IEM

श्रद्धावान लभते ज्ञानम्

INSTITUTE OF ENGINEERING & MANAGEMENT

Good Education, Good Jobs

Object Oriented Programming Mini Project
Student Information System

By

Soumyajit Chowdhury
22021002002008(176)

Samrat Dawn
12020002017040(165)

Students' Information System

Suppose that you have to create a student's information system for the enrolled students in the order of their first names. But the problem is that they may take admission in a random fashion. So, you have to maintain these records using a linked list where each node represents a student.

Each student node is an object containing his/her name, roll number (unique), marks and grade. The students' roll numbers are generated in the order of their admission. There are three operations –

To add a student's records (roll number, name, marks, grade)

To delete a student's record

To search a student's record based on their first name, last name and roll number

To modify a student's record (e.g., the marks of a particular student need to be changed, the name has been misspelt, etc.)

The linked list records are to be stored in a file. Also, the operations which are performed on the register, are to be saved in a log file. Each log file entry consists of two fields –

Operation name

Timestamp of the operation

The deleted records are to be stored in a stack such that the delete operations can be undone.

You have to design the information system using Java Swing. You should use different components of Java Swing to enhance the look and feel of the designed system. There are no restrictions to design your system layout.

Algorithm

Linear Search

Linear searching techniques are the simplest technique. In this technique, the items are searched one by one. This procedure is also applicable for unsorted data set.

```
procedure linear_search(list, value)
  for each item in the list
    if item == value then
      return the item's location
    end if
  end for
end procedure
```

Modified Binary Search

When the list is sorted, we can use the binary search technique to find items on the list. In this procedure, the entire list is divided into two sub-lists. If the item is found in the middle position, it returns the location, otherwise jumps to either left or right sub-list and do the same process again until finding the item or exceed the range. Modified it to get the location where I can insert a new element.

```
procedure binary_search(list, value, low, high)
  mid = abs(low+high)/2
  if low == high then
    if value – mid < 0 then
      return mid
    else
      return mid+1
  else
    if value- mid == 0 then
      return mid+1
    else if then
      return binary_search(list, value, low, mid-1)
    else
      return binary_search(list, value, mid+1, high)
```

Data Structure

Stack

Stack is a linear data structure which follows a particular order in which the operations are performed. The order may be LIFO(Last In First Out) or FILO(First In Last Out).

ArrayList

ArrayList is a part of the Java collection framework and it is a class of java.util package. It provides us with dynamic arrays in Java.

LinkedList

Linked List is a part of the Collection framework present in java.util package. This class is an implementation of the LinkedList data structure which is a linear data structure where the elements are not stored in contiguous locations and every element is a separate object with a data part and address part.

Source Code

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.io.*;
import java.util.*;
import java.time.LocalDateTime;

class Student {
    private int roll;
    private String name;
    private String mark;
    private String grade;

    public Student() {
        this.roll = -1;
        this.name = "Default";
        this.mark = "0";
        this.grade = "F";
    }

    public Student(int roll, String name, String mark, String grade) {
        name.trim();
        mark.trim();
        name = name.toUpperCase();
        grade = grade.toUpperCase();

        if (name.isEmpty() || mark.isEmpty() || grade.isEmpty() || Integer.parseInt(mark) > 100
                || Integer.parseInt(mark) < 0) {
            this.roll = -1;
            this.name = "Default";
            this.mark = "0";
            this.grade = "F";
        } else {
            this.roll = roll;
            this.name = name;
            this.mark = mark;
            this.grade = grade;
        }
    }
    public String getName() {
        return this.name;
    }

    public int getRoll() {
        return this.roll;
    }

    public String getMark() {
        return this.mark;
    }

    public String getGrade() {
        return this.grade;
    }

    public void setName(String name) {
        this.name = name;
    }

    public void setMark(String mark) {
        this.mark = mark;
    }

    public void setGrade(String grade) {
        this.grade = grade;
    }
}

class System_Search {
    public static int System_Linear_Search(LinkedList<Student> students, int roll) {
        for (Student student : students) {
            if (student.getRoll() == roll) {
                return students.indexOf(student);
            }
        }
        return -1;
    }

    public static int System_Linear_Search(LinkedList<Student> students, Student entry) {
        for (Student student : students) {
            if ((entry.getName().compareTo(student.getName())) < 0) {
                return students.indexOf(student);
            }
        }
        return students.size();
    }

    public static int System_Binary_Search(LinkedList<Student> students, Student entry, int low, int
high) {
        int mid = (low + high) / 2;
        if (low == high) {
            if ((entry.getName().compareTo(students.get(mid).getName())) < 0) {
                return mid;
            } else {
                return mid + 1;
            }
        } else {
            if ((entry.getName().compareTo(students.get(mid).getName())) == 0) {
                return mid + 1;
            } else if ((entry.getName().compareTo(students.get(mid).getName())) < 0) {
                return System_Binary_Search(students, entry, low, mid - 1);
            } else {
                return System_Binary_Search(students, entry, mid + 1, high);
            }
        }
    }
}

class Add_Student_Panel extends JPanel {
    JPanel panel_header;
    JPanel panel_body;
    JPanel panel_footer;

    JTextField name_text;
    JTextField mark_text;
    JTextField grade_text;

    JButton add_details;
    System_Dialog add_dialog;

    public Add_Student_Panel(Student_Information_System MainFrame) {
        panel_header = new JPanel();
        panel_body = new JPanel();
        panel_footer = new JPanel();

        name_text = new JTextField();
        mark_text = new JTextField();
        grade_text = new JTextField();

        add_details = new JButton("Add Details");
        add_dialog = new System_Dialog(MainFrame, this);

        setLayout(new BorderLayout());
        add(panel_header, BorderLayout.NORTH);
        add(panel_body, BorderLayout.CENTER);
        add(panel_footer, BorderLayout.SOUTH);
        add(new JPanel(), BorderLayout.EAST);
        add(new JPanel(), BorderLayout.WEST);

        panel_header.add(new JLabel("Enter Student Details to Add"));
        panel_header.setBorder(BorderFactory.createLineBorder(Color.black));

        panel_body.setLayout(new BoxLayout(this.panel_body, BoxLayout.Y_AXIS));
        panel_body.add(new JLabel("Enter Student Name"));
        panel_body.add(name_text);
        panel_body.add(new JLabel("Enter Marks"));
        panel_body.add(mark_text);
        panel_body.add(new JLabel("Enter Grade"));
        panel_body.add(grade_text);
        panel_footer.setLayout(new GridLayout(2, 1));
        panel_footer.add(add_details);
        panel_footer.add(new Back_Button(MainFrame));

        add_details.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent event) {
                add_dialog.setVisible(true);
            }
        });
    }
}

class Delete_Student_Panel extends JPanel {
    JPanel panel_header;
    JPanel panel_body;
    JPanel panel_footer;

    JLabel name;
    JLabel mark;
    JLabel grade;

    JPanel searchArea;
    JPanel resultArea;

    JLabel roll_heading;
    JTextField roll_text;
    JButton search;

    JButton delete_details;
    System_Dialog delete_dialog;

    LinkedList<Student> students;
    System_Dialog error;

    public static Stack<Student> deleted_students = new Stack<>();

    public Delete_Student_Panel(Student_Information_System MainFrame) {
        panel_header = new JPanel();
        panel_body = new JPanel();
        panel_footer = new JPanel();

        name = new JLabel("-");
        mark = new JLabel("-");
        grade = new JLabel("-");

        searchArea = new JPanel();
        resultArea = new JPanel();

        roll_heading = new JLabel("Enter Roll No. ");
        roll_text = new JTextField();
        search = new JButton("Search");

        delete_details = new JButton("Delete Details");
        delete_dialog = new System_Dialog(MainFrame, this);
```

```
students = new LinkedList<>();
error = new System_Dialog(MainFrame, "Invalid Credentials");

setLayout(new BorderLayout());
add(panel_header, BorderLayout.NORTH);
add(panel_body, BorderLayout.CENTER);
add(panel_footer, BorderLayout.SOUTH);
add(new JPanel(), BorderLayout.EAST);
add(new JPanel(), BorderLayout.WEST);

panel_header.add(new JLabel("Enter Student Details to Delete"));
panel_header.setBorder(BorderFactory.createLineBorder(Color.black));

searchArea.setLayout(new BoxLayout(this.searchArea, BoxLayout.X_AXIS));
searchArea.add(roll_heading);
searchArea.add(roll_text);
searchArea.add(search);

resultArea.setLayout(new GridLayout(4, 2));
resultArea.add(new JLabel("Student Name"));
resultArea.add(name);
resultArea.add(new JLabel("Student Marks"));
resultArea.add(mark);
resultArea.add(new JLabel("Student Grade"));
resultArea.add(grade);
resultArea.setVisible(false);

panel_body.setLayout(new BorderLayout());
panel_body.add(searchArea, BorderLayout.NORTH);
panel_body.add(resultArea, BorderLayout.CENTER);

panel_footer.setLayout(new GridLayout(2, 1));
panel_footer.add(delete_details);
panel_footer.add(new Back_Button(MainFrame));

delete_details.setEnabled(false);
delete_details.addActionListener(new ActionListener() {
   public void actionPerformed(ActionEvent event) {
      delete_dialog.setVisible(true);
   }
});
search.addActionListener(new ActionListener() {
   public void actionPerformed(ActionEvent event) {
      try {
         Scanner scanner_students = new Scanner(new File("file_system\\student_details.txt"));
         while (scanner_students.hasNextLine()) {
            String[] elements = scanner_students.nextLine().split(",");
            Student record = new Student(Integer.parseInt(elements[0]), elements[1], elements[2],
                    elements[3]);
            students.add(record);
         }
         scanner_students.close();

         int roll;
```

```
                try {
                    roll = Integer.parseInt(roll_text.getText().trim());
                } catch (Exception exception) {
                    roll = -1;
                }

                int position = System_Search.System_Linear_Search(students, roll);

                if (position == -1 || roll == -1) {
                    error = new System_Dialog(MainFrame, "Invalid Credentials");
                    error.setVisible(true);
                    return;
                }
                delete_details.setEnabled(true);
                resultArea.setVisible(true);
                name.setText(students.get(position).getName());
                mark.setText(students.get(position).getMark());
                grade.setText(students.get(position).getGrade());
            } catch (Exception exception) {
                System.out.println(exception);
            }
        }
    });
  }
}

class Modify_Student_Panel extends JPanel {
    JPanel panel_header;
    JPanel panel_body;
    JPanel panel_footer;

    JPanel searchArea;
    JPanel resultArea;

    JLabel roll_heading;
    JTextField roll_text;
    JButton search;

    JTextField name;
    JTextField mark;
    JTextField grade;

    JButton modify_details;
    System_Dialog modify_dialog;

    LinkedList<Student> students;
    System_Dialog error;

    public Modify_Student_Panel(Student_Information_System MainFrame) {
        panel_header = new JPanel();
        panel_body = new JPanel();
        panel_footer = new JPanel();

        searchArea = new JPanel();
        resultArea = new JPanel();
```

```
roll_heading = new JLabel("Enter Roll No. ");
roll_text = new JTextField();
search = new JButton("Search");

name = new JTextField();
mark = new JTextField();
grade = new JTextField();

modify_details = new JButton("Modify Details");
modify_dialog = new System_Dialog(MainFrame, this);

students = new LinkedList<>();
error = new System_Dialog(MainFrame, "Invalid Credentials");

setLayout(new BorderLayout());
add(panel_header, BorderLayout.NORTH);
add(panel_body, BorderLayout.CENTER);
add(panel_footer, BorderLayout.SOUTH);
add(new JPanel(), BorderLayout.EAST);
add(new JPanel(), BorderLayout.WEST);

panel_header.add(new JLabel("Enter Student Details to Modify"));
panel_header.setBorder(BorderFactory.createLineBorder(Color.black));

searchArea.setLayout(new BoxLayout(this.searchArea, BoxLayout.X_AXIS));
searchArea.add(roll_heading);
searchArea.add(roll_text);
searchArea.add(search);

resultArea.setLayout(new BoxLayout(this.resultArea, BoxLayout.Y_AXIS));
resultArea.add(new JLabel("Student Name"));
resultArea.add(name);
resultArea.add(new JLabel("Student Marks"));
resultArea.add(mark);
resultArea.add(new JLabel("Student Grade"));
resultArea.add(grade);
resultArea.setVisible(false);

panel_body.setLayout(new BorderLayout());
panel_body.add(searchArea, BorderLayout.NORTH);
panel_body.add(resultArea, BorderLayout.CENTER);

panel_footer.setLayout(new GridLayout(2, 1));
panel_footer.add(modify_details);
panel_footer.add(new Back_Button(MainFrame));

modify_details.setEnabled(false);
modify_details.addActionListener(new ActionListener() {
   public void actionPerformed(ActionEvent event) {
      modify_dialog.setVisible(true);
   }
});

search.addActionListener(new ActionListener() {
```

```
            public void actionPerformed(ActionEvent event) {
                try {
                    Scanner scanner_students = new Scanner(new File("file_system\\student_details.txt"));
                    while (scanner_students.hasNextLine()) {
                        String[] elements = scanner_students.nextLine().split(",");
                        Student record = new Student(Integer.parseInt(elements[0]), elements[1], elements[2],
                                elements[3]);
                        students.add(record);
                    }
                    scanner_students.close();

                    int roll;
                    try {
                        roll = Integer.parseInt(roll_text.getText().trim());
                    } catch (Exception exception) {
                        roll = -1;
                    }

                    int position = System_Search.System_Linear_Search(students, roll);

                    if (position == -1 || roll == -1) {
                        error = new System_Dialog(MainFrame, "Invalid Credentials");
                        error.setVisible(true);
                        return;
                    }
                    modify_details.setEnabled(true);
                    resultArea.setVisible(true);
                    name.setText(students.get(position).getName());
                    mark.setText(students.get(position).getMark());
                    grade.setText(students.get(position).getGrade());
                } catch (Exception exception) {
                    System.out.println(exception);
                }
            }
        });
    }
}

class Search_Student_Panel extends JPanel {
    JPanel panel_header;
    JPanel panel_body;
    JPanel panel_footer;

    JLabel name;
    JLabel mark;
    JLabel grade;

    JPanel searchArea;
    JPanel resultArea;

    JLabel roll_heading;
    JTextField roll_text;
    JButton search;

    LinkedList<Student> students;
```

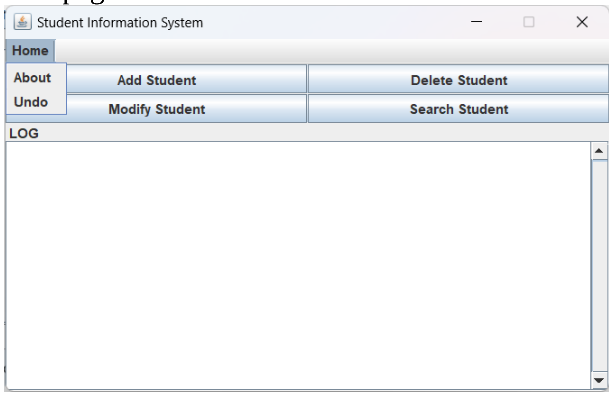
```
System_Dialog error;

public Search_Student_Panel(Student_Information_System MainFrame) {
    panel_header = new JPanel();
    panel_body = new JPanel();
    panel_footer = new JPanel();

    name = new JLabel("-");
    mark = new JLabel("-");
    grade = new JLabel("-");

    searchArea = new JPanel();
    resultArea = new JPanel();

    roll_heading = new JLabel("Enter Roll No. ");
    roll_text = new JTextField();
    search = new JButton("Search Details");

    students = new LinkedList<>();
    error = new System_Dialog(MainFrame, "Invalid Credentials");

    setLayout(new BorderLayout());
    add(panel_header, BorderLayout.NORTH);
    add(panel_body, BorderLayout.CENTER);
    add(panel_footer, BorderLayout.SOUTH);
    add(new JPanel(), BorderLayout.EAST);
    add(new JPanel(), BorderLayout.WEST);

    panel_header.add(new JLabel("Enter Student Details to Search"));
    panel_header.setBorder(BorderFactory.createLineBorder(Color.black));

    searchArea.setLayout(new BoxLayout(this.searchArea, BoxLayout.X_AXIS));
    searchArea.add(roll_heading);
    searchArea.add(roll_text);
    searchArea.add(search);

    resultArea.setLayout(new GridLayout(4, 2));
    resultArea.add(new JLabel("Student Name"));
    resultArea.add(name);
    resultArea.add(new JLabel("Student Marks"));
    resultArea.add(mark);
    resultArea.add(new JLabel("Student Grade"));
    resultArea.add(grade);
    resultArea.setVisible(false);

    panel_body.setLayout(new BorderLayout());
    panel_body.add(searchArea, BorderLayout.NORTH);
    panel_body.add(resultArea, BorderLayout.CENTER);

    panel_footer.setLayout(new GridLayout(2, 1));
    panel_footer.add(new JPanel());
    panel_footer.add(new Back_Button(MainFrame));

    search.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent event) {
                try {
                    Scanner scanner_students = new Scanner(new File("file_system\\student_details.txt"));
                    while (scanner_students.hasNextLine()) {
                        String[] elements = scanner_students.nextLine().split(",");
                        Student record = new Student(Integer.parseInt(elements[0]), elements[1], elements[2],
                                elements[3]);
                        students.add(record);
                    }
                    scanner_students.close();

                    int roll;
                    try {
                        roll = Integer.parseInt(roll_text.getText().trim());
                    } catch (Exception exception) {
                        roll = -1;
                    }

                    int position = System_Search.System_Linear_Search(students, roll);

                    if (position == -1 || roll == -1) {
                        error = new System_Dialog(MainFrame, "Invalid Credentials");
                        error.setVisible(true);
                        return;
                    }
                    resultArea.setVisible(true);
                    name.setText(students.get(position).getName());
                    mark.setText(students.get(position).getMark());
                    grade.setText(students.get(position).getGrade());
                } catch (Exception exception) {
                    System.out.println(exception);
                }
            }
        });
    }
}

class Back_Button extends JButton {
    Back_Button(Student_Information_System MainFrame) {
        setText("Back");
        addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent event) {
                System_Body_Panel.refresh_body_panel(MainFrame, new System_Body_Panel());
            }
        });
    }
}

class System_Menu extends JMenuBar {
    JMenu menu_1;
    JMenuItem menu_item_1;
    JMenuItem menu_item_2;
    System_Dialog menu_dialog;

    LinkedList<Student> students;
    System_Dialog error;
    Student entry;

    public System_Menu(Student_Information_System MainFrame) {
        menu_1 = new JMenu("Home");
        menu_item_1 = new JMenuItem("About");
        menu_item_2 = new JMenuItem("Undo");
        menu_dialog = new System_Dialog(MainFrame, this);

        students = new LinkedList<>();

        menu_1.add(menu_item_1);
        menu_1.add(menu_item_2);
        add(menu_1);

        menu_item_1.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent event) {
                menu_dialog.setVisible(true);
            }
        });
        menu_item_2.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent event) {
                try {
                    if (Delete_Student_Panel.deleted_students.isEmpty()) {
                        error = new System_Dialog(MainFrame, "No Elements To Recover");
                        error.setVisible(true);
                        return;
                    }

                    entry = Delete_Student_Panel.deleted_students.pop();

                    Scanner scanner_students = new Scanner(new File("file_system\\student_details.txt"));
                    while (scanner_students.hasNextLine()) {
                        String[] elements = scanner_students.nextLine().split(",");
                        Student record = new Student(Integer.parseInt(elements[0]), elements[1], elements[2],
                                elements[3]);
                        students.add(record);
                    }
                    scanner_students.close();

                    int position = System_Search.System_Binary_Search(students, entry, 0, students.size() -
1);

                    students.add(position, entry);

                    FileWriter write_student = new FileWriter("file_system\\student_details.txt");
                    for (Student student : students) {
                        write_student.write(student.getRoll() + "," + student.getName() + "," +
student.getMark() + ","
                                + student.getGrade() + "\n");
                    }
                    write_student.close();

                    FileWriter write_log = new FileWriter("file_system\\log_file.txt", true);
                    write_log.write("Restored Information of " + entry.getName() + "\t" +
LocalDateTime.now() + "\n");
```

```
                write_log.close();

                Scanner scanner_info = new Scanner(new File("file_system\\file_info.txt"));
                int info = Integer.parseInt(scanner_info.nextLine()) + 1;
                scanner_info.close();

                FileWriter write_info = new FileWriter("file_system\\file_info.txt");
                write_info.write("" + info);
                write_info.close();

                System_Body_Panel.refresh_body_panel(MainFrame, new System_Body_Panel());
            } catch (Exception exception) {
                System.out.println(exception);
            }
        }
    });
  }
}

class System_Dialog extends JDialog {
  JButton ok;
  JButton cancle;
  System_Dialog error;
  LinkedList<Student> students;
  boolean keep_track;

  public System_Dialog(Student_Information_System MainFrame, String errorMessage) {
    super(MainFrame, "Error", true);
    setSize(200, 75);
    setContentPane(new JLabel(errorMessage));
  }

  public System_Dialog(Student_Information_System MainFrame, System_Menu menuBar) {
    super(MainFrame, menuBar.menu_item_1.getText(), true);
    ok = new JButton("OK");
    cancle = new JButton("Cancle");
    setSize(200, 75);
    setContentPane(new JLabel("This is a Mini Project."));
  }

  public System_Dialog(Student_Information_System MainFrame, Add_Student_Panel addStudent)
{
    super(MainFrame, addStudent.add_details.getText(), true);
    ok = new JButton("OK");
    cancle = new JButton("Cancle");

    setSize(300, 100);
    setLayout(new FlowLayout());

    add(new JLabel("Are You Sure?"));
    add(ok);
    add(cancle);

    students = new LinkedList<>();
```

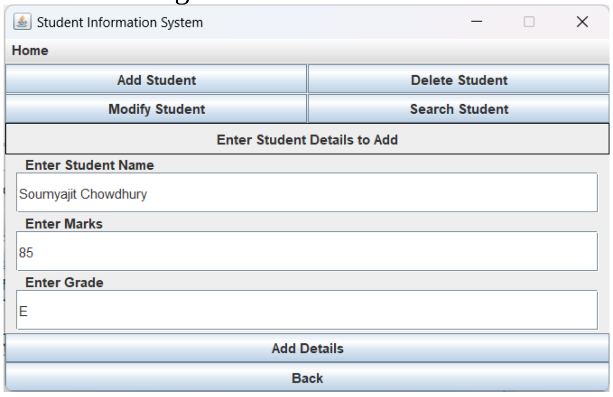
```
        ok.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent event) {
                try {
                    int roll = 0;
                    Scanner scanner_students = new Scanner(new File("file_system\\student_details.txt"));
                    while (scanner_students.hasNextLine()) {
                        String[] elements = scanner_students.nextLine().split(",");
                        Student record = new Student(Integer.parseInt(elements[0]), elements[1], elements[2],
                                elements[3]);
                        roll = roll>record.getRoll()?roll:record.getRoll();
                        students.add(record);
                    }
                    scanner_students.close();

                    Student entry = new Student(roll+1, addStudent.name_text.getText(),
addStudent.mark_text.getText(),
                            addStudent.grade_text.getText());

                    if (entry.getRoll() == -1) {
                        error = new System_Dialog(MainFrame, "Invalid Credentials");
                        error.setVisible(true);
                        return;
                    }

                    int position = System_Search.System_Binary_Search(students, entry, 0, students.size() -
1);

                    students.add(position, entry);

                    FileWriter write_student = new FileWriter("file_system\\student_details.txt");
                    for (Student student : students) {
                        write_student.write(student.getRoll() + "," + student.getName() + "," +
student.getMark() + ","
                                + student.getGrade() + "\n");
                    }
                    write_student.close();

                    FileWriter write_log = new FileWriter("file_system\\log_file.txt", true);
                    write_log.write("Added Information of " + entry.getName() + "\t" +
LocalDateTime.now() + "\n");
                    write_log.close();

                    FileWriter write_info = new FileWriter("file_system\\file_info.txt");
                    write_info.write("" + entry.getRoll());
                    write_info.close();

                    setVisible(false);

                    System_Body_Panel.refresh_body_panel(MainFrame, new System_Body_Panel());
                } catch (Exception exception) {
                    System.out.println(exception);
                }
            }
        });
        cancle.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent event) {
                setVisible(false);
            }
        });
    }

    public System_Dialog(Student_Information_System MainFrame, Delete_Student_Panel
deleteStudent) {
        super(MainFrame, deleteStudent.delete_details.getText(), true);
        ok = new JButton("OK");
        cancle = new JButton("Cancle");

        setSize(300, 100);
        setLayout(new FlowLayout());

        add(new JLabel("Are You Sure?"));
        add(ok);
        add(cancle);

        students = new LinkedList<>();

        ok.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent event) {
                try {
                    int roll = Integer.parseInt(deleteStudent.roll_text.getText());

                    Scanner scanner_students = new Scanner(new File("file_system\\student_details.txt"));
                    while (scanner_students.hasNextLine()) {
                        String[] elements = scanner_students.nextLine().split(",");
                        Student record = new Student(Integer.parseInt(elements[0]), elements[1], elements[2],
                                elements[3]);
                        students.add(record);
                    }
                    scanner_students.close();

                    int position = System_Search.System_Linear_Search(students, roll);

                    if (position == -1) {
                        error = new System_Dialog(MainFrame, "Invalid Credentials");
                        error.setVisible(true);
                        return;
                    }
                    FileWriter write_student = new FileWriter("file_system\\student_details.txt");
                    for (Student student : students) {
                        if (students.get(position).equals(student)) {
                            Delete_Student_Panel.deleted_students.push(student);
                            continue;
                        }
                        write_student.write(student.getRoll() + "," + student.getName() + "," +
student.getMark()
                                + "," + student.getGrade() + "\n");
                    }
                    write_student.close();

                    FileWriter write_log = new FileWriter("file_system\\log_file.txt", true);
                    write_log.write("Deleted Information of " + students.get(position).getName() + "\t"
                            + LocalDateTime.now() + "\n");
                    write_log.close();

                    Scanner scanner_info = new Scanner(new File("file_system\\file_info.txt"));
                    int info = Integer.parseInt(scanner_info.nextLine()) - 1;
                    scanner_info.close();

                    FileWriter write_info = new FileWriter("file_system\\file_info.txt");
                    write_info.write("" + info);
                    write_info.close();

                    setVisible(false);

                    System_Body_Panel.refresh_body_panel(MainFrame, new System_Body_Panel());
                } catch (Exception exception) {
                    System.out.println(exception);
                }
            }
        });
        cancle.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent event) {
                setVisible(false);
            }
        });
    }

    public System_Dialog(Student_Information_System MainFrame, Modify_Student_Panel
modifyStudent) {
        super(MainFrame, modifyStudent.modify_details.getText(), true);
        ok = new JButton("OK");
        cancle = new JButton("Cancle");
        setSize(300, 100);
        setLayout(new FlowLayout());
        add(new JLabel("Are You Sure?"));
        add(ok);
        add(cancle);

        students = new LinkedList<>();

        ok.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent event) {
                try {
                    int roll = Integer.parseInt(modifyStudent.roll_text.getText());

                    Scanner scanner_students = new Scanner(new File("file_system\\student_details.txt"));
                    while (scanner_students.hasNextLine()) {
                        String[] elements = scanner_students.nextLine().split(",");
                        Student record = new Student(Integer.parseInt(elements[0]), elements[1], elements[2],
                                elements[3]);
                        students.add(record);
                    }
                    scanner_students.close();

                    int position = System_Search.System_Linear_Search(students, roll);
                    if (position == -1) {
                        error = new System_Dialog(MainFrame, "Invalid Credentials");
                        error.setVisible(true);
                        return;
                    }

                    students.get(position).setName(modifyStudent.name.getText());
                    students.get(position).setMark(modifyStudent.mark.getText());
                    students.get(position).setGrade(modifyStudent.grade.getText());

                    FileWriter write_student = new FileWriter("file_system\\student_details.txt");
                    for (Student student : students) {
                        write_student.write(student.getRoll() + "," + student.getName() + "," +
student.getMark() + ","
                                + student.getGrade() + "\n");
                    }
                    write_student.close();

                    FileWriter write_log = new FileWriter("file_system\\log_file.txt", true);
                    write_log.write("Modified Information of " + students.get(position).getName() + "\t"
                            + LocalDateTime.now() + "\n");
                    write_log.close();

                    setVisible(false);

                    System_Body_Panel.refresh_body_panel(MainFrame, new System_Body_Panel());
                } catch (Exception exception) {
                    System.out.println(exception);
                }
            }
        });
        cancle.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent event) {
                setVisible(false);
            }
        });
    }
}

class System_Head_Panel extends JPanel {
    JButton add_student;
    JButton delete_student;
    JButton modify_student;
    JButton search_student;

    public System_Head_Panel(Student_Information_System MainFrame) {
        add_student = new JButton("Add Student");
        delete_student = new JButton("Delete Student");
        modify_student = new JButton("Modify Student");
        search_student = new JButton("Search Student");

        setLayout(new GridLayout(2, 4));
        add(add_student);
        add(delete_student);
        add(modify_student);
        add(search_student);

        add_student.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent event) {
                System_Body_Panel.refresh_body_panel(MainFrame, new
Add_Student_Panel(MainFrame));
            }
        });
        delete_student.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent event) {
                System_Body_Panel.refresh_body_panel(MainFrame, new
Delete_Student_Panel(MainFrame));
            }
        });
        modify_student.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent event) {
                System_Body_Panel.refresh_body_panel(MainFrame, new
Modify_Student_Panel(MainFrame));
            }
        });
        search_student.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent event) {
                System_Body_Panel.refresh_body_panel(MainFrame, new
Search_Student_Panel(MainFrame));
            }
        });
    }
}

class System_Body_Panel extends JPanel {
    JLabel header;
    JTextArea textArea;
    JScrollPane scrollableTextArea;
    File file;
    Scanner scanner;

    public System_Body_Panel() {
        header = new JLabel(" LOG");
        textArea = new JTextArea();
        scrollableTextArea = new JScrollPane(textArea);

        setLayout(new BorderLayout());
        textArea.setEditable(false);
        scrollableTextArea.setVerticalScrollBarPolicy(JScrollPane.VERTICAL_SCROLLBAR_ALWA
YS);

        try {
            file = new File("file_system\\log_file.txt");
            scanner = new Scanner(file);
            while (scanner.hasNextLine()) {
                textArea.append(scanner.nextLine());
                textArea.append("\n");
            }
        } catch (Exception exception) {
            System.out.println(exception);
        }

        scanner.close();
        add(header, BorderLayout.NORTH);
        add(scrollableTextArea, BorderLayout.CENTER);
    }

    public static void refresh_body_panel(Student_Information_System MainFrame, JPanel 
BodyPanel) {
        MainFrame.remove(MainFrame.body_panel);
        MainFrame.revalidate();
        MainFrame.repaint();
        MainFrame.body_panel = BodyPanel;
        MainFrame.add(MainFrame.body_panel, BorderLayout.CENTER);
        MainFrame.revalidate();
        MainFrame.repaint();
    }
}

class Student_Information_System extends JFrame {
    JPanel head_panel;
    JPanel body_panel;

    public Student_Information_System(String app_name) {
        super(app_name);
        head_panel = new System_Head_Panel(this);
        body_panel = new System_Body_Panel();

        setSize(550, 350);
        setLayout(new BorderLayout());

        setJMenuBar(new System_Menu(this));
        add(head_panel, BorderLayout.NORTH);
        add(body_panel, BorderLayout.CENTER);

        setResizable(false);
        setVisible(true);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}

public class User_Interface {
    public static void main(String[] args) {
        SwingUtilities.invokeLater(
                new Runnable() {
                    public void run() {
                        new Student_Information_System("Student Information System");
                    }
                });
    }
}
```

Output

Homepage



Add Student Page

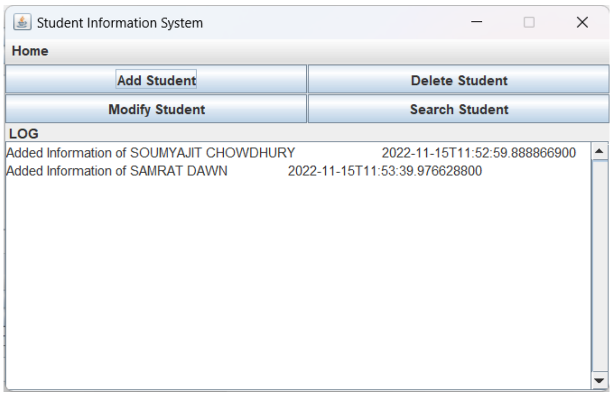


Enter Student Details to Add

Enter Student Name: Soumyajit Chowdhury

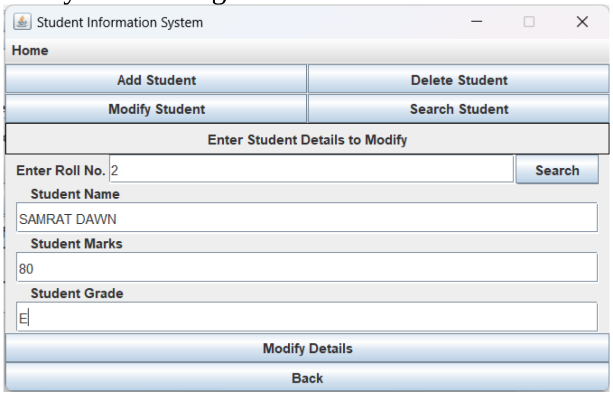
Enter Marks: 85

Enter Grade: E



```
Added Information of SOUMYAJIT CHOWDHURY          2022-11-15T11:52:59.888866900
Added Information of SAMRAT DAWN          2022-11-15T11:53:39.976628800
```

Modify Student Page



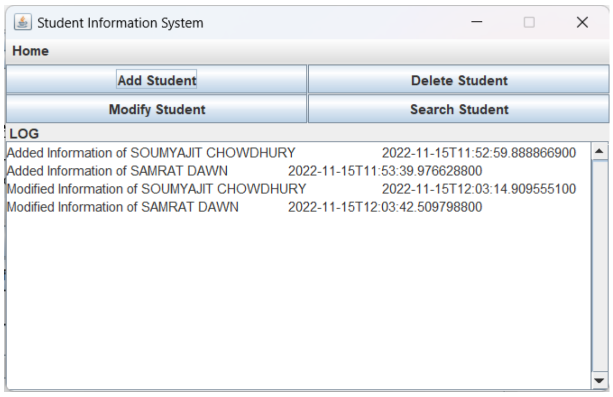
Enter Student Details to Modify

Enter Roll No. 2

Student Name: SAMRAT DAWN

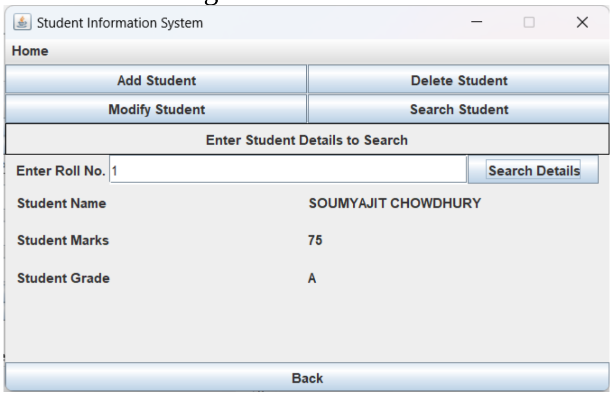
Student Marks: 80

Student Grade: E



```
Added Information of SOUMYAJIT CHOWDHURY          2022-11-15T11:52:59.888866900
Added Information of SAMRAT DAWN          2022-11-15T11:53:39.976628800
Modified Information of SOUMYAJIT CHOWDHURY          2022-11-15T12:03:14.909555100
Modified Information of SAMRAT DAWN          2022-11-15T12:03:42.509798800
```

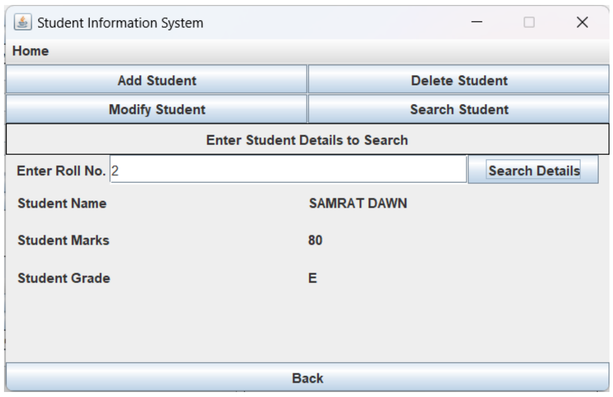
Search Student Page



Enter Student Details to Search

Enter Roll No. 1

Student Name	SOUMYAJIT CHOWDHURY
Student Marks	75
Student Grade	A

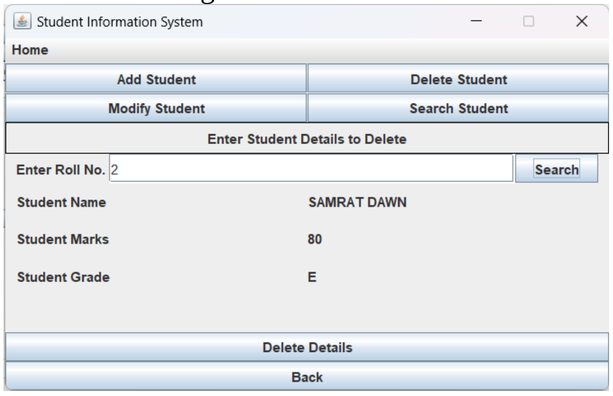


Enter Student Details to Search

Enter Roll No. 2

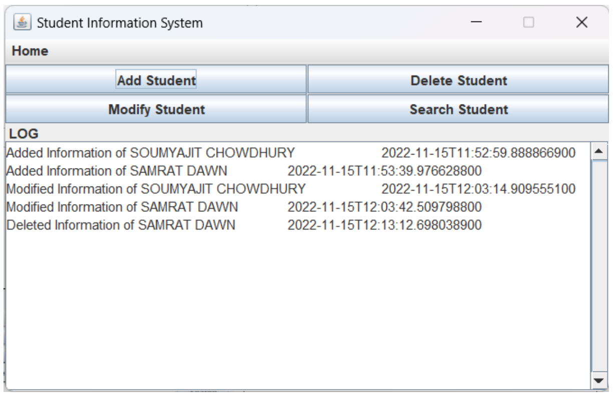
Student Name	SAMRAT DAWN
Student Marks	80
Student Grade	E

Delete Student Page



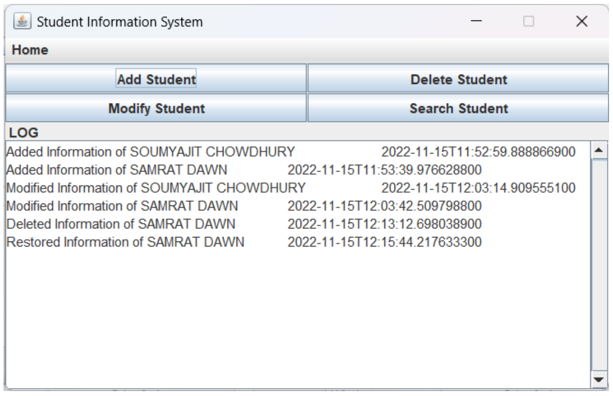
Enter Roll No. 2

Student Name	SAMRAT DAWN
Student Marks	80
Student Grade	E



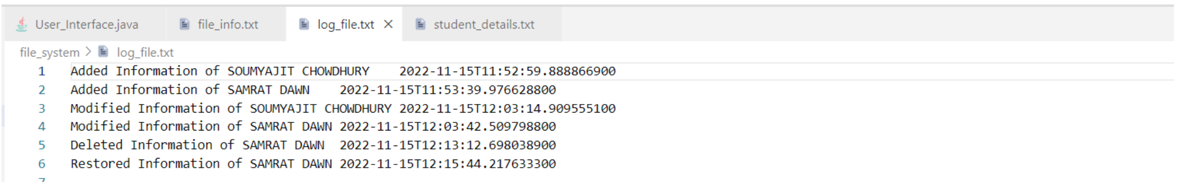
```
Added Information of SOUMYAJIT CHOWDHURY        2022-11-15T11:52:59.888866900
Added Information of SAMRAT DAWN        2022-11-15T11:53:39.976628800
Modified Information of SOUMYAJIT CHOWDHURY        2022-11-15T12:03:14.909555100
Modified Information of SAMRAT DAWN        2022-11-15T12:03:42.509798800
Deleted Information of SAMRAT DAWN        2022-11-15T12:13:12.698038900
```

Restore Information

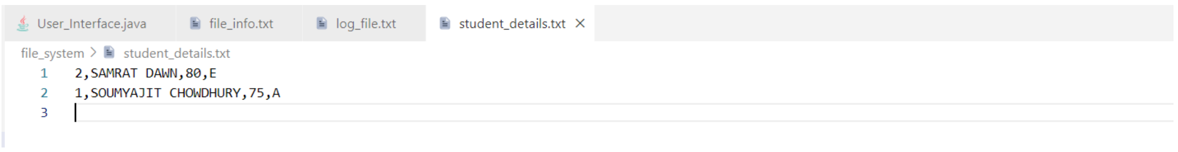


```
Added Information of SOUMYAJIT CHOWDHURY        2022-11-15T11:52:59.888866900
Added Information of SAMRAT DAWN        2022-11-15T11:53:39.976628800
Modified Information of SOUMYAJIT CHOWDHURY        2022-11-15T12:03:14.909555100
Modified Information of SAMRAT DAWN        2022-11-15T12:03:42.509798800
Deleted Information of SAMRAT DAWN        2022-11-15T12:13:12.698038900
Restored Information of SAMRAT DAWN        2022-11-15T12:15:44.217633300
```

File Information



```
Added Information of SOUMYAJIT CHOWDHURY    2022-11-15T11:52:59.888866900
Added Information of SAMRAT DAWN    2022-11-15T11:53:39.976628800
Modified Information of SOUMYAJIT CHOWDHURY 2022-11-15T12:03:14.909555100
Modified Information of SAMRAT DAWN 2022-11-15T12:03:42.509798800
Deleted Information of SAMRAT DAWN  2022-11-15T12:13:12.698038900
Restored Information of SAMRAT DAWN 2022-11-15T12:15:44.217633300
```



```
2,SAMRAT DAWN,80,E
1,SOUMYAJIT CHOWDHURY,75,A
```