ASSIGNMENT 1
Creating a Web Page Using CSS, HTML, and JavaScript

Name: G.Kalyani Krishna
Roll No: 22011102017
Date: 06.08.2024

Aim

To create a simple web page using inline CSS for styling and inline JavaScript for interactivity.

Introduction

Web development involves the use of various languages to create interactive and visually appealing websites. HTML (Hypertext Markup Language) provides the structure, CSS (Cascading Style Sheets) handles the presentation and layout, while JavaScript adds interactivity and dynamic behavior to the page. In this assignment, we will create a basic web page using inline CSS and JavaScript for demonstration purposes.

Algorithm

1. Create an HTML file to define the structure of the web page.
2. Use inline CSS to style the elements of the web page within the HTML file.
3. Add inline JavaScript to implement interactivity (e.g., button click events).
4. Open the web page in a browser and validate the functionality.

Code

HTML Code with Inline CSS and JavaScript

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale
       =1.0">
```

```html
    <title>Simple Web Page</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f0f0f0;
            text-align: center;
            margin: 20px;
        }

        h1 {
            color: #333;
        }

        p {
            font-size: 18px;
        }

        button {
            padding: 10px 20px;
            background-color: #008CBA;
            color: white;
            border: none;
            cursor: pointer;
        }

        button:hover {
            background-color: #005f73;
        }
    </style>
</head>
<body>
    <h1>Welcome to My Web Page</h1>
    <p>This is a simple web page created using HTML, inline CSS, and inline JavaScript.</p>
    <button onclick="changeText()">Click Me!</button>
    <p id="message"></p>

    <script>
        function changeText() {
            document.getElementById('message').textContent = "You clicked the button!";
        }
    </script>
</body>
</html>
```

Output

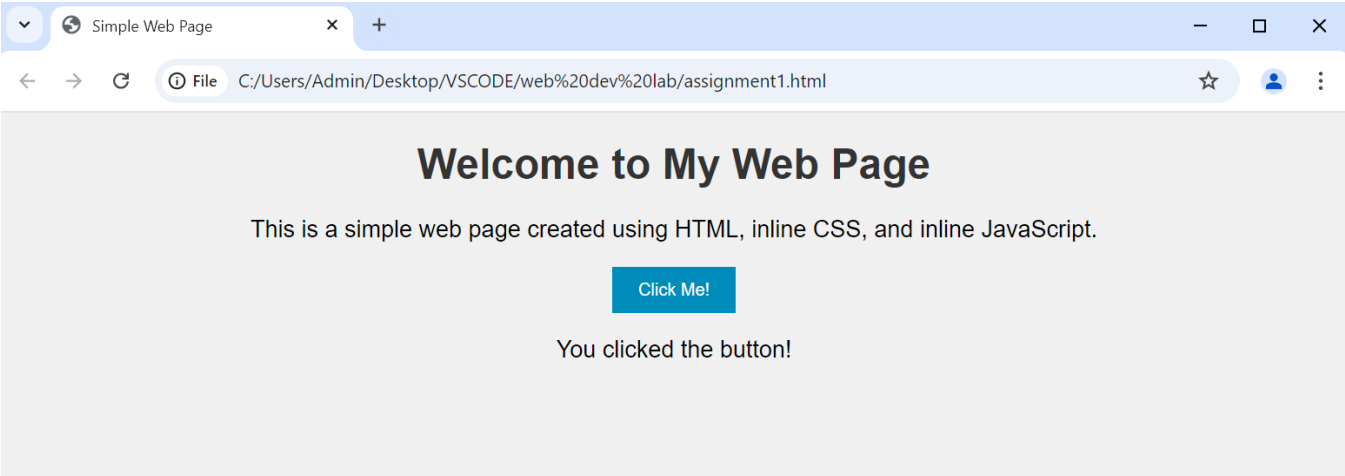

Figure 1: Initial Web Page with a Button

The web page starts with a header, paragraph, and a button. Upon clicking the button, the message changes dynamically using inline JavaScript.

Result

The web page was successfully created using HTML, inline CSS, and inline JavaScript. It demonstrates the integration of structure, styling, and interactivity in web development.

ASSIGNMENT 2
React JS Program to Switch Between Layouts

Name: G.Kalyani Krishna
Roll No: 22011102017
Date: 13.08.2024

Aim

To create a React.js program that switches between two different layouts when a button is clicked.

Installation

To set up React.js and create the application, use the following commands:

```
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.3/
   install.sh | bash
source ~/.bashrc
nvm install 18
nvm use 18
node -v
```

Then create the React application with the following commands:

```
npx create-react-app switch-layouts
cd switch-layouts
```

Code

Layout1.js

```
import React from 'react';

const Layout1 = ({ children }) => (
  <div style={{ padding: '20px', background: '#f0f0f0' }}>
    <header>
      <h1>Layout 1</h1>
    </header>
    <main>{children}</main>
    <footer>Footer for Layout 1</footer>
```

```
  </div>
);

export default Layout1;
```

Layout2.js

```
import React from 'react';

const Layout2 = ({ children }) => (
  <div style={{ padding: '20px', background: '#e0e0e0' }}>
    <header>
      <h1>Layout 2</h1>
    </header>
    <main>{children}</main>
    <footer>Footer for Layout 2</footer>
  </div>
);

export default Layout2;
```

App.js

```
import React, { useState } from 'react';
import Layout1 from './Layout1';
import Layout2 from './Layout2';

const App = () => {
  const [layout, setLayout] = useState('layout1');

  const toggleLayout = () => {
    setLayout(prevLayout => (prevLayout === 'layout1' ? 'layout2' :
       'layout1'));
  };

  return (
    <div>
      <button onClick={toggleLayout}>
        Switch to {layout === 'layout1' ? 'Layout 2' : 'Layout 1'}
      </button>
      {layout === 'layout1' ? (
        <Layout1>
          <p>This is the content in Layout 1.</p>
        </Layout1>
      ) : (
        <Layout2>
```

```
            <p>This is the content in Layout 2.</p>
          </Layout2>
        )}
      </div>
    );
};

export default App;
```

Execution

Run the application with the following command:

```
npm start
```

Output

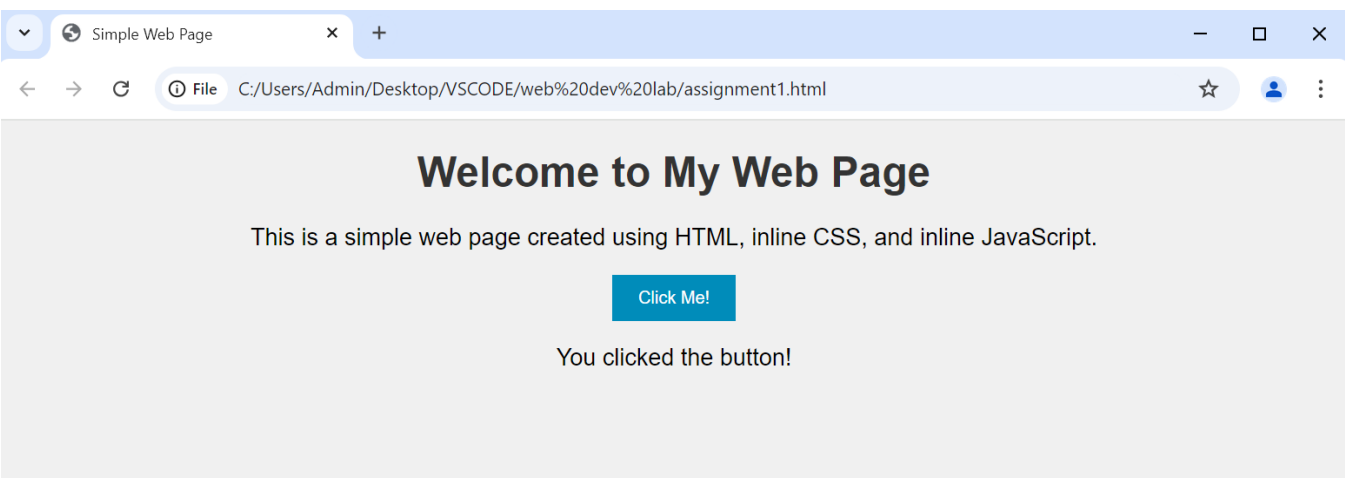

Figure 1: Layout 1

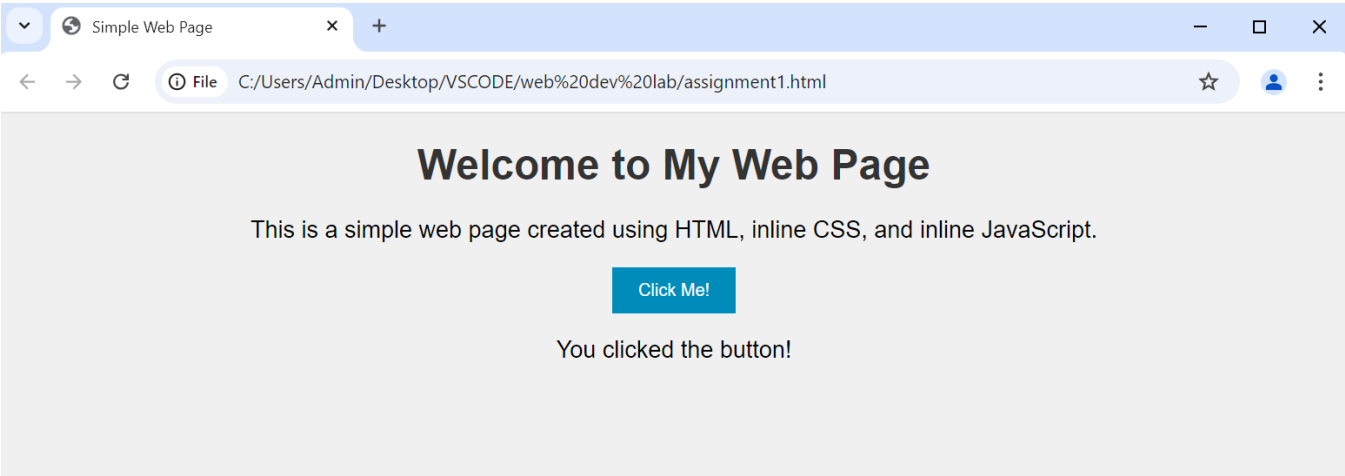

Figure 2: Layout 2

ASSIGNMENT 3
React JS Program to Implement Search Filter

Name: G.Kalyani Krishna
Roll No: 22011102017
Date: 13.08.2024

Aim

To create a React.js program that implements a search filter to filter through a list of items.

Introduction

React.js is a powerful JavaScript library used for building user interfaces. One common feature in web applications is the search filter, which allows users to easily find items in a list. This assignment demonstrates how to create a search filter in a React.js application using state management and event handling.

Installation

To set up React.js and create the application, use the following commands:

```
npx create-react-app search-filter
cd search-filter
```

Code

SearchFilter.js

```
import React, { useState } from 'react';

const SearchFilter = () => {
  // Initial list of items
  const items = [
    'Apple',
    'Banana',
    'Orange',
    'Grapes',
```

```
    'Mango',
    'Blueberry',
    'Strawberry',
  ];

  // State for the search query
  const [query, setQuery] = useState('');

  // Handle the change in the search input
  const handleChange = (event) => {
    setQuery(event.target.value);
  };

  // Filter items based on the search query
  const filteredItems = items.filter(item =>
    item.toLowerCase().includes(query.toLowerCase())
  );

  return (
    <div>
      <h1>Search Filter</h1>
      <input
        type="text"
        placeholder="Search items..."
        value={query}
        onChange={handleChange}
      />
      <ul>
        {filteredItems.length > 0 ? (
          filteredItems.map((item, index) => <li key={index}>{item}<
             /li>)
        ) : (
          <li>No items found</li>
        )}
      </ul>
    </div>
  );
};

export default SearchFilter;
```

App.js

```
import React from 'react';
import SearchFilter from './SearchFilter';
import './App.css';

```

```
const App = () => {
  return (
    <div className="App">
      <SearchFilter />
    </div>
  );
};

export default App;
```

App.css

```
.App {
  text-align: center;
}

input {
  padding: 10px;
  font-size: 16px;
  width: 300px;
}

ul {
  list-style-type: none;
  padding: 0;
}

li {
  padding: 5px;
  font-size: 18px;
}
```

Execution

Run the application with the following command:

```
npm start
```

Output

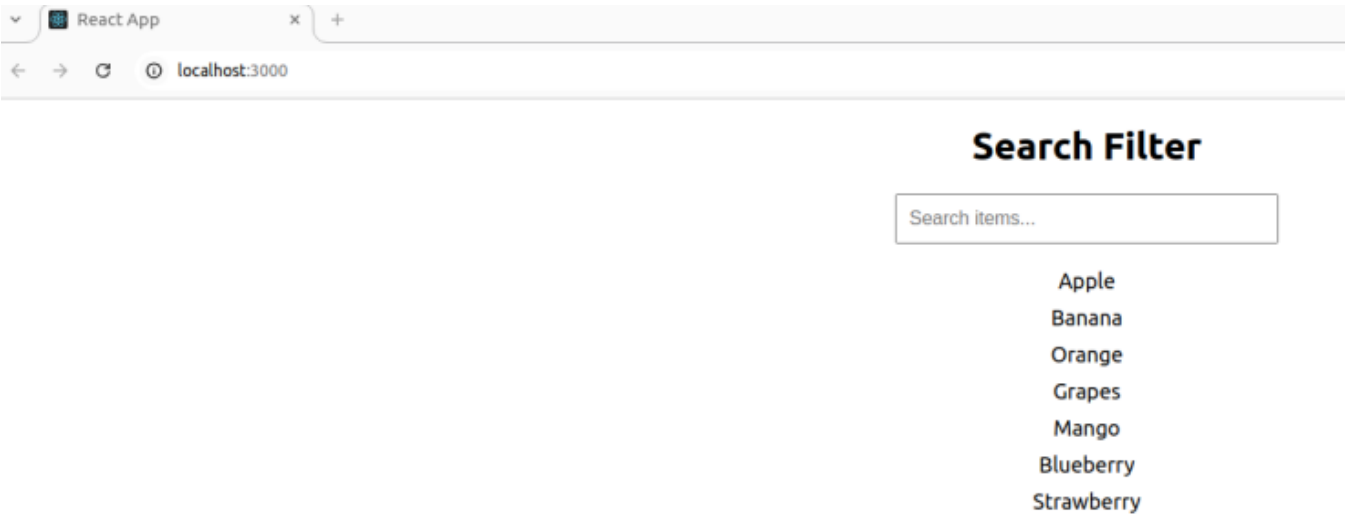

Figure 1: Search Filter page - Initial View

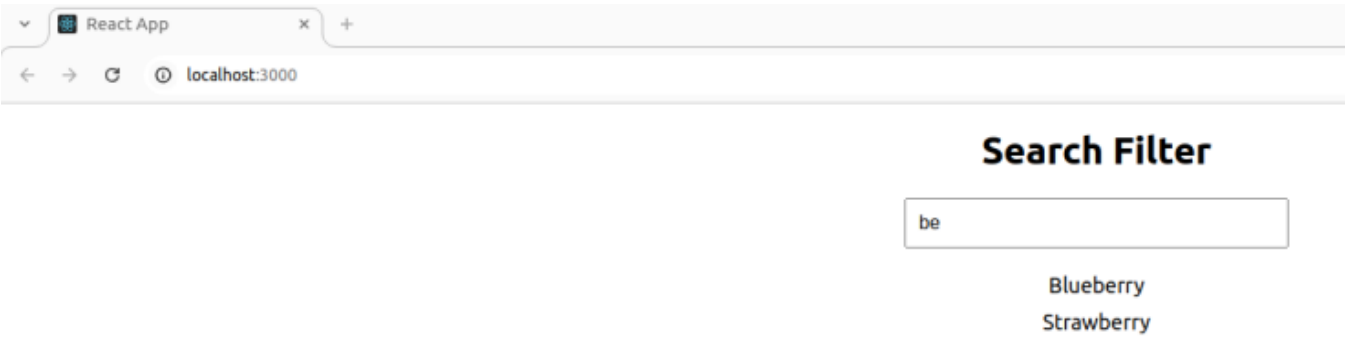

Figure 2: Search Filter page - After Searching

ASSIGNMENT 4
React JS Program to Create a Form

Name: G.Kalyani Krishna
Roll No: 22011102017
Date: 20.08.2024

Aim

To create a React.js program that implements a real-time form for user input.

Introduction

React.js is a popular JavaScript library used for building user interfaces. This assignment focuses on creating a real-time form that captures user inputs and displays the data dynamically. This provides immediate feedback to users, enhancing their interaction with the application.

Installation

To set up React.js and create the application, use the following commands:

```
npx create-react-app realtime-form
cd realtime-form
```

Code

RealTimeForm.js

```
import React, { useState } from 'react';

const RealTimeForm = () => {
  // State to manage form fields
  const [formData, setFormData] = useState({
    name: '',
    email: '',
    age: '',
    message: ''
  });
```

```

// Handle input change
const handleChange = (event) => {
  const { name, value } = event.target;
  setFormData({
    ...formData,
    [name]: value
  });
};

return (
  <div>
    <h1>Real-Time Form</h1>
    <form>
      <div>
        <label>
          Name:
          <input
            type="text"
            name="name"
            value={formData.name}
            onChange={handleChange}
          />
        </label>
      </div>
      <div>
        <label>
          Email:
          <input
            type="email"
            name="email"
            value={formData.email}
            onChange={handleChange}
          />
        </label>
      </div>
      <div>
        <label>
          Age:
          <input
            type="number"
            name="age"
            value={formData.age}
            onChange={handleChange}
          />
        </label>
      </div>
```

```
        <div>
          <label>
            Message:
            <textarea
              name="message"
              value={formData.message}
              onChange={handleChange}
            />
          </label>
        </div>
      </form>
      <h2>Form Data</h2>
      <pre>{JSON.stringify(formData, null, 2)}</pre>
    </div>
  );
};

export default RealTimeForm;
```

App.js

```
import React from 'react';
import RealTimeForm from './RealTimeForm';
import './App.css'; // Optional: Add some styles if needed

const App = () => {
  return (
    <div className="App">
      <RealTimeForm />
    </div>
  );
};

export default App;
```

App.css

```
.App {
  text-align: center;
}

form {
  display: inline-block;
  text-align: left;
}

```

```
div {
  margin-bottom: 10px;
}

label {
  display: block;
  margin-bottom: 5px;
}

input, textarea {
  width: 100%;
  padding: 8px;
  font-size: 16px;
}
```

Execution

Run the application with the following command:

```
npm start
```

Output

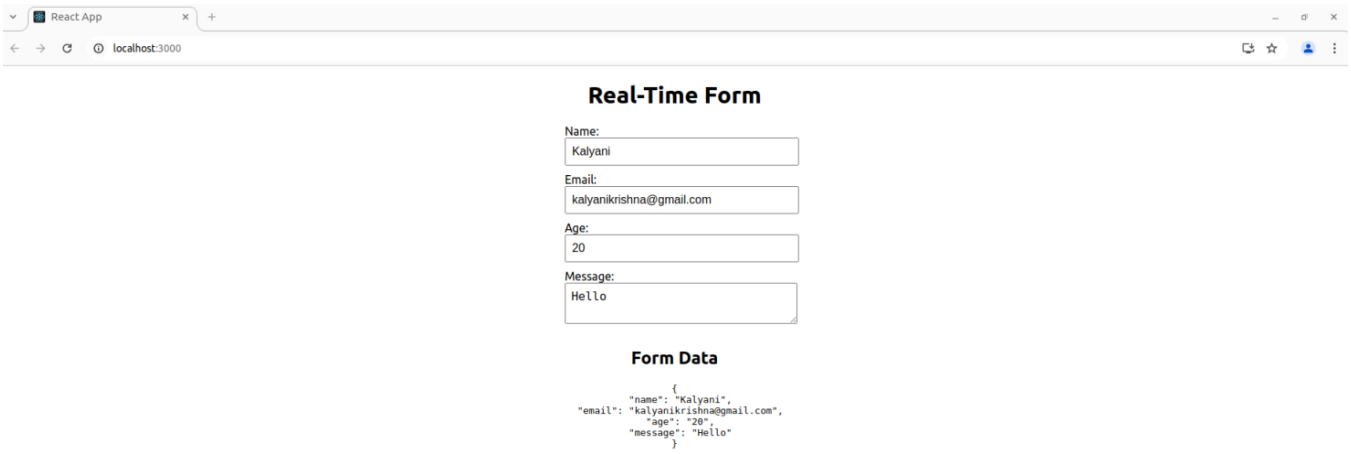

Figure 1: Real-Time Form Page

ASSIGNMENT 5
Implementation of a Navigation Menu

Name: G.Kalyani Krishna
Roll No: 22011102017
Date: 20.08.2024

Aim

To implement a navigation menu in a React.js application using React Router for page navigation.

Introduction

Navigation menus are essential for web applications, allowing users to access different pages or sections of the application easily. This assignment focuses on creating a simple navigation menu using React Router, enabling seamless navigation between different components such as Home, About, and Contact pages.

Installation

To set up the application and install the necessary dependencies, use the following commands:

```
npx create-react-app navigation-menu
cd navigation-menu
npm install react-router-dom
```

Code

Home.js

```
import React from 'react';

const Home = () => {
  return (
    <div>
      <h2>Home Page</h2>
      <p>Hello! My name is Kalyani. I am a third-year IoT student.
         Welcome to my web technology lab assignment!</p>
```

```
    </div>
  );
};

export default Home;
```

About.js

```
import React from 'react';

const About = () => {
  return (
    <div>
      <h2>About Page</h2>
      <p>My name is Kalyani, and I am a third-year student specializing in the Internet of Things (IoT). This website is part of my web technology lab assignment where I am learning to create React applications and implement various features such as routing and real-time updates.</p>
    </div>
  );
};

export default About;
```

Contact.js

```
import React from 'react';

const Contact = () => {
  return (
    <div>
      <h2>Contact Page</h2>
      <p>If you have any questions or just want to get in touch, you can contact me at:</p>
      <p>Email: <a href="mailto:kalyani@gmail.com">kalyani@gmail.com</a></p>
    </div>
  );
};

export default Contact;
```

App.js

```
import React from 'react';
import { BrowserRouter as Router, Route, Routes } from 'react-router
   -dom';
import Home from './Home';
import About from './About';
import Contact from './Contact';
import Navigation from './Navigation';
import './App.css'; // Optional: Add some styles if needed

const App = () => {
  return (
    <Router>
      <div className="App">
        <Navigation />
        <Routes>
          <Route path="/" element={<Home />} />
          <Route path="/about" element={<About />} />
          <Route path="/contact" element={<Contact />} />
        </Routes>
      </div>
    </Router>
  );
};

export default App;
```

App.css

```
.App {
  text-align: center;
  font-family: Arial, sans-serif;
}

nav {
  background: #282c34;
  padding: 1em;
}

ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
}

li {
  display: inline;
  margin-right: 15px;
```

```
}

a {
  color: #61dafb;
  text-decoration: none;
  font-weight: bold;
}

a:hover {
  text-decoration: underline;
}

h2 {
  color: #282c34;
}

p {
  color: #333;
  font-size: 18px;
}
```

Execution

Run the application with the following command:

```
npm start
```

Output

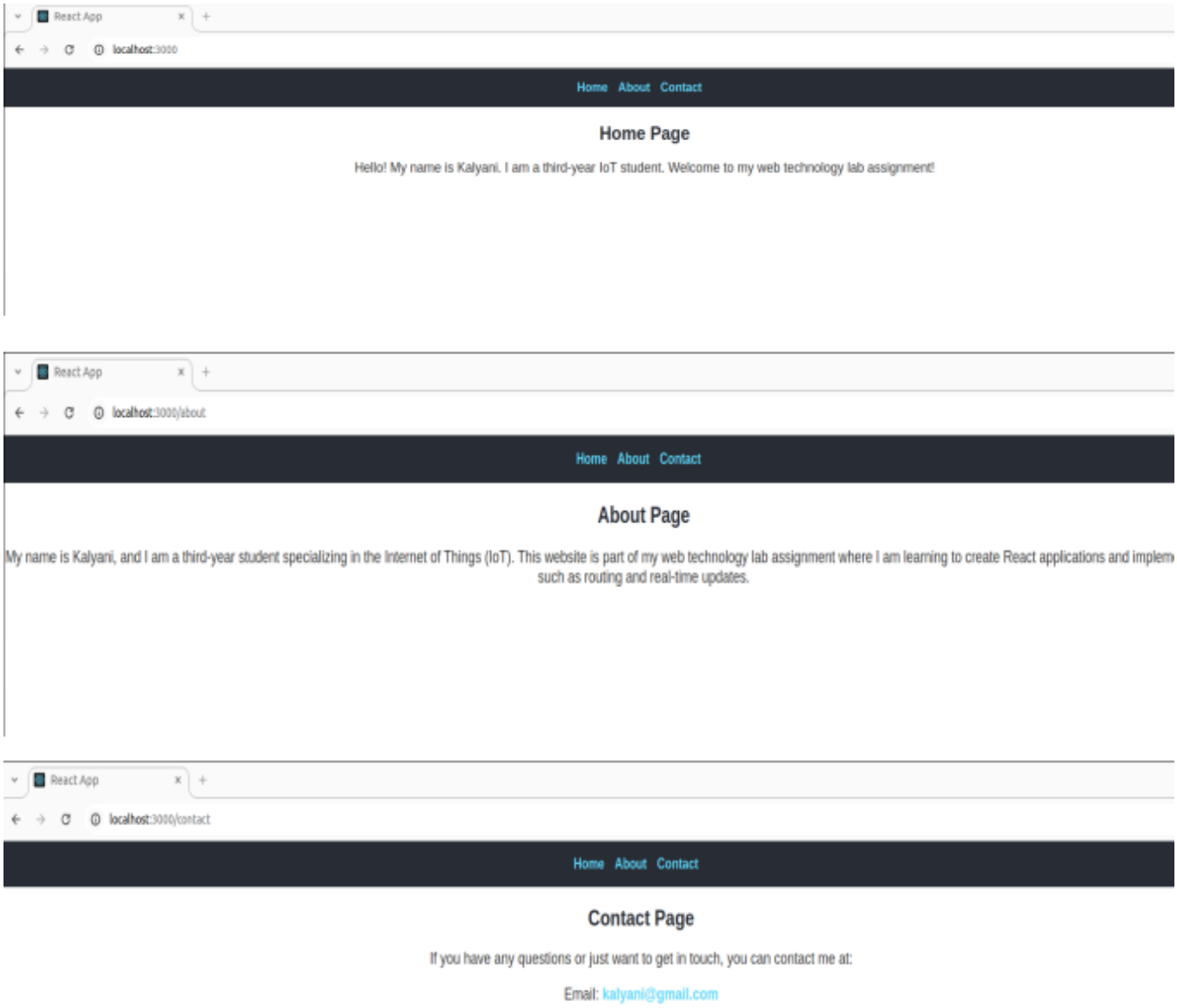

Figure 1: Navigation Menu in the Application

ASSIGNMENT 6
Creation of a Video Player

Name: G.Kalyani Krishna
Roll No: 22011102017
Date: 27.08.2024

Aim

To create a video player using React that includes custom controls for play/pause, volume adjustment, and progress tracking.

Introduction

Video players are an integral part of modern web applications, allowing users to watch and interact with video content seamlessly. This assignment focuses on creating a simple yet functional video player using React, enabling users to play, pause, adjust volume, and seek through the video.

Installation

To set up the application and install the necessary dependencies, use the following commands:

```
npx create-react-app video-player
cd video-player
```

Code

VideoPlayer.js

```
import React, { useRef, useState } from 'react';

const VideoPlayer = () => {
  const videoRef = useRef(null); // Reference to the video element
  const [isPlaying, setIsPlaying] = useState(false); // State to
     manage playback

  const handlePlayPause = () => {
    if (videoRef.current.paused) {
```

```
      videoRef.current.play();
      setIsPlaying(true);
    } else {
      videoRef.current.pause();
      setIsPlaying(false);
    }
  };

  const handleVolumeChange = (event) => {
    videoRef.current.volume = event.target.value;
  };

  const handleProgressChange = (event) => {
    videoRef.current.currentTime = event.target.value;
  };

  return (
    <div className="video-player">
      <video
        ref={videoRef}
        width="640"
        controls={false} // Custom controls
        src="https://www.w3schools.com/html/mov_bbb.mp4" // Replace
           with your video URL
      >
        Your browser does not support the video tag.
      </video>
      <div className="controls">
        <button onClick={handlePlayPause}>
          {isPlaying ? 'Pause' : 'Play'}
        </button>
        <input
          type="range"
          min="0"
          max={videoRef.current ? videoRef.current.duration : 100}
          step="0.1"
          onChange={handleProgressChange}
        />
        <input
          type="range"
          min="0"
          max="1"
          step="0.1"
          onChange={handleVolumeChange}
        />
      </div>
    </div>
```

```
  );
};

export default VideoPlayer;
```

App.css

```
.video-player {
  text-align: center;
}

video {
  display: block;
  margin: 0 auto;
}

.controls {
  margin-top: 10px;
}

button {
  padding: 10px;
  font-size: 16px;
}

input[type="range"] {
  width: 300px;
}
```

App.js

```
import React from 'react';
import VideoPlayer from './VideoPlayer';
import './App.css';

const App = () => {
  return (
    <div className="App">
      <h1>Video Player</h1>
      <VideoPlayer />
    </div>
  );
};

export default App;
```

Execution

Run the application with the following command:

```
npm start
```

Output

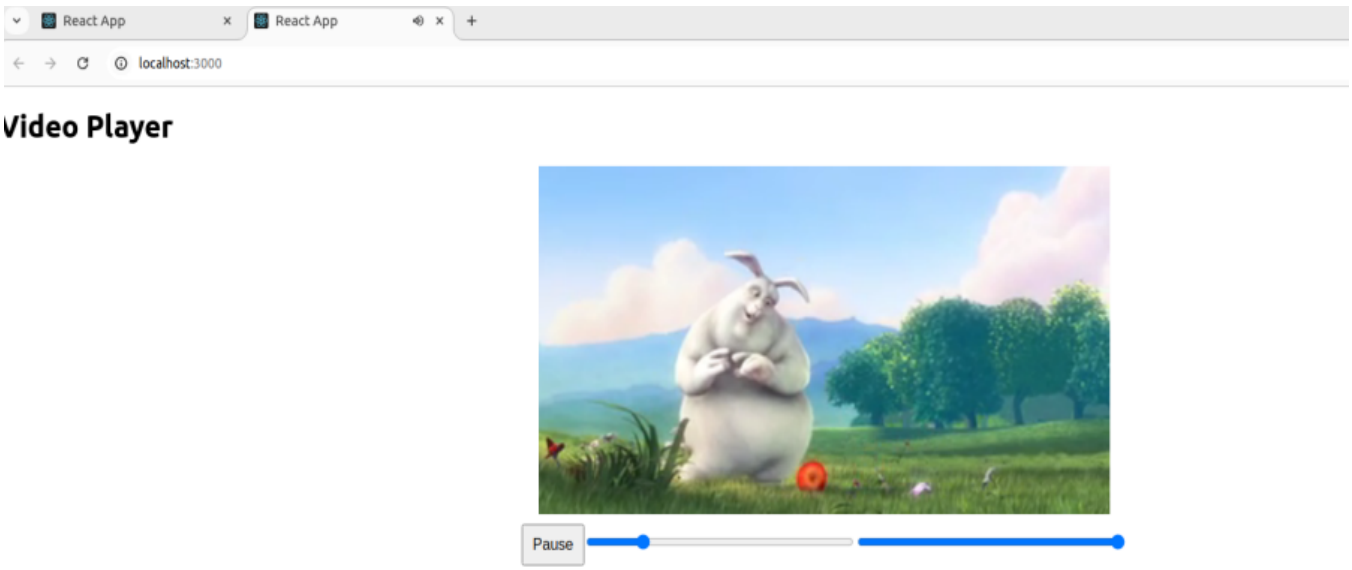

Figure 1: Video Player Interface

ASSIGNMENT 7
Creation of a Text Editor

Name: G.Kalyani Krishna
Roll No: 22011102017
Date: 27.08.2024

Aim

To create a text editor using React that allows users to format text with various styles, such as headers, bold, italic, and underline.

Introduction

Text editors are essential tools for content creation and formatting in modern web applications. This assignment focuses on building a simple text editor using React and the Draft.js library, enabling users to apply various text styles easily.

Installation

To set up the application and install the necessary dependencies, use the following commands:

```
npx create-react-app text-editor
cd text-editor
npm install draft-js
```

Code

Editor.js

```
import React, { useState } from 'react';
import { Editor, EditorState, RichUtils } from 'draft-js';
import 'draft-js/dist/Draft.css'; // Import default Draft.js styles
import './Editor.css'; // Import custom styles

const TextEditor = () => {
  const [editorState, setEditorState] = useState(EditorState.
     createEmpty());
```

```

  const handleKeyCommand = (command) => {
    const newState = RichUtils.handleKeyCommand(editorState, command);
    if (newState) {
      setEditorState(newState);
      return 'handled';
    }
    return 'not-handled';
  };

  const toggleBlockType = (blockType) => {
    setEditorState(RichUtils.toggleBlockType(editorState, blockType));
  };

  const toggleInlineStyle = (inlineStyle) => {
    setEditorState(RichUtils.toggleInlineStyle(editorState, inlineStyle));
  };

  return (
    <div className="editor-container">
      <div className="editor-controls">
        <button onClick={() => toggleBlockType('header-one')}>H1</button>
        <button onClick={() => toggleBlockType('header-two')}>H2</button>
        <button onClick={() => toggleBlockType('blockquote')}>Blockquote</button>
        <button onClick={() => toggleInlineStyle('BOLD')}>Bold</button>
        <button onClick={() => toggleInlineStyle('ITALIC')}>Italic</button>
        <button onClick={() => toggleInlineStyle('UNDERLINE')}>Underline</button>
      </div>
      <Editor
        editorState={editorState}
        handleKeyCommand={handleKeyCommand}
        onChange={setEditorState}
        placeholder="Start typing..."
      />
    </div>
  );
};

```

```
export default TextEditor;
```

Editor.css

```
.editor-container {
  border: 1px solid #ddd;
  padding: 10px;
  max-width: 800px;
  margin: 20px auto;
}

.editor-controls {
  margin-bottom: 10px;
}

button {
  margin-right: 5px;
  padding: 5px;
  font-size: 14px;
}

.DraftEditor-root {
  border: 1px solid #ddd;
  min-height: 200px;
  padding: 10px;
}
```

App.js

```
import React from 'react';
import TextEditor from './Editor';
import './App.css';

const App = () => {
  return (
    <div className="App">
      <h1>React Text Editor</h1>
      <TextEditor />
    </div>
  );
};

export default App;
```

App.css

```
.App {
  text-align: center;
}
```

Execution

Run the application with the following command:

```
npm start
```

Output

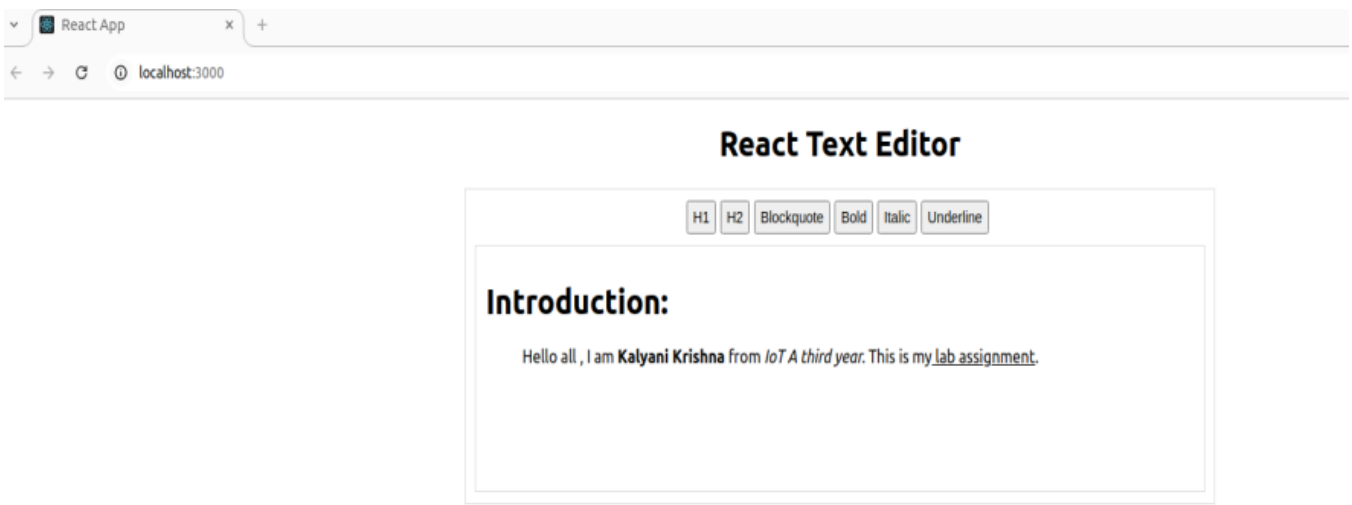

Figure 1: Text Editor Interface

ASSIGNMENT 8
Creation of a Basic HTML Webpage

Name: G.Kalyani Krishna
Roll No: 22011102017
Date: 03.09.2024

Aim

To create a basic HTML webpage that showcases the use of HTML and CSS to enhance presentation and user experience.

Introduction

Creating a well-designed webpage involves the integration of HTML for structure and CSS for styling. This assignment focuses on building a visually appealing webpage that incorporates modern design elements, ensuring an engaging user interface.

Code

index.html

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>My Basic Webpage</title>
    <link rel="stylesheet" href="styles.css">
    <link href="https://fonts.googleapis.com/css2?family=Roboto:wght@400;700&display=swap" rel="stylesheet">
</head>
<body>
    <header>
        <h1>Welcome to My Basic Webpage</h1>
        <nav>
            <ul>
```

```html
                <li><a href="#about">About</a></li>
                <li><a href="#services">Services</a></li>
                <li><a href="#portfolio">Portfolio</a></li>
                <li><a href="#contact">Contact</a></li>
            </ul>
        </nav>
    </header>
    <section id="about">
        <h2>About Us</h2>
        <p>This webpage is designed to showcase my skills in HTML and CSS.</p>
    </section>
    <section id="services">
        <h2>Our Services</h2>
        <ul>
            <li>Web Development</li>
            <li>Graphic Design</li>
            <li>SEO Optimization</li>
        </ul>
    </section>
    <section id="portfolio">
        <h2>Our Portfolio</h2>
        <div class="gallery">
            <img src="https://via.placeholder.com/150" alt="Project 1">
            <img src="https://via.placeholder.com/150" alt="Project 2">
            <img src="https://via.placeholder.com/150" alt="Project 3">
        </div>
    </section>
    <section id="contact">
        <h2>Contact Us</h2>
        <form>
            <label for="name">Name:</label>
            <input type="text" id="name" name="name" required>
            <label for="email">Email:</label>
            <input type="email" id="email" name="email" required>
            <label for="message">Message:</label>
            <textarea id="message" name="message" required></textarea>
            <button type="submit">Send Message</button>
        </form>
    </section>
    <footer>
        <p>&copy; 2024 My Basic Webpage. All rights reserved.</p>
    </footer>
```

```html
</body>
</html>
```

styles.css

```css
body {
    font-family: 'Roboto', sans-serif;
    margin: 0;
    padding: 0;
    background-color: #f4f4f4;
    color: #333;
}
header {
    background: #007BFF;
    color: #fff;
    padding: 20px 0;
    text-align: center;
}
nav ul {
    list-style: none;
    padding: 0;
}
nav ul li {
    display: inline;
    margin: 0 15px;
}
nav a {
    color: #fff;
    text-decoration: none;
    font-weight: bold;
}
section {
    padding: 20px;
    margin: 10px;
    background: #fff;
    border-radius: 5px;
    box-shadow: 0 2px 5px rgba(0,0,0,0.1);
}
.gallery {
    display: flex;
    justify-content: space-around;
}
footer {
    text-align: center;
    padding: 10px 0;
    background: #007BFF;
    color: #fff;
```

```
}
```

Execution

To view the webpage, open the `index.html` file in a web browser. Ensure that the `styles.css` file is in the same directory as the HTML file for the styles to apply correctly.

Output

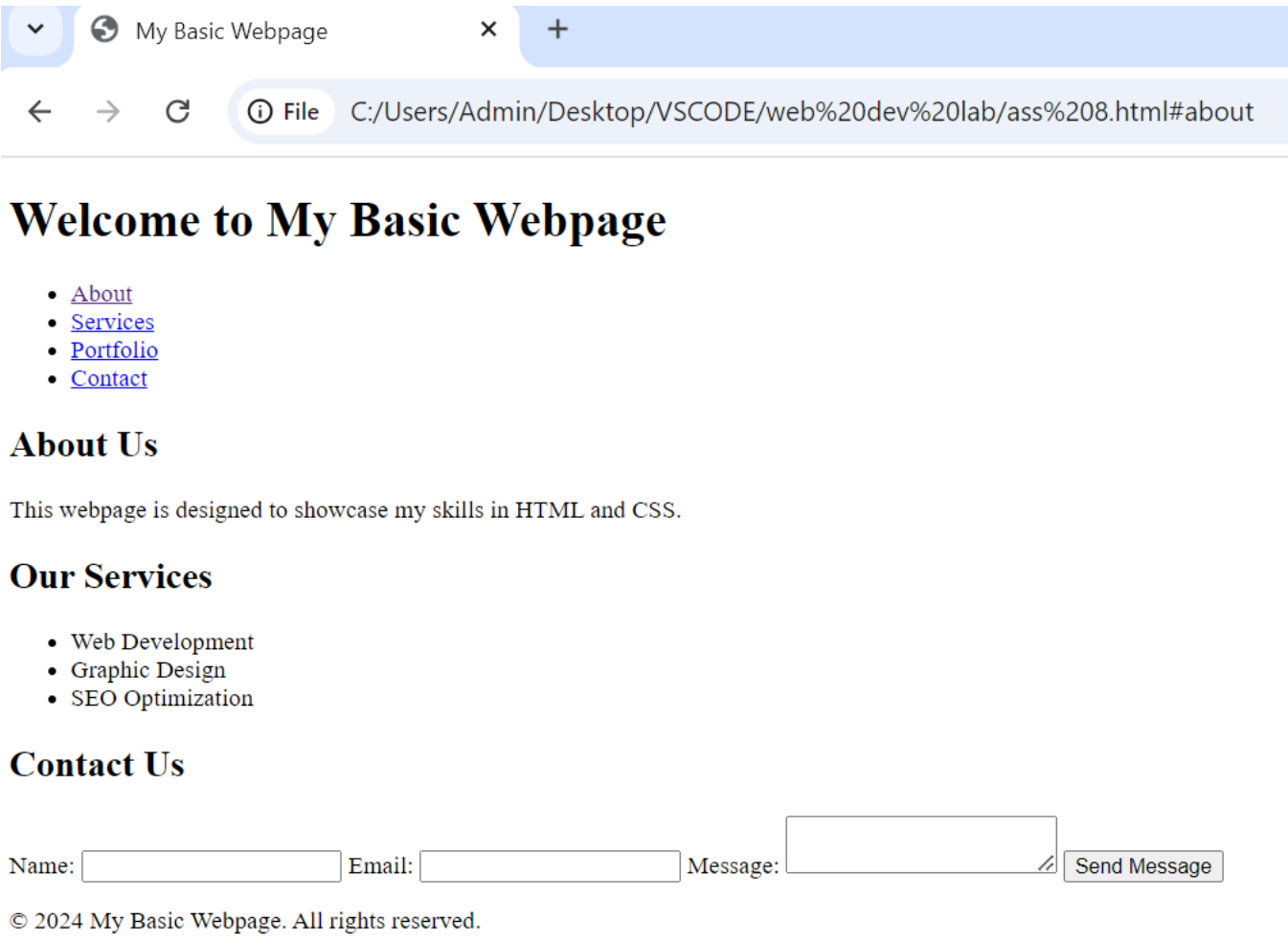

Figure 1: Basic HTML Webpage Interface

ASSIGNMENT 9
Form Validation in Web Development

Name: G.Kalyani Krishna
Roll No: 22011102017
Date: 10.09.2024

Aim

To understand and implement form validation using HTML5 attributes and JavaScript, ensuring that user inputs follow specified criteria before form submission.

Introduction

Form validation is a key aspect of web development that ensures data integrity and prevents malicious input from being submitted. In this assignment, we focus on validating a user registration form with fields like name, email, password, and phone number using both HTML5 validation attributes and custom JavaScript functions for enhanced validation.

Code

index.html

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale
       =1.0">
    <title>Form Validation</title>
    <link rel="stylesheet" href="styles.css">
    <script src="scripts.js" defer></script>
</head>
<body>
    <header>
        <h1>Registration Form</h1>
    </header>
    <section id="form-section">
```

```html
        <form id="registration-form" action="#" method="post" onsubmit="return validateForm()">
            <label for="name">Name:</label>
            <input type="text" id="name" name="name" required pattern="[A-Za-z\s]+" title="Name should contain only letters.">

            <label for="email">Email:</label>
            <input type="email" id="email" name="email" required>

            <label for="password">Password:</label>
            <input type="password" id="password" name="password" required minlength="8" title="Password must be at least 8 characters long.">

            <label for="phone">Phone:</label>
            <input type="tel" id="phone" name="phone" required pattern="\d{10}" title="Enter a valid 10-digit phone number.">

            <button type="submit">Register</button>
        </form>
    </section>
</body>
</html>
```

scripts.js

```javascript
function validateForm() {
    let name = document.getElementById('name').value;
    let email = document.getElementById('email').value;
    let password = document.getElementById('password').value;
    let phone = document.getElementById('phone').value;

    // Simple regex for validating email format
    let emailPattern = /^[a-zA-Z0-9._-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,6}$/;
    if (!emailPattern.test(email)) {
        alert("Please enter a valid email address.");
        return false;
    }

    // Password length check
    if (password.length < 8) {
        alert("Password must be at least 8 characters long.");
        return false;
    }
```

```

    // Simple regex for validating phone number
    let phonePattern = /^\d{10}$/;
    if (!phonePattern.test(phone)) {
        alert("Please enter a valid 10-digit phone number.");
        return false;
    }

    return true;
}
```

styles.css

```
body {
    font-family: 'Arial', sans-serif;
    margin: 0;
    padding: 0;
    background-color: #f4f4f4;
    color: #333;
}

header {
    background-color: #007BFF;
    color: white;
    text-align: center;
    padding: 20px 0;
}

section {
    margin: 20px auto;
    width: 50%;
    background-color: white;
    padding: 20px;
    border-radius: 5px;
    box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
}

label {
    display: block;
    margin: 10px 0 5px;
}

input {
    width: 100%;
    padding: 10px;
    margin: 5px 0 15px;
    border-radius: 5px;
```

```
    border: 1px solid #ccc;
}

button {
    width: 100%;
    padding: 10px;
    background-color: #007BFF;
    color: white;
    border: none;
    border-radius: 5px;
    cursor: pointer;
}

button:hover {
    background-color: #0056b3;
}
```

Output

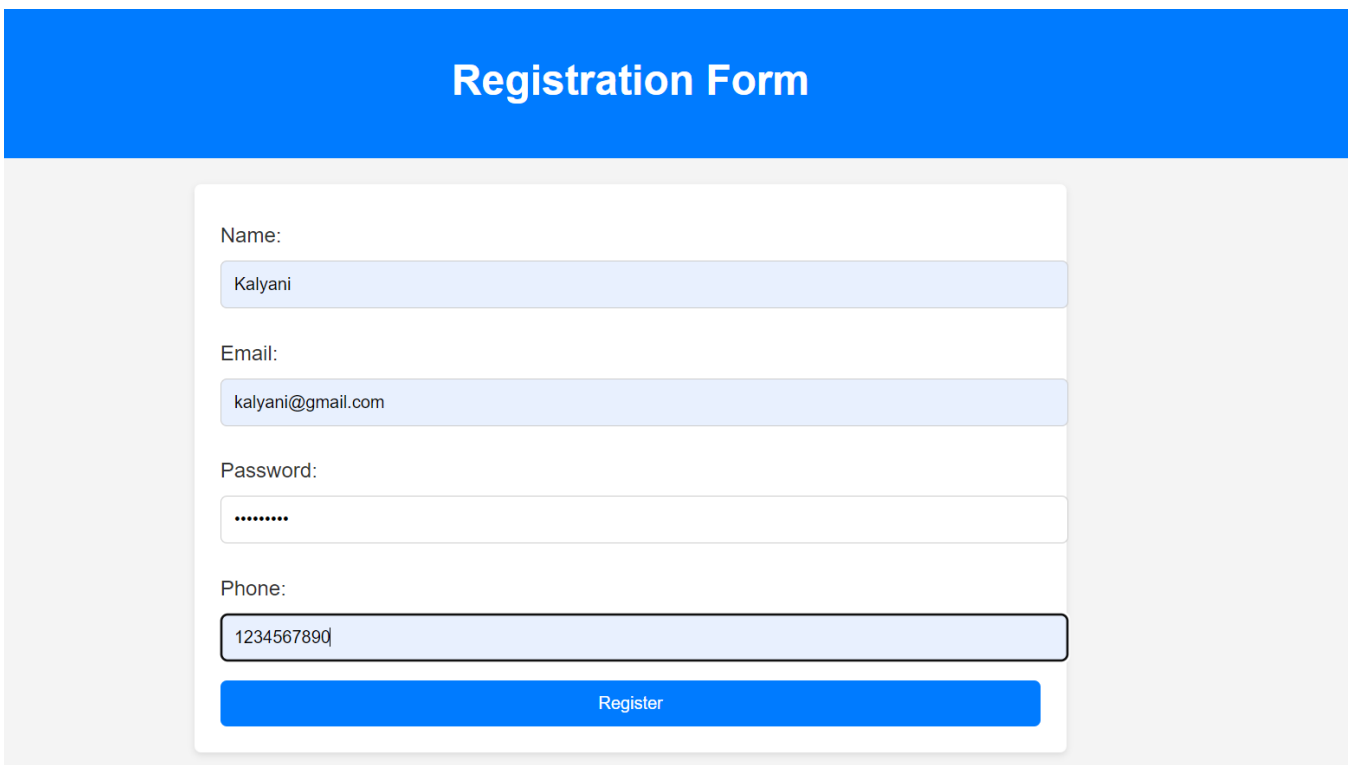

Figure 1: Form Validation Example

ASSIGNMENT 10
Fetching Data from Server with AJAX

Name: G.Kalyani Krishna
Roll No: 22011102017
Date: 17.09.2024

Aim

To implement a web page with form validation using HTML5 and JavaScript, and demonstrate AJAX to fetch data asynchronously.

Introduction

In this assignment, we learn about implementing basic form validation using both HTML5 attributes and JavaScript for additional validation logic. We also demonstrate how to use AJAX to load content asynchronously into a web page without refreshing it.

Code

ajax.html

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>AJAX Fetch Example</title>
</head>
<body>
    <h1>AJAX Data Fetch Example</h1>
    <button id="fetchDataBtn">Fetch Data</button>
    <div id="result"></div>

    <script>
        document.getElementById("fetchDataBtn").addEventListener("click", function() {
```

```
            var xhr = new XMLHttpRequest();
            xhr.open('GET', 'https://jsonplaceholder.typicode.com/posts/1', true);
            xhr.onload = function () {
                if (xhr.status === 200) {
                    var data = JSON.parse(xhr.responseText);
                    document.getElementById("result").innerHTML = `
                        <h3>Title: ${data.title}</h3>
                        <p>Body: ${data.body}</p>
                    `;
                } else {
                    document.getElementById("result").innerHTML = "Error: Unable to fetch data";
                }
            };
            xhr.send();
        });
    </script>
</body>
</html>
```

Output

Figure 1: AJAX Fetching Example

ASSIGNMENT 11
Implementation of a Semester Fee Form

Name: G.Kalyani Krishna
Roll No: 22011102017
Date: 01.10.2024

Aim

To design and implement a semester fee form that collects user data and calculates the total fee based on user inputs.

Introduction

The semester fee form is an essential tool for educational institutions to streamline fee collection and management. This assignment involves creating a web-based form that allows students to input their personal information, course selection, and associated fees. The form will validate the inputs and calculate the total fee dynamically using JavaScript.

Code

index.html

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale
       =1.0">
    <title>Semester Fee Form</title>
    <link rel="stylesheet" href="styles.css">
    <script src="scripts.js" defer></script>
</head>
<body>
    <header>
        <h1>Semester Fee Form</h1>
    </header>
    <section id="form-section">
```

```html
        <form id="fee-form" action="#" method="post" onsubmit="return calculateTotal()">
            <label for="student-name">Student Name:</label>
            <input type="text" id="student-name" name="student-name" required>

            <label for="course">Course:</label>
            <select id="course" name="course" required>
                <option value="" disabled selected>Select your course</option>
                <option value="btech">B.Tech - INR 2000</option>
                <option value="mtech">M.Tech - INR 2500</option>
                <option value="mba">MBA - INR 3000</option>
            </select>

            <label for="scholarship">Scholarship (%) :</label>
            <input type="number" id="scholarship" name="scholarship" min="0" max="100" value="0">

            <button type="submit">Calculate Fee</button>
        </form>
        <div id="result"></div>
    </section>
</body>
</html>
```

scripts.js

```javascript
function calculateTotal() {
    const courseFees = {
        btech: 2000,
        mtech: 2500,
        mba: 3000
    };

    let course = document.getElementById('course').value;
    let scholarship = document.getElementById('scholarship').value;

    if (course && scholarship >= 0) {
        let fee = courseFees[course];
        let discount = (scholarship / 100) * fee;
        let totalFee = fee - discount;

        document.getElementById('result').innerHTML = `Total Fee: INR ${totalFee.toFixed(2)}`;
        return false; // Prevent form submission
    } else {
```

```
        alert("Please fill in all fields correctly.");
        return false;
    }
}
```

styles.css

```
body {
    font-family: 'Arial', sans-serif;
    margin: 0;
    padding: 0;
    background-color: #f4f4f4;
    color: #333;
}

header {
    background-color: #007BFF;
    color: white;
    text-align: center;
    padding: 20px 0;
}

section {
    margin: 20px auto;
    width: 50%;
    background-color: white;
    padding: 20px;
    border-radius: 5px;
    box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
}

label {
    display: block;
    margin: 10px 0 5px;
}

input, select {
    width: 100%;
    padding: 10px;
    margin: 5px 0 15px;
    border-radius: 5px;
    border: 1px solid #ccc;
}

button {
    width: 100%;
    padding: 10px;
```

```
    background-color: #007BFF;
    color: white;
    border: none;
    border-radius: 5px;
    cursor: pointer;
}

button:hover {
    background-color: #0056b3;
}

#result {
    margin-top: 20px;
    font-size: 1.2em;
    color: #333;
}
```

Output

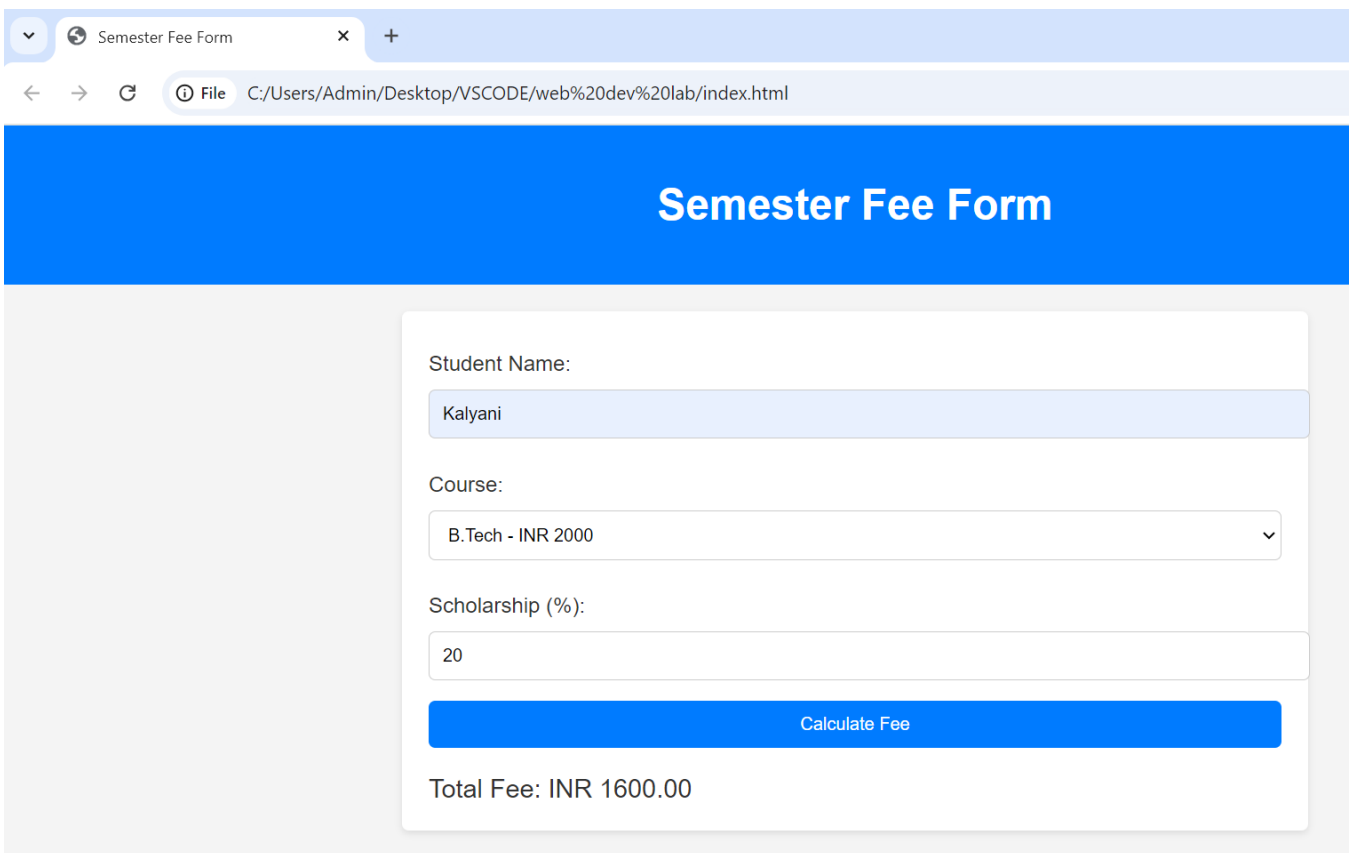

Figure 1: Semester Fee Form Example