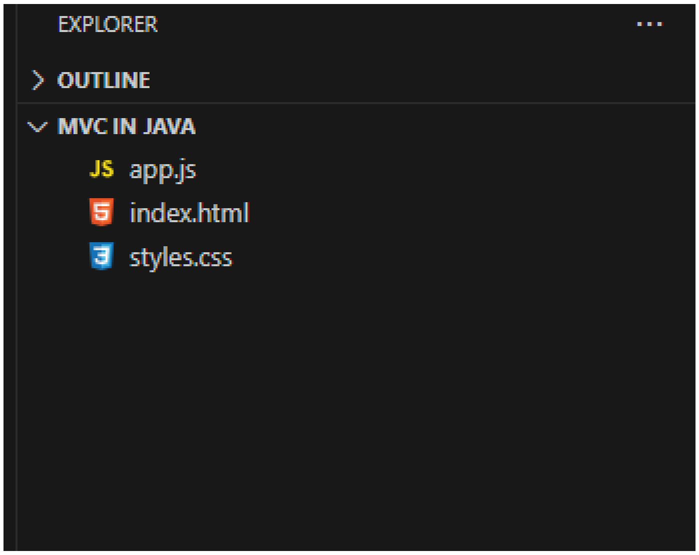
Explanation Mvc Pattern

1. **HTML (View)**:
   - Create a basic layout to display the app.
   - Include input fields for entering the student name and roll number.
   - Add a button to trigger the "Add Student" functionality.
   - Provide an empty list element (<ul>) to display the list of students.
2. **CSS**:
   - Style the app for a clean and modern look.
   - Add spacing, alignment, and colors for better user experience.
3. **JavaScript**:
   - **Model (Data)**:
     - Define a Student class with attributes: name and rollNo.
   - **View (Display)**:
     - The StudentView class manages all DOM manipulations (displaying students or showing messages).
     - Provide methods to update the list of students and alert messages.
   - **Controller (Logic)**:
     - The StudentController class connects the user interactions with the data and view.
     - Define a method to add a new student to the list.
     - Bind the "Add Student" button to a function that reads input, creates a Student object, and updates the view.
4. **Initialization**:
   - Create instances of StudentView and StudentController to link the MVC components.
   - Automatically bind the controller's methods to the UI elements.

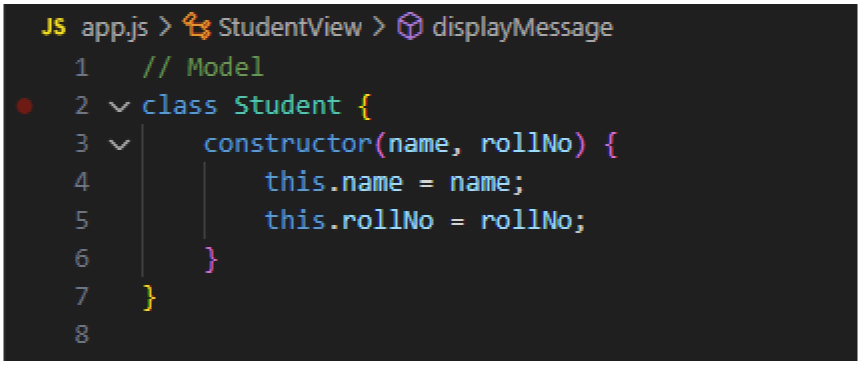
How It Works

- The user enters a name and roll number and clicks the "Add Student" button.
- The **Controller** reads the input, creates a new **Student (Model)**, and adds it to the array.
- The **View** updates the displayed list of students and shows an alert message for feedback.

Code:



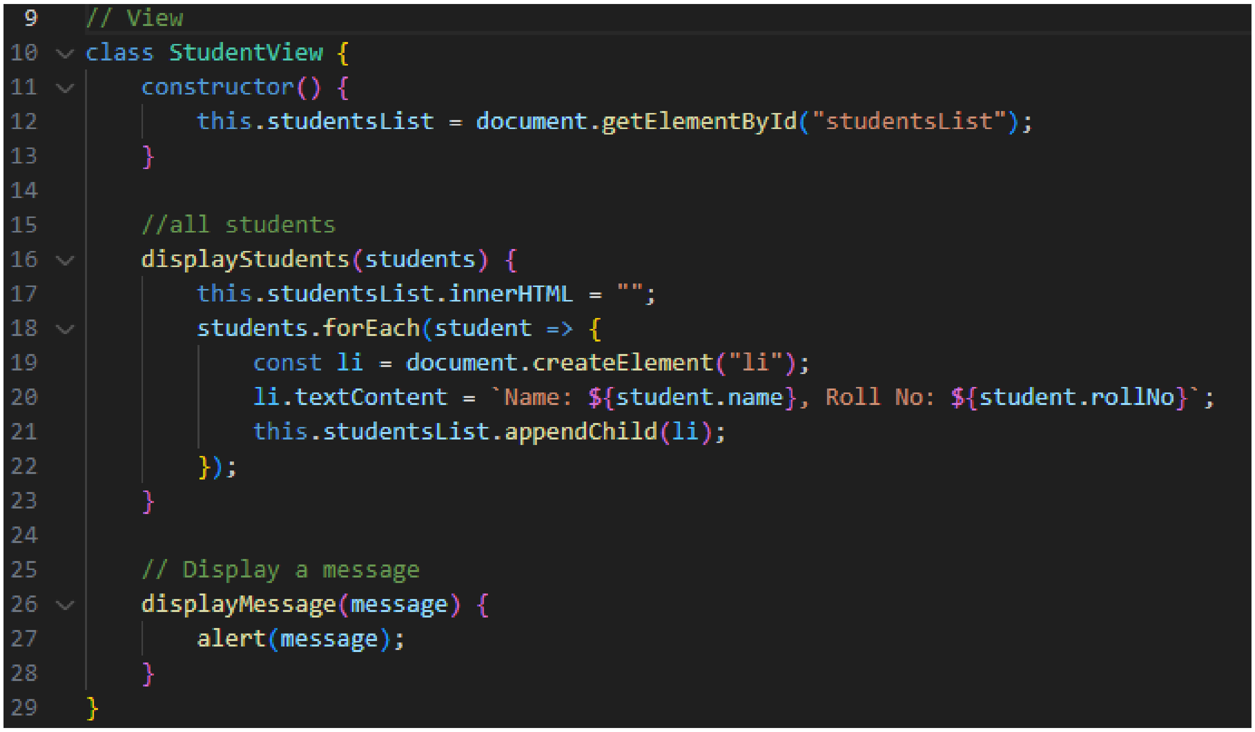
Model:



```
// Model
class Student {
    constructor(name, rollNo) {
        this.name = name;
        this.rollNo = rollNo;
    }
}

```

View:

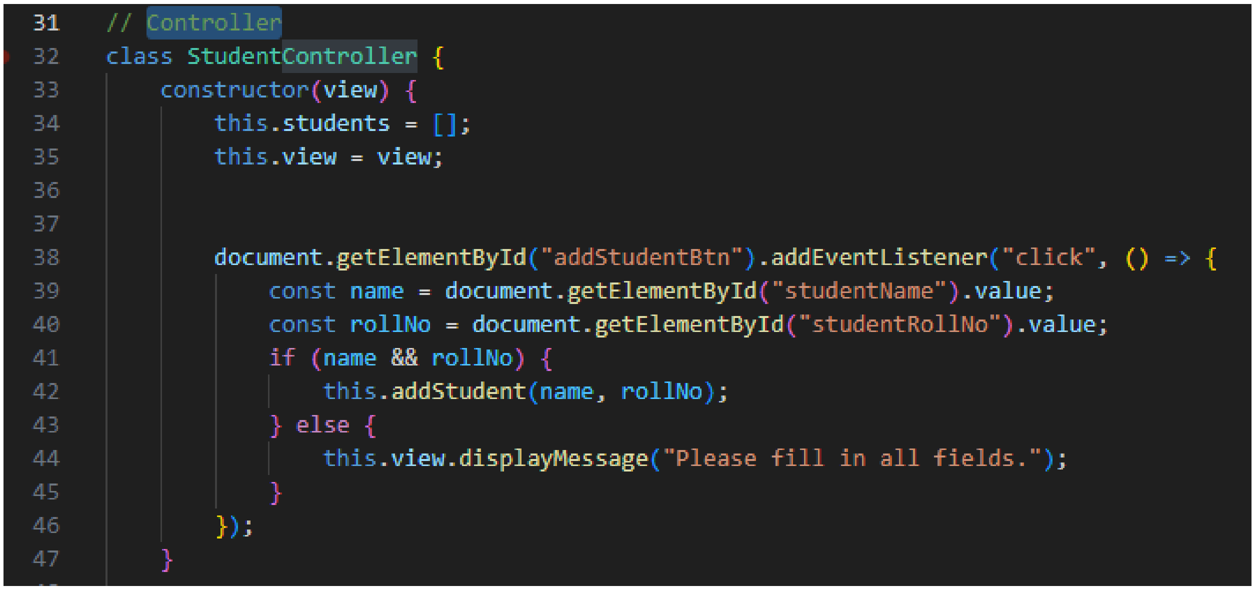


```javascript
// View
class StudentView {
    constructor() {
        this.studentsList = document.getElementById("studentsList");
    }

    //all students
    displayStudents(students) {
        this.studentsList.innerHTML = "";
        students.forEach(student => {
            const li = document.createElement("li");
            li.textContent = `Name: ${student.name}, Roll No: ${student.rollNo}`;
            this.studentsList.appendChild(li);
        });
    }

    // Display a message
    displayMessage(message) {
        alert(message);
    }
}
```

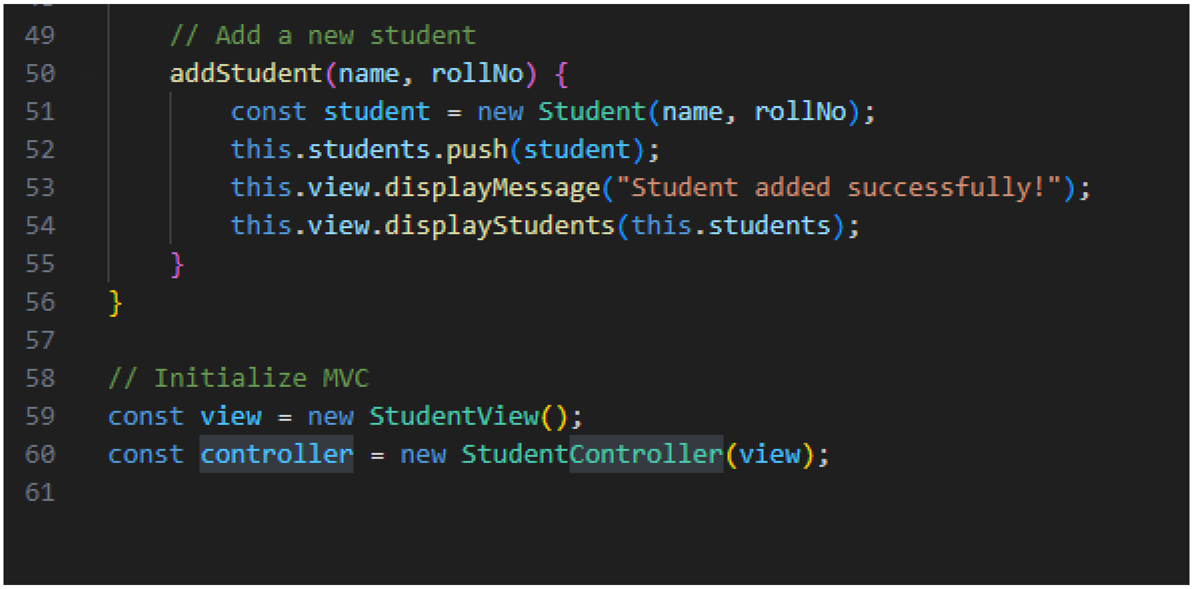
Controller:



```javascript
// Controller
class StudentController {
    constructor(view) {
        this.students = [];
        this.view = view;


        document.getElementById("addStudentBtn").addEventListener("click", () => {
            const name = document.getElementById("studentName").value;
            const rollNo = document.getElementById("studentRollNo").value;
            if (name && rollNo) {
                this.addStudent(name, rollNo);
            } else {
                this.view.displayMessage("Please fill in all fields.");
            }
        });
    }
```

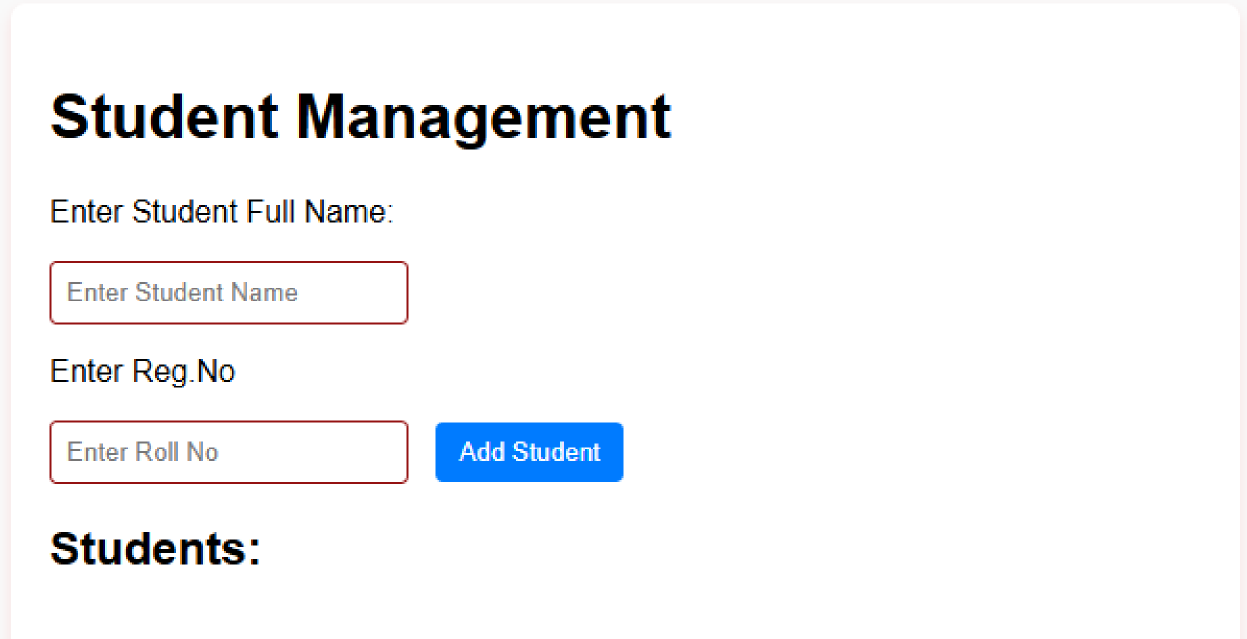
Initialize MVC:



```
    // Add a new student
    addStudent(name, rollNo) {
        const student = new Student(name, rollNo);
        this.students.push(student);
        this.view.displayMessage("Student added successfully!");
        this.view.displayStudents(this.students);
    }
}

// Initialize MVC
const view = new StudentView();
const controller = new StudentController(view);
```

Output:



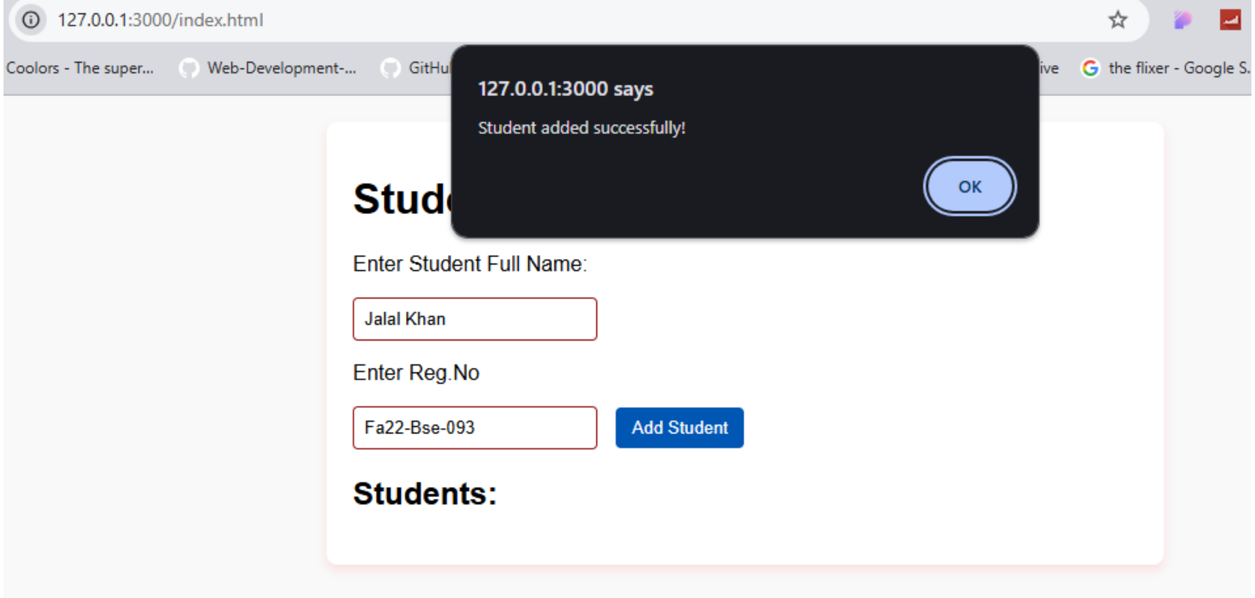
Student Management

Enter Student Full Name:

Enter Reg.No

Students:

Fill:



127.0.0.1:3000 says

Student added successfully!

Stud

Enter Student Full Name:

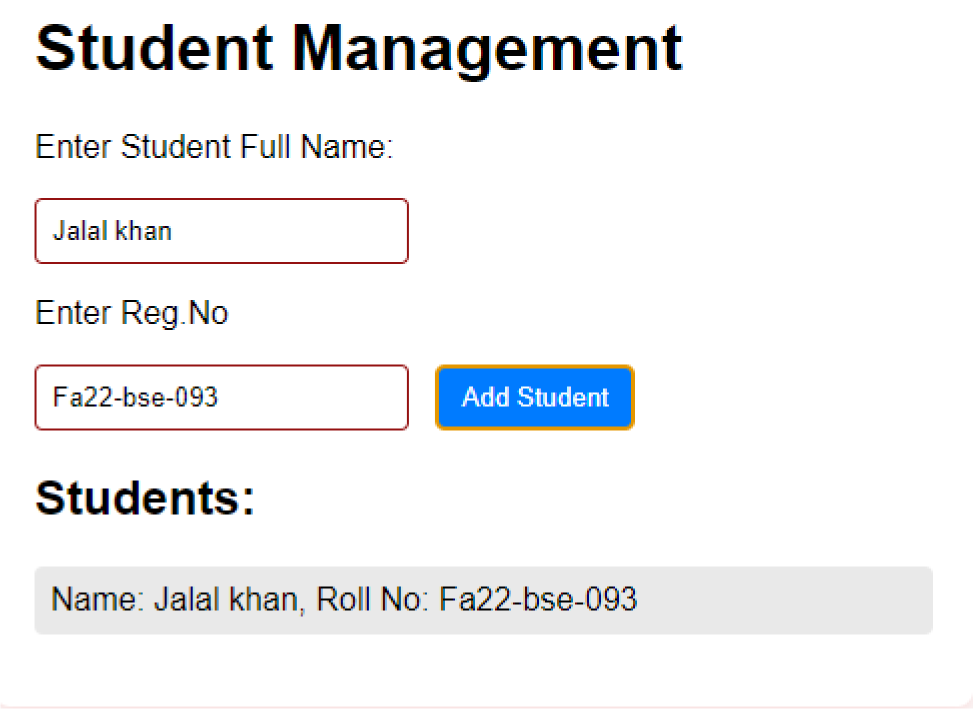
Jalal Khan

Enter Reg.No

Fa22-Bse-093

Students:

Final Output:



Student Management

Enter Student Full Name:

Jalal khan

Enter Reg.No

Fa22-bse-093

Students:

Name: Jalal khan, Roll No: Fa22-bse-093