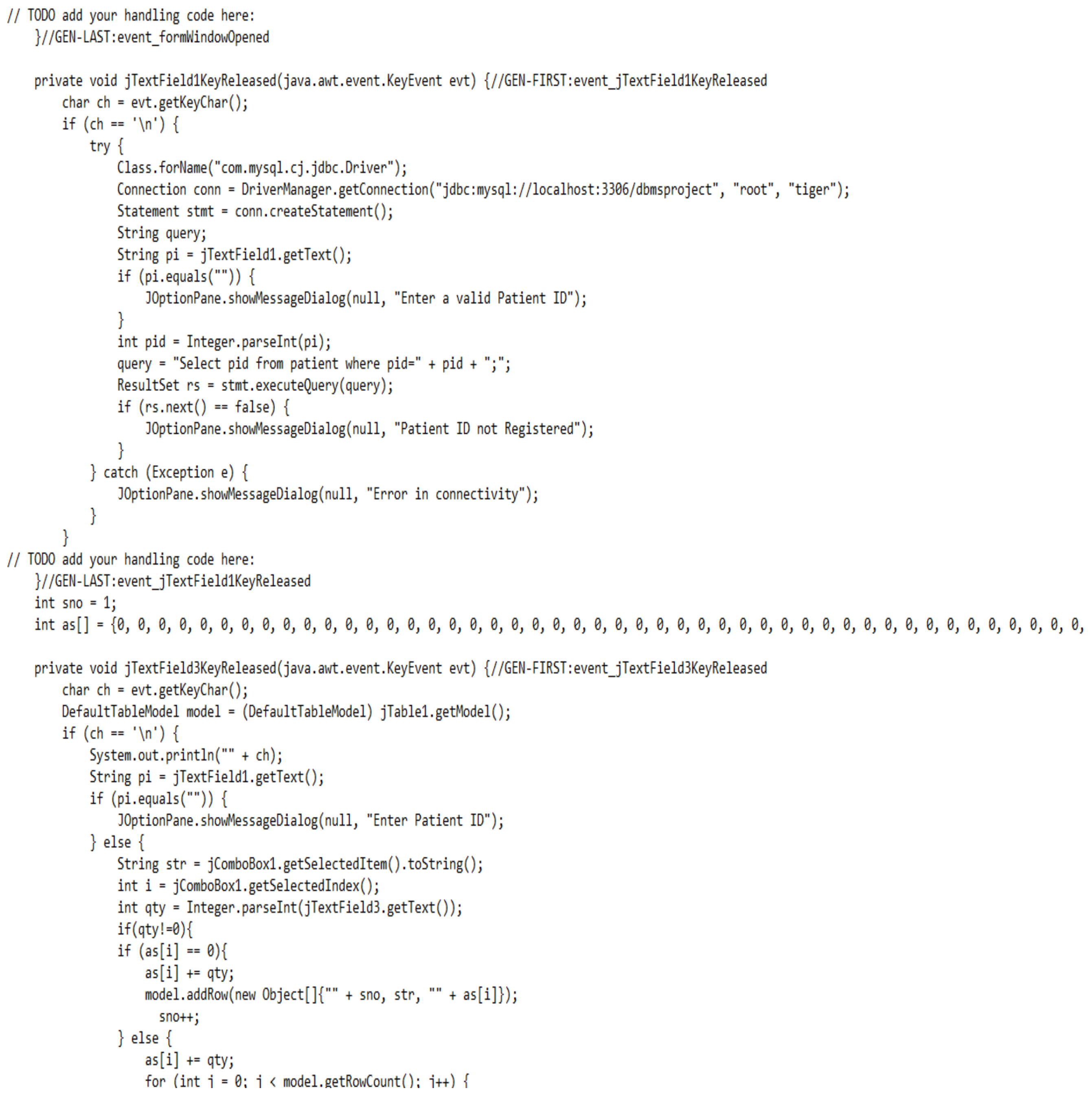
- Manage patient. account opening and updating
- Provide medication for patients
- Manage own profile



```
int as[] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0};
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_jButton1ActionPerformed
   String s=jTextField1.getText();
   String sp=jTextField10.getText();
    if (s.equals("") && sp.equals("")) {
        JOptionPane.showMessageDialog(this, "Please enter data in mandatory fields");
    } else {
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/dbmsproject?autoReconnect=true&useSSL=false", "root",
            Statement stmt = conn.createStatement();
            String query;
            String pid, fname, lname, gender, dob, bldgrp, hno, street, city, state, phn1, phn2, email;
            int doc_id;
            fname = jTextField1.getText();
            lname = jTextField15.getText();
            pid = jTextField9.getText();
            gender = "Male";
            if (jRadioButton2.isSelected() == true) {
                gender = "Female";
            }
            int a = jComboBox3.getSelectedIndex() + 1;
            int count=0;
            query = "Select COUNT(empid) from employee where deptid=" + a + ";";
            System.out.println(""+query);
            ResultSet rs = stmt.executeQuery(query);
            if(rs.next()){
            count = rs.getInt(1);
            }
            if (count == 0) {
                JOptionPane.showMessageDialog(null, "No Doctor Available!");
            } else {
                query = "Select empid from employee where deptid=" + a + ";";
                rs = stmt.executeQuery(query);
                System.out.println(""+query);
                as[a-1] = (as[a - 1] + 1) % (count+1);
                int b = as[a - 1];
                System.out.println(""+b);
                for (int x = b; x >0; x--) {
                    rs.next();
                }
                doc_id = rs.getInt(1);
                System.out.println(""+doc_id);
                dob = jTextField10.getText();
            bldgrp = jComboBox1.getSelectedItem().toString();
            hno = jTextField4.getText();
```

4.4 Nurse module:

- Manage patient. account opening and updating
- Allot bed, ward, cabin for patients
- Provide medication according to patient prescription
- Manage own profile



```
// TODO add your handling code here:
    }//GEN-LAST:event_formWindowOpened

    private void jTextField1KeyReleased(java.awt.event.KeyEvent evt) {//GEN-FIRST:event_jTextField1KeyReleased
        char ch = evt.getKeyChar();
        if (ch == '\n') {
            try {
                Class.forName("com.mysql.cj.jdbc.Driver");
                Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/dbmsproject", "root", "tiger");
                Statement stmt = conn.createStatement();
                String query;
                String pi = jTextField1.getText();
                if (pi.equals("")) {
                    JOptionPane.showMessageDialog(null, "Enter a valid Patient ID");
                }
                int pid = Integer.parseInt(pi);
                query = "Select pid from patient where pid=" + pid + ";";
                ResultSet rs = stmt.executeQuery(query);
                if (rs.next() == false) {
                    JOptionPane.showMessageDialog(null, "Patient ID not Registered");
                }
            } catch (Exception e) {
                JOptionPane.showMessageDialog(null, "Error in connectivity");
            }
        }
// TODO add your handling code here:
    }//GEN-LAST:event_jTextField1KeyReleased
    int sno = 1;
    int as[] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,

    private void jTextField3KeyReleased(java.awt.event.KeyEvent evt) {//GEN-FIRST:event_jTextField3KeyReleased
        char ch = evt.getKeyChar();
        DefaultTableModel model = (DefaultTableModel) jTable1.getModel();
        if (ch == '\n') {
            System.out.println("" + ch);
            String pi = jTextField1.getText();
            if (pi.equals("")) {
                JOptionPane.showMessageDialog(null, "Enter Patient ID");
            } else {
                String str = jComboBox1.getSelectedItem().toString();
                int i = jComboBox1.getSelectedIndex();
                int qty = Integer.parseInt(jTextField3.getText());
                if(qty!=0){
                if (as[i] == 0){
                    as[i] += qty;
                    model.addRow(new Object[]{"" + sno, str, "" + as[i]});
                      sno++;
                } else {
                    as[i] += qty;
                    for (int j = 0; j < model.getRowCount(); j++) {
```

4.5 Pharmacist module:

- Maintain medicine
- Keep records of hospitals stock medicines and status
- Manage medicine categories
- Watch prescription of patient
- Provide medication to prescriptions

```
// TODO add your handling code here:
    }//GEN-LAST:event_jButton3ActionPerformed

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
         * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {
            java.util.logging.Logger.getLogger(PatientsTests.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {
            java.util.logging.Logger.getLogger(PatientsTests.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {
            java.util.logging.Logger.getLogger(PatientsTests.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {
            java.util.logging.Logger.getLogger(PatientsTests.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new PatientsTests().setVisible(true);
            }
        });
    }
```

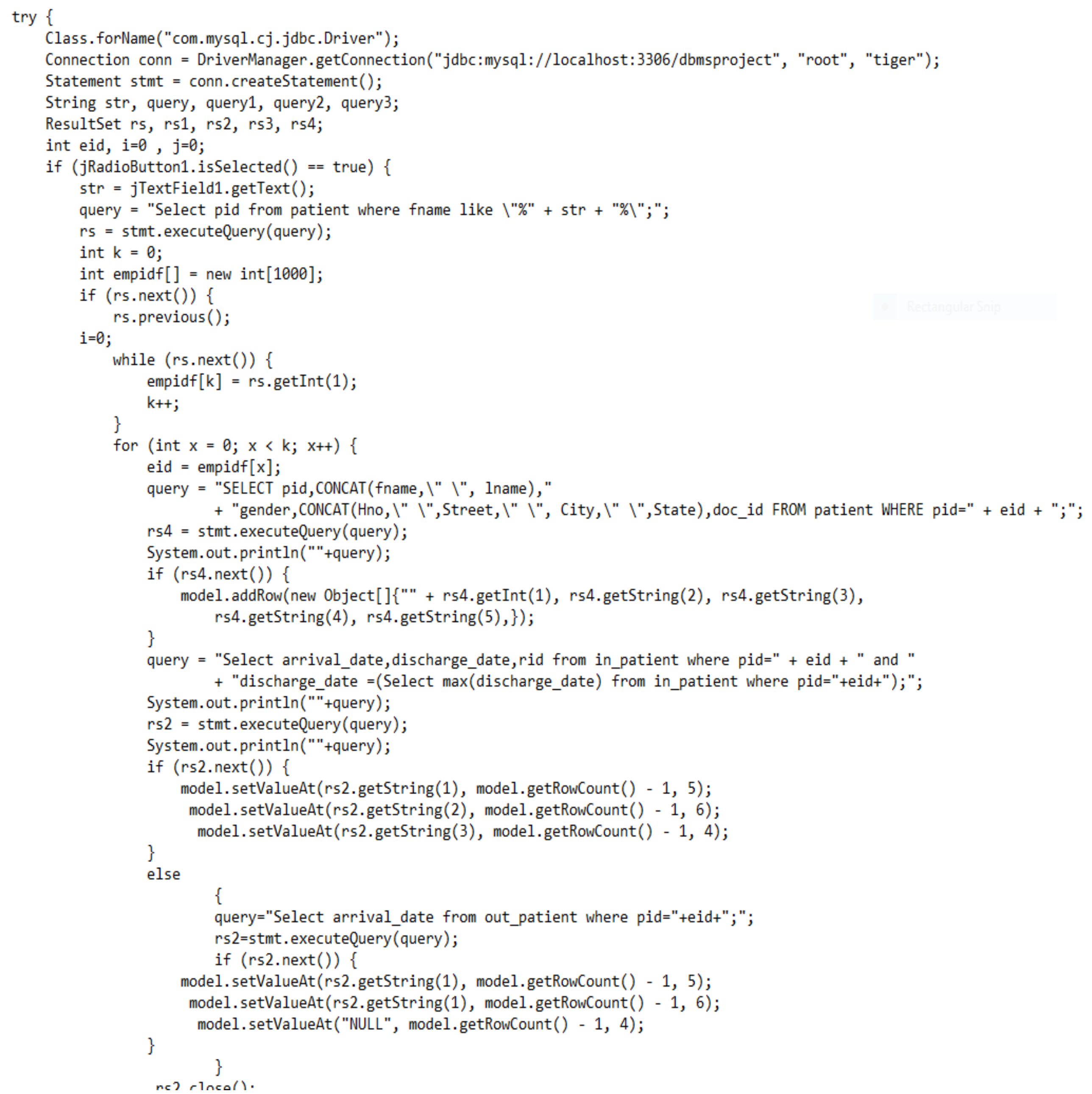
4.6 Laboratorist module:

- Watch prescription list
- Upload diagnostic report
- Preview of report files. like xray images, ct scan, mri reports
- Manage own profile

```
// TODO add your handling code here:
    }//GEN-LAST:event_jButton3ActionPerformed

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
         * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {
            java.util.logging.Logger.getLogger(PatientsTests.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {
            java.util.logging.Logger.getLogger(PatientsTests.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {
            java.util.logging.Logger.getLogger(PatientsTests.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {
            java.util.logging.Logger.getLogger(PatientsTests.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new PatientsTests().setVisible(true);
            }
        });
    }
```

4.7 Accountant module:

- Create invoice for payment
- Order invoice to patient
- Take cash payment
- Watch payment history of patients
- Manage own profile

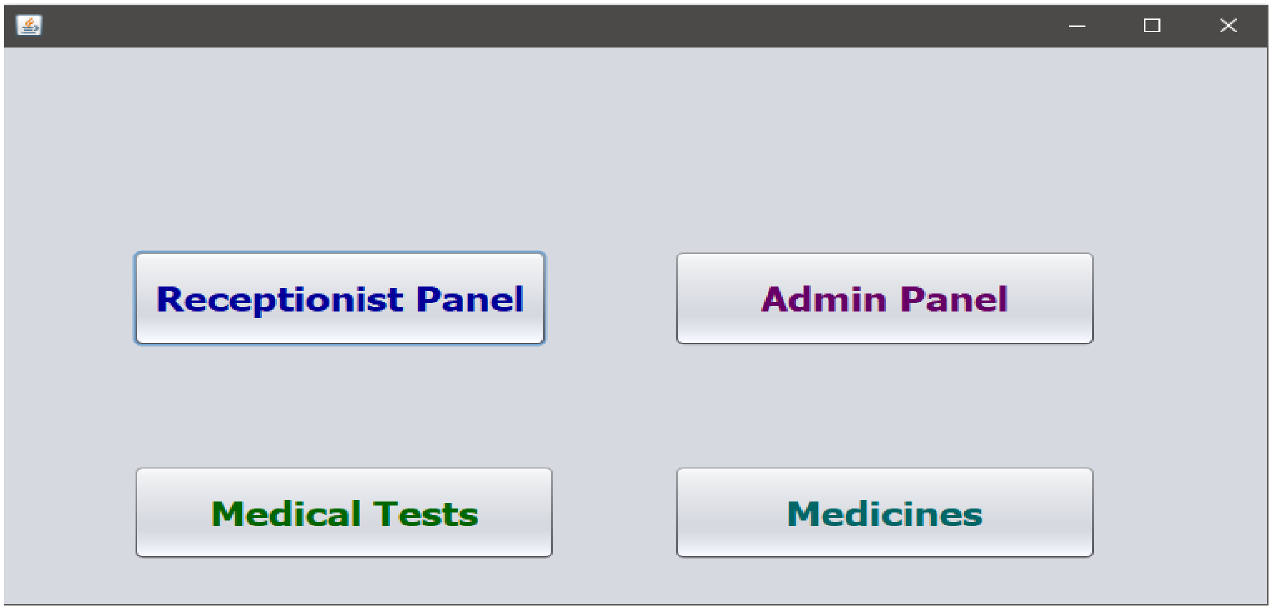


```
try {
    Class.forName("com.mysql.cj.jdbc.Driver");
    Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/dbmsproject", "root", "tiger");
    Statement stmt = conn.createStatement();
    String str, query, query1, query2, query3;
    ResultSet rs, rs1, rs2, rs3, rs4;
    int eid, i=0 , j=0;
    if (jRadioButton1.isSelected() == true) {
        str = jTextField1.getText();
        query = "Select pid from patient where fname like \"%" + str + "%\";";
        rs = stmt.executeQuery(query);
        int k = 0;
        int empidf[] = new int[1000];
        if (rs.next()) {
            rs.previous();
        i=0;
            while (rs.next()) {
                empidf[k] = rs.getInt(1);
                k++;
            }
            for (int x = 0; x < k; x++) {
                eid = empidf[x];
                query = "SELECT pid,CONCAT(fname,\" \", lname),"
                        + "gender,CONCAT(Hno,\" \",Street,\" \", City,\" \",State),doc_id FROM patient WHERE pid=" + eid + ";";
                rs4 = stmt.executeQuery(query);
                System.out.println(""+query);
                if (rs4.next()) {
                    model.addRow(new Object[]{"" + rs4.getInt(1), rs4.getString(2), rs4.getString(3),
                        rs4.getString(4), rs4.getString(5),});
                }
                query = "Select arrival_date,discharge_date,rid from in_patient where pid=" + eid + " and "
                        + "discharge_date =(Select max(discharge_date) from in_patient where pid="+eid+");";
                System.out.println(""+query);
                rs2 = stmt.executeQuery(query);
                System.out.println(""+query);
                if (rs2.next()) {
                    model.setValueAt(rs2.getString(1), model.getRowCount() - 1, 5);
                     model.setValueAt(rs2.getString(2), model.getRowCount() - 1, 6);
                      model.setValueAt(rs2.getString(3), model.getRowCount() - 1, 4);
                }
                else
                        {
                        query="Select arrival_date from out_patient where pid="+eid+";";
                        rs2=stmt.executeQuery(query);
                        if (rs2.next()) {
                    model.setValueAt(rs2.getString(1), model.getRowCount() - 1, 5);
                     model.setValueAt(rs2.getString(1), model.getRowCount() - 1, 6);
                      model.setValueAt("NULL", model.getRowCount() - 1, 4);
                }
                        }
                 rs2.close();
```

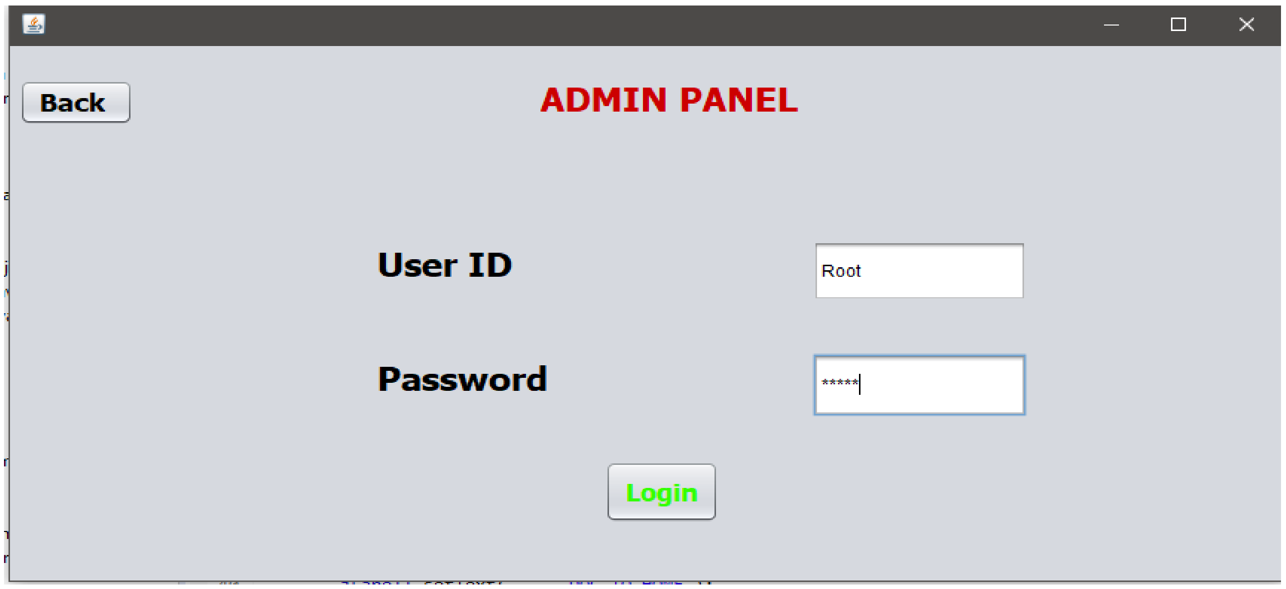
CHAPTER 5

RESULT

5.1 MAIN PAGE



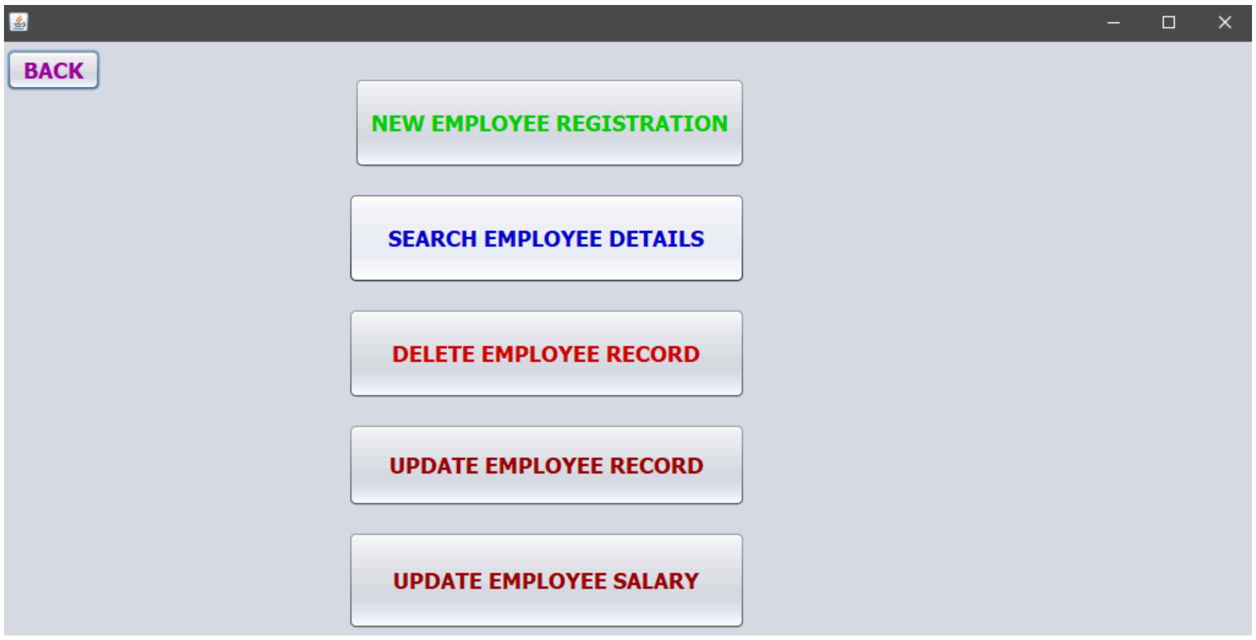
5.2 ADMIN LOGIN



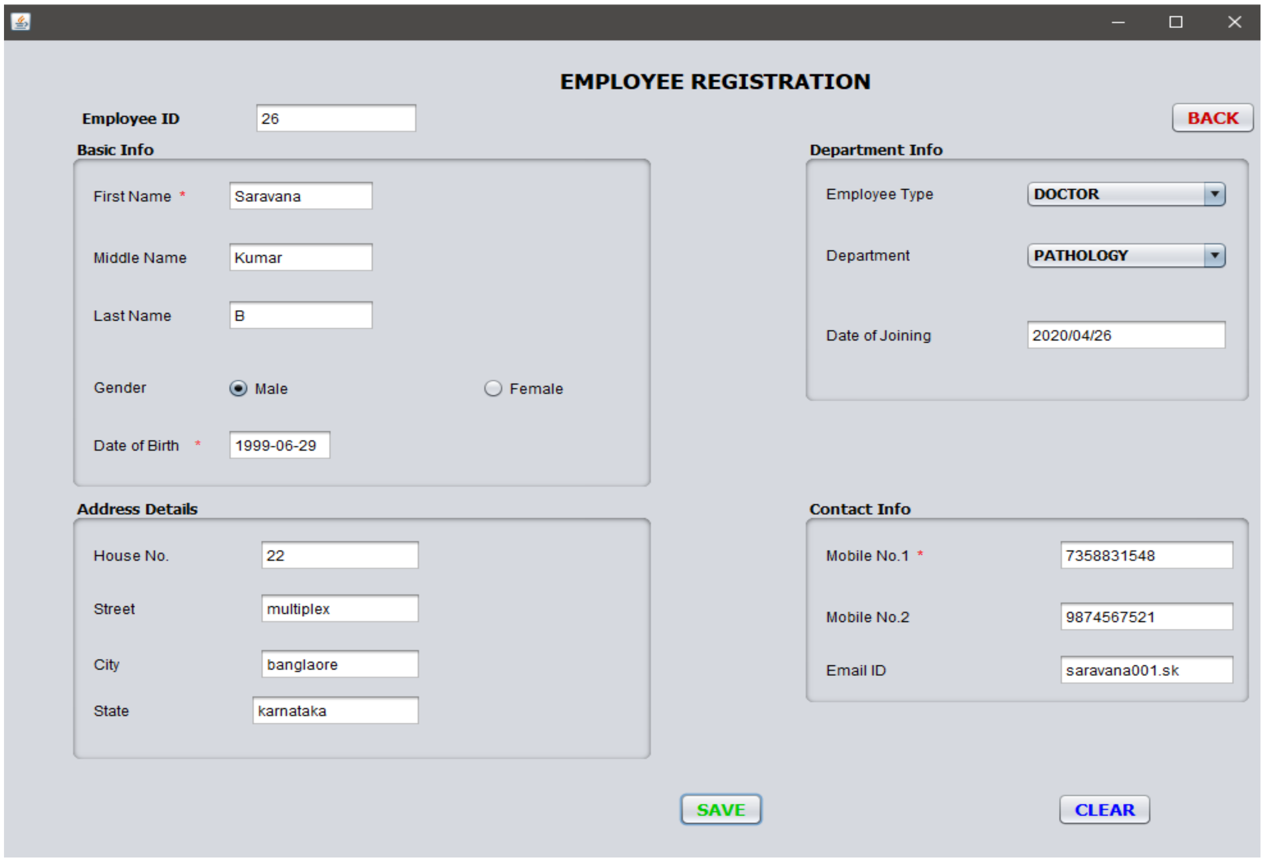
User ID Root

Password *****

5.3 ADMIN LOGGED IN PAGE



5.4 NEW EMPLOYEE


EMPLOYEE REGISTRATION

Employee ID: 26

Basic Info

First Name *	Saravana
Middle Name	Kumar
Last Name	B
Gender	[✓ Male] Female
Date of Birth *	1999-06-29

Department Info

Employee Type	DOCTOR
Department	PATHOLOGY
Date of Joining	2020/04/26

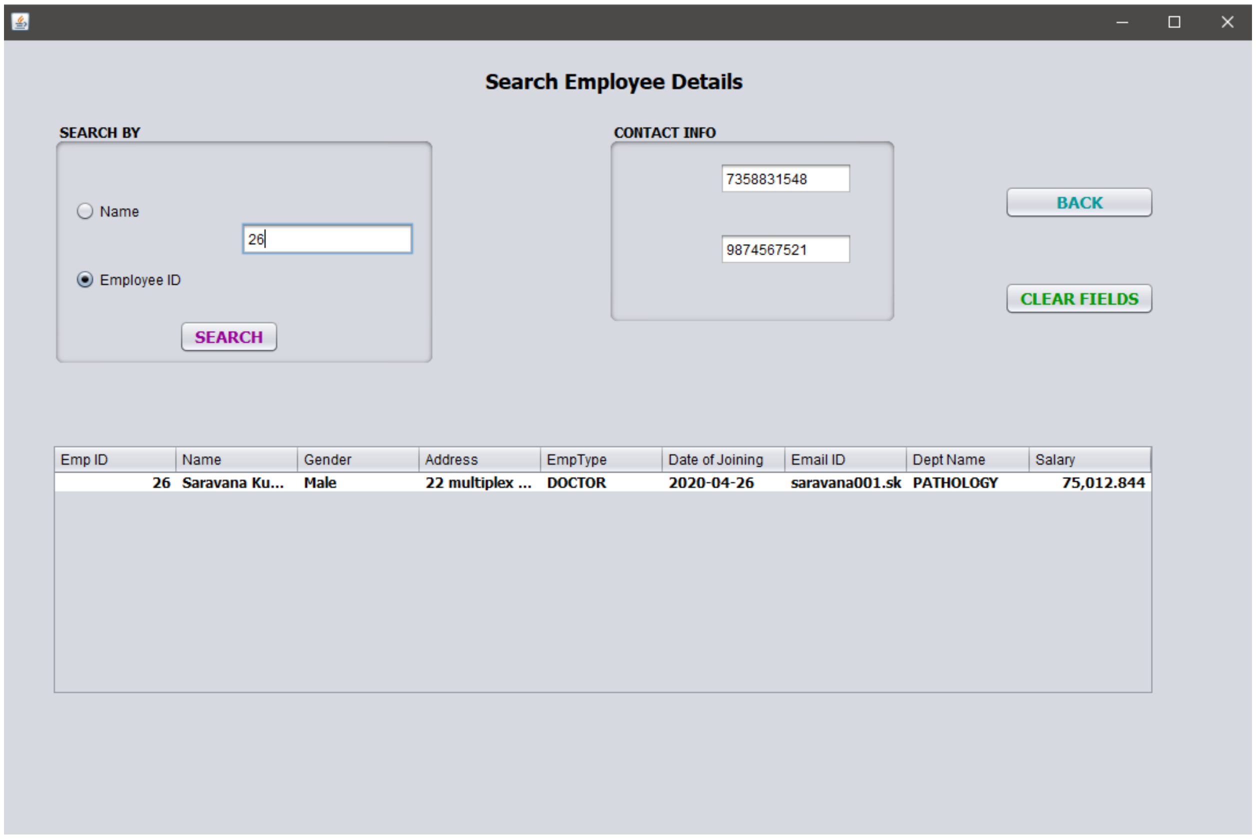
Address Details

House No.	22
Street	multiplex
City	banglaore
State	karnataka

Contact Info

Mobile No.1 *	7358831548
Mobile No.2	9874567521
Email ID	saravana001.sk

5.5 SEARCH EMPLOYEE



Search Employee Details

SEARCH BY

○ Name

● Employee ID

26

SEARCH

CONTACT INFO

7358831548

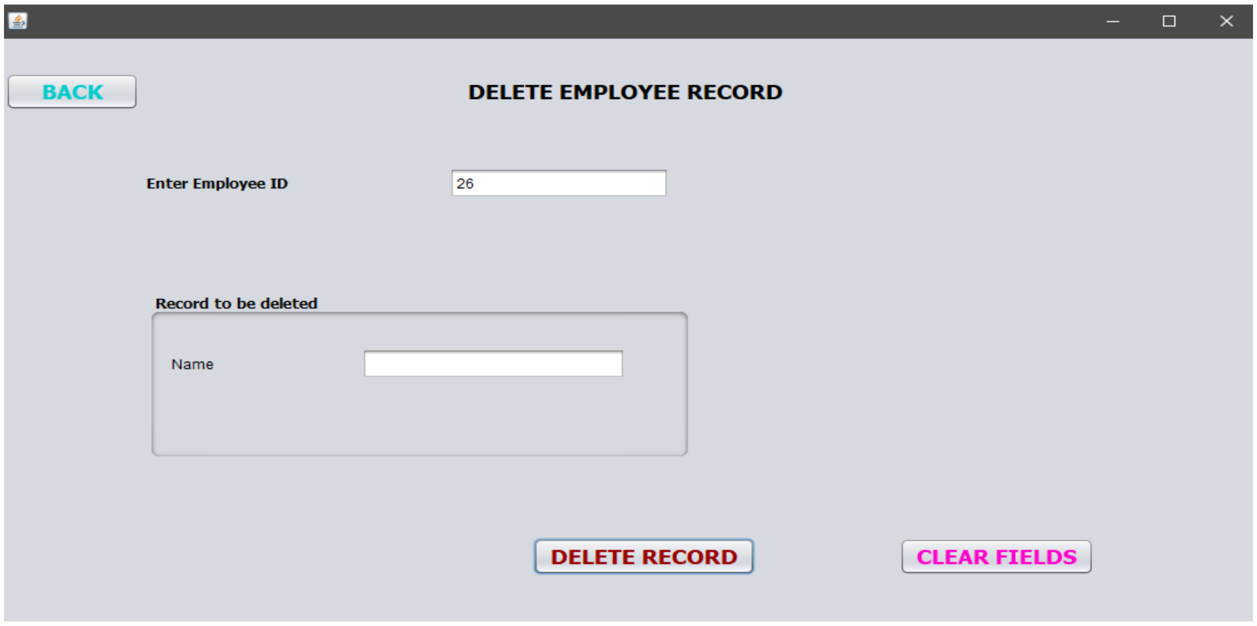
9874567521

BACK

CLEAR FIELDS

Emp ID	Name	Gender	Address	EmpType	Date of Joining	Email ID	Dept Name	Salary
26	Saravana Ku...	Male	22 multiplex ...	DOCTOR	2020-04-26	saravana001.sk	PATHOLOGY	75,012.844

5.6 DELETE EMPLOYEE



BACK

DELETE EMPLOYEE RECORD

Enter Employee ID 26

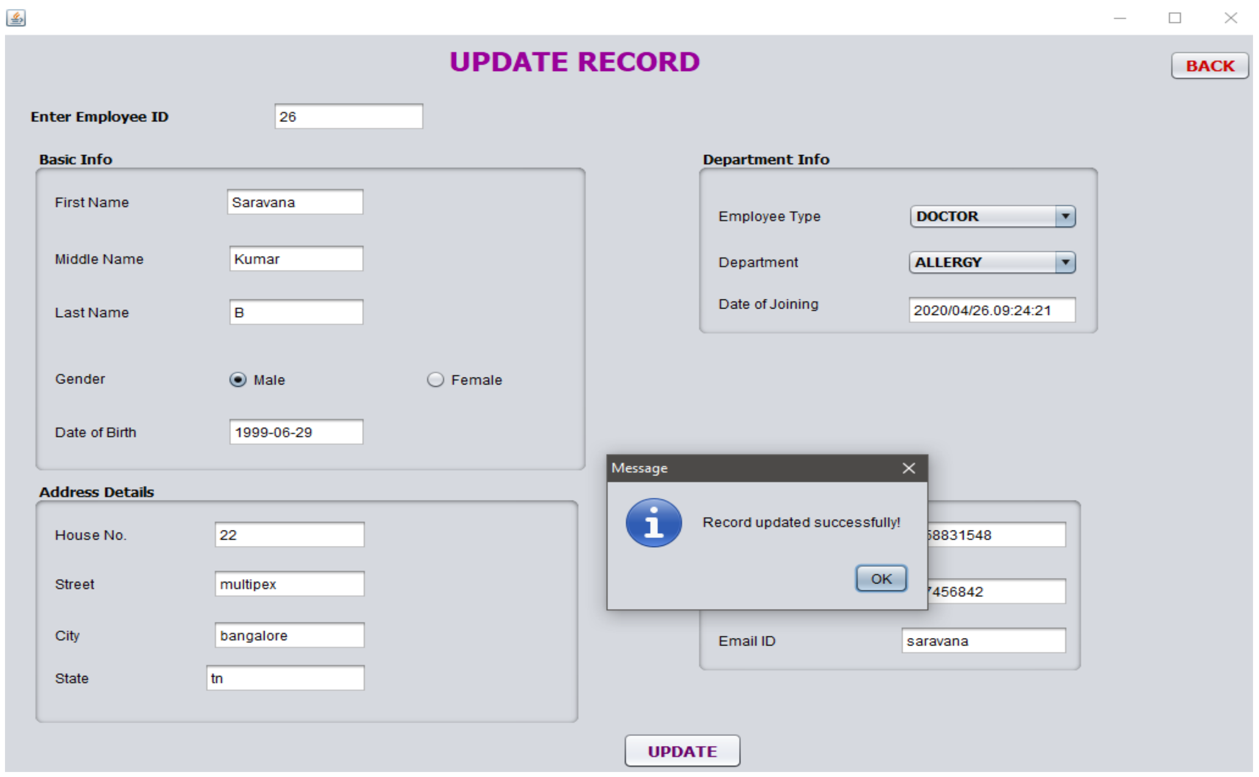
Record to be deleted

Name

DELETE RECORD

CLEAR FIELDS

5.7 UPDATE EMPLOYEE



UPDATE RECORD

BACK

Enter Employee ID 26

Basic Info

First Name	Saravana
Middle Name	Kumar
Last Name	B
Gender	[✓ Male] Female
Date of Birth	1999-06-29

Department Info

Employee Type	DOCTOR
Department	ALLERGY
Date of Joining	2020/04/26.09:24:21

Address Details

House No.	22
Street	multipex
City	bangalore
State	tn

	58831548
	7456842
Email ID	saravana

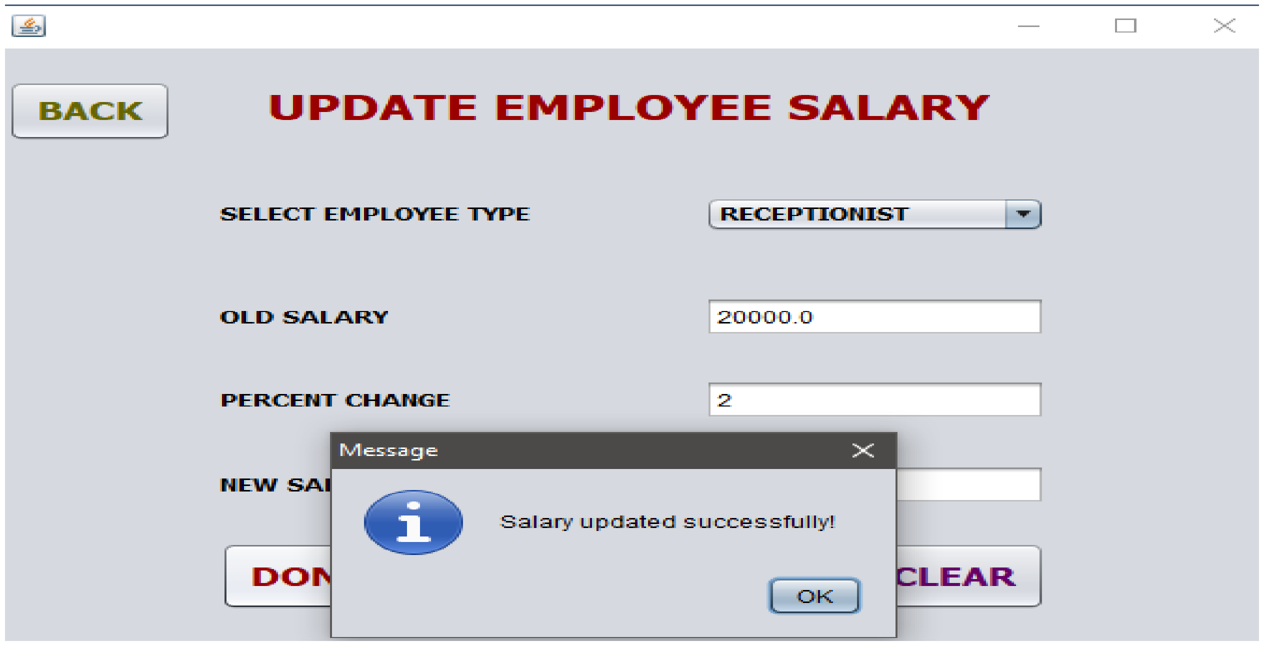
Message

Record updated successfully!

OK

UPDATE

5.8 UPDATE SALARY



BACK

UPDATE EMPLOYEE SALARY

SELECT EMPLOYEE TYPE	RECEPTIONIST
OLD SALARY	20000.0
PERCENT CHANGE	2
NEW SAL	

DON

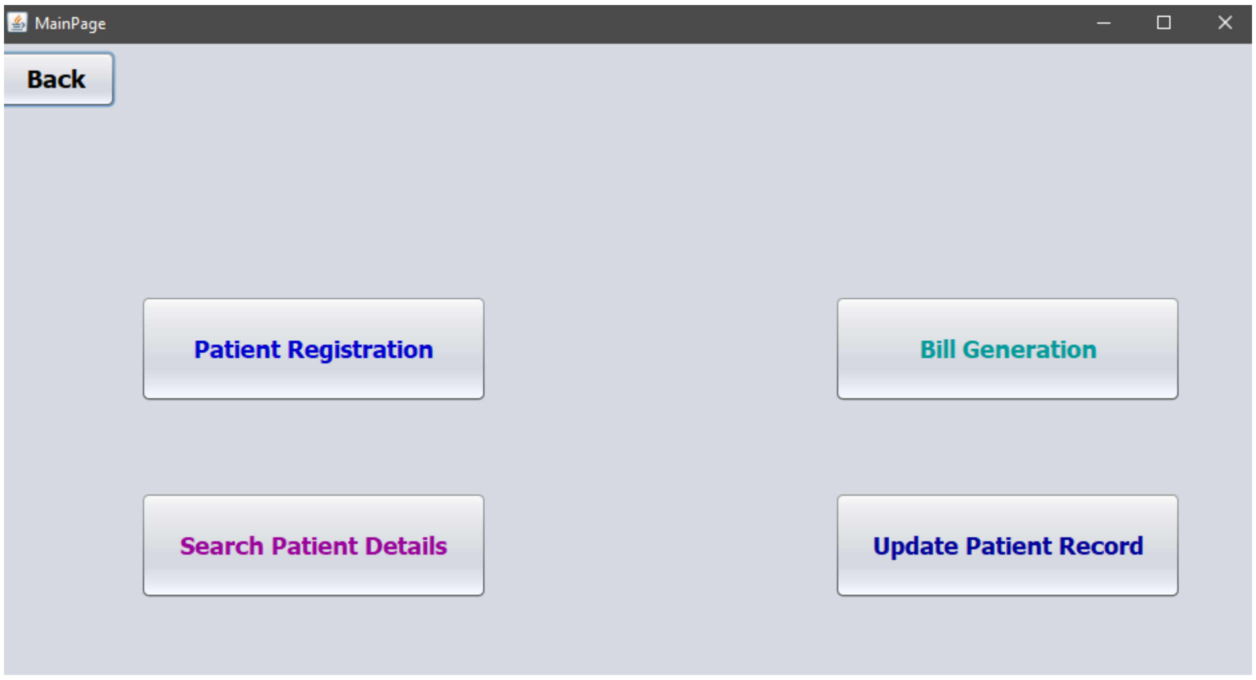
CLEAR

Message

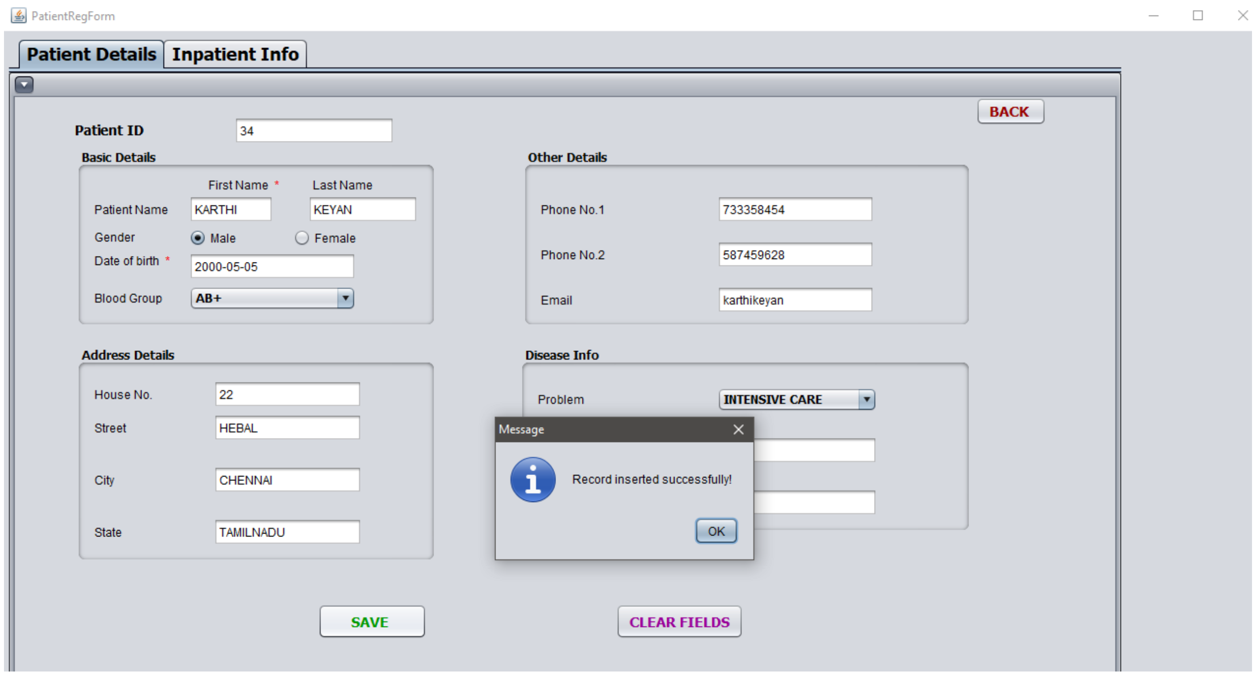
Salary updated successfully!

OK

5.9 RECEPTIONIST PANEL



5.9 PATIENT REGISTRATION



Patient ID 34

Basic Details

	First Name *	Last Name
Patient Name	KARTHI	KEYAN
Gender	[✓ Male]	Female
Date of birth *	2000-05-05	
Blood Group	AB+	

Other Details

Phone No.1	733358454
Phone No.2	587459628
Email	karthikeyan

Address Details

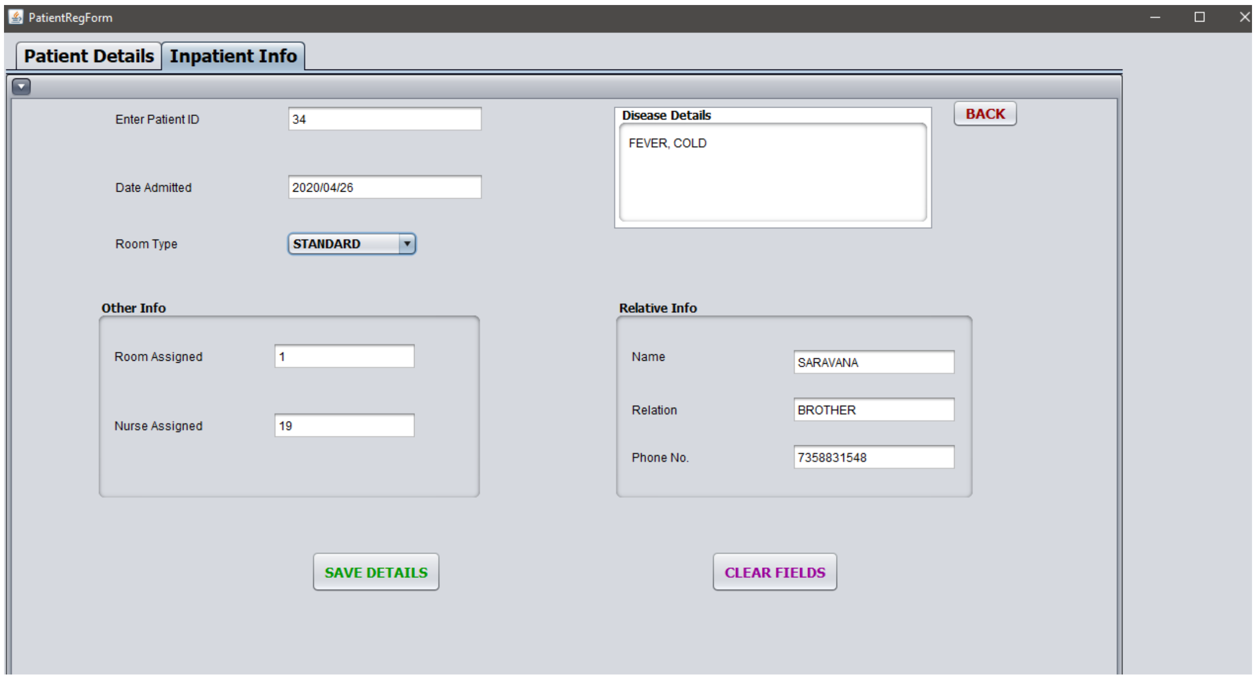
House No.	22
Street	HEBAL
City	CHENNAI
State	TAMILNADU

Disease Info

Problem	INTENSIVE CARE

Record inserted successfully!

5.10 IN_PATIENT DETAILS



Enter Patient ID	34
Date Admitted	2020/04/26
Room Type	STANDARD

Disease Details

FEVER, COLD

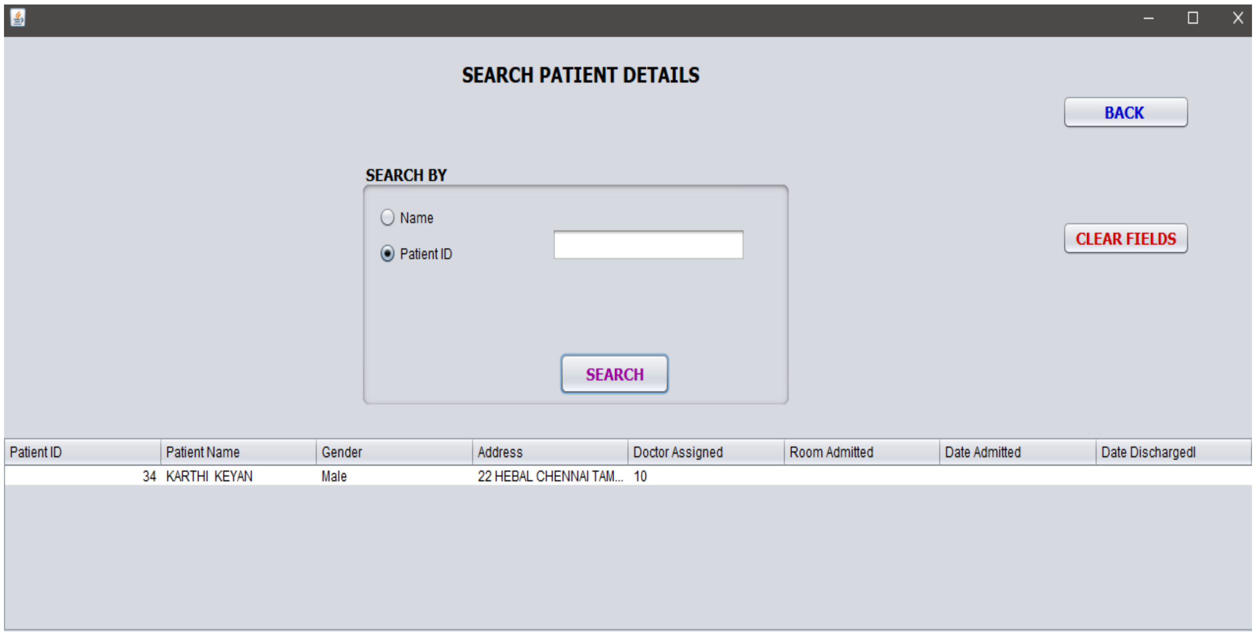
Other Info

Room Assigned	1
Nurse Assigned	19

Relative Info

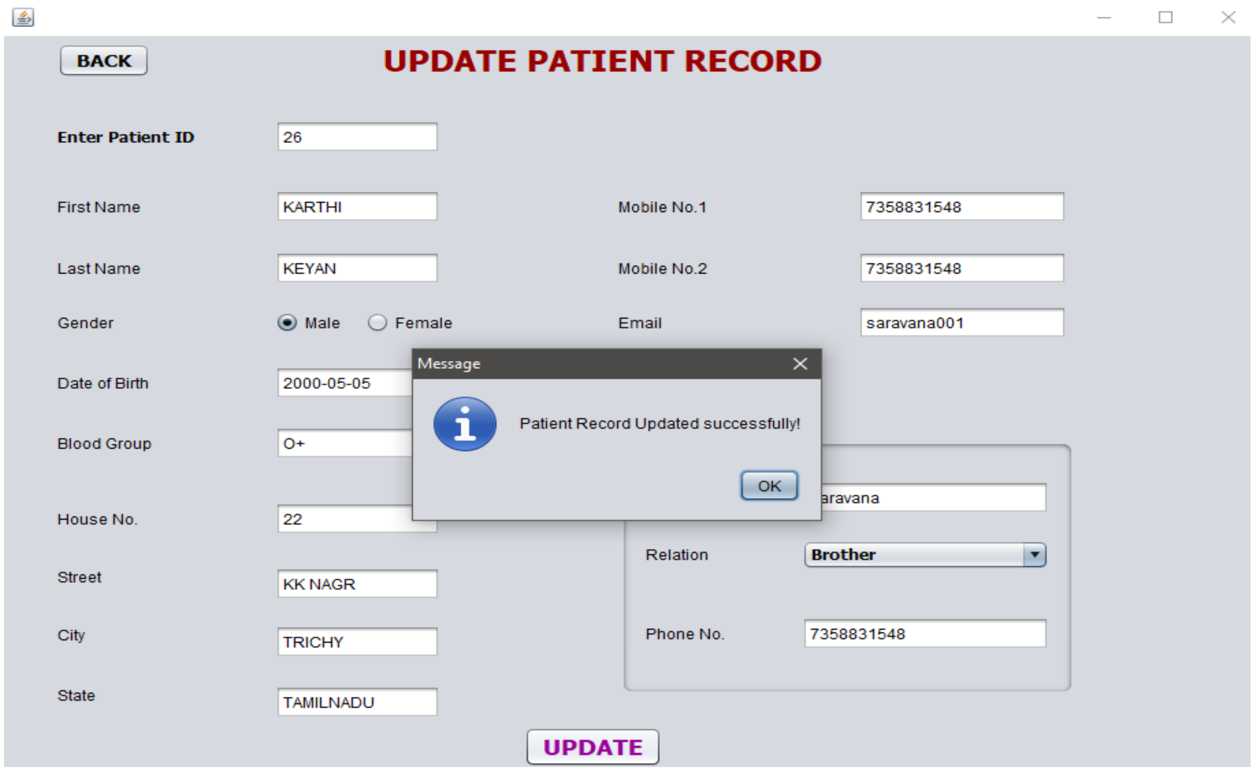
Name	SARAVANA
Relation	BROTHER
Phone No.	7358831548

5.11 SEARCHING PATIENT



Patient ID	Patient Name	Gender	Address	Doctor Assigned	Room Admitted	Date Admitted	Date Dischargedl
34	KARTHI KEYAN	Male	22 HEBAL CHENNAI TAM...	10			

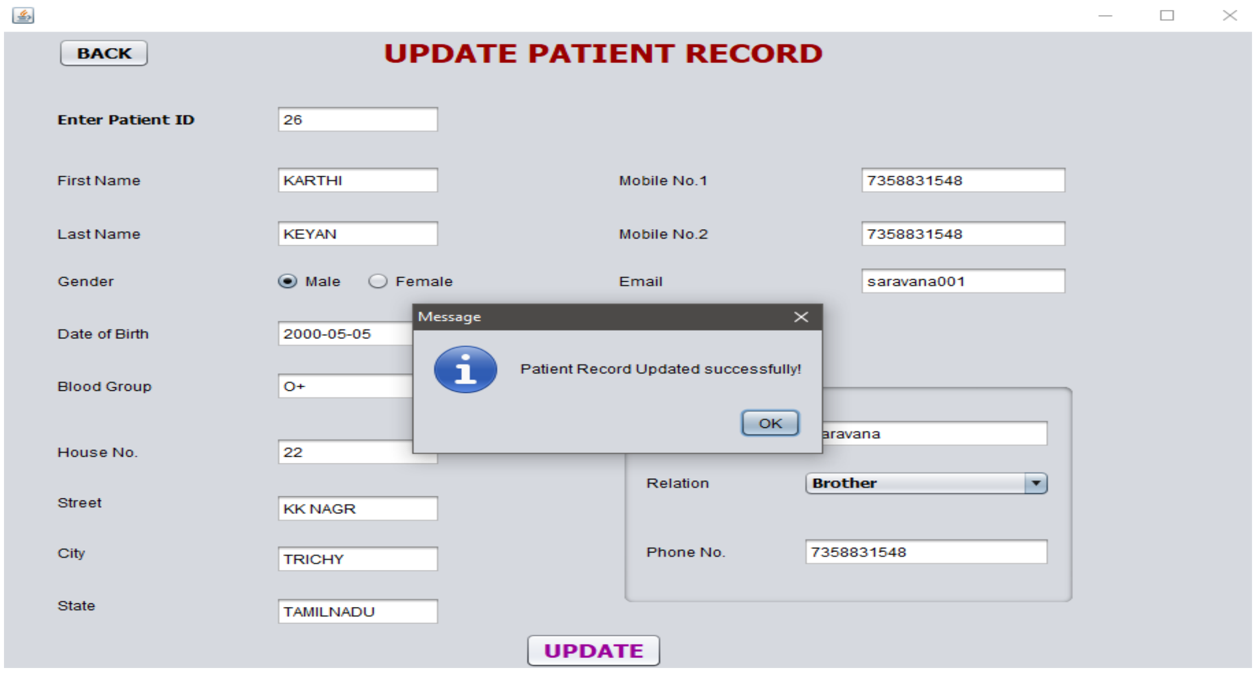
5.11 UPDATE PATIENT



Enter Patient ID	26		
First Name	KARTHI	Mobile No.1	7358831548
Last Name	KEYAN	Mobile No.2	7358831548
Gender	[✓ Male] Female	Email	saravana001
Date of Birth	2000-05-05		
Blood Group	O+		aravana
House No.	22	Relation	Brother
Street	KK NAGR	Phone No.	7358831548
City	TRICHY		
State	TAMILNADU		

Patient Record Updated successfully!

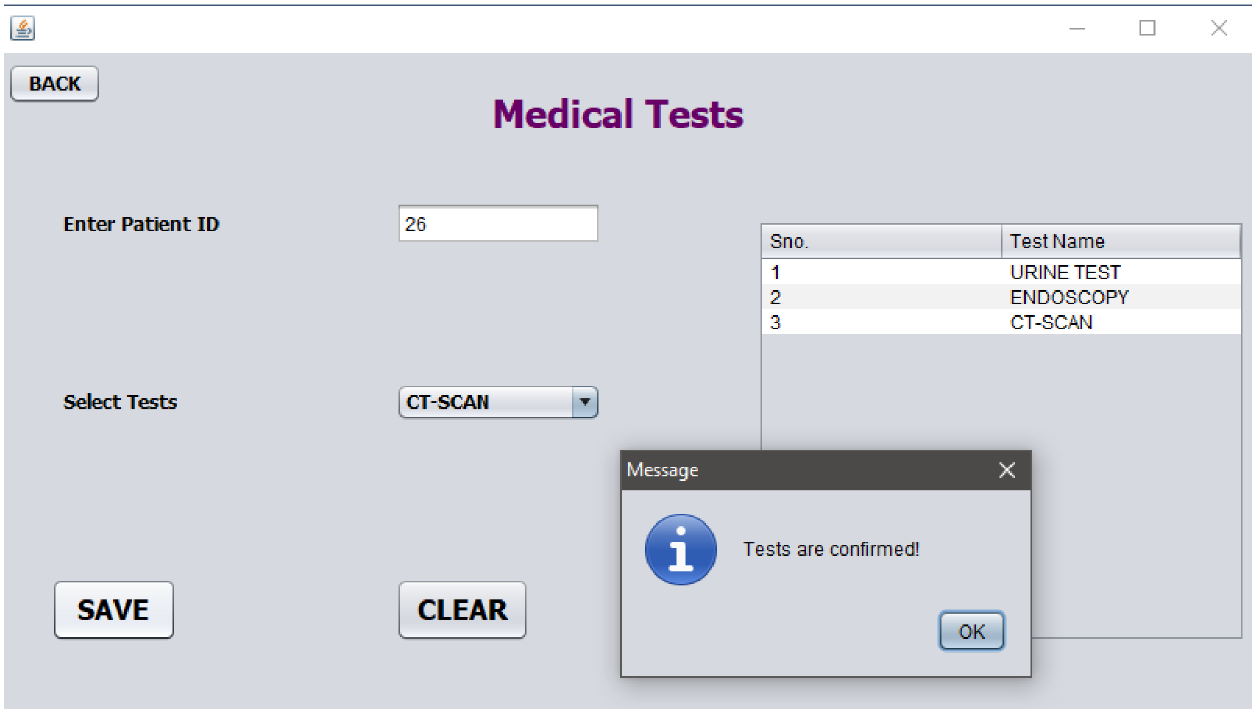
5.12 MEDICINE



Enter Patient ID	26		
First Name	KARTHI	Mobile No.1	7358831548
Last Name	KEYAN	Mobile No.2	7358831548
Gender	[✓ Male] Female	Email	saravana001
Date of Birth	2000-05-05		
Blood Group	O+		aravana
House No.	22	Relation	Brother
Street	KK NAGR	Phone No.	7358831548
City	TRICHY		
State	TAMILNADU		

Patient Record Updated successfully!

5.13 MEDICINE TEST



Medical Tests

Enter Patient ID 26

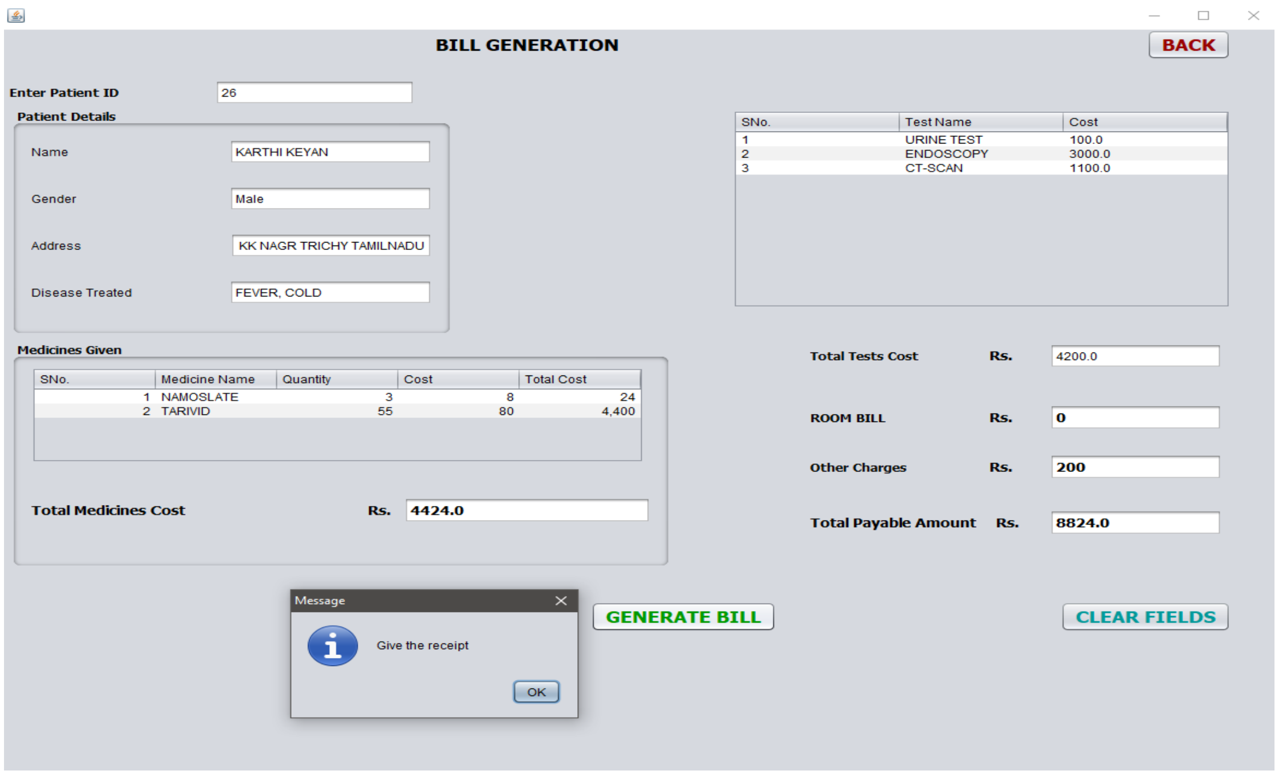
Select Tests CT-SCAN

Sno.	Test Name
1	URINE TEST
2	ENDOSCOPY
3	CT-SCAN

Message

Tests are confirmed!

5.14 BILL GENERATION



BILL GENERATION

Enter Patient ID 26

Patient Details

Name	KARTHI KEYAN
Gender	Male
Address	KK NAGR TRICHY TAMILNADU
Disease Treated	FEVER, COLD

SNo.	Test Name	Cost
1	URINE TEST	100.0
2	ENDOSCOPY	3000.0
3	CT-SCAN	1100.0

Medicines Given

SNo.	Medicine Name	Quantity	Cost	Total Cost
1	NAMOSLATE	3	8	24
2	TARIVID	55	80	4,400

Total Medicines Cost Rs. 4424.0

Total Tests Cost	Rs.	4200.0
ROOM BILL	Rs.	0
Other Charges	Rs.	200
Total Payable Amount	Rs.	8824.0

Message

Give the receipt

CHAPTER 6

CONCLUSION

Taking into account all the mentioned details, we can make the conclusion that inevitable part of the lifecycle of the modern medical institution. It automates numerous daily operations and enables smooth interactions of the users. Developing the hospital system software is a great opportunity to create the distinct, efficient and fast delivering healthcare model. Implementation of this project helps to store all the kinds of records, provide coordination and user communication, implement policies, improve day-to-day operations, arrange the supply chain, manage financial and human resources, and market hospital services. This beneficial decision covers the needs of the patients, staff and hospital authorities and simplifies their interactions. It has become the usual approach to manage the hospital.

CHAPTER 7

REFERENCES

1. Athula Wijewickrama and Soemon Takakuwa, 'Outpatient appointment scheduling in a multi facility system,” Proceedings of the 2008 Winter Simulation Conference, Miami, FL, USA, December 7-10, 2008, IEEE, 2008.
2. Chongiun Yan and .Iiafu Tang. ”Sequential appoinment scheduling problem with general patient choice,” Proceeding of the 11th World Congress on Intelligent Control and Automation, Shenyang, China, 29 June4 July, 2014, IEEE. 2015.
3. Yeo Sy Mey and Suresh Sankaranarayanan, "Near Field Communication Based Patient Appointment," 2013 International Conference on Cloud & Ubiquitous Computing & Emerging Technologies, Pune, India, 15-16 November, 2013, IEEE, 2014.
4. Shamkant B.Navathe, Ramez Elmasri, Fundamentals of Database Systems, 7'” ed. U.S.State: Pearson Education, 2017.
5. Dr.R.Nageswara Rao, Core Python Programming, New Delhi: Dreamtech Press, 2017.
6. Alan D.Moore, Mastering GUI Programming with Python, Birmingham UK: Pack! Publishing, 2017