

School of Optoelectronic Information and Computer
Engineering
University of Shanghai for Science and Technology

JAVA PROGRAMMING AND DEVELOPMENT

EXPERIMENT REPORT



Name Yue Zhu

Student Number 1615050118

Course Code 12002910

Instructor Bingxue Zhang

Table of Contents

Experiment Name: Student Information Management System.....	3
1. Purpose of the experiment.....	3
2. Experimental equipment.....	3
3. Experimental principle	3
4. Record of the experimental process.....	3
4.1 Requirements analysis	3
4.2 System design.....	3
4.3 Code implementation.....	5
5. Results and Analysis.....	16
5.1 Debugging and testing.....	16
6. Self-assessment.....	17
6.1 Course Harvest	17
6.2 Deficiencies and follow-up development work.....	17

Experiment Name: "Student Information Management System"

JAVA program development

1. Purpose of the experiment

1. Familiar with the use of Java development tools
2. Master the process of writing, running and testing Java programs.
3. Write programs using the object-oriented features of the Java language.
4. Master the common libraries of Java.

2. Experimental equipment

Hardware: PC

Software: Java Development Toolkit, IDE(Eclipse, IntelliJ IDEA, Netbeans)

3. Experimental principle

Java program language features, library features, object-oriented methods, etc.

4. Record of the experimental process

4.1 Requirements Analysis

(1) Students

- a. Student Registration
- b. Student login
- c. Display of students' personal information and query of course grades

(2) Teachers

- a. Teacher Registration
- b. Teacher login
- c. Teacher interface
 1. Addition of student information
 2. Query of student information
 3. Updating student information
 4. Deletion of student information
 5. Addition of student grade
 6. Query of student grade
 7. Updating student grade
 8. Deletion of student grade

4.2 System Design

(1) Database

- a. Create database "mydb"
- b. Create tables
 1. student table: record student number (primary key), name, password, which can be used when a student registers and logs in (In the design, only students who have had their

student numbers entered by the School Academic Affairs Office can register. Otherwise, they cannot register and will be notified that the student has not yet been entered.)

2. inf: record students' personal information: student number (primary key), name, gender, address, telephone, major, class, which is used when teachers add, delete, modify and check students' information and display students' personal information

3. teacher: record the teacher's ID (primary key), name, and password for the teacher to use when logging in and registering

4. grade: record students' grade information: [student number (primary key), subject (primary key)](composite key), grade for teachers to use when adding, deleting, modifying and checking student information

(2) Database class

Encapsulation of database connection, opening, adding, deleting, modifying, querying, and closing operations

Database.open() Open the database

Database.update(sql) Additions, deletions, modifications, and checks

Database.close() Close the database

(3) MainUI class

Main Interface: Login and registration screen for students and teachers

(4) StudentRegister class

Student registration interface: Students enter the student number, name and password set by themselves, and the program will first determine whether the student number exists in the inf table (that is, whether the teacher has entered the student), if the student number does not exist, the prompt message "The student number has not been entered" will not be registered. If it has been entered, it is necessary to judge whether the name corresponding to the student number is consistent with the inf form, if it is inconsistent, the prompt message "The student number is inconsistent with the name" will not be entered. If it is the same, the information will be entered into the student table. This ensures that a student can only register if the teacher enters the student, i.e. the number and ID of the students in both the student and inf tables are the same.

(5) TeacherRegister class

Teacher registration interface: Teachers can log in to the teacher interface only after entering their work number, name and password.

(6) StudentPage class

After students log in, they can see their personal information and check the results of each course

(7) TeacherPage class

The interface displayed after the teacher logs in, where you can choose to add, delete, modify, and check students' information and grades

(8) StudentAdd class

Enter a new student's personal information

(9) StudentDelete class

Delete a profile that already has a student

(10) StudentUpdate class

Update information for an existing student

(11) StudentQuery class

Query a student's information

(12)StudentGAdd class

Enter your student number, select a course, and add

(13)StudentGDelete class

Enter the student number, select the course, and delete

(14)StudentGUpdate class

Enter your student number, select a course, and update

(15)StudentGQuery class

Enter the student number, select the course, and search

4.3 Code Implementation

//An encapsulated class for database operations

```
public class Database {
    //Database connection object
    private static Connection con=null;
    //Open a database connection
    public static boolean open() {
        try {
            //Load the driver
            Class.forName("com.mysql.cj.jdbc.Driver");
            //Connect to the database
            String
            conStr="jdbc:mysql://localhost/mydb?userSSL=true&useUnicode=true&characterEncoding=UTF8&serverTimezone=GMT";
            con=DriverManager.getConnection(conStr, "root", "zhuyue");
            return true;
        } catch (Exception e) {
            return false;
        }
    }
    //Query operations
    public static ResultSet query(String sql) {
        Statement st=null;
        ResultSet rs=null;
        try {
            st=con.createStatement();
            rs=st.executeQuery(sql);
        } catch (Exception e) {
            System.out.println("Database query error ");
            System.out.println(s);
            e.printStackTrace();
            rs=null;
        }
        return rs;
    }
    //Add, delete and modify operations
    public static int update(String sql) {
        int affectRows=0;
        Statement st=null;
        try {
            st=con.createStatement();
            affectRows=st.executeUpdate(sql);
        } catch (SQLException ex) {
            ex.printStackTrace();
            affectRows=st. EXECUTE_FAILED;
        }
        return affectRows;
    }
}
```

```

//Close the database connection
public static void close() {
    try {
        if(con!=null)
            con.close();
    }catch(Exception e) {
        System.out.println("Close error");
    }
}

}

//The implementation class of the main interface
public class MainUI extends JFrame implements ActionListener{

    //Define the component
    JButton jb1,jb2=null;
    JPanel jp1,jp2,jp3,jp4=null;
    JTextField jtf=null;
    JLabel jlb1,jlb2=null;
    ButtonGroup btnGroup=null;
    JRadioButton jrbStu=null;
    JRadioButton jrbTea=null;
    JPasswordField jpf=null;

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        MainUI main=new MainUI();
        main.setTitle("Student Information Management System");
    }
    Constructor
    public MainUI() {

        //Create a component
        jb1=new JButton("Register");
        jb2=new JButton("Login");
        Set up a listener
        jb1.addActionListener(this);
        jb2.addActionListener(this);

        jp1=new JPanel();
        jp2=new JPanel();
        jp3=new JPanel();
        jp4=new JPanel();

        jlb1=new JLabel("ID:    ");
        jlb2=new JLabel("PWD:  ");

        jtf=new JTextField(10);
        jpf=new JPasswordField(10);

        btnGroup = new ButtonGroup();
        jrbStu = new JRadioButton("Student");
        jrbTea = new JRadioButton("Teacher");
        jrbStu.setSelected(true);
        btnGroup.add(jrbStu);
        btnGroup.add(jrbTea);

        Added to JPanel
        jp1.add(jrbStu);
        jp1.add(jrbTea);

        jp2.add(jlb1);
        jp2.add(jtf);

        jp3.add(jlb2);

```

```

        jp3.add(jpf);

        jp4.add(jb1);
        jp4.add(jb2);

        Join JFrame
        this.getContentPane().add(jp1);
        this.getContentPane().add(jp2);
        this.getContentPane().add(jp3);
        this.getContentPane().add(jp4);

        this.setLayout(new GridLayout(4,1)); // Select Grid Layout Manager
        this.setSize(300,200);
        this.setLocation(400, 200);
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        this.setVisible(true);

    }

    public void clear()
    {
        jtf.setText("");
        jpf.setText("");
    }

    public void actionPerformed(ActionEvent e) { // Event detection

        if(e.getActionCommand()=="login")
        {
            if(jrbStu.isSelected())
            {
                try {
                    stuLogin();//student login

                } catch (Exception ex) {}

            } else if(jrbTea.isSelected()){
                try {

                    teaLogin();//teacher login
                } catch (Exception ex) {}

            }

        } else if(e.getActionCommand()=="register ")
        {
            if(jrbStu.isSelected())
            {
                stuRegister();
            } else if(jrbTea.isSelected()){
                teaRegister()
            }

        }

    }

    //Non-empty judgment
    public boolean notEmpty()
    {

        boolean a=false;
        char[] p =jpf.getPassword();
        if(jtf.getText().isEmpty())
        {
            JOptionPane.showMessageDialog(null,"Please enter your ID! ", " Prompt message",JOptionPane.WARNING_MESSAGE);

```

```

        }else if(new String(p).isEmpty())
        {
            JOptionPane.showMessageDialog(null," Please enter password! ", " Prompt message
            ",JOptionPane.WARNING_MESSAGE);
        }else {
            a=true;
        }
        return a;
    }

//Student login
public void stuLogin() throws Exception{
    String id=jtf.getText();
    char[] p =jpf.getPassword();
    String pwd=new String (p);

    if(notEmpty()) {
        Database.open();
        String sql="select * from student where id='"+id+"'";
        ResultSet rs=Database.query(sql);
        if(!rs.next()) {
            JOptionPane.showMessageDialog(null, " This ID has not been registered yet! \n
            Please log in again.", "Prompt message", JOptionPane.ERROR_MESSAGE);
            clear(); // Clear the input field
        }else{
            if(pwd.equals(rs.getString("pwd"))){
                JOptionPane.showMessageDialog(null," Login successful! ", " Prompt message
                ",JOptionPane.WARNING_MESSAGE);
                this.dispose();

                StudentPage stuP=new StudentPage(rs);
            }else {
                JOptionPane.showMessageDialog(null," Username or password is incorrect!
                Please re-enter", "Prompt Message", JOptionPane.ERROR_MESSAGE);
                clear();
            }
        }
        Database.close();
    }

}

//Teacher login
public void teaLogin() throws Exception{
    String id=jtf.getText();
    char[] p =jpf.getPassword();
    String pwd=new String (p);
    if(notEmpty()) {
        Database.open();
        String sql="select * from teacher where id='"+id+"'";
        ResultSet rs=Database.query(sql);
        if(!rs.next()) {
            JOptionPane.showMessageDialog(null, "This ID has not been registered!\nPlease log in or
            register again", "Prompt message", JOptionPane.ERROR_MESSAGE);
            clear();
        }else{
            if(pwd.equals(rs.getString("pwd"))){
                JOptionPane.showMessageDialog(null," Login successful!! ", " Prompt Message
                ",JOptionPane.WARNING_MESSAGE);
                this.dispose();
                TeacherPage teaP=new TeacherPage();
            }else {

```



```

        JOptionPane.showMessageDialog(null," Username or password is incorrect !\nPlease
        input again", "Prompt Message", JOptionPane.ERROR_MESSAGE);
        clear();
    }
}
Database.close();
}

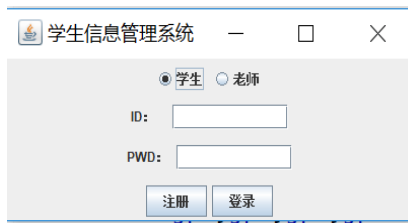
//Student Registration
public void stuRegister() {

    StudentRegister stuR=new StudentRegister();

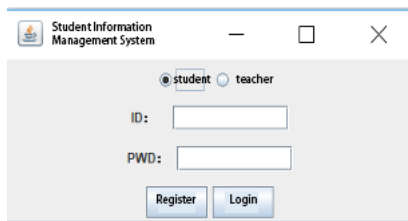
}

//Teacher Registration
public void teaRegister(){
    TeacherRegister teaR=new TeacherRegister();
}
}

```



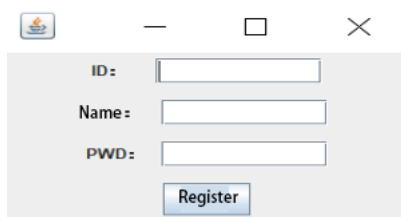
4.3.1 System login



Translated version



4.3.2 Register



Translated version

```

//Student interface
public class StudentPage extends JFrame implements ActionListener{

    JButton jb=null;
    JPanel jp1,jp2,jp3,jp4,jp5,jp6,jp7,jp8,jp9=null;
    JLabel jlb1,jlb2,jlb3,jlb4,jlb5,jlb6=null;
    JComboBox jcb=null;
    String name;
    String clas;
    String major;
    String id;
    public StudentPage(ResultSet rs1) throws SQLException{
        //Create a component
        jb=new JButton("Query");

        Set up a listener
        jb.addActionListener(this);

        jp1=new JPanel();
        jp2=new JPanel();
        jp3=new JPanel();
        jp4=new JPanel();
        jp5=new JPanel();
        jp6=new JPanel();
        jp7=new JPanel();
        jp8=new JPanel();
        jp9=new JPanel();

        Database.open();
        id=rs1.getString("id");
        ResultSet rs=Database.query("SELECT * FROM inf WHERE id='"+id+"'");
        if(rs.next()){

            name=rs.getString("name");
            clas=rs.getString("class");
            major=rs.getString("major");

        }

        jlb1=new JLabel("Student number: "+id+"");
        jlb2=new JLabel("Course: ");
        jlb3=new JLabel("Name: "+name+"");
        jlb4=new JLabel ("Class: "+clas+"");
        jlb5=new JLabel("Major: "+major+"");
        jlb6=new JLabel();

        Database.close();

        String[] subject={"java","Operating System","Probability Theory","Discrete Mathematics"};
        jcb=new JComboBox(subject);
        jcb.setMaximumRowCount(4);
        jcb.setSize(20, 10);

        //Added to JPanel
        jp1.setLayout(new GridLayout(4,1));
        jp2.setLayout(new GridLayout(3,1));

        jp1.add(jp3);
        jp1.add(jp4);
        jp1.add(jp5);
        jp1.add(jp6);

```

```

        jp2.add(jp7);
        jp2.add(jp8);
        jp2.add(jp9);

        jp3.add(jlb1);
        jp4.add(jlb3);
        jp5.add(jlb4);
        jp6.add(jlb5);

        jp7.add(jlb2);
        jp7.add(jcb);

        jp8.add(jb);
        jp9.add(jlb6);

        Join JFrame
        setLayout(new BorderLayout()); // Select Grid Layout Manager
        getContentPane().add(jp1,BorderLayout.WEST);
        getContentPane().add(jp2,BorderLayout.EAST);
        setSize(300,200);
        setLocation(400, 200);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setVisible(true);

    }

    public void query() throws SQLException{

        String subject=jcb.getSelectedItem().toString().trim();
        int a=0;

        Database.open();

        ResultSet rs=Database.query("SELECT * FROM grade WHERE id='"+id+"' and
        subject='"+subject+"'");
        if(rs.next()){

            jlb6.setText(rs.getString("score"));

        }else

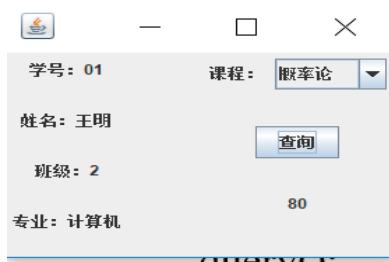
            JOptionPane.showMessageDialog(this," The grades for this course have not been entered
            yet!"," Prompt dialog box", JOptionPane.INFORMATION_MESSAGE);

        Database.close();
    }

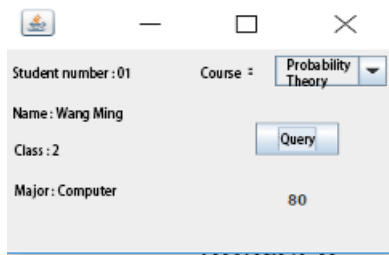
    public void actionPerformed(ActionEvent e) { /
        try {
            query();
        } catch(SQLException ee) {}

    }
}

```



4.3.3 Student interface



Translated version

//Teacher interface
 public class TeacherPage extends JFrame implements ActionListener{

```

    JMenuBar, jMenuBar;
    JMenu jMenu1;
    JMenu jMenu2;
    JMenuItem1;
    JMenuItem2;
    JMenuItem3;
    JMenuItem4;
    JMenuItem5;
    JMenuItem6;
    JMenuItem item7;
    JMenuItem item8;

    StudentAdd zengjia;
    StudentQuery chaxun;
    StudentUpdate gengxin;
    StudentDelete shanchu;
    StudentGAdd gzengjia;
    StudentGQuery gchaxun;
    StudentGUpdate ggengxin;
    StudentGDelete gshanchu;

    public TeacherPage() {
        jMenuBar = new JMenuBar();

        jMenu1 = new JMenu("Student Information");
        jMenu2 = new JMenu("Grade Information");

        item1 = new JMenuItem("Enter");
        item2 = new JMenuItem("Query");
        item3 = new JMenuItem("Update");
        item4 = new JMenuItem("Delete");
        item5 = new JMenuItem("Enter");
        item6 = new JMenuItem("Query");
        item7 = new JMenuItem("Update");
        item8 = new JMenuItem("Delete");

        item1.addActionListener(this);
        item2.addActionListener(this);
        item3.addActionListener(this);
        item4.addActionListener(this);
        item5.addActionListener(this);

```

```

        item6.addActionListener(this);
        item7.addActionListener(this);
        item8.addActionListener(this);

        zengjia=new StudentAdd();
        chaxun=new StudentQuery();
        gengxin=new StudentUpdate();
        shanchu=new StudentDelete();

        gzengjia=new StudentGAdd();
        gchaxun=new StudentGQuery();
        ggengxin=new StudentGUpdate();
        gshanchu=new StudentGDelete();

        jMenu1.add(item1);
        jMenu1.add(item2);
        jMenu1.add(item3);
        jMenu1.add(item4);
        jMenu2.add(item5);
        jMenu2.add(item6);
        jMenu2.add(item7);
        jMenu2.add(item8);

        jMenuBar.add(jMenu1);
        jMenuBar.add(jMenu2);
        Here's how to add a title bar
        this.setJMenuBar(jMenuBar);

        this.setSize(800,600);
        this.setLocation(400, 200);
        this.setVisible(true);
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE); //Set the guarantee that the JVM also exits
when the window is closed
    }

    public void actionPerformed(ActionEvent e) {
        if(e.getSource()==item1) {
            getContentPane().removeAll();
            getContentPane().add(zengjia,"center");
            getContentPane().repaint();
            getContentPane().validate();
        }
        if(e.getSource()==item2) {
            getContentPane().removeAll();
            getContentPane().add(chaxun, "Center");
            getContentPane().repaint();
            getContentPane().validate();
        }
        if(e.getSource()==item3) {
            getContentPane().removeAll();
            getContentPane().add(gengxin,"Center");
            getContentPane().repaint();
            getContentPane().validate();
        }
        if(e.getSource()==item4) {
            getContentPane().removeAll();
            getContentPane().add(shanchu,"center");
            getContentPane().repaint();
            getContentPane().validate();
        }
        if(e.getSource()==item5) {
            getContentPane().removeAll();
            getContentPane().add(gzengjia,"center");
            getContentPane().repaint();
        }
    }

```

```

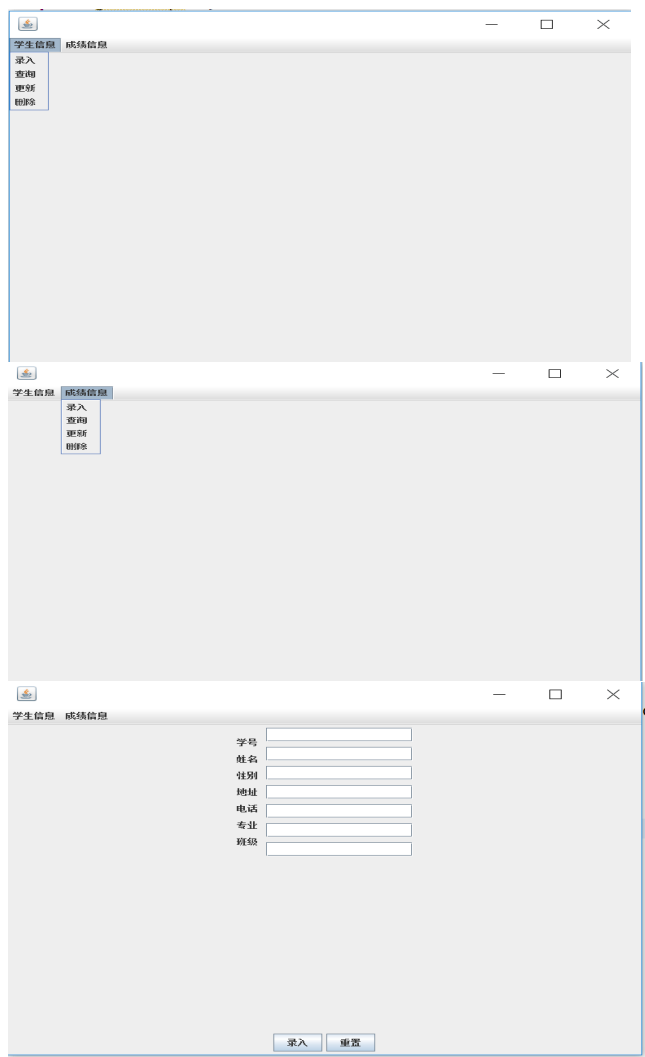
        getContentPane().validate();
    }

    if(e.getSource()==item6) {
        getContentPane().removeAll();
        getContentPane().add(gchaxun, "Center");
        getContentPane().repaint();
        getContentPane().validate();
    }

    if(e.getSource()==item7) {
        getContentPane().removeAll();
        getContentPane().add(ggengxin,"center");
        getContentPane().repaint();
        getContentPane().validate();
    }

    if(e.getSource()==item8) {
        getContentPane().removeAll();
        getContentPane().add(gshanchu,"center");
        getContentPane().repaint();
        getContentPane().validate();
    }
}

```



学生信息 成绩信息

请输入学号: 查询

姓名

性别

地址

电话

专业

班级

学生信息 成绩信息

请输入待更新学生的学号: 确定

姓名

性别

地址

电话

专业

班级

更新

学生信息 成绩信息

请输入待更新学生的学号: 确定

姓名

性别


地址

电话

专业

班级

提示消息

 更新成功!

确定

更新

学生信息 成绩信息

学号

科目

成绩

java

录入 重置



5. Results and Analysis

5.1 Debugging and Testing

1. Registration and login of students and teachers
 2. Student grade inquiry
 3. Teachers' additions, deletions, corrections and checks on students' grades
 4. Teachers add, delete, modify, and review student information
- After debugging, testing, all functions are completed

6. Self-assessment

6.1 Course Gains

1. Learn to write programs using a graphical user interface
2. Learn to use databases
3. Learn the idea of encapsulation of classes

6.2 Deficiencies and Follow-up Development

The disadvantage is that the number of courses in this design is fixed, but in practice, each student can choose a variety of different courses, so the fixed courses will be changed to variable in the future, and it is also necessary to consider adding the course number for each course, and in what form the variable courses should be displayed in the GUI interface.