IPSA SCIENTIA POTESTAS EST

est.2009

HORIZON
CAMPUS

HOSPITAL MANAGEMENT SYSTEM

Object oriented program

Supervisor Name: **Dr. Samanthi Wickramasinghe.**

Name: **H.P. Malith Kalinga Heenatigala.**

This project proposal is submitted in partial fulfilment of the requirement of the Degree of Bachelor of Science (Hons) in Information Technology of the Horizon Campus

Contents

01. Background of the Study/Current System

I am an undergraduate of horizon campus and I am doing the Bachelor of Science (Hons) in Information Technology degree program. As 2nd year student I have to study object oriented programming as a subject. In that case I am doing this project to meet subject materials.

In here we are going to develop a hospital management system using java, MYSQL and NetBeans IDE 8.2.

At present we can identify lot of Management systems are making easy our task. Here we have to develop a management system with our subject materials. So our group members discussed a topic and we choose to build a hospital management system with initial functions.

Then we identify who are the actors involving with this system, what are the main functions we should develop, themes, colours and how to develop. In there we were identify mainly three actors with a hospital. Admin, doctor and patient. With the system patient can do nothing and admin and doctor only can only use it. We decided to develop relevant logins, admins role, Doctors role, hospital information and a flash screen. Here admin can add, delete, edit doctors data and patient data. Doctor can add patient diagnosis data. And delete edit print patients history.

02.Scope of the Project

2.1 Aim(s)

The aim of project is develop a Management System for Hospital.

Present today Management system is a fashion. We aimed to fulfil our project on time and we will try to provide the best possible service with this system.

2.2. Objectives

- To develop a hospital management system with java, MySQL and NetBeans IDE 8.2.
- To provide maximum service to hospital.
- To less time consuming.
- To give a reliable service to patient.
- To identify main functions and build them.
- To provide a best possible service with this system.

2.3. Major Processes

Identify main actors who are involving with this system.

- Admin
- Doctor

Identify what are the roles of relevant actor.

- Admin can add patient data.
- Admin can add Doctor Data.
- Admin can delete patient data. □ Admin can delete Doctor Data □ Admin can edit patient data.
- Admin can edit Doctor Data

- Doctor can add patient diagnosis data.
- Doctor can delete patient’s history. □ Doctor can edit patient’s history.
- Doctor can print patient’s history.

Identify main functions with relevant actors.

- Admin can add, delete, and edit patient data.
    - ✓ Patient ID
    - ✓ Name
    - ✓ Phone
    - ✓ Address
    - ✓ Gender
    - ✓ Age
    - ✓ Blood group
    - ✓ Major Disease Differed Earlier

- Admin can add, delete, and edit Doctor Data.
    - ✓ Doctor ID
    - ✓ Name
    - ✓ Phone
    - ✓ Address
    - ✓ Type
    - ✓ Password

- Doctor can add patient diagnosis data.
    - ✓ Symptoms
    - ✓ Diagnosis
    - ✓ Medicines
    - ✓ Ward required
    - ✓ Ward type

- Doctor can delete, edit, print patient’s history.
    - ✓ Symptoms
    - ✓ Diagnosis
    - ✓ Medicines
    - ✓ Ward required
    - ✓ Ward type

Test the functions.

- After coding we want to test the all the functionality of the system.

Re-built and serve the system.

- If any of functionality is missing or having bugs re-build the functions and test again.

03. Software and hardware requirements

Software requirements: NetBeans IDE 8.2

XAMPP

Operating System (Microsoft windows 10)

Languages: java

MySQL

Hardware requirements: laptop (Minimum Intel core i3 processor)

Minimum 4GB ram

Keyboard

Mouse

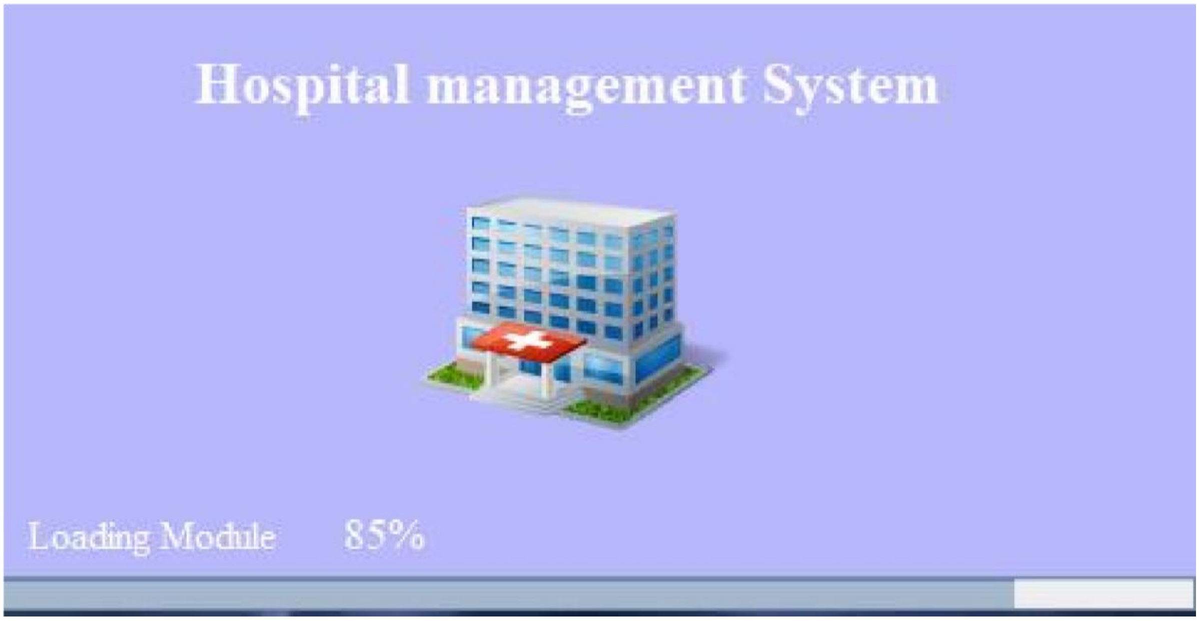
04.External interface requirements

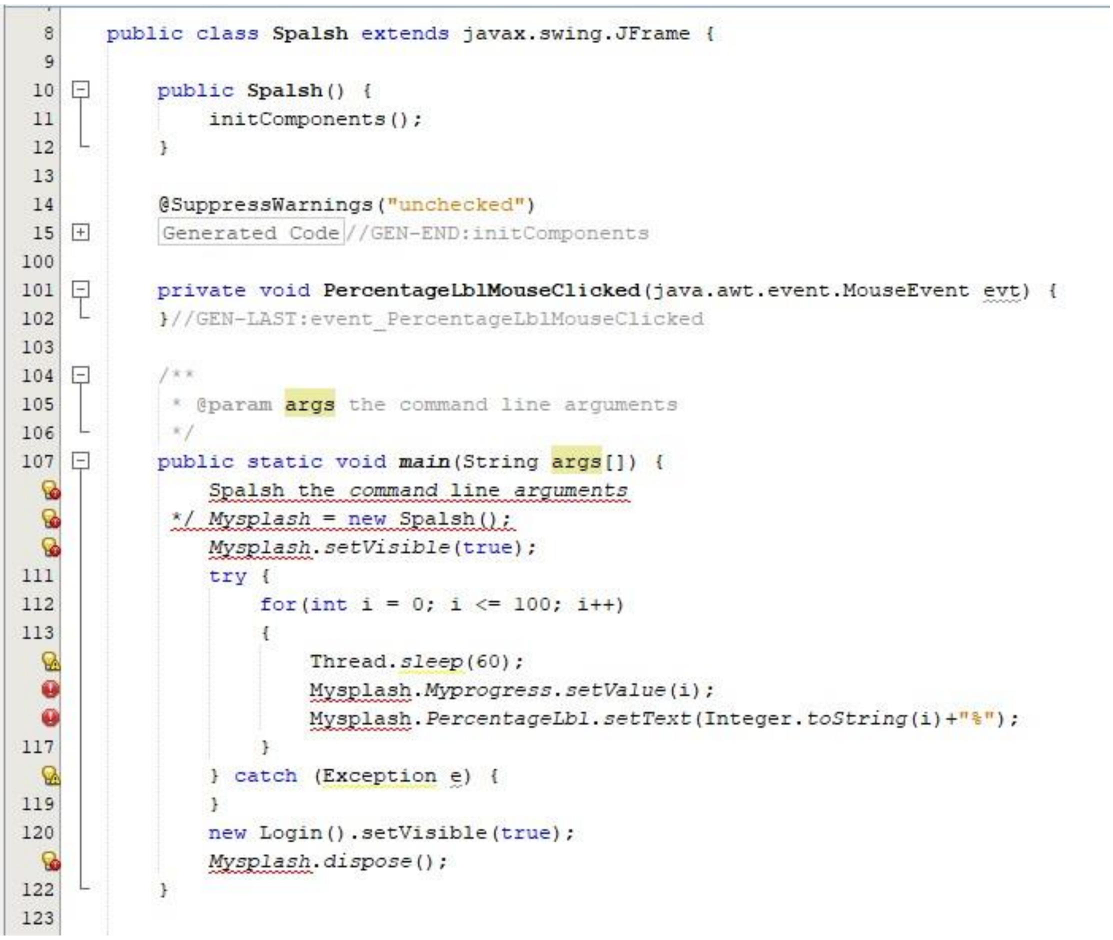
Mainly we are going to build 7 interfaces.

1. Splash screen
2. Login interface
3. Admin – patients data interface
4. Admin – doctor data interface
5. Doctor – add diagnosis interface
6. Doctor – patient history interface
7. Hospital information interface

05.Data capturing interfaces and layouts

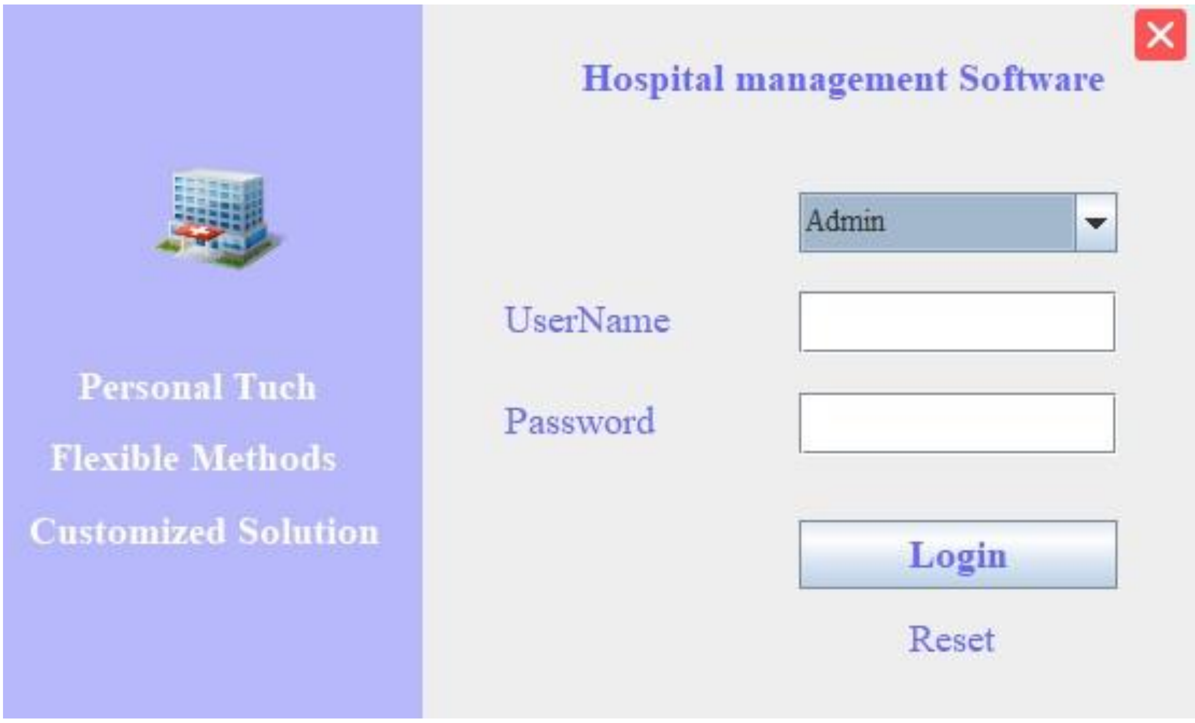
5.1. Splash screen

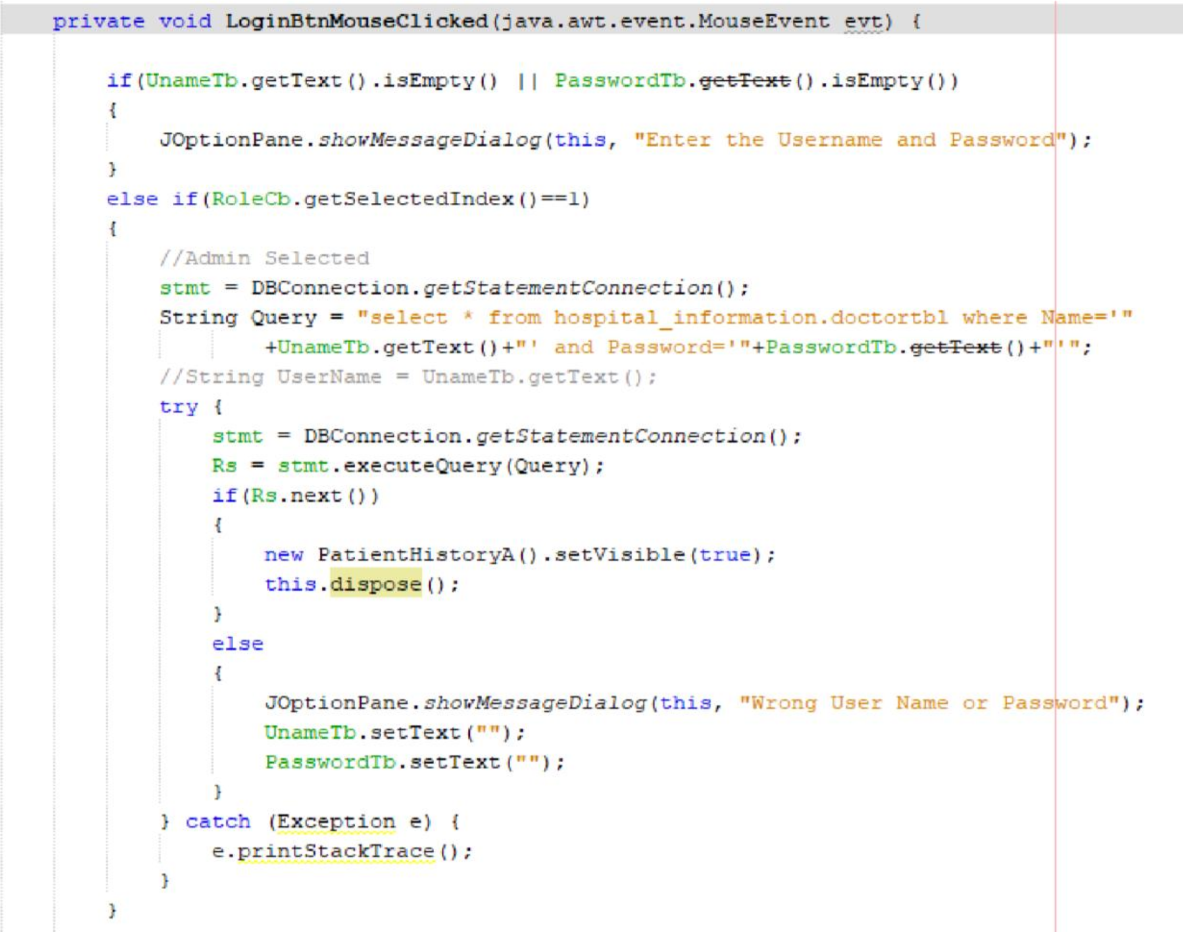




```
public class Spalsh extends javax.swing.JFrame {

    public Spalsh() {
        initComponents();
    }

    @SuppressWarnings("unchecked")
    Generated Code//GEN-END:initComponents

    private void PercentageLblMouseClicked(java.awt.event.MouseEvent evt) {
    }//GEN-LAST:event_PercentageLblMouseClicked

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        Spalsh the command line arguments
     */ Mysplash = new Spalsh();
        Mysplash.setVisible(true);
        try {
            for(int i = 0; i <= 100; i++)
            {
                Thread.sleep(60);
                Mysplash.Myprogress.setValue(i);
                Mysplash.PercentageLbl.setText(Integer.toString(i)+"%");
            }
        } catch (Exception e) {
        }
        new Login().setVisible(true);
        Mysplash.dispose();
    }

```

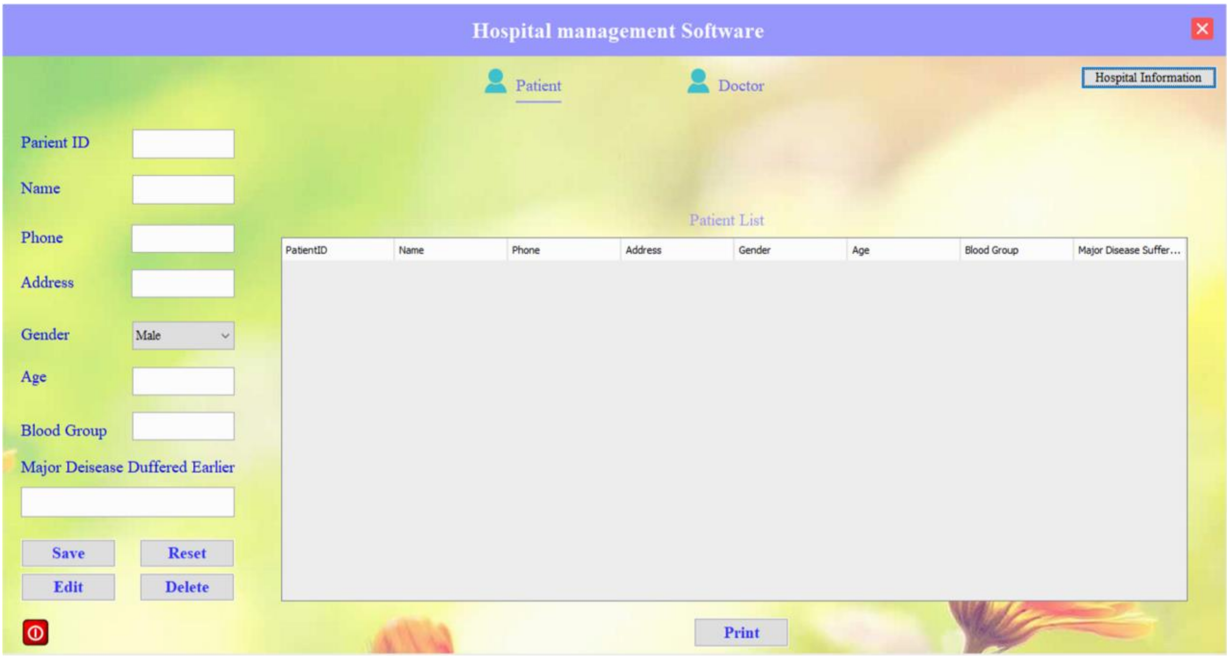
5.2. Login interface

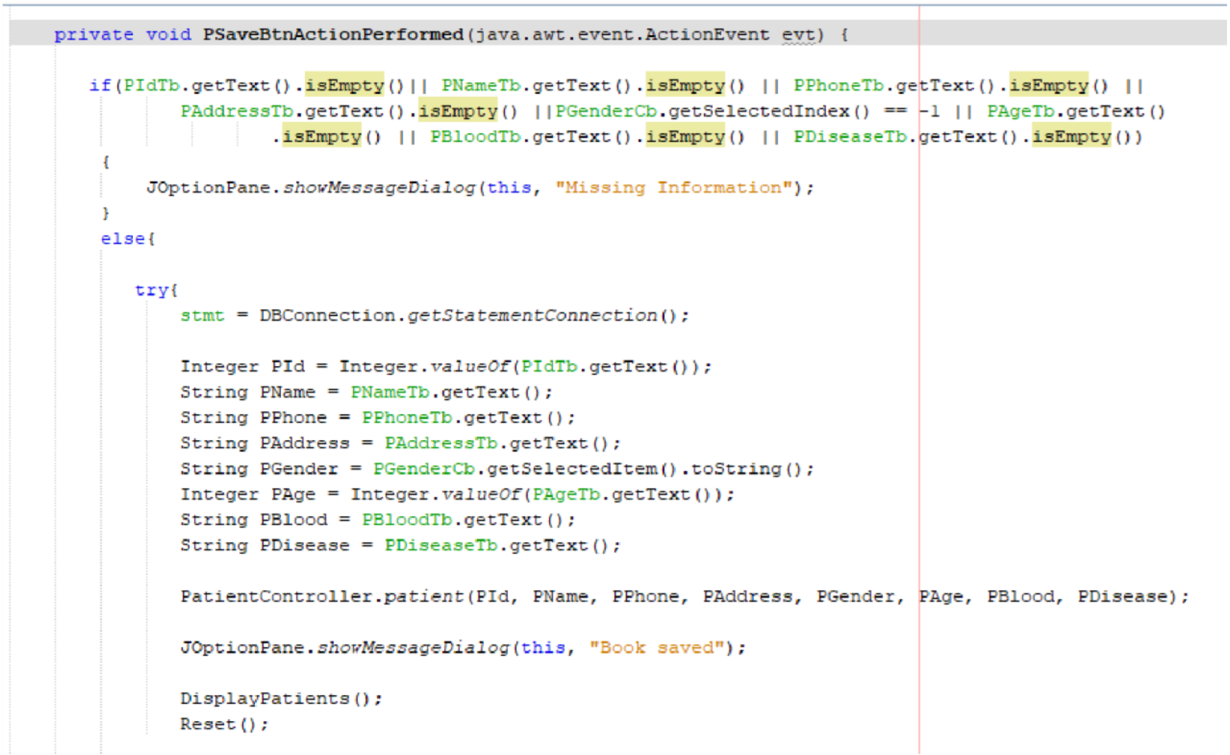




```
private void LoginBtnMouseClicked(java.awt.event.MouseEvent evt) {

    if(UnameTb.getText().isEmpty() || PasswordTb.~~getText~~().isEmpty())
    {
        JOptionPane.showMessageDialog(this, "Enter the Username and Password");
    }
    else if(RoleCb.getSelectedIndex()==1)
    {
        //Admin Selected
        stmt = DBConnection.getStatementConnection();
        String Query = "select * from hospital_information.doctortbl where Name='"
                +UnameTb.getText()+"' and Password='"+PasswordTb.~~getText~~()+"'";
        //String UserName = UnameTb.getText();
        try {
            stmt = DBConnection.getStatementConnection();
            Rs = stmt.executeQuery(Query);
            if(Rs.next())
            {
                new PatientHistoryA().setVisible(true);
                this.dispose();
            }
            else
            {
                JOptionPane.showMessageDialog(this, "Wrong User Name or Password");
                UnameTb.setText("");
                PasswordTb.setText("");
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
```

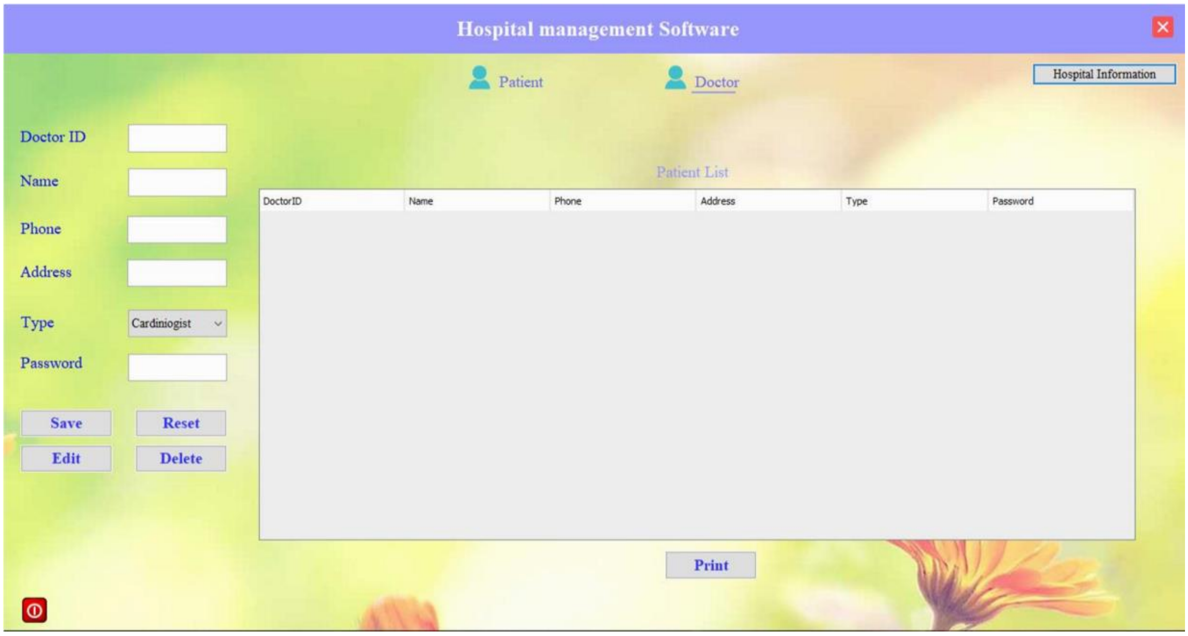
53. Admin – patients data interface



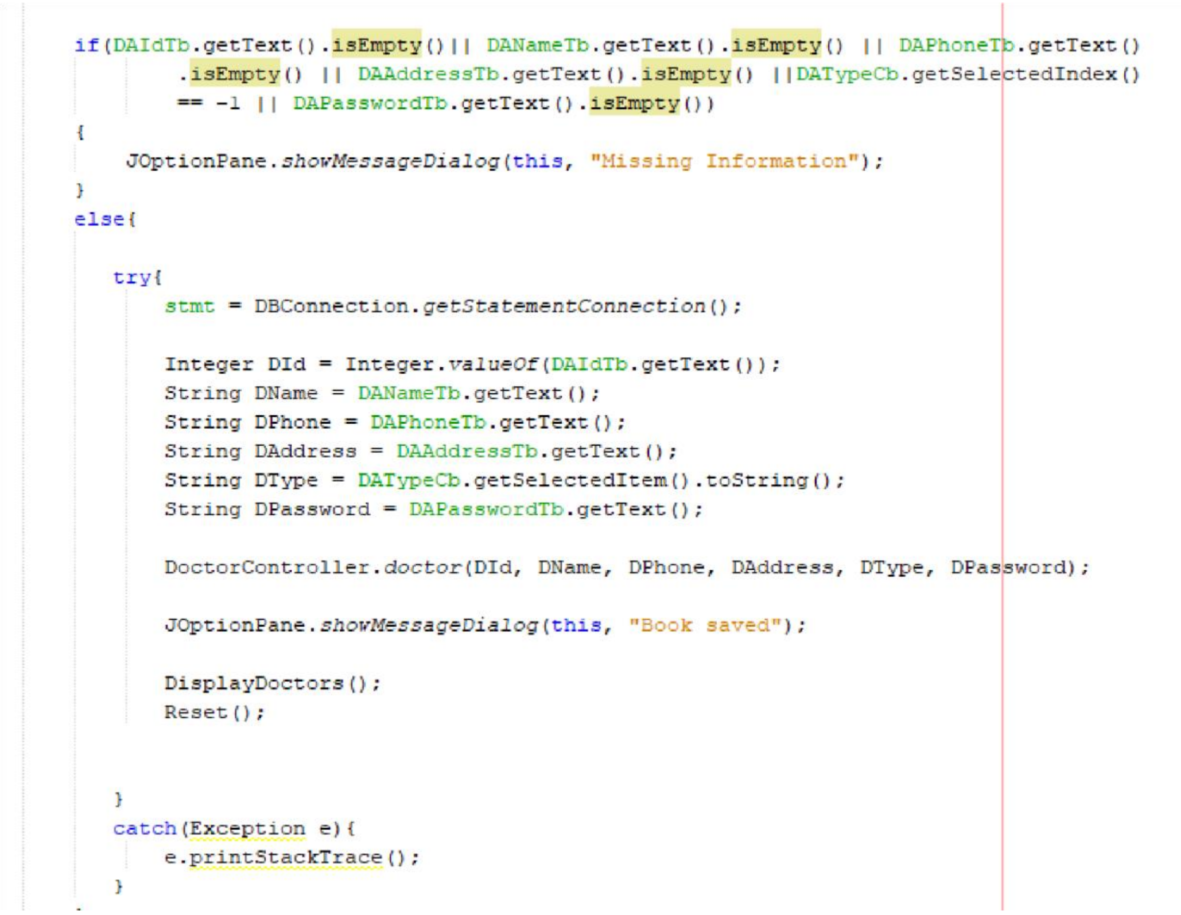


```
    private void PSaveBtnActionPerformed(java.awt.event.ActionEvent evt) {

        if(PIdTb.getText().isEmpty()|| PNameTb.getText().isEmpty() || PPhoneTb.getText().isEmpty() ||
                PAddressTb.getText().isEmpty() ||PGenderCb.getSelectedIndex() == -1 || PAgeTb.getText()
                        .isEmpty() || PBloodTb.getText().isEmpty() || PDiseaseTb.getText().isEmpty())
         {
             JOptionPane.showMessageDialog(this, "Missing Information");
         }
         else{

             try{
                 stmt = DBConnection.getStatementConnection();

                 Integer PId = Integer.valueOf(PIdTb.getText());
                 String PName = PNameTb.getText();
                 String PPhone = PPhoneTb.getText();
                 String PAddress = PAddressTb.getText();
                 String PGender = PGenderCb.getSelectedItem().toString();
                 Integer PAge = Integer.valueOf(PAgeTb.getText());
                 String PBlood = PBloodTb.getText();
                 String PDisease = PDiseaseTb.getText();

                 PatientController.patient(PId, PName, PPhone, PAddress, PGender, PAge, PBlood, PDisease);

                 JOptionPane.showMessageDialog(this, "Book saved");

                 DisplayPatients();
                 Reset();
```

5.4. Admin – doctor data interface

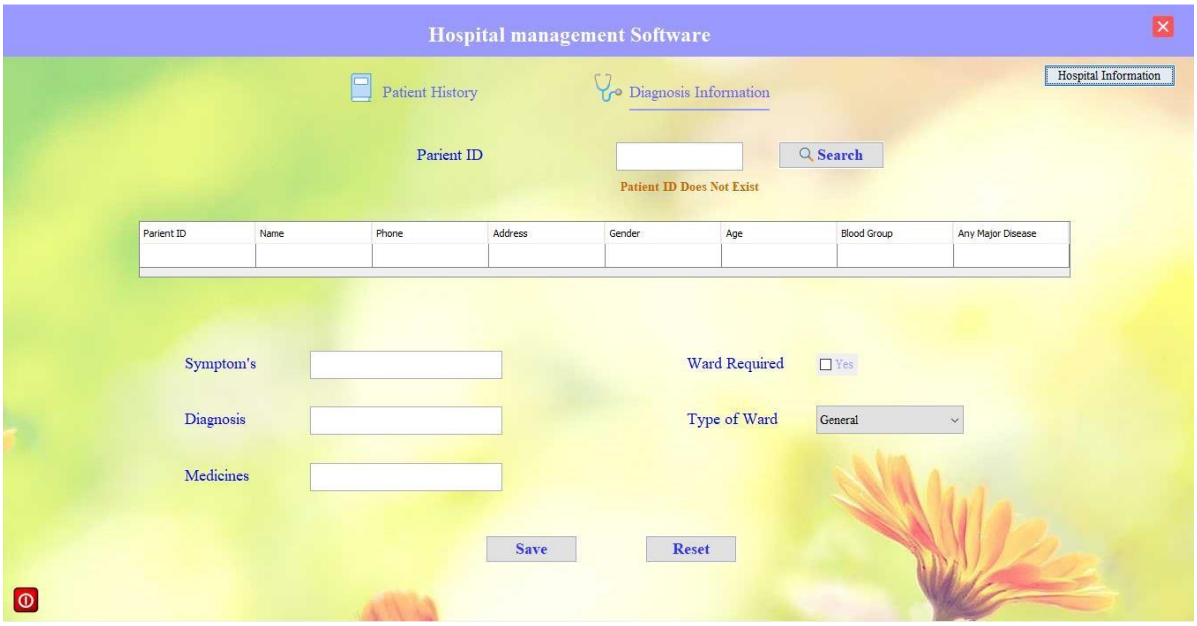


Type: Cardiniogist



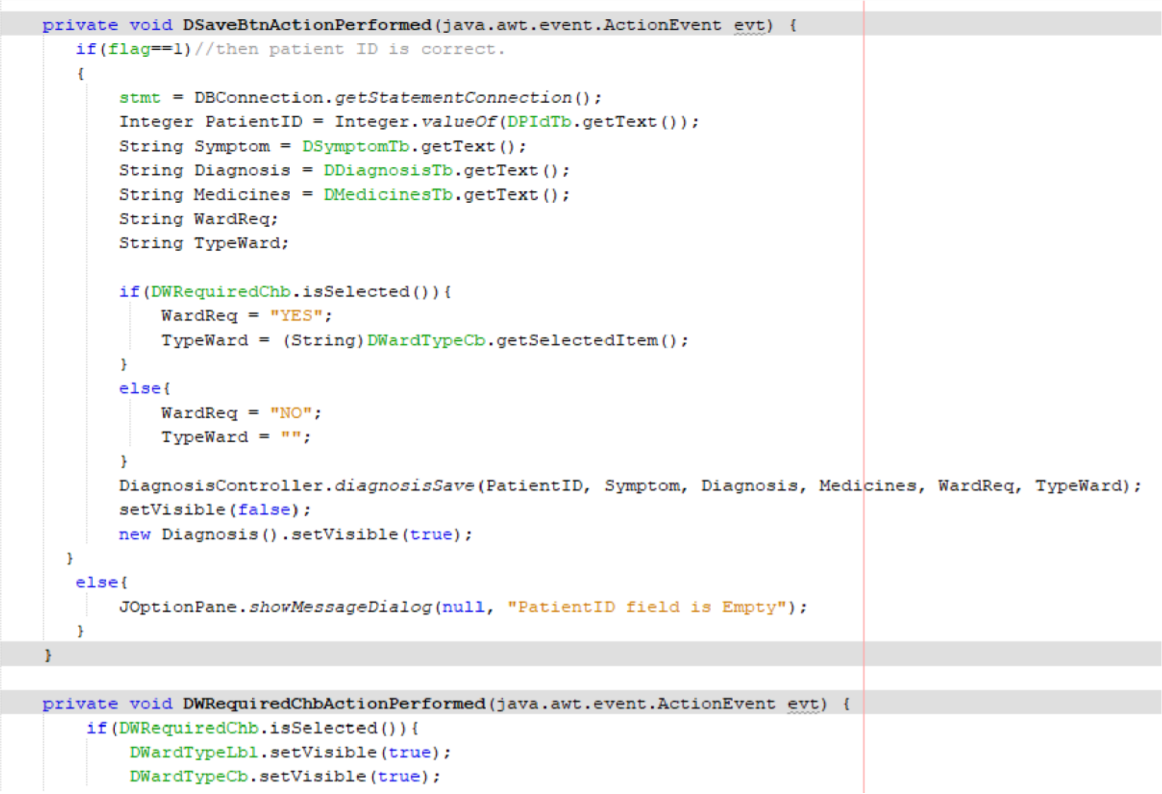
```
if(DAIdTb.getText().isEmpty()|| DANameTb.getText().isEmpty() || DAPhoneTb.getText()
        .isEmpty() || DAAddressTb.getText().isEmpty() ||DATypeCb.getSelectedIndex()
        == -1 || DAPasswordTb.getText().isEmpty())
{
    JOptionPane.showMessageDialog(this, "Missing Information");
}
else{

   try{
       stmt = DBConnection.getStatementConnection();

       Integer DId = Integer.valueOf(DAIdTb.getText());
       String DName = DANameTb.getText();
       String DPhone = DAPhoneTb.getText();
       String DAddress = DAAddressTb.getText();
       String DType = DATypeCb.getSelectedItem().toString();
       String DPassword = DAPasswordTb.getText();

       DoctorController.doctor(DId, DName, DPhone, DAddress, DType, DPassword);

       JOptionPane.showMessageDialog(this, "Book saved");

       DisplayDoctors();
       Reset();


   }
   catch(Exception e){
       e.printStackTrace();
   }
```

5.5 Doctor – add diagnosis interface



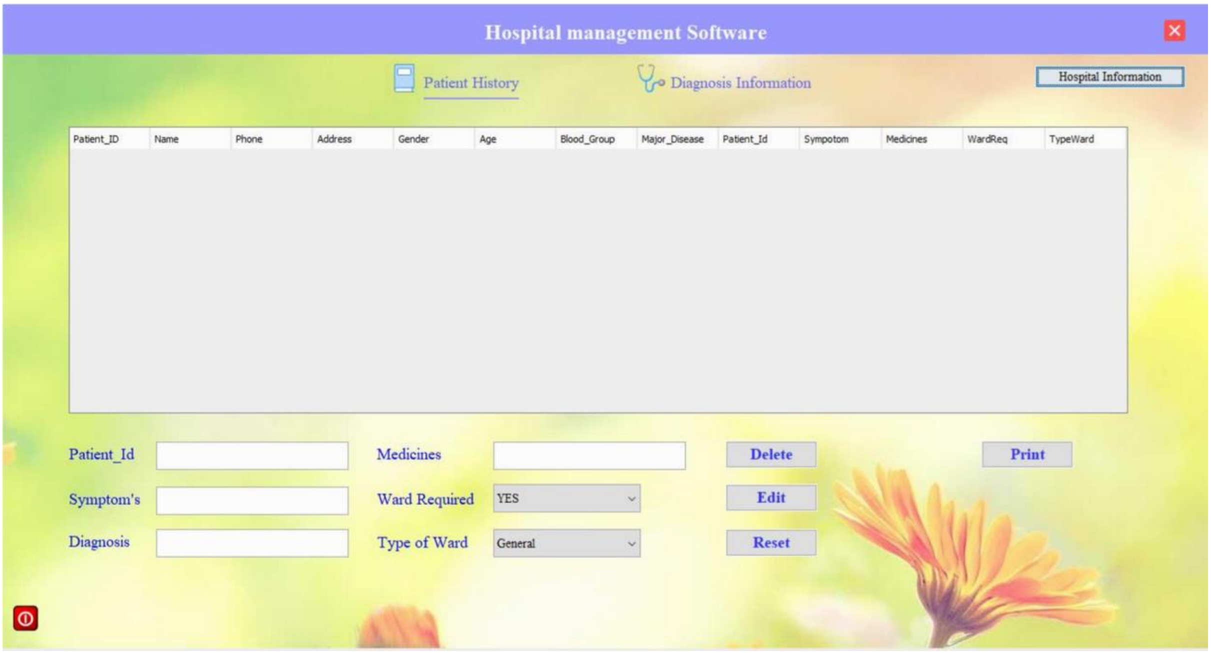
Patient ID Does Not Exist

Parient ID	Name	Phone	Address	Gender	Age	Blood Group	Any Major Disease



```
private void DSaveBtnActionPerformed(java.awt.event.ActionEvent evt) {
    if(flag==1)//then patient ID is correct.
    {
        stmt = DBConnection.getStatementConnection();
        Integer PatientID = Integer.valueOf(DPIdTb.getText());
        String Symptom = DSymptomTb.getText();
        String Diagnosis = DDiagnosisTb.getText();
        String Medicines = DMedicinesTb.getText();
        String WardReq;
        String TypeWard;

        if(DWRequiredChb.isSelected()){
            WardReq = "YES";
            TypeWard = (String)DWardTypeCb.getSelectedItem();
        }
        else{
            WardReq = "NO";
            TypeWard = "";
        }
        DiagnosisController.diagnosisSave(PatientID, Symptom, Diagnosis, Medicines, WardReq, TypeWard);
        setVisible(false);
        new Diagnosis().setVisible(true);
    }
    else{
        JOptionPane.showMessageDialog(null, "PatientID field is Empty");
    }
}

private void DWRequiredChbActionPerformed(java.awt.event.ActionEvent evt) {
    if(DWRequiredChb.isSelected()){
        DWardTypeLbl.setVisible(true);
        DWardTypeCb.setVisible(true);
```

5.5 Doctor – patient history interface



Patient_ID	Name	Phone	Address	Gender	Age	Blood_Group	Major_Disease	Patient_Id	Sympotom	Medicines	WardReq	TypeWard

Patient_Id

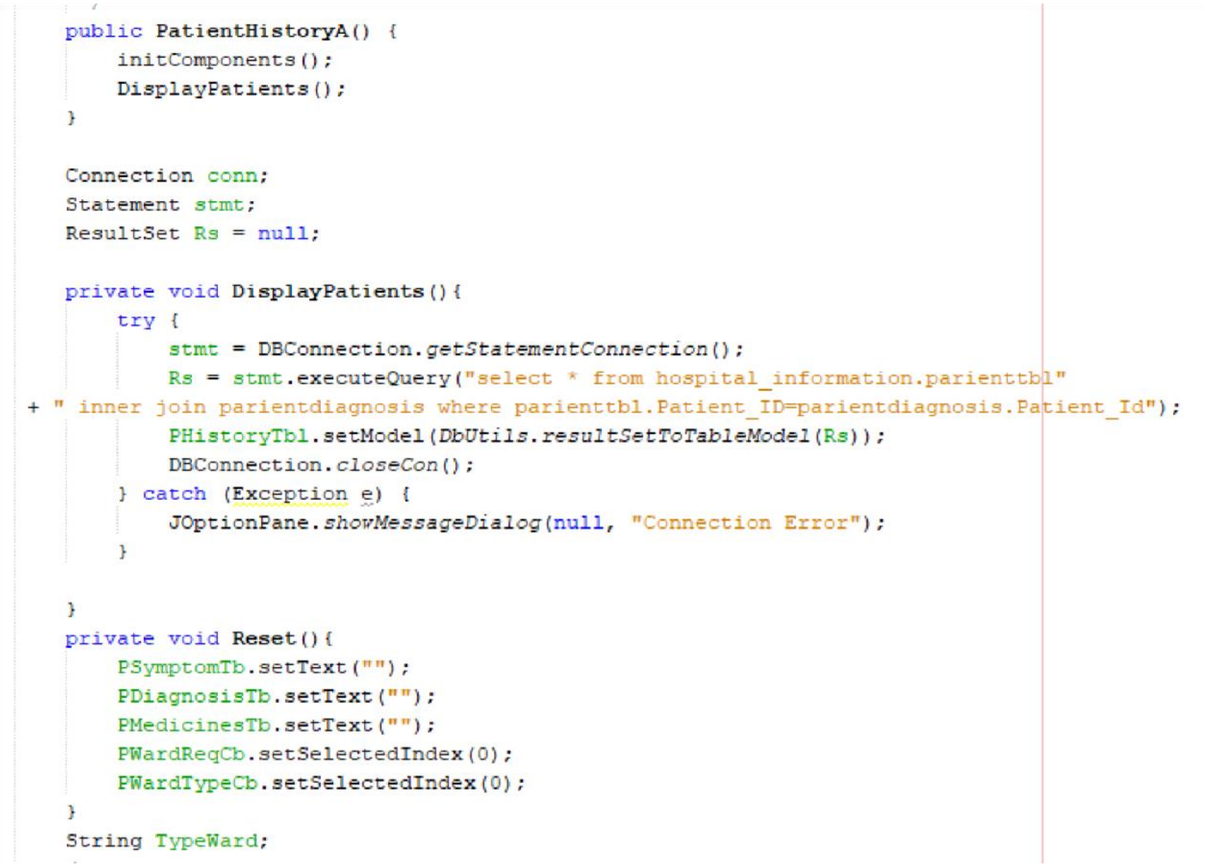
Symptom's

Diagnosis

Medicines

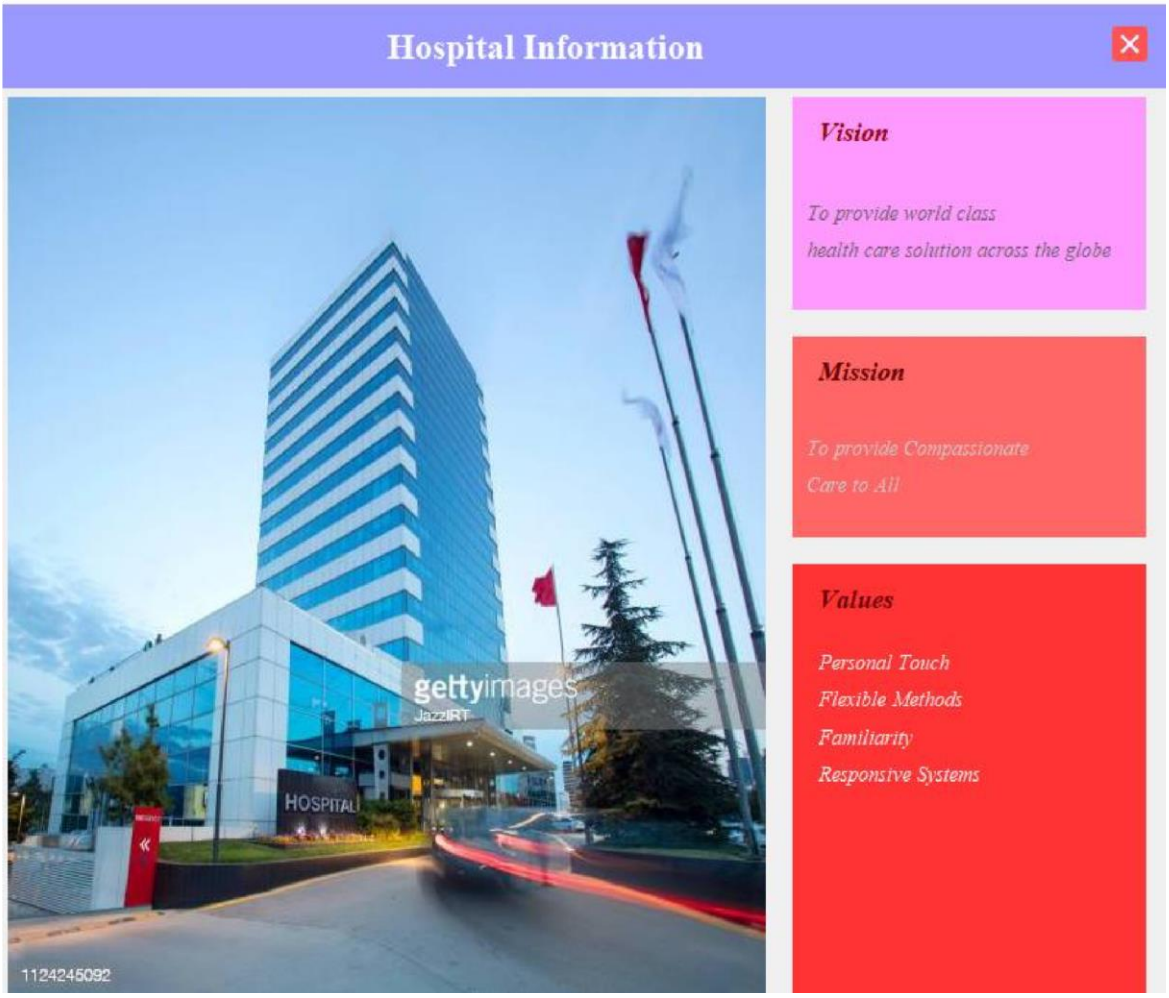
Ward Required: YES

Type of Ward: General



```
    public PatientHistoryA() {
        initComponents();
        DisplayPatients();
    }

    Connection conn;
    Statement stmt;
    ResultSet Rs = null;

    private void DisplayPatients(){
        try {
            stmt = DBConnection.getStatementConnection();
            Rs = stmt.executeQuery("select * from hospital_information.parienttbl"
+ " inner join parientdiagnosis where parienttbl.Patient_ID=parientdiagnosis.Patient_Id");
            PHistoryTbl.setModel(DbUtils.resultSetToTableModel(Rs));
            DBConnection.closeCon();
        } catch (Exception e) {
            JOptionPane.showMessageDialog(null, "Connection Error");
        }

    }
    private void Reset(){
        PSymptomTb.setText("");
        PDiagnosisTb.setText("");
        PMedicinesTb.setText("");
        PWardReqCb.setSelectedIndex(0);
        PWardTypeCb.setSelectedIndex(0);
    }
    String TypeWard;
```

5.6 Hospital information interface



Hospital Information

Vision

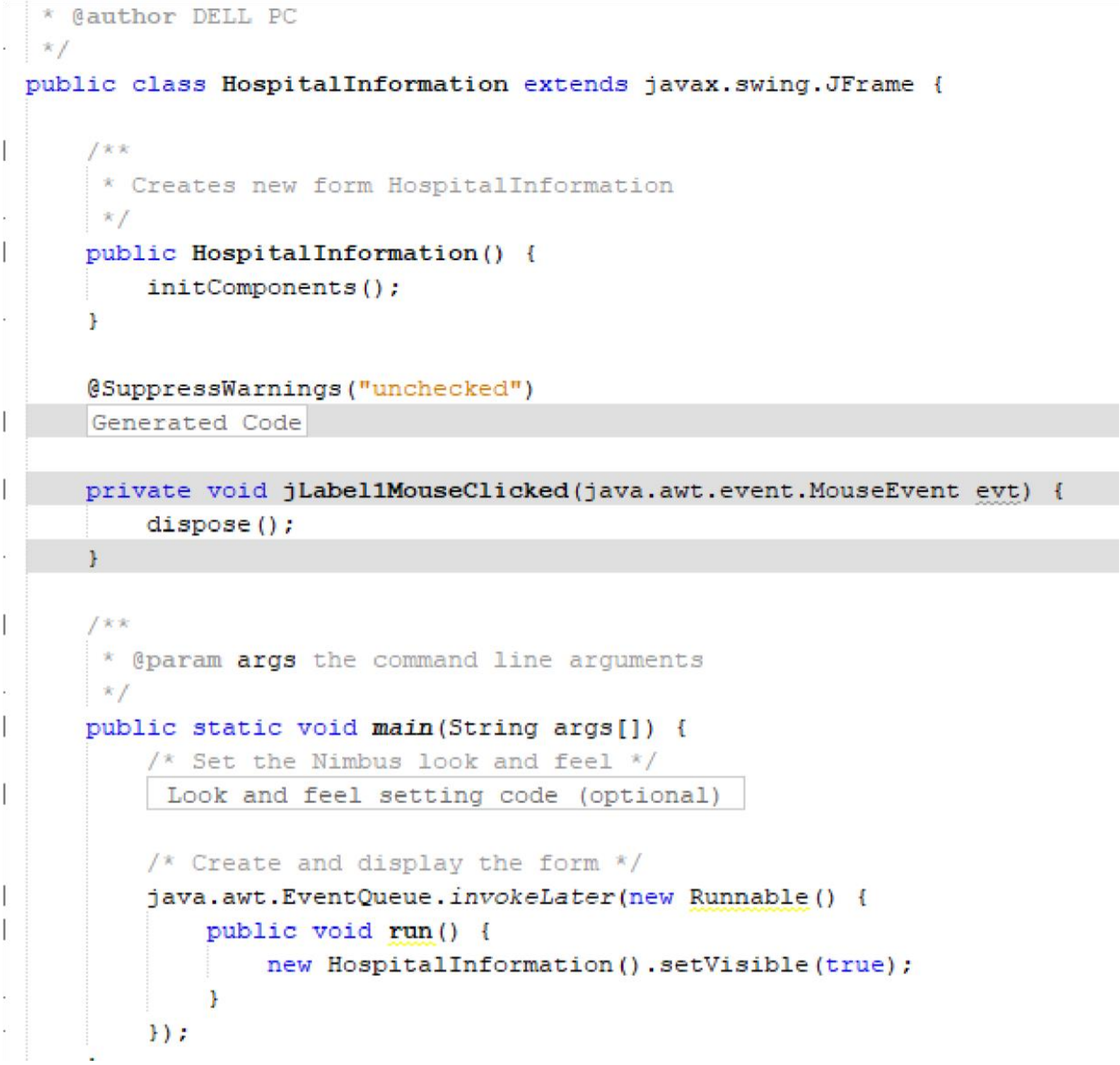
To provide world class health care solution across the globe

Mission

To provide Compassionate Care to All

Values

Personal Touch
Flexible Methods
Familiarity
Responsive Systems



```
 * @author DELL PC
 */
public class HospitalInformation extends javax.swing.JFrame {

    /**
     * Creates new form HospitalInformation
     */
    public HospitalInformation() {
        initComponents();
    }

    @SuppressWarnings("unchecked")
    Generated Code

    private void jLabel1MouseClicked(java.awt.event.MouseEvent evt) {
        dispose();
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        Look and feel setting code (optional)

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