VISVESVARAYA TECHNOLOGICAL UNIVERSITY

JNANA SANGAMA BELGAVI – 590014

MINI PROJECT REPORT ON

"MEDILAB"

Submitted in partial fulfilment for the requirement of 5th semester for the

Degree of Bachelor of Engineering in

COMPUTER SCIENCE & ENGINEERING

For the Academic Year 2021-22

SUBMITTED BY:

SAHANA N[1DB19CS124]

Under the guidance of

Mrs.SHRUTHI.G	**Dr.PARVATHI.C**
Asst.Professor	**Assoc.Professor**
Dept of CSE	**Dept of CSE**

DON BOSCO INSTITUTE OF TECHNOLOGY

BENGALURU-560074

DON BOSCO INSTITUTE OF TECHNOLOGY

Kumbalagodu, Bengaluru -560074

CERTIFICATE

This is to certify that the Project Report entitled "**MEDILAB**" is a bonafide Project work carried out by **SAHANA N(1DB19CS124),** in partial fulfillment of 5th semester for the Degree of **Bachelor of Engineering in Computer Science and Engineering** of Visvesvaraya Technological University, Belagavi, during the academic year 2021-22. It is certified that all corrections/suggestions indicated for Internal Assessments have been incorporated with the degree mentioned.

Signature of Guide		**Signature of HOD**
Dr.Parvathi.C	**Mrs Shruthi .G**	**Prof. B.S UMASHANKAR**
Assoc.Professor,	Asst.professor	Professor & Head
Dept. of CSE,	Dept of CSE	Dept. of CSE .
Bengaluru	Bengaluru	Bengaluru

External Viva

Name of the Examiners **Signature with Date**

1. ____________________ ____________________

2. ____________________ ____________________

DON BOSCO INSTITUTE OF TECHNOLOGY

Kumbalagodu, Bengaluru -560074

DECLARATION

I, **SAHANA N (1DB19CS124),** student of fifth semester B.E, Department of Computer Science and Engineering, Don Bosco Institute of Technology, Kumbalagodu, Bengaluru, declare, that the Project Work entitled **"MEDILAB"** has been carried out by and submitted in partial fulfilment of the requirement of 5th semester Aug 2021-Jan 2022. The matter embodied in this report has been submitted to any university or institute for the award of any other degree or diploma.

Place: Bengaluru SAHANA N

Date: [1DB19CS124]

ACKNOWLEDGEMENT

At the various stages in making the mini project, a number of people have given me invaluable comment on the manuscript. I take this opportunity to express my deepest gratitude and appreciation to all those who helped me directly or indirectly towards the successful completion of this project.

I would like to thank our Principal **Dr.HEMADRI NAIDU T**, Don Bosco Institute of Technology for his support though out this project.

I express my whole hearted gratitude to **Prof. B.S. UMASHANKAR**, who is our respectable Head of Dept. of Computer Science. I wish to acknowledge for his valuable help and encouragement.

In this regard I owe a heartfelt gratitude to my guide **Mrs.Shruthi.G** , Ass.Professor and **Dr.Parvathi.C**, Assoc. Professor, of Department of Computer Science and Engineering, for her timely advice on the project and regular assistance throughout the project work. I would also like to thank the teaching and non-teaching staff members of Department of Computer Science and Engineering for their corporation.

ABSTRACT

MEDILAB pathology software belong to the class of application software intended for storage and management of information obtained in the course of the work of the laboratory. The systems are used to control and manage the information related to consultant hospitals, doctors, patient, test results, reports These services are to be provided in an efficient, cost effective manner, with the goa l of reducing the time and resources currently required for such tasks.

A significant part of the operation of any Pathology Lab involves the acquisition, management and timely retrieval of great volumes of information. This information typically involves; information of consultant hospitals and doctors, patient personal information, tests information and reports for respective tests.

CONTENTS

Chapter 1

INTRODUCTION

1.1 Aim

To generate the pathology reports for diagnostic centres.

1.2 Problem Statement

To provide facile, high quality and ethical information services that enables the pathology lab staff to maintain the data of all the hospitals, doctors and patients which come under their lab. This pathology system maintains the patient information and their consultant hospitals and doctors and also stores the conducted tests results.

1.2 Objectives of the Project

The objectives of the **MEDILAB** are as follows:

- To store complete patient record.
- Retrieve old information of a patient.
- Helps in reducing the number of lost records of the patient and improve the data retrieval.
- Easier for staff of the clinic to retrieve back the record of existing patient history.
- Manages all the laboratory services provided by the clinic.

1.4 Organization of the Report

This chapter deals with the Introduction and organization of the project report. Chapter 2 discusses the Specific to the problem-Requirement Analysis-Design. Chapter 3 discusses the Design and Implementation. Chapter 4 include results and snapshot and. Chapter 5 gives the conclusion and future enhancement of the project.

Chapter 2

REQUIREMENT ANALYSIS

2.1 REQUIREMENTS FOR PROJECT

- Data Base Management textbook (Ramez ELmasri and Shamkant B Navathe)
- Guidelines by lectures
- Knowledge about **Oracle 10 g** application and **Eclipse for java**.
- Requirement gathered from Adithya pathology lab, Shimoga.
- Used Microsoft Teams application due to this pandemic situation.

2.2 Oracle SQL:

Structured Query Language (SQL) is the set of statements with which all the programs and users access data in an Oracle database. Application programs and Oracle tools often allow users access to the database without using SQL directly, but these applications in turn must use SQL when executing the user’s request. An Oracle database is a collection of data treated as a unit. The purpose of a database is to store and retrieve related information. A database server is the key to solving the problems of information management. In general, a server reliably manages a large amount of data in a multiuser environment so that many users can concurrently access the same data. All this is accomplished while delivering high performance. A database server also prevents unauthorized access and provides efficient solutions for failure recovery.

Oracle Database is the first database designed for enterprise grid computing, the most flexible and cost-effective way to manage information and applications. Enterprise grid computing creates large pools of industry-standard, modular storage and servers. With this architecture, each new system can be rapidly provisioned from the pool of components. There is no need for peak workloads, because capacity can be easily added or reallocated from the resource pools as needed.

The database has logical structures and physical structures. Because the physical and logical structures are separate, the physical storage of data can be managed without affecting the access to logical storage structures.

2.2.1 ORACLE DATABASE ARCHITECTURE:

An Oracle database is a collection of data treated as a unit. The purpose of a database

The grid style of computing treats collections of similar IT resources holistically as a single pool, while exploiting the distinct nature of individual resources within the pool. To address simultaneously the problems of monolithic systems and fragmented resources, grid computing achieves a balance between the benefits of holistic resource management and flexible independent resource control. IT resources managed in a grid include:

- Infrastructure: the hardware and software that create a data storage and program execution environment
- Applications: the program logic and flow that define specific business processes
- Information: the meanings inherent in all different types of data used to conduct business

On the path toward this grand vision of grid computing, companies need real solutions to support their incremental moves toward a more flexible and more productive IT architecture. The Oracle Database 10g family of software products implement much of the core grid technology to get companies started. And Oracle delivers this grid computing functionality in the context of holistic enterprise architecture, providing a robust security infrastructure, centralized management, intuitive, powerful development tools, and universal access. Oracle Database 10g includes:

- Oracle Database 10g
- Oracle Application Server 10*g*
- Oracle Enterprise Manager 10*g*
- Oracle Collaboration Suite 10*g*

Although the grid features of Oracle 10g span all of the products listed above, this discussion will focus on the grid computing capabilities of Oracle Database 10g.

CHAPTER 3:

DESIGN AND IMPLEMENTATION

3.1 System design:

The system design process involves developing several modules of the system at different levels of abstractions like:

- Hospital Module
- Doctor Module
- Patient Module
- Test Module
- Report Module

3.2 ER DIAGRAM

Here we put ER diagram and schema diagram of medilab pathology database. It contains the connection i.e., relation between the entities and the participation ratio. And primary key is underlined as we see in figure and foreign keys are the keys that relate to primary key of other table represented by connecting to that table.

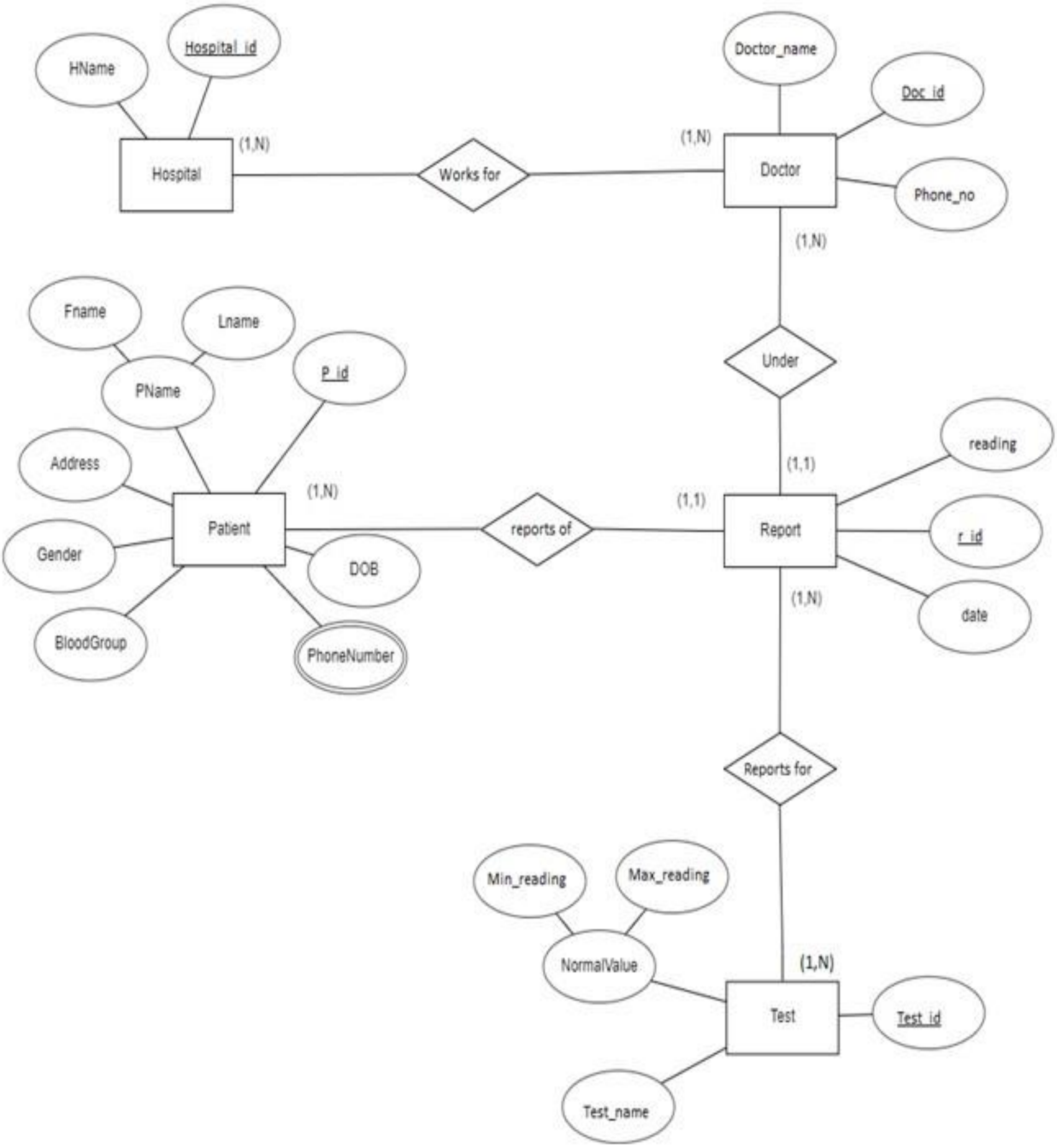


Fig : 3.1 ER DIAGRAM

3.3 SCHEME DIAGRAM

3.4 PSEUDO CODE:

3.4.1: Mainpage

This is the pseudo code for main page which has components directing the user to required module page.

```
public static void main(String[] args) {

    EventQueue.invokeLater(new Runnable() {

     public void run() {

          try {

     Mainpage frame = new Mainpage();
    frame.setVisible(true);
                                    }

      catch (Exception e) {

      e.printStackTrace();

                        }

                }

        });

    }

    public Mainpage() {
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setBounds(200, 150, 500, 400);
```

```
contentPane = new JPanel();
contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
setContentPane(contentPane);
contentPane.setLayout(null);


 JButton btnHospital = new JButton("Hospital Information");
btnHospital.setFont(new Font("Tahoma", Font.BOLD, 13));


 btnHospital.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent arg0) {


                        Hospital h = new Hospital();

                    h.setVisible(true);

                }

        });

        btnHospital.setBounds(25, 100, 200, 50);
        contentPane.add(btnHospital);


        JButton btnDoctor = new JButton("Doctor");
        btnDoctor.addActionListener(new ActionListener() {
                public void actionPerformed(ActionEvent arg0) {
                Doctor d = new Doctor();

                        d.setVisible(true);

                }

        });
```

```
btnDoctor.setFont(new Font("Tahoma", Font.BOLD, 13));
btnDoctor.setBounds(25, 175, 200, 50);
contentPane.add(btnDoctor);


JButton btnPatientInfo = new JButton("Patient Info");
 btnPatientInfo.addActionListener(new ActionListener() {
  public void actionPerformed(ActionEvent arg0) {
                      Patient p = new Patient();
                       p.setVisible(true);

            }

});

btnPatientInfo.setFont(new Font("Tahoma", Font.BOLD, 13));
btnPatientInfo.setBounds(250, 100, 200, 50);
contentPane.add(btnPatientInfo);


JButton btnTest = new JButton("Test");

btnTest.addActionListener(new      ActionListener()      {
            public void actionPerformed(ActionEvent arg0) {
                        Test t = new Test();

                        t.setVisible(true);

            }

});

btnTest.setFont(new Font("Tahoma", Font.BOLD, 13));
btnTest.setBounds(250, 175, 200, 50);
contentPane.add(btnTest);
```

```
                JButton btnReport = new JButton("Report");
                btnReport.addActionListener(new ActionListener() {
                public void actionPerformed(ActionEvent arg0) {
                        Report r = new Report();
                            r.setVisible(true);



                     }

                });

                btnReport.setFont(new Font("Tahoma", Font.BOLD, 13));
                btnReport.setBounds(145, 250, 200, 50);
                contentPane.add(btnReport);
                txtHospitalDatabase = new JTextField();
                txtHospitalDatabase.setText("HOSPITAL DATABASE");
                txtHospitalDatabase.setHorizontalAlignment
                                                    (SwingConstants.CENTER);
                txtHospitalDatabase.setForeground(Color.BLACK);
                txtHospitalDatabase.setFont(new Font("Times New Roman", Font.BOLD, 17));
                txtHospitalDatabase.setEditable(false);
                txtHospitalDatabase.setColumns(10);
                txtHospitalDatabase.setBackground(Color.YELLOW);
                txtHospitalDatabase.setBounds(75, 27, 350, 42);
                contentPane.add(txtHospitalDatabase);

           }

      }
```

3.4.2: DB CODE:

The following code is used for database connectivity.

```
Packagehospitaldb;
 Import      java.sql.Connection;
import java.sql.DriverManager;
import          java.sql.ResultSet;
import java.sql.SQLException;

import javax.swing.table.TableModel;

public class DB {
 private static Connection con;

        public static Connection getConnection() {

                // step1 load the driver class

                try {

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

                        Class.forName("oracle.jdbc.driver.OracleDriver");

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

                        // step2 create the connection object

con = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe", "TECHACAD", "sahana");

                        System.out.println("DatabaseConnected");
        } catch (ClassNotFoundException e) {
                e.printStackTrace();                }
                catch (SQLException e) {
                e.printStackTrace();
                }

                return con;


        }


        public static void main(String args[]){

                getConnection();

        }

}
```

3.4.3: VIEW CODE

To view contents of table following pseudo code is used

The same pseudo code is used in all pages. We have shown it for the Hospital page.

```
public void actionPerformed(ActionEvent arg0) {

                        try
                        {
                                String query="select * from hospital ORDER BY Hospital_id";
                                PreparedStatement pst1=con.prepareStatement(query);
                                ResultSet rs=pst1.executeQuery();
                                table1.setModel(DbUtils.resultSetToTableModel(rs));
                        }
                        catch(Exception e1)
                        {
                                e1.printStackTrace();
```

```
                                  }


                        }
```

3.4.4: INSERT CODE :

To insert content into the table following pseudo code is used

The same pseudo code is used in all pages, here we are showing the insert code of Hospital page.

```
public void actionPerformed(ActionEvent e) {
                              ResultSet id;
                              String name = null;
              name = nametextField.getText();

                               try {
                                 Class.forName("oracle.jdbc.driver.OracleDriver");
               Connection con2=DriverManager.getConnection("jdbc:oracle:thin
                                          :@localhost:1521:xe","system","admin");

                                 String sql1 = "SELECT MAX(Hospital_id) FROM HOSPITAL";
                         PreparedStatement pst1=con2.prepareStatement(sql1);

                                 id=pst1.executeQuery();
                                  String id1 = null;
                                 if(id.next())
                                  id1 = id.getString(1);
                                 if(id1==null) {
                                   id1="0";
                                 }
```

```
        int id2 = Integer.parseInt(id1);
        id2=id2+1;
        if (name.isEmpty()==false)
        {
        String sql = "insert into hospital values('"+id2+"','"+name+"')";

        PreparedStatement pst=con2.prepareStatement(sql);
        pst.executeQuery();
        JOptionPane.showMessageDialog(null, "Data Inserted Successfully");
        }
        else
        {
JOptionPane.showMessageDialog(null,"Hospital name not inserted");
}

     } catch(Exception e1)  {  System.out.println(e1);}
  }
```

CHAPTER 4:

RESULT AND SNAPSHOTS

- The snapshots [4.1-4.6] are user interface pages which give a view of how the pages look for a user.
- The snapshots [4.7a-4.7e] are the INSERT view of the table.
- The snapshots [4.8] are the VIEW view of the table.

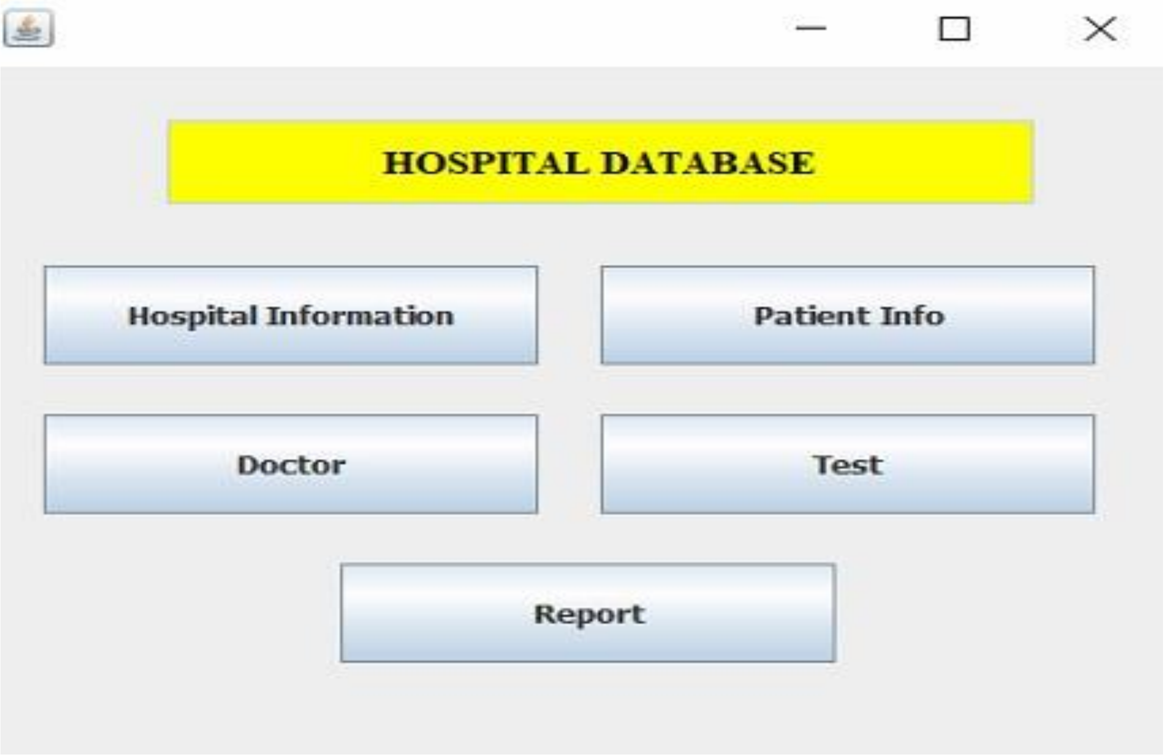


Fig. 4.1: Snapshot of Mainpage

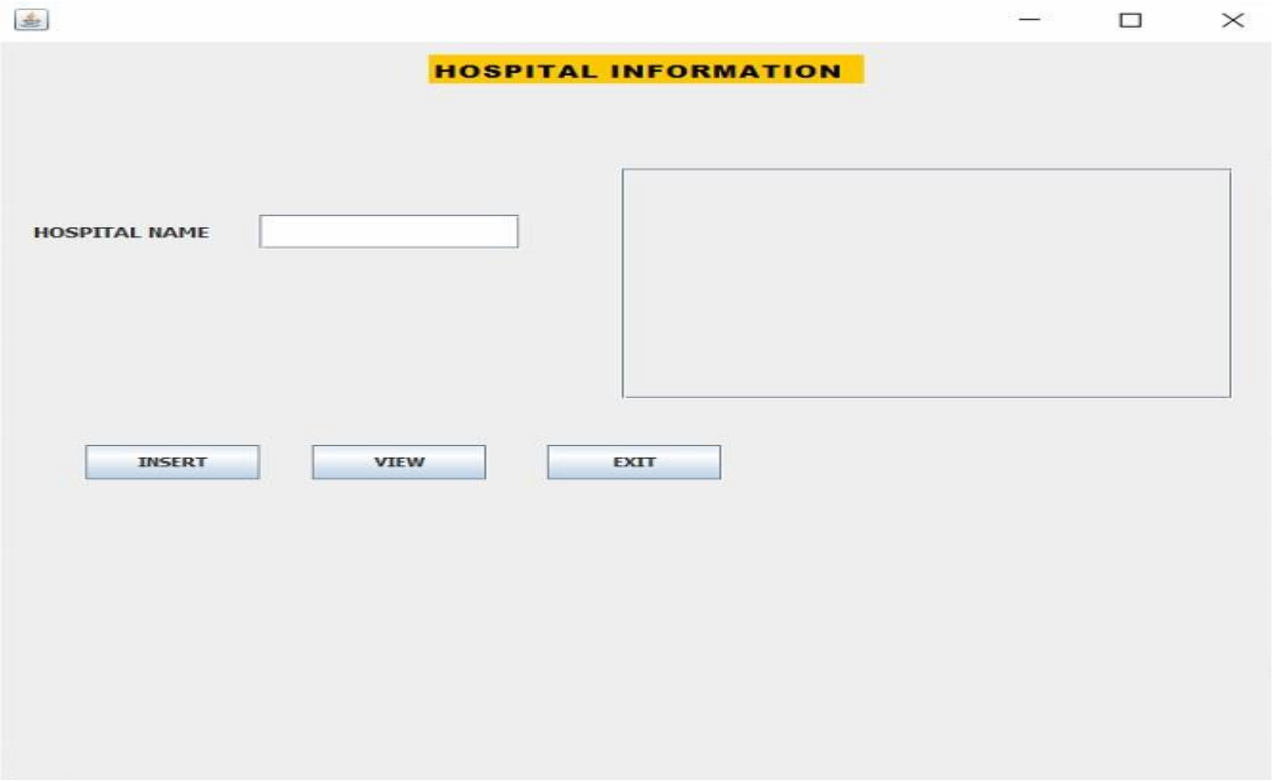


Fig. 4.2: Snapshot of Hospital Information page

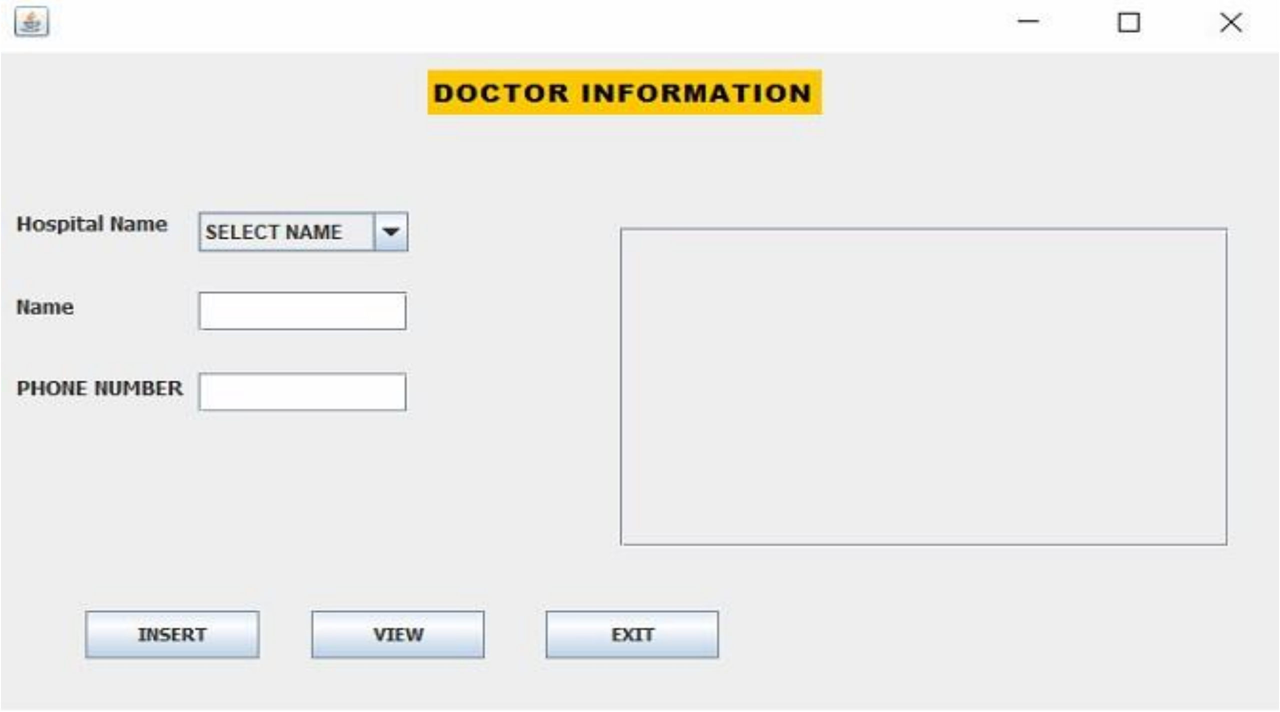


Fig. 4.3: Snapshot of Doctor Information page

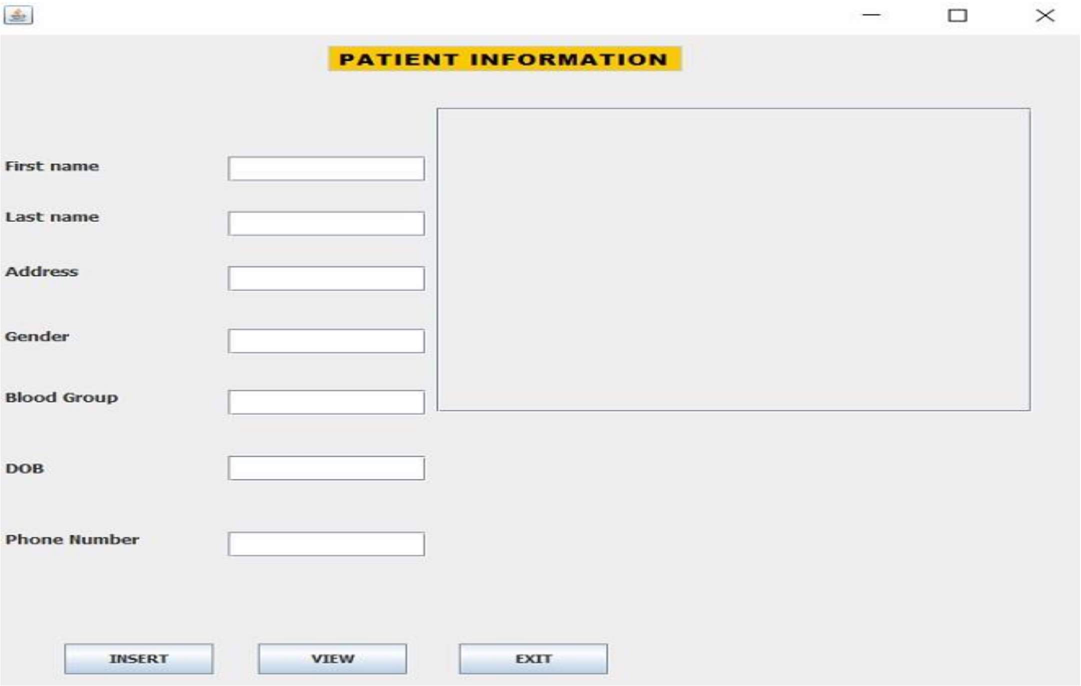


Fig. 4.4: Snapshot of Patient Information page

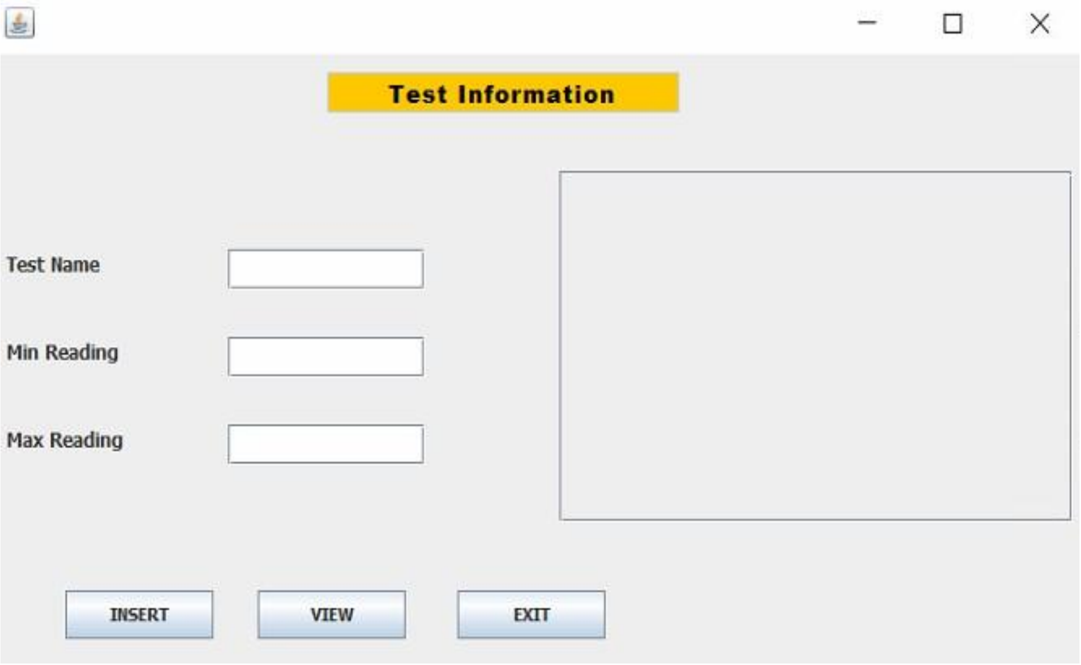
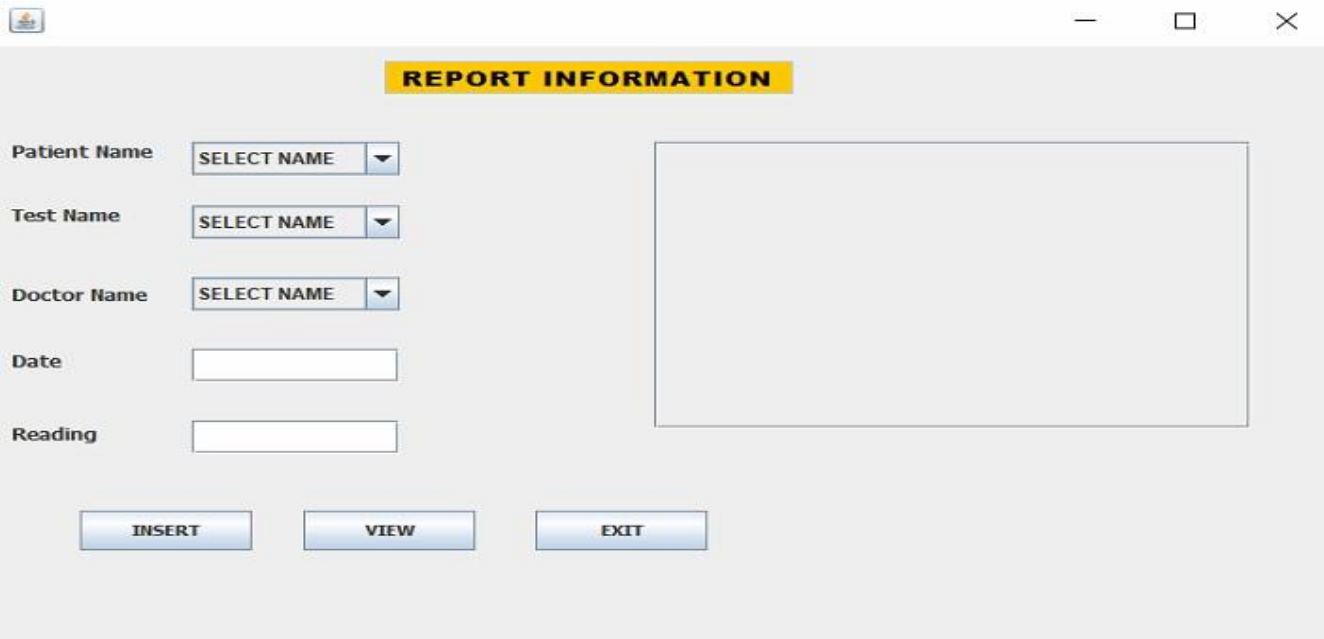


Fig. 4.5: Snapshot of Test Information page



REPORT INFORMATION

Patient Name SELECT NAME

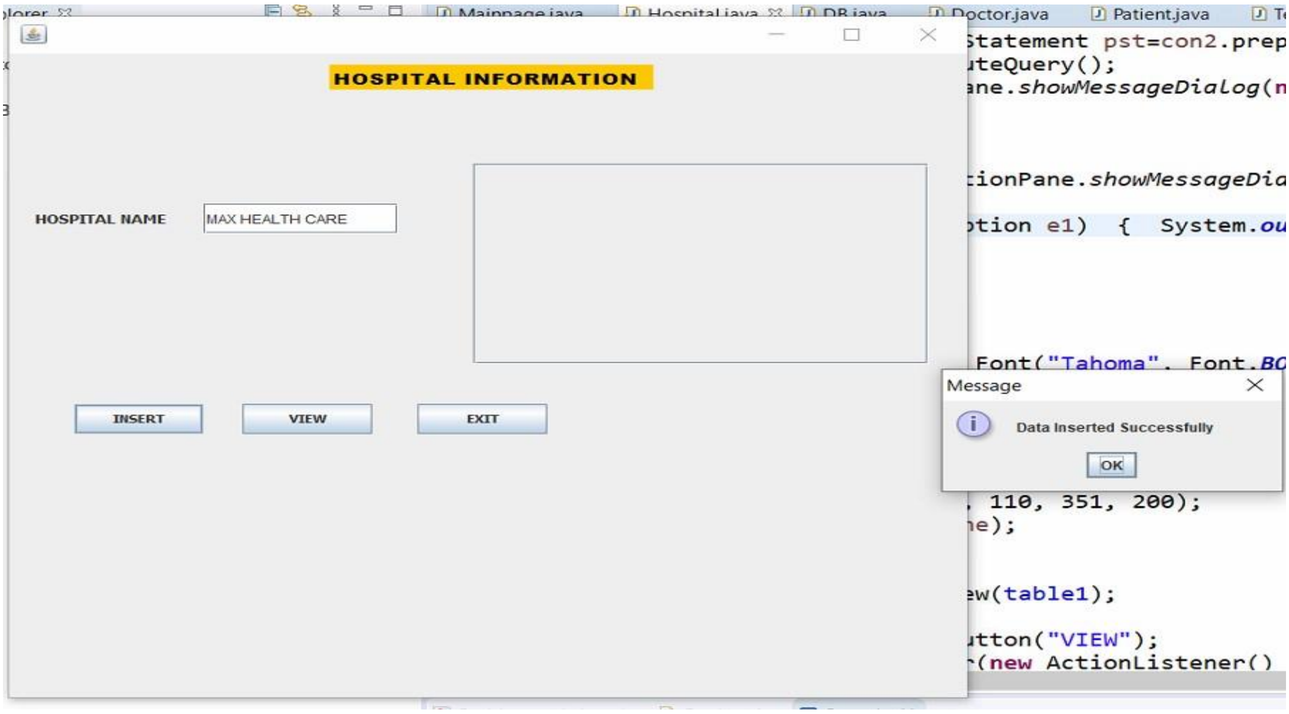
Test Name SELECT NAME

Doctor Name SELECT NAME

Date

Reading

Fig. 4.6: Snapshot of Report Information page



HOSPITAL INFORMATION

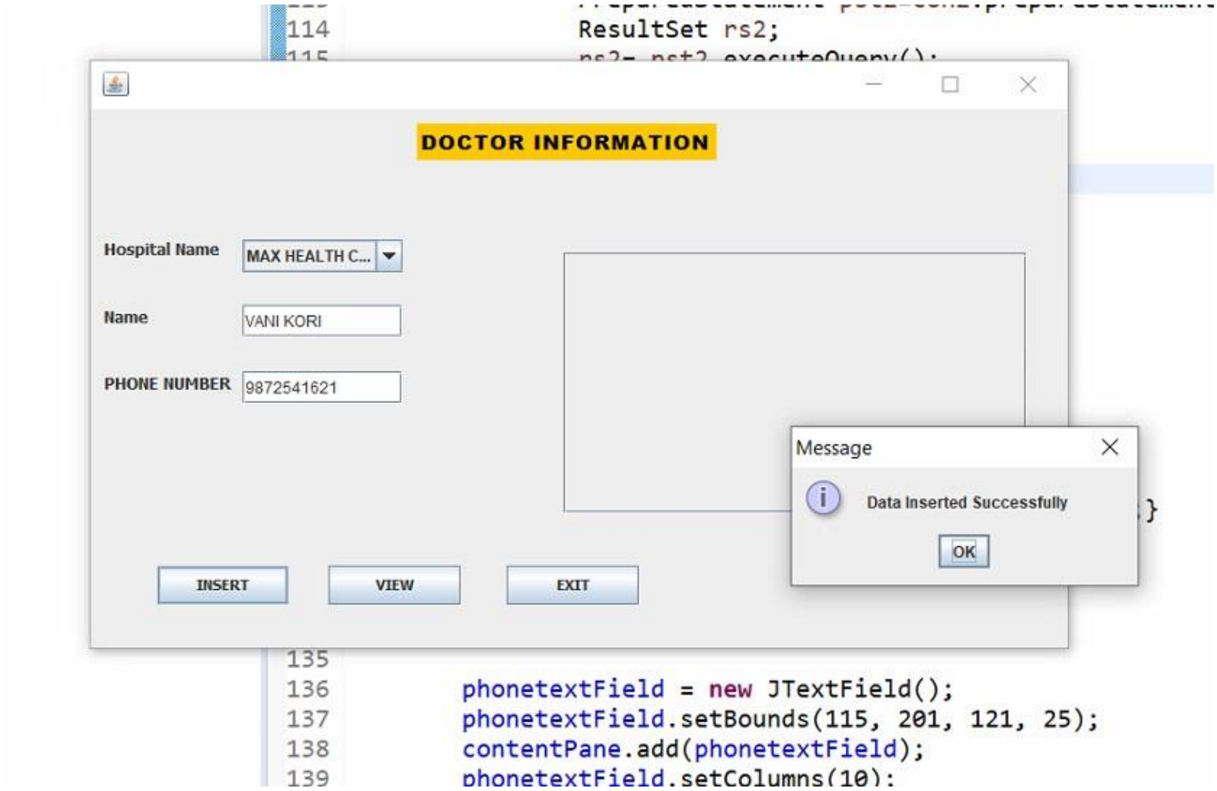
HOSPITAL NAME MAX HEALTH CARE

Message

Data Inserted Successfully

```
Statement pst=con2.prep
uteQuery();
ane.showMessageDialog(n

tionPane.showMessageDia
ption e1)  {  System.ou

Font("Tahoma", Font.BO

, 110, 351, 200);
ne);

ew(table1);

utton("VIEW");
r(new ActionListener()
```

Fig. 4.7 a: Snapshot of INSERT operation



```
ResultSet rs2;
```

DOCTOR INFORMATION

Hospital Name: MAX HEALTH C...

Name: VANI KORI

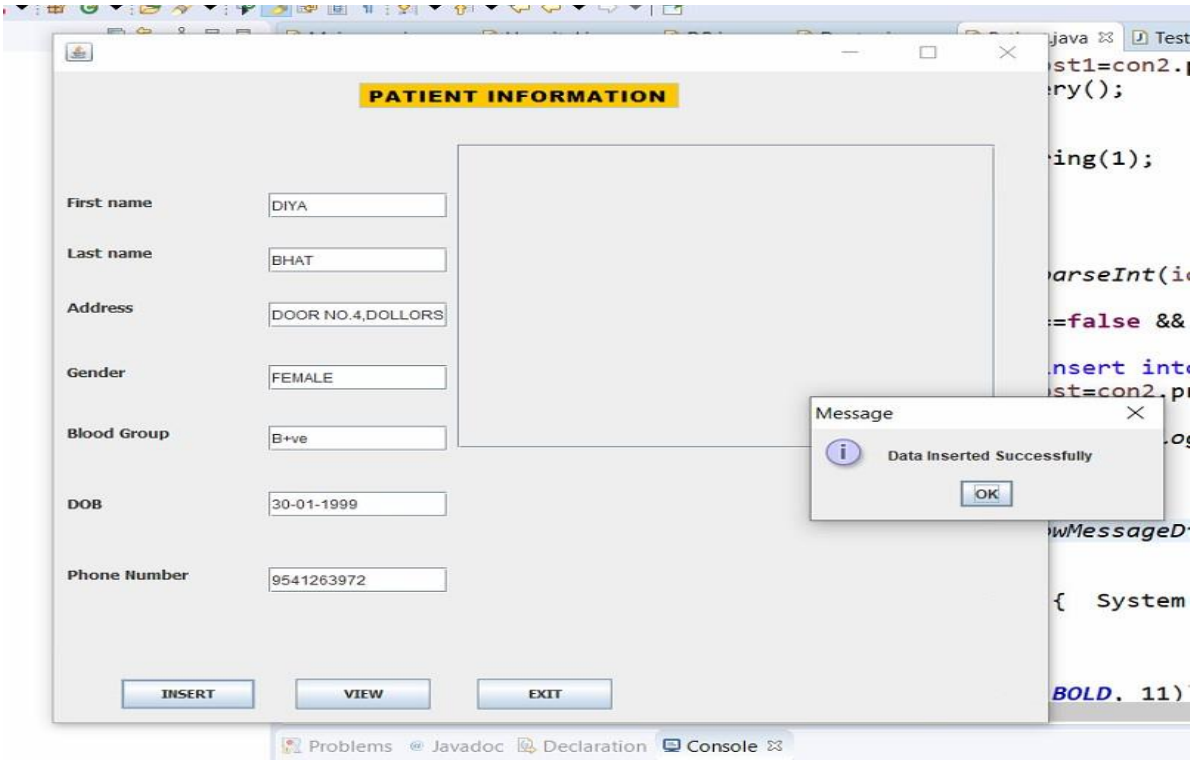
PHONE NUMBER: 9872541621

Message

Data Inserted Successfully

```
phonetextField = new JTextField();
phonetextField.setBounds(115, 201, 121, 25);
contentPane.add(phonetextField);
phonetextField.setColumns(10);
```

Fig. 4.7 b: Snapshot of INSERT operation



PATIENT INFORMATION

First name: DIYA

Last name: BHAT

Address: DOOR NO.4,DOLLORS

Gender: FEMALE

Blood Group: B+ve

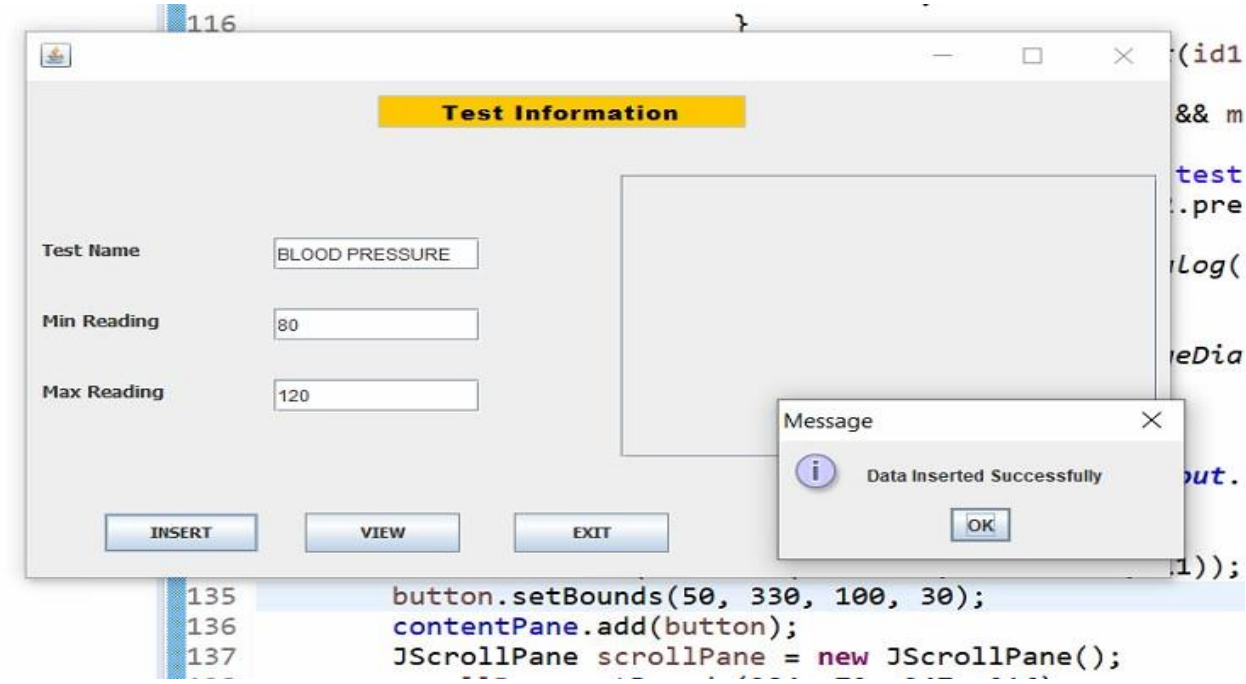
DOB: 30-01-1999

Phone Number: 9541263972

Message

Data Inserted Successfully

Fig. 4.7 c: Snapshot of INSERT operation



Test Information

Test Name: BLOOD PRESSURE

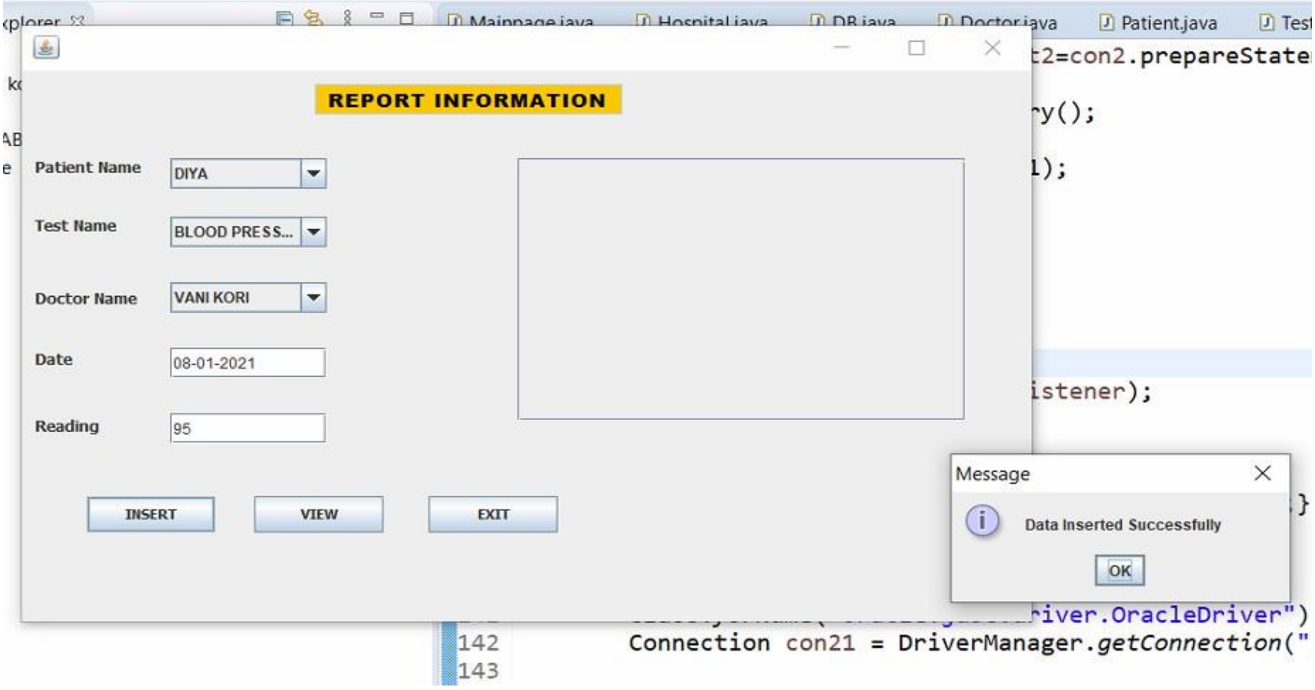
Min Reading: 80

Max Reading: 120

Message: Data Inserted Successfully

```
button.setBounds(50, 330, 100, 30);
contentPane.add(button);
JScrollPane scrollPane = new JScrollPane();
```

Fig. 4.7 d: Snapshot of INSERT operation



REPORT INFORMATION

Patient Name: DIYA

Test Name: BLOOD PRESS...

Doctor Name: VANI KORI

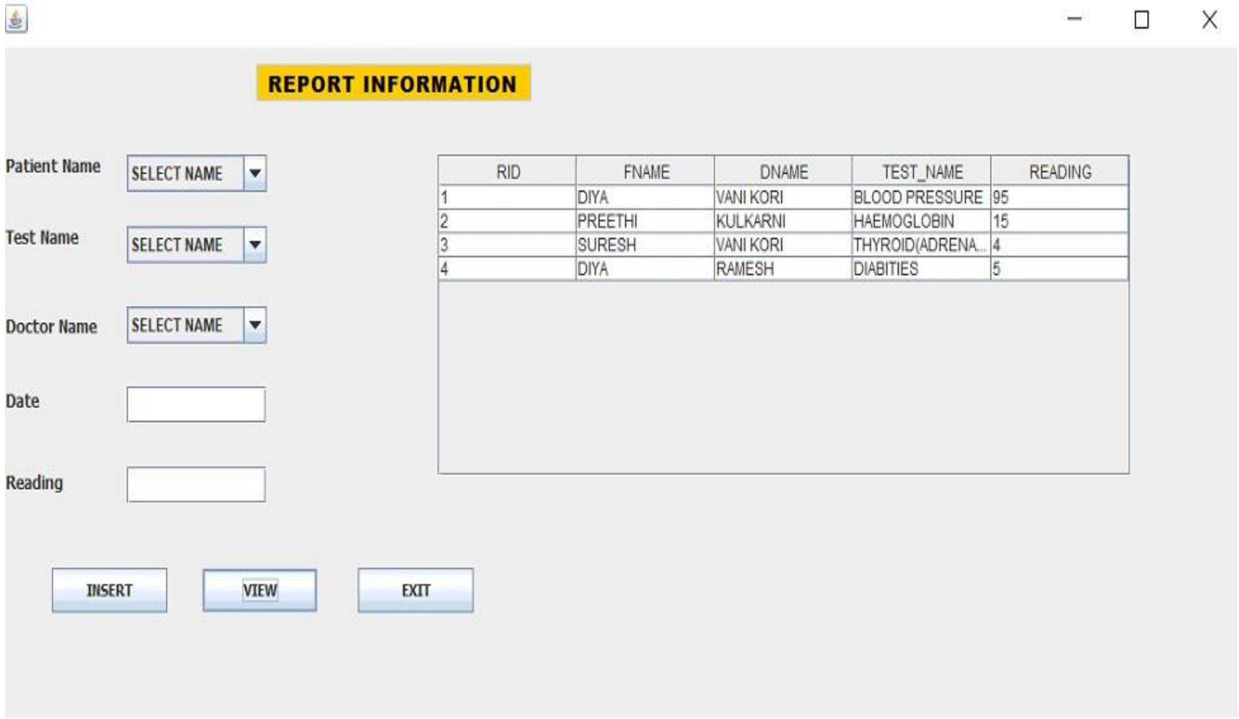
Date: 08-01-2021

Reading: 95

Message: Data Inserted Successfully

```
Connection con21 = DriverManager.getConnection("
```

Fig 4.7 e: Snapshot of INSERT operation



REPORT INFORMATION

Patient Name: SELECT NAME

Test Name: SELECT NAME

Doctor Name: SELECT NAME

Date

Reading

RID	FNAME	DNAME	TEST_NAME	READING
1	DIYA	VANI KORI	BLOOD PRESSURE	95
2	PREETHI	KULKARNI	HAEMOGLOBIN	15
3	SURESH	VANI KORI	THYROID(ADRENA...	4
4	DIYA	RAMESH	DIABITIES	5

Fig. 4.8: Snapshot of VIEW operation

CHAPTER 5:

CONCLUSION

The **Medilab** pathology software is all about the modernizing a hospital through the use of technology.This laboratory system is quite reliable and is proven on many stages. All the basic requirements of the hospital are provided in this hospital system in order to manage it perfectly and large amount of data can also be stored. It gives many facilities like searching for the detail of patient, billing facilities as well as the creation of test reports. So it's an important system for modern days. The old system has many faults like they have to keep the record of the patients using papers. It was difficult to find the record patient but our system can overcome this disadvantage.

Future Enhancement:

- We can enhance the system by including a module where the lab gets information on how many tests conducted per day/month/year.
- We can also enhance to include Histo-pathology report where parts of organs are sent under testing.

CHAPTER 6:

REFERENCE

1. RamezElmasri and Shamkant B Navathe, “Database systems Models,

   Languages, Design and Application Programming”, Pearson, 7th Edition, 2017.

2. Herbert Schildt: AVA the Complete Reference,7th/9th Edition,

   Tata McGraw Hill, 2007.

3. Nptel “Programming in java” course videos, notes and demonstration.

   (url - NPTEL :: Computer Science and Engineering - NOC:Programming in Java)

4. Stackoverflow help in-

   How to view database resultset in Java swing - Stack Overflow