Titles

Assignment Name – Student Registration System

Build by – Shashikant Sahu

Batch – 1st March 2025

Table of Contents

Introduction

Assignment Overview

A simple web application built using HTML, CSS, and JavaScript DOM to manage student registration records. Users can register student information, view registered records, and perform edit and delete operations with data persistence using browser local storage.

Organized File Structure

The project follows a clean and modular file organization, making it easy to maintain, scale, and understand:

Description of Files:

- **index.html**
  Contains the complete structure of the Student Registration System webpage, including:
  - A header with the system title and description.
  - A form section to input student details such as Name, ID, Email, and Contact.
  - A display section using a dynamic HTML table to list registered students.
  - Links to external resources like CSS and JavaScript files within the <head> and before the closing <body> tag.
- **style.css**
  Defines the entire visual appearance of the application using:
  - **Flexbox** for form layout and spacing.
  - **Table styling** for data presentation.
  - Custom colors, borders, paddings, and hover effects.
  - Dynamic scrollable content in the table for overflow data.
- **script.js**
  Contains all JavaScript logic using **pure DOM methods**, including:
  - Form input validation (regex for email, numeric fields, and name).
  - Creating, updating, and deleting student records dynamically.
  - Storing and retrieving data using **localStorage** to preserve records on page reload.
  - No innerHTML-based manipulation—everything is built using document.createElement() and DOM APIs.

Creativity and Presentation

The Student Registration System is built with usability and functionality in mind, ensuring a smooth and engaging user experience for registering and managing student records.

Design Approach:

- Minimalistic & Clean UI
  Uses a simple, uncluttered layout with clear input fields, subtle colors, and consistent font styling to enhance readability.

Key UI Elements:

- Header Section
  Displays the title and brief description of the system at the top, setting the purpose clearly.
- Registration Form
  Allows users to input student details like Name, Student ID, Email, and Contact No. Fields are validated to prevent incorrect data.
- Student Records Table
  A dynamically generated table that shows all registered students. Each row includes Edit and Delete buttons for user control.

---

1. Technologies Used

Technology	Purpose
HTML5	Webpage structure and semantic layout
CSS3	Styling, form layout, table design
JavaScript (DOM)	Logic for adding, editing, deleting records; validation; localStorage

2. Code

.html code :-

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Student Registration System</title>
    <link rel="stylesheet" href="style.css">
</head>

<body>
    <header>
        <h1 class="heading">Student Registration System</h1>
        <p>Register, edit, and manage student information with ease.</p>
    </header>

    <main>
        <section class="form-section">
            <form id="registrationForm">
                <input type="text" id="name" placeholder="Student Name" required>
                <input type="number" id="studentId" placeholder="Student ID" required>
                <input type="email" id="email" placeholder="Email ID" required>
                <input type="number" id="contact" placeholder="Contact No." required>
                <button id="submit" type="submit">Register</button>
            </form>
        </section>

        <section class="display-section">
            <h2 class="heading">Registered Students</h2>
            <table id="studentTable">
                <thead>
```

```
                <tr>
                    <th>Name</th>
                    <th>Student ID</th>
                    <th>Email</th>
                    <th>Contact</th>
                    <th>Actions</th>
                </tr>
            </thead>
            <tbody></tbody>
        </table>
    </section>
</main>

<script src="javaScript.js"></script>
</body>

</html>
```

.css code :-

```
body {
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0;
    background-color: #f2f2f2;
  }

  header {
    background-color: #4CAF50;
    color: white;
    padding: 1rem;
    text-align: center;
  }

  main {
    padding: 2rem;
    max-width: 800px;
    margin: auto;
  }

  form {
    display: flex;
    flex-direction: column;
    gap: 1rem;
    background: #fff;
    padding: 1.5rem;
    border-radius: 8px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
  }

  input, button {
    padding: 0.8rem;
    border: 1px solid #ccc;
    border-radius: 5px;
  }
```

```
button {
  background-color: #4CAF50;
  color: white;
  cursor: pointer;
  font-weight: bold;
}

button:hover {
  background-color: #23a62a;
}

table {
  width: 100%;
  margin-top: 2rem;
  border-collapse: collapse;
  background: white;
  border-radius: 8px;
  overflow: hidden;
}

th, td {
  padding: 0.75rem;
  text-align: left;
  border-bottom: 1px solid #ddd;
}

th {
  background-color: #4CAF50;
  color: white;
}

tbody {
  display: block;
  max-height: 300px;
  overflow-y: auto;
}
```

```
 thead, tbody tr {
   display: table;
   width: 100%;
   table-layout: fixed;
 }

 .actions button {
   margin-right: 0.5rem;
   padding: 0.4rem 0.8rem;
   font-size: 0.9rem;
 }
 .heading {
   text-align: center;
 }
```

.js code :-

```
// Ensure the script runs after the full HTML document has been loaded
document.addEventListener('DOMContentLoaded', function () {

    // Get form and input field references
    const form = document.getElementById('registrationForm');
    const nameInput = document.getElementById('name');
    const studentIdInput = document.getElementById('studentId');
    const emailInput = document.getElementById('email');
    const contactInput = document.getElementById('contact');
    const studentTable = document.getElementById('studentTable').getElementsByTagName('tbody')[0];

    // Load student data from local storage or initialize an empty array
    let students = JSON.parse(localStorage.getItem('students')) || [];

    // Function to render all student records in the table
    function renderTable() {
      studentTable.innerHTML = ''; // Clear the table

      students.forEach((student, index) => {
```

```
    const row = document.createElement('tr'); // Create a new table row

    // Fill row with student data
    row.innerHTML = `
      <td>${student.name}</td>
      <td>${student.studentId}</td>
      <td>${student.email}</td>
      <td>${student.contact}</td>
      <td class="actions"></td>
    `;

    const actionsCell = row.querySelector('.actions'); // Select the actions cell

    // Create Edit button
    const editButton = document.createElement('button');
    editButton.textContent = 'Edit';
    editButton.addEventListener('click', () => editStudent(index));
    actionsCell.appendChild(editButton); // Add Edit button

    // Create Delete button
    const deleteButton = document.createElement('button');
    deleteButton.textContent = 'Delete';
    deleteButton.addEventListener('click', () => deleteStudent(index));
    actionsCell.appendChild(deleteButton); // Add Delete button

    studentTable.appendChild(row); // Add row to the table
  });
}

// Validate input fields with regex
function validateInput(name, studentId, email, contact) {
  const nameRegex = /^[A-Za-z ]+$/;
  const idRegex = /^\d+$/;
  const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
  const contactRegex = /^\d+$/;

  return (
    nameRegex.test(name) &&
```

```
    idRegex.test(studentId) &&
    emailRegex.test(email) &&
    contactRegex.test(contact)
  );
}

// Handle form submission
form.addEventListener('submit', function (e) {
  e.preventDefault(); // Prevent form from reloading the page

  const name = nameInput.value.trim();
  const studentId = studentIdInput.value.trim();
  const email = emailInput.value.trim();
  const contact = contactInput.value.trim();

  if (!name || !studentId || !email || !contact) {
    alert('All fields are required.');
    return;
  }

  if (!validateInput(name, studentId, email, contact)) {
    alert('Please enter valid details.');
    return;
  }

  if (form.dataset.editIndex) {
    // Update existing student record
    students[form.dataset.editIndex] = { name, studentId, email, contact };
    delete form.dataset.editIndex;
  } else {
    // Add new student record
    students.push({ name, studentId, email, contact });
  }

  // Save to local storage and refresh table
  localStorage.setItem('students', JSON.stringify(students));
  form.reset(); // Clear form
  renderTable(); // Re-render table
```

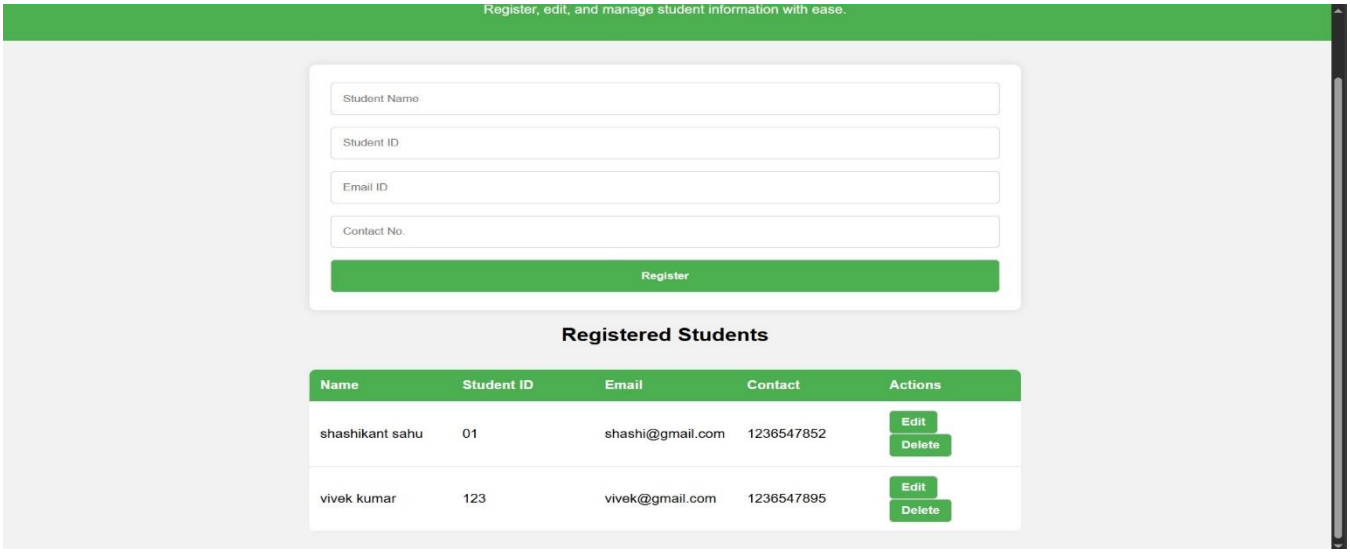
```
  });

  // Delete a student record
  function deleteStudent(index) {
    if (confirm('Are you sure you want to delete this record?')) {
      students.splice(index, 1); // Remove from array
      localStorage.setItem('students', JSON.stringify(students)); // Update storage
      renderTable(); // Refresh table
    }
  }

  // Load student data into the form for editing
  function editStudent(index) {
    const student = students[index];
    nameInput.value = student.name;
    studentIdInput.value = student.studentId;
    emailInput.value = student.email;
    contactInput.value = student.contact;
    form.dataset.editIndex = index; // Store the index for editing
  }

  renderTable(); // Initial render on page load
});
```

3. Screenshot :-



- **GitHub Link:- https://github.com/shashikant021/Student-Registration-System**