Programming Assignment Unit 7

Godfrey Ouma

University of the People

CS 1102: Programming 1

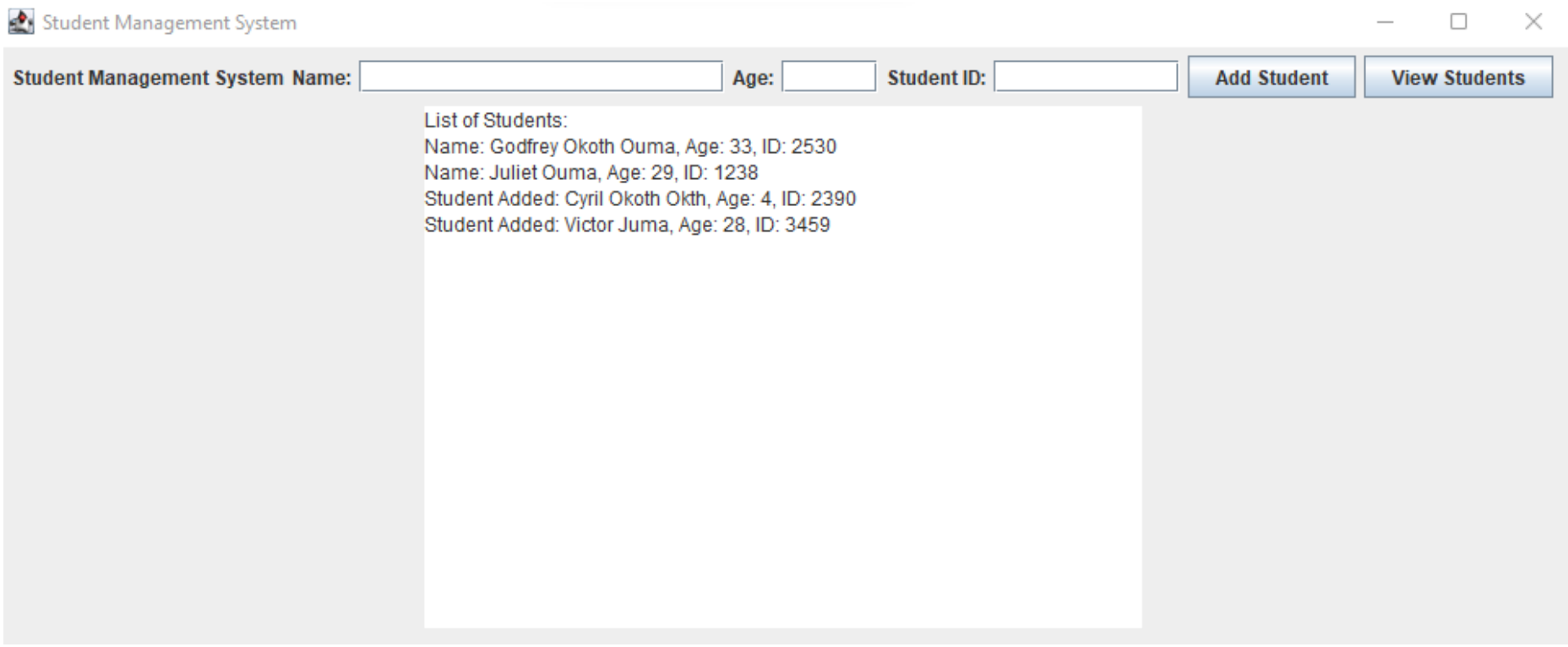
Ruth Alabi

January 4, 2024

```java
package cs;
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.ArrayList;
public class StudentManagementSystem extends JFrame {
    private ArrayList<Student> students; // List to store student records

    // GUI components
    private JLabel titleLabel;
    private JTextField nameField, ageField, idField;
    private JButton addButton, viewButton;
    private JTextArea displayArea;

    public StudentManagementSystem() {
        setTitle("Student Management System");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new FlowLayout());

        // Initialize the list of students
        students = new ArrayList<>();

        // Components initialization
        titleLabel = new JLabel("Student Management System");
        JLabel nameLabel = new JLabel("Name:");
        JLabel ageLabel = new JLabel("Age:");
        JLabel idLabel = new JLabel("Student ID:");
        nameField = new JTextField(20);
        ageField = new JTextField(5);
        idField = new JTextField(10);
        addButton = new JButton("Add Student");
        viewButton = new JButton("View Students");
        displayArea = new JTextArea(20, 40);

        // Add components to the frame
        add(titleLabel);
        add(nameLabel);
        add(nameField);
        add(ageLabel);
        add(ageField);
        add(idLabel);
        add(idField);
        add(addButton);
        add(viewButton);
        add(displayArea);

        // Event listeners
        addButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                addStudent();
            }
        });

        viewButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                viewStudents();
            }
        });
```

```
        pack();
        setVisible(true);
    }

    // Method to add student details to the list
    private void addStudent() {
        String name = nameField.getText();
        int age = Integer.parseInt(ageField.getText());
        int id = Integer.parseInt(idField.getText());
        students.add(new Student(name, age, id));
        displayArea.append("Student Added: " + name + ", Age: " + age + ", ID: " + id +
"\n");
        clearFields();
    }

    // Method to display all students in the JTextArea
    private void viewStudents() {
        displayArea.setText("");
        if (students.isEmpty()) {
            displayArea.append("No students available.");
        } else {
            displayArea.append("List of Students:\n");
            for (Student student : students) {
                displayArea.append("Name: " + student.getName() + ", Age: " +
student.getAge() + ", ID: " + student.getId() + "\n");
            }
        }
    }

    // Method to clear input fields after adding a student
    private void clearFields() {
        nameField.setText("");
        ageField.setText("");
        idField.setText("");
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                new StudentManagementSystem();
            }
        });
    }
}

class Student {
    private String name;
    private int age;
    private int id;

    public Student(String name, int age, int id) {
        this.name = name;
        this.age = age;
        this.id = id;
    }

    public String getName() {
        return name;
```

```
    }

    public int getAge() {
        return age;
    }

    public int getId() {
        return id;
    }
}
```


Student Management System
Student Management System Name:
Age:
Student ID:
Add Student
View Students
List of Students:
Name: Godfrey Okoth Ouma, Age: 33, ID: 2530
Name: Juliet Ouma, Age: 29, ID: 1238
Student Added: Cyril Okoth Okth, Age: 4, ID: 2390
Student Added: Victor Juma, Age: 28, ID: 3459

Student Management System GUI Application Documentation

Purpose:

The Student Management System GUI application is designed to assist administrators in managing student records efficiently. It provides a user-friendly interface for adding new students, viewing student details, and storing essential information such as name, age, and student ID.

Functionality:

Components:

1. **titleLabel:** Displays the title "Student Management System" at the top of the interface.
2. **nameField:** Allows the user to input the student's name.
3. **ageField:** Enables the user to input the student's age.
4. **idField:** Accepts the input of the student's ID.
5. **addButton:** Button to add a new student to the records.
6. **viewButton:** Button to view all students' details.
7. **displayArea:** JTextArea to display the list of students or messages.

Event Handlers:

1. **addButton ActionListener:** Handles the addition of a new student by capturing input from nameField, ageField, and idField. It then creates a new Student object and adds it to the students' list. Displays a confirmation message in the displayArea.

2. **viewButton ActionListener:** Displays all stored student records in the displayArea.

Methods:

1. **addStudent():** Method to add a new student to the students' list. It validates and retrieves input from text fields, creates a new Student object, and updates the displayArea with the student's details.
2. **viewStudents():** Displays the list of all students' details in the displayArea.
3. **clearFields():** Clears the input fields after adding a new student.

Student Class:

The **Student** class defines the blueprint for a Student object with attributes like name, age, and ID. It includes accessor methods to retrieve these attributes.

Usage:

1. **Adding a Student:**
    - Enter the student's name, age, and ID in the respective text fields.
    - Click the "Add Student" button to add the student to the records.
    - The student's details will be displayed in the text area.
2. **Viewing Students:**
    - Click the "View Students" button to display the list of all stored students' details in the text area.

Usage Example:

1. Adding a Student:
    - Name: [Enter Student's Name]
    - Age: [Enter Student's Age]
    - ID: [Enter Student's ID]
    - Click "Add Student"
2. Viewing Students:
    - Click "View Students" to display the list of students and their details.

Notes:

- Ensure valid inputs are provided for age and ID.
- Empty fields may cause errors; fill all fields before adding a new student.

Conclusion:

The Student Management System GUI application provides an intuitive interface for managing student records. It allows for the addition of new students and viewing all stored student details, providing administrators with an easy-to-use tool for managing student information.