**Question 1: Patient Tracker:**

**Index.html:**

**Code:**

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Patient Tracker Application</title>

  <style>

    body {

      font-family: Arial, sans-serif;

      margin: 20px;

    }

    .form-group {

      margin-bottom: 15px;

    }

    label {

      display: block;

      margin-bottom: 5px;

    }

    input, select, button {

      padding: 8px;

      width: 100%;

      box-sizing: border-box;

    }

    table {

      width: 100%;

      border-collapse: collapse;

      margin-top: 20px;

    }

    table, th, td {

      border: 1px solid #ccc;

    }

    th, td {

      padding: 10px;

      text-align: left;

    }

    .delete-btn {

      color: white;

      background-color: red;

      border: none;

      padding: 5px 10px;

      cursor: pointer;

    }

    .delete-btn:hover {

      background-color: darkred;

    }

  </style>

</head>

<body>

  <h1>Patient Tracker Application</h1>

  <form id="patientForm">

    <div class="form-group">

      <label for="name">Name:</label>

      <input type="text" id="name" name="name" required>

    </div>

    <div class="form-group">

      <label for="age">Age:</label>

      <input type="number" id="age" name="age" required>

    </div>

    <div class="form-group">

      <label for="gender">Gender:</label>

      <select id="gender" name="gender" required>

        <option value="">Select</option>

        <option value="Male">Male</option>

        <option value="Female">Female</option>

        <option value="Other">Other</option>

      </select>

    </div>

    <div class="form-group">

      <label for="phone">Phone Number:</label>

      <input type="tel" id="phone" name="phone" required>

    </div>

    <div class="form-group">

      <label for="bloodGroup">Blood Group:</label>

      <select id="bloodGroup" name="bloodGroup" required>

        <option value="">Select</option>

        <option value="A+">A+</option>

        <option value="A-">A-</option>

        <option value="B+">B+</option>

        <option value="B-">B-</option>

        <option value="AB+">AB+</option>

        <option value="AB-">AB-</option>

        <option value="O+">O+</option>

        <option value="O-">O-</option>

      </select>

    </div>

    <div class="form-group">

      <label for="email">Email:</label>

      <input type="email" id="email" name="email" required>

    </div>

    <button type="submit">Add Patient</button>

  </form>

  <table id="patientTable">

    <thead>

      <tr>

        <th>Name</th>

        <th>Age</th>

        <th>Gender</th>

        <th>Phone</th>

        <th>Blood Group</th>

        <th>Email</th>

        <th>Actions</th>

      </tr>

    </thead>

    <tbody>

      <!-- Patient records will be dynamically added here -->

    </tbody>

  </table>

  <script src="index.js"></script>

</body>

</html>

**Index.js:**

**Code:**

document.addEventListener('DOMContentLoaded', function () {

      const patientForm = document.getElementById('patientForm');

      const patientTableBody = document.querySelector('#patientTable tbody');

      loadFromStorage();

      fetchMockData();

      patientForm.addEventListener('submit', function (e) {

        e.preventDefault();

        const name = document.getElementById('name').value.trim();

        const age = document.getElementById('age').value.trim();

        const gender = document.getElementById('gender').value.trim();

        const phone = document.getElementById('phone').value.trim();

        const bloodGroup = document.getElementById('bloodGroup').value.trim();

        const email = document.getElementById('email').value.trim();

        if (!validateInput(name, age, gender, phone, bloodGroup, email)) {

          return;

        }

        const patient = { name, age, gender, phone, bloodGroup, email };

        addPatientToTable(patient);

        saveToStorage(patient);

        showNotification('Patient added successfully!');

        patientForm.reset();

      });

      // Fetch mock data

      function fetchMockData() {

        const xhr = new XMLHttpRequest();

        xhr.open('GET', 'https://jsonplaceholder.typicode.com/users', true);

        xhr.onload = function () {

          if (xhr.status === 200) {

            const patients = JSON.parse(xhr.responseText);

            patients.forEach(patient => {

              addPatientToTable({

                name: patient.name,

                age: 25,

                gender: 'Male',

                phone: '1234567890',

                bloodGroup: 'O+',

                email: patient.email,

              });

            });

          }

        };

        xhr.send();

      }

      function addPatientToTable(patient) {

        const row = document.createElement('tr');

        row.innerHTML = `

          <td>${patient.name}</td>

          <td>${patient.age}</td>

          <td>${patient.gender}</td>

          <td>${patient.phone}</td>

          <td>${patient.bloodGroup}</td>

          <td>${patient.email}</td>

          <td><button class="delete-btn">Delete</button></td>

        `;

        patientTableBody.appendChild(row);

        row.querySelector('.delete-btn').addEventListener('click', function () {

          deletePatient(row, patient);

        });

      }

      function validateInput(name, age, gender, phone, bloodGroup, email) {

        const nameRegex = /^[A-Za-z\s]+$/;

        const phoneRegex = /^\d{10}$/;

        const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;

        if (!nameRegex.test(name)) {

          alert('Invalid name format! Only letters and spaces are allowed.');

          return false;

        }

        if (age <= 0 || age > 120 || isNaN(age)) {

          alert('Invalid age! Age must be a number between 1 and 120.');

          return false;

        }

        if (!gender) {

          alert('Please select a gender.');

          return false;

        }

        if (!phoneRegex.test(phone)) {

          alert('Invalid phone number! Must be exactly 10 digits.');

          return false;

        }

        if (!bloodGroup) {

          alert('Please select a blood group.');

          return false;

        }

        if (!emailRegex.test(email)) {

          alert('Invalid email format!');

          return false;

        }

        return true;

      }

      function deletePatient(row, patient) {

        row.remove();

        const patients = JSON.parse(localStorage.getItem('patients')) || [];

        const updatedPatients = patients.filter(

          p =>

            p.name !== patient.name ||

            p.age !== patient.age ||

            p.email !== patient.email

        );

        localStorage.setItem('patients', JSON.stringify(updatedPatients));

        showNotification('Patient deleted successfully!');

      }

      function showNotification(message) {

        if (Notification.permission === 'granted') {

          new Notification(message);

        } else if (Notification.permission !== 'denied') {

          Notification.requestPermission().then(permission => {

            if (permission === 'granted') {

              new Notification(message);

            }

          });

        }

      }

      function saveToStorage(patient) {

        const patients = JSON.parse(localStorage.getItem('patients')) || [];

        patients.push(patient);

        localStorage.setItem('patients', JSON.stringify(patients));

      }

      function loadFromStorage() {

        const patients = JSON.parse(localStorage.getItem('patients')) || [];

        patients.forEach(patient => {

          addPatientToTable(patient);

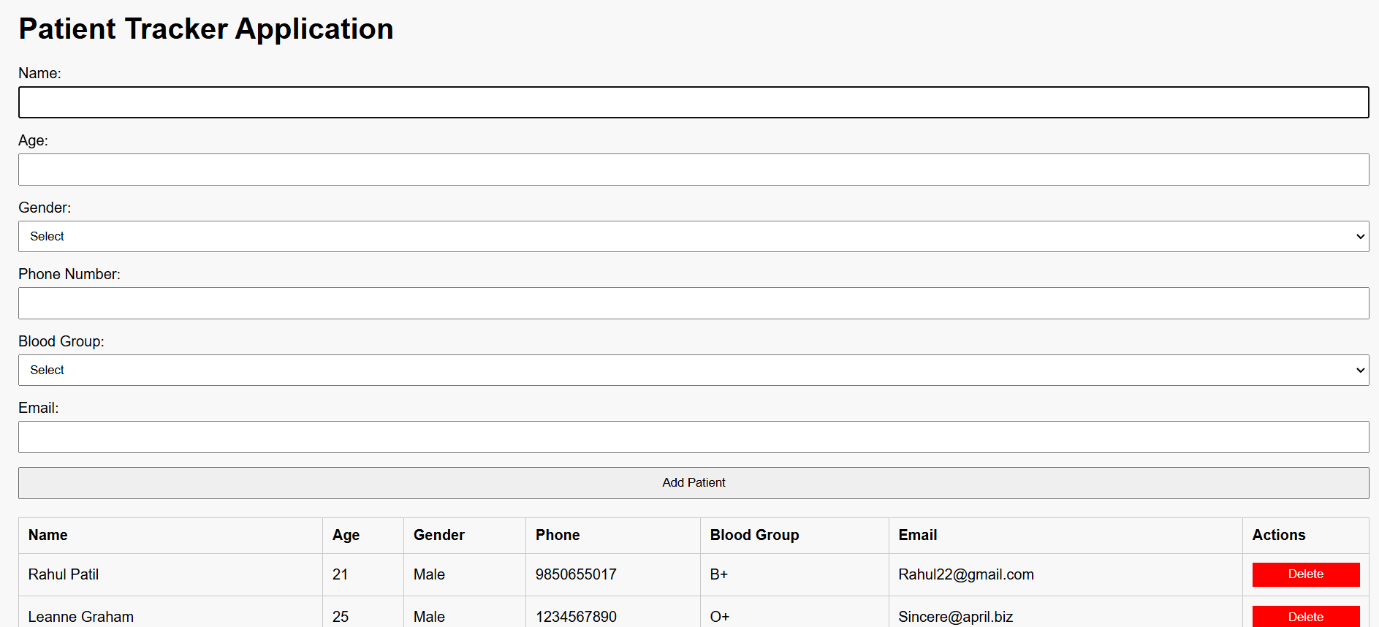
        });

      }

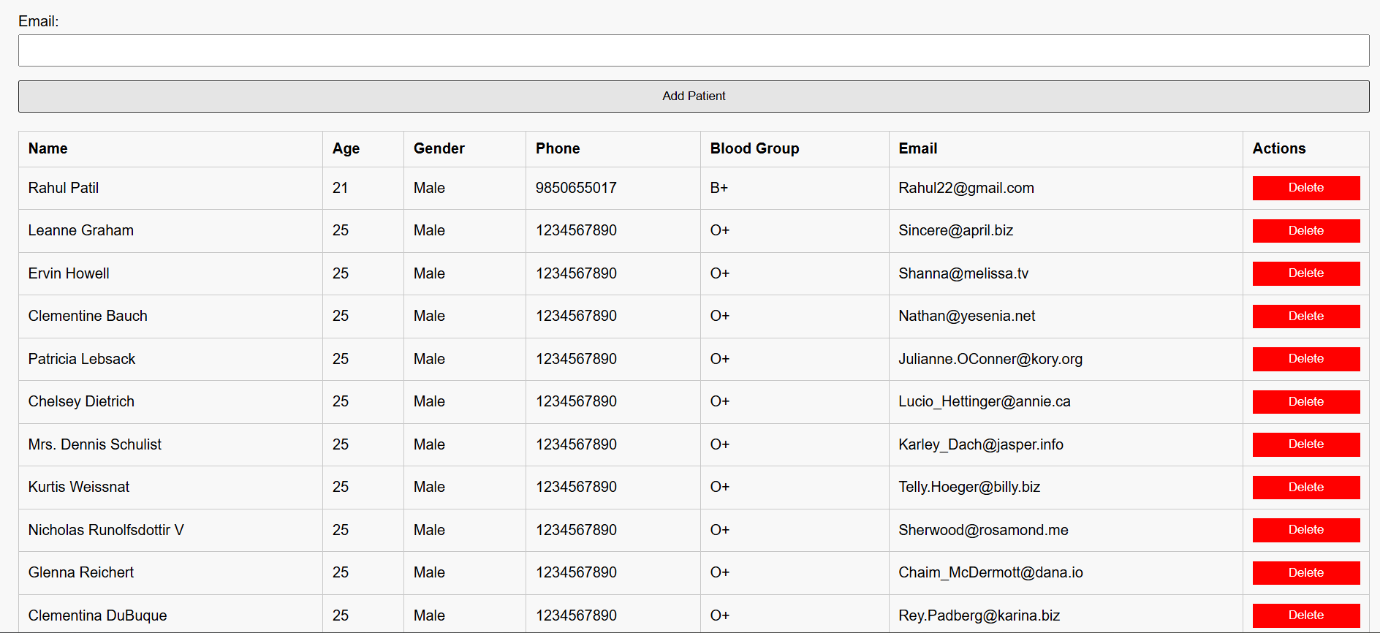
    });

**Outputs:**

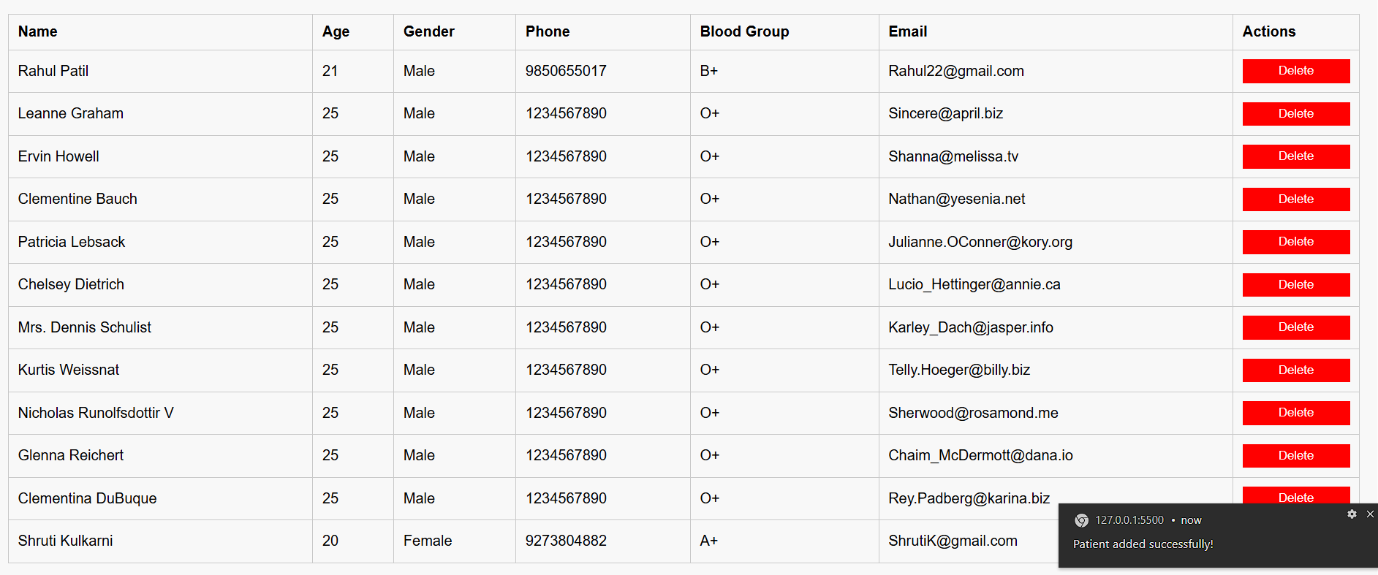
**HTML Structure:**



**Sample Data using AJAX:**



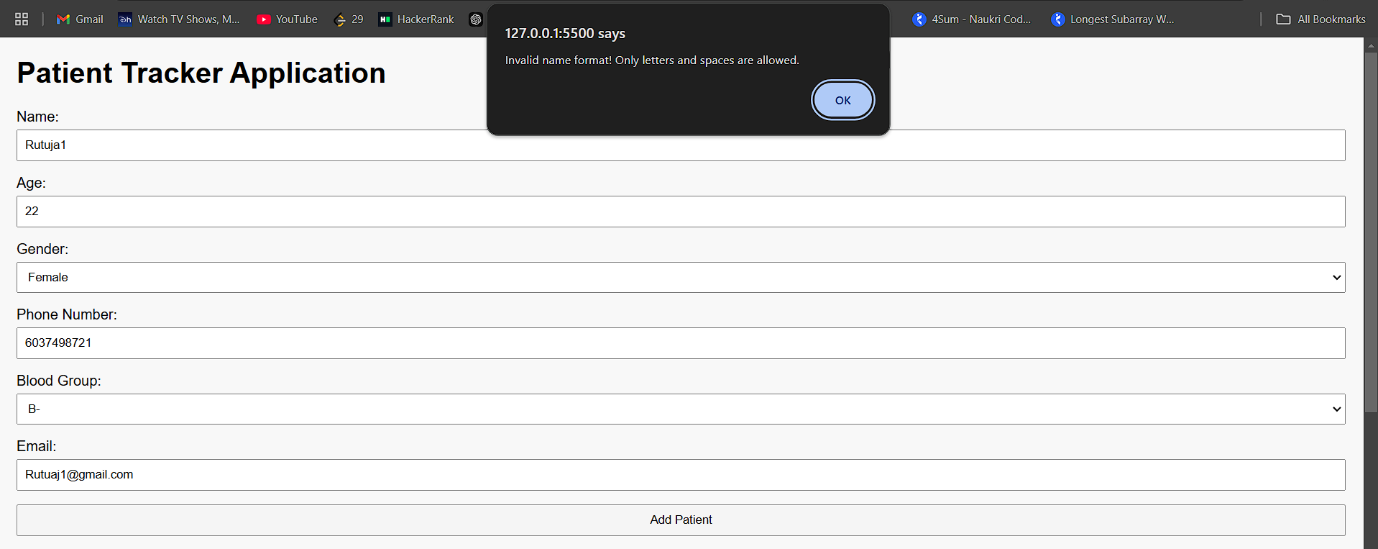
**DOM Manipulation and Alert using Notification Api (New Patient Added):**

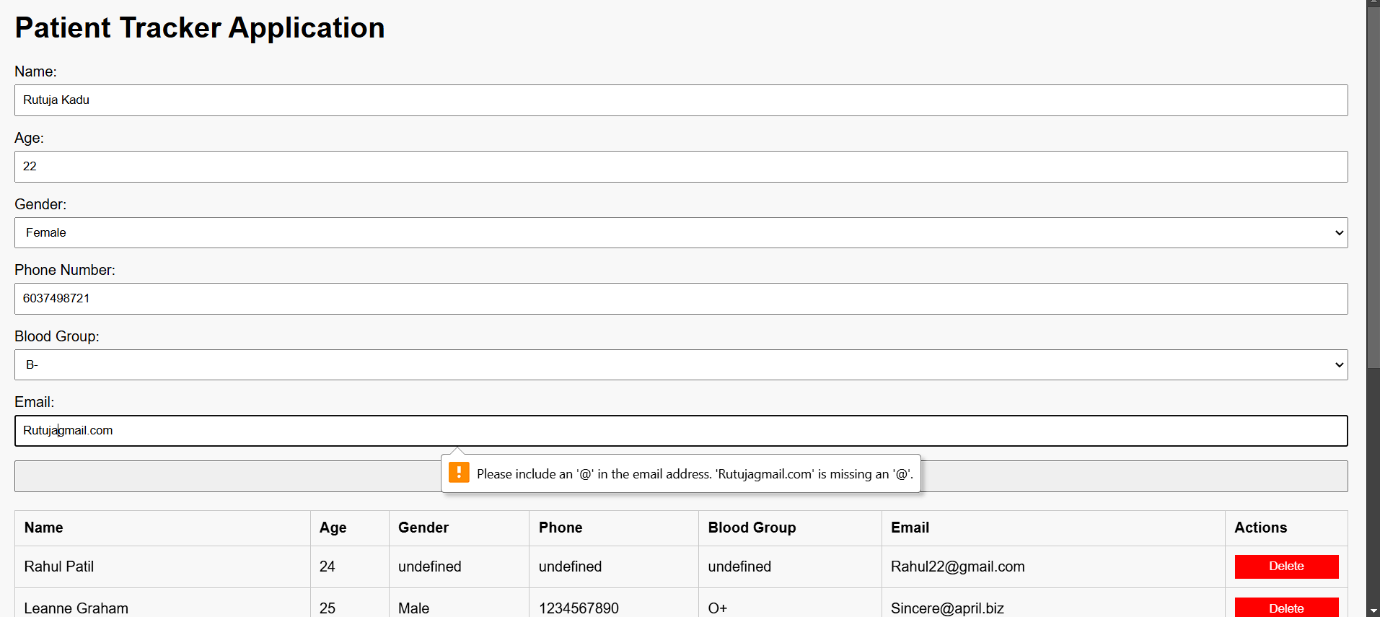


**Storage Api usage:**



**Validation using Regex:**





**JSON data Handling:**

