JavaScript_Day – 1_Week2_Hands_On -- Harshitha Kamatam

Problem 1

Problem Statement:

Personal Notes Saver using LocalStorage (Level-1)

Scenario
You are building a simple web page where users can write daily notes and save them in their browser without using a server.

Requirements

- **A textarea for writing notes.**
- **A Save button (using onclick).**
- **A Clear button.**
- **Notes must:**

  Be stored in localStorage

  Automatically load when the page refreshes
- **Display stored note on page load.**

Technical Constraints

- **Must use:**
  - **onclick inline event**
  - **localStorage.setItem()**
  - **localStorage.getItem()**
  - **localStorage.removeItem()**
- **No backend/database.**
- **Pure HTML + JavaScript only.**
- **Data stored as key-value pair.**

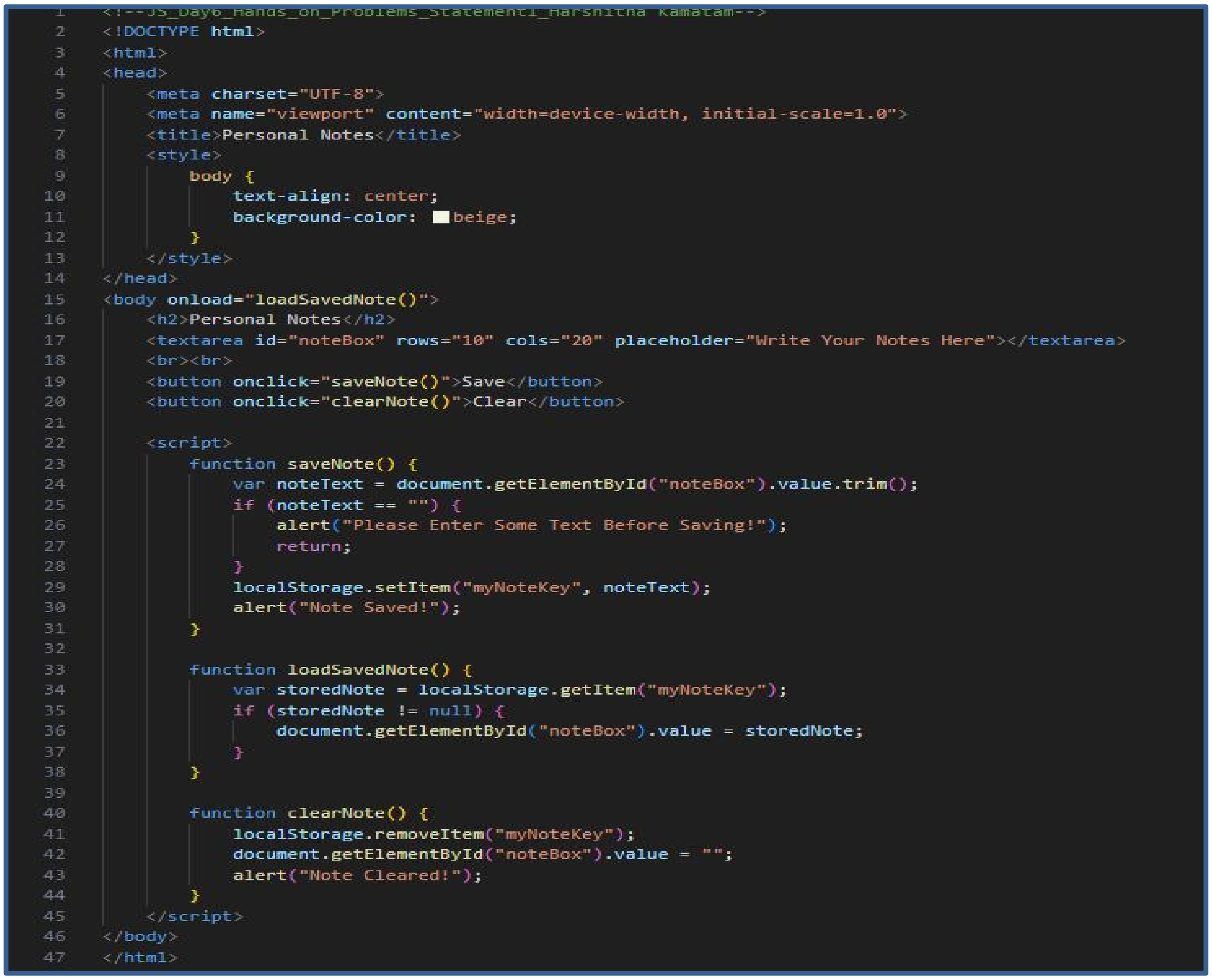
Learning Outcome

You should be able to:

- **Inline event handling (onclick)**
- **Browser Storage APIs**
- **Storing & retrieving key-value data**
- **Page load event handling**
- **Basic DOM manipulation**

Source Code:

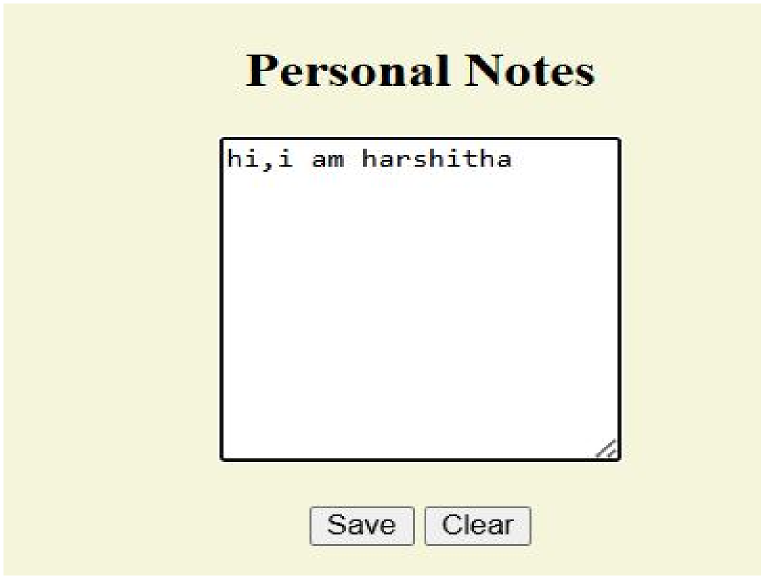
File Name: Index.html

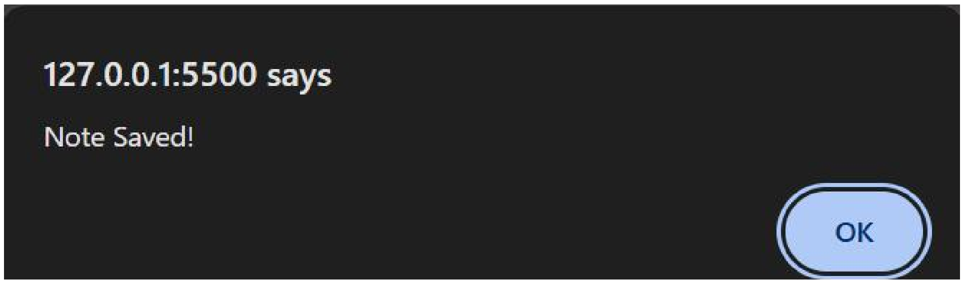


```html
<!--JS_Day6_Hands_on_Problems_Statement1_Harshitha Kamatam-->
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Personal Notes</title>
    <style>
        body {
            text-align: center;
            background-color: beige;
        }
    </style>
</head>
<body onload="loadSavedNote()">
    <h2>Personal Notes</h2>
    <textarea id="noteBox" rows="10" cols="20" placeholder="Write Your Notes Here"></textarea>
    <br><br>
    <button onclick="saveNote()">Save</button>
    <button onclick="clearNote()">Clear</button>

    <script>
        function saveNote() {
            var noteText = document.getElementById("noteBox").value.trim();
            if (noteText == "") {
                alert("Please Enter Some Text Before Saving!");
                return;
            }
            localStorage.setItem("myNoteKey", noteText);
            alert("Note Saved!");
        }

        function loadSavedNote() {
            var storedNote = localStorage.getItem("myNoteKey");
            if (storedNote != null) {
                document.getElementById("noteBox").value = storedNote;
            }
        }

        function clearNote() {
            localStorage.removeItem("myNoteKey");
            document.getElementById("noteBox").value = "";
            alert("Note Cleared!");
        }
    </script>
</body>
</html>
```

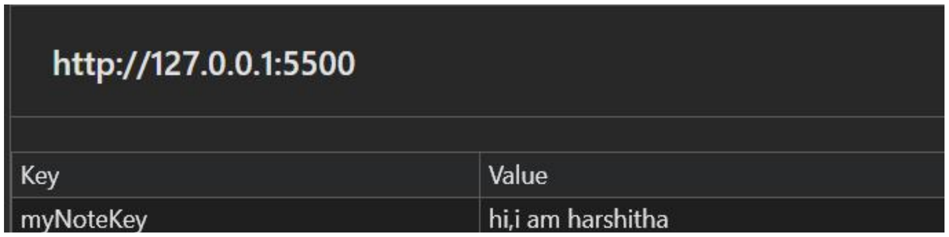
Output:

Output File: index.html

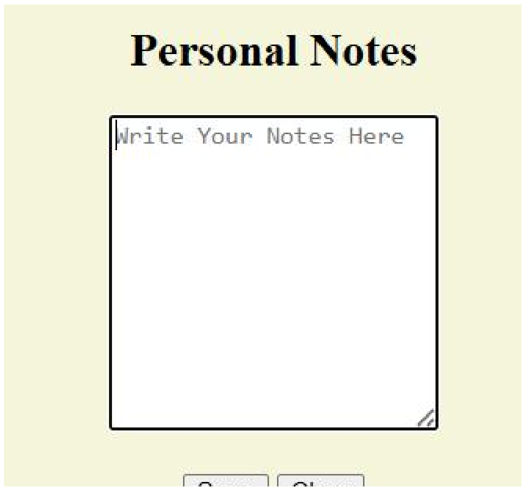


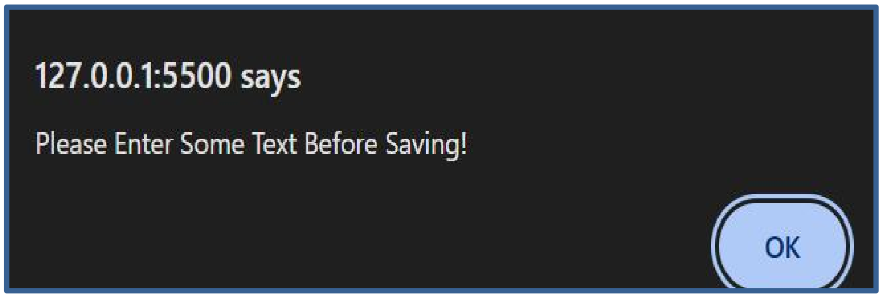

127.0.0.1:5500 says

Note Saved!


http://127.0.0.1:5500

Key	Value
myNoteKey	hi,i am harshitha


Personal Notes


127.0.0.1:5500 says

Please Enter Some Text Before Saving!

Explanation:

This code creates a simple **Personal Notes app** using a textarea and buttons. When you click **Save**, it stores your note in the browser's localStorage, so it stays even after refreshing the page. When the page loads, it automatically shows the saved note, and if you click **Clear**, it deletes the note. It also shows an alert if you try to save an empty note.

Problem 2

Problem Statement:

Live Form Validation with Events (Level-1)

Scenario

Create a simple registration form that validates user input when fields change.

Requirements

- **Fields:**
  - **Name**
  - **Email**
  - **Age**
- **Use:**
  - **onchange**
  - **onclick**
- **Validate:**
  - **Name cannot be empty**
  - **Email must contain "@"**
  - **Age must be greater than 18**
- **Display validation message dynamically.**
- **Store valid user data in sessionStorage.**

Technical Constraints

- **Must use inline onchange events.**
- **Store valid data using:**
  - **sessionStorage.setItem()**
- **No external libraries.**
- **Use basic JavaScript only.**

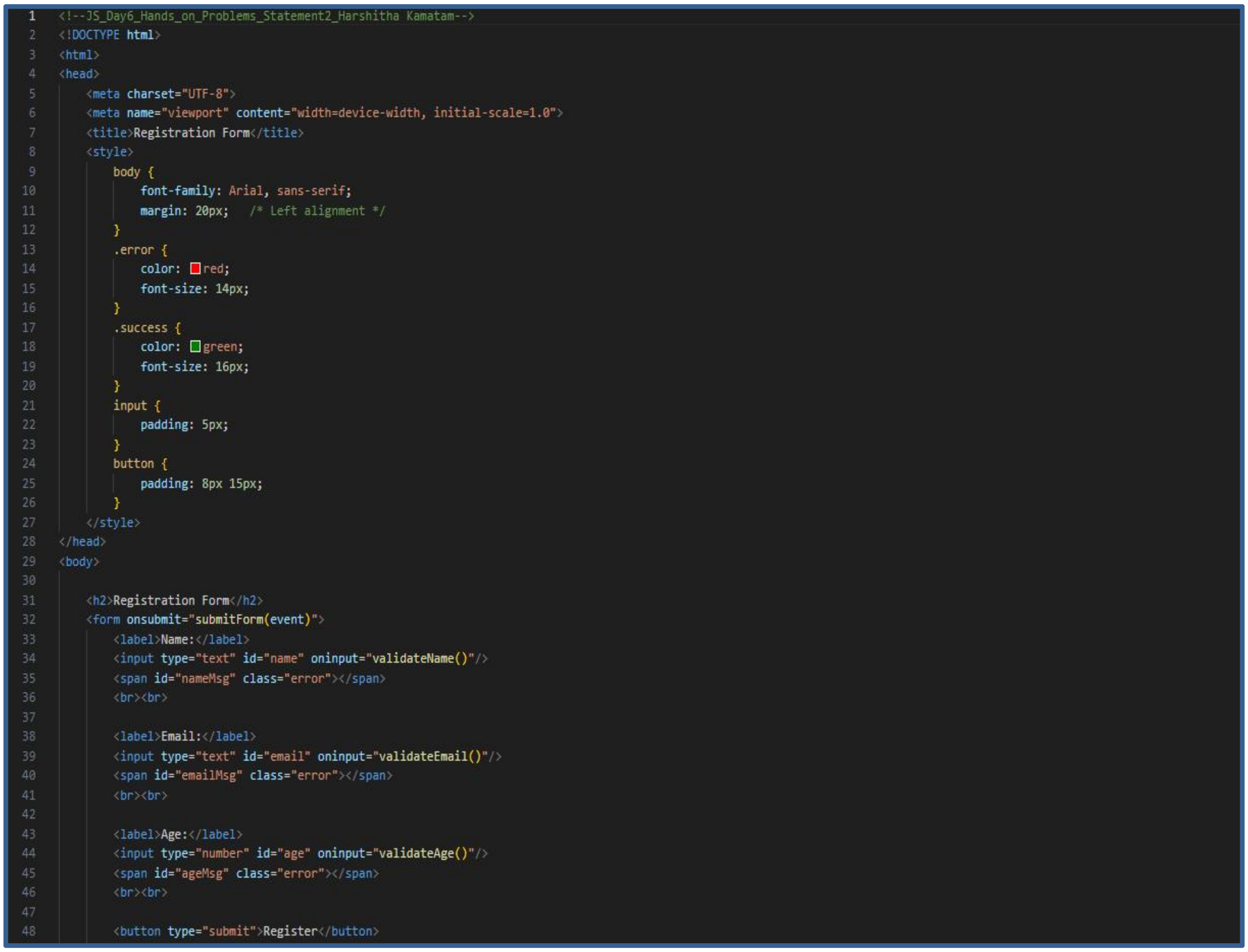
Learning Outcome

You will be able to:

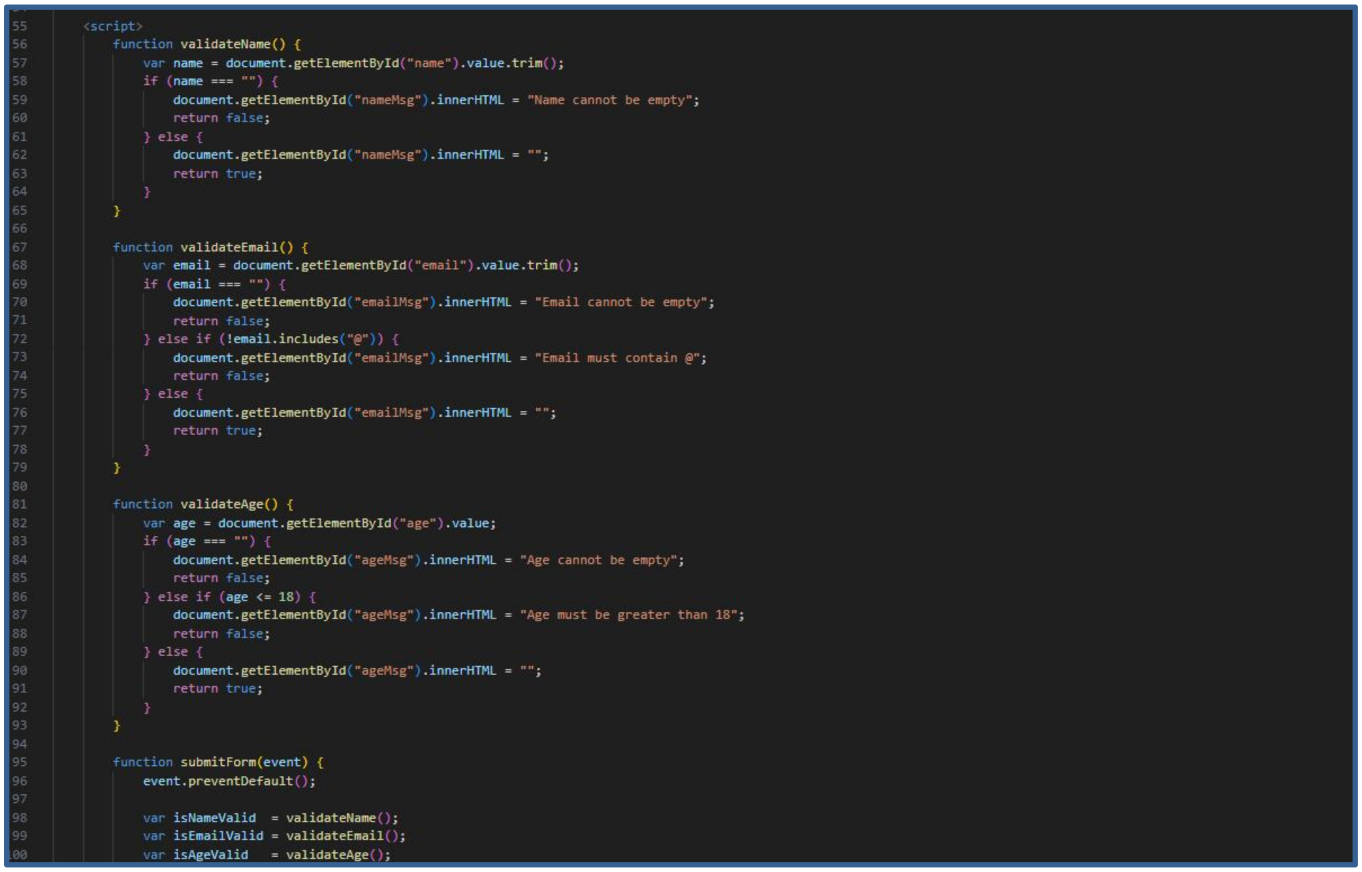
- **onchange event usage**
- **Form validation logic**
- **Difference between localStorage and sessionStorage**
- **Dynamic DOM updates**

Source Code:

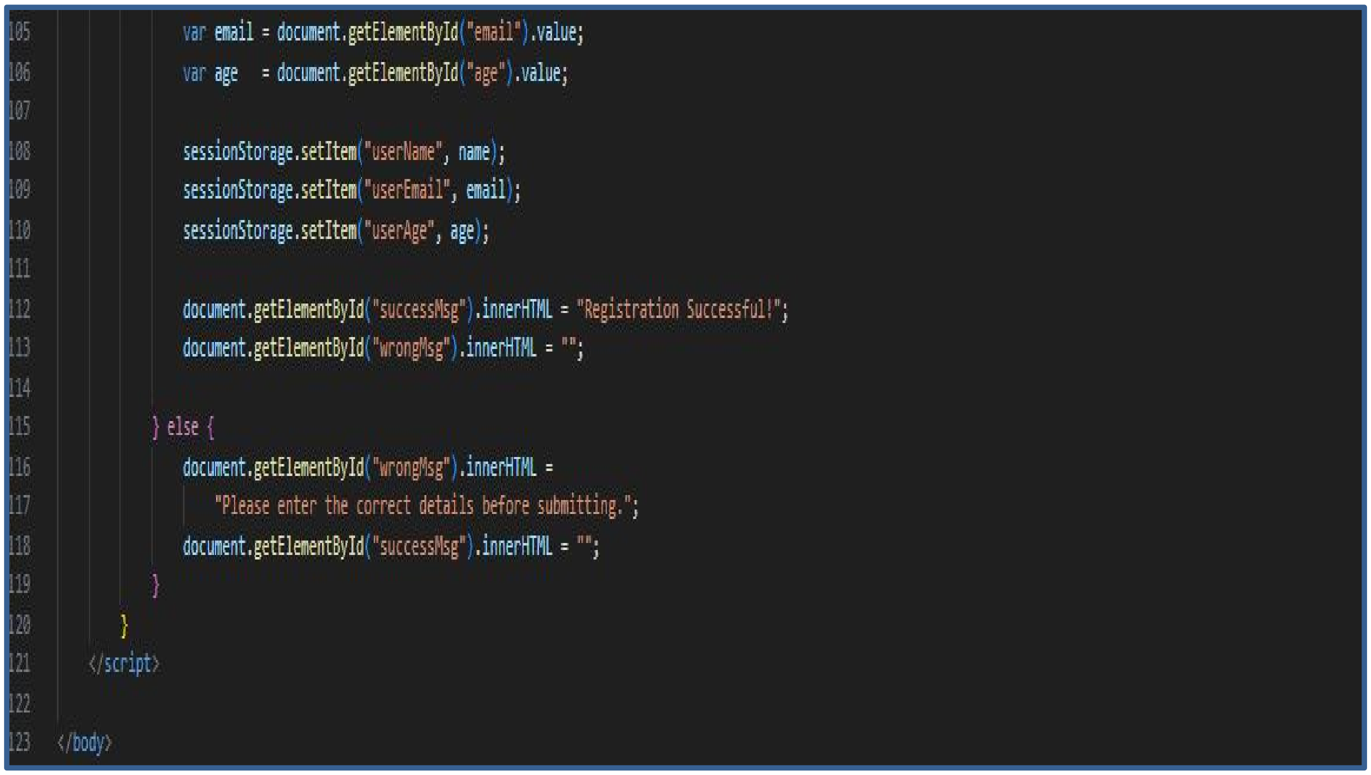
File Name: index.html



```html
<!--JS_Day6_Hands_on_Problems_Statement2_Harshitha Kamatam-->
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Registration Form</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 20px;   /* Left alignment */
        }
        .error {
            color: red;
            font-size: 14px;
        }
        .success {
            color: green;
            font-size: 16px;
        }
        input {
            padding: 5px;
        }
        button {
            padding: 8px 15px;
        }
    </style>
</head>
<body>

    <h2>Registration Form</h2>
    <form onsubmit="submitForm(event)">
        <label>Name:</label>
        <input type="text" id="name" oninput="validateName()"/>
        <span id="nameMsg" class="error"></span>
        <br><br>

        <label>Email:</label>
        <input type="text" id="email" oninput="validateEmail()"/>
        <span id="emailMsg" class="error"></span>
        <br><br>

        <label>Age:</label>
        <input type="number" id="age" oninput="validateAge()"/>
        <span id="ageMsg" class="error"></span>
        <br><br>

        <button type="submit">Register</button>
```



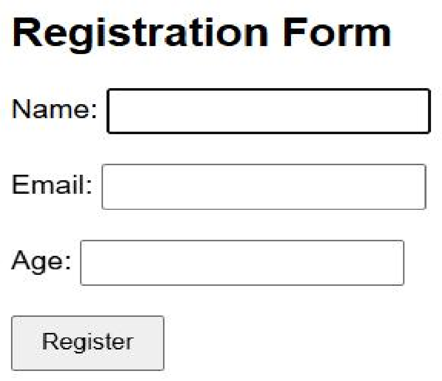
```
    <script>
        function validateName() {
            var name = document.getElementById("name").value.trim();
            if (name === "") {
                document.getElementById("nameMsg").innerHTML = "Name cannot be empty";
                return false;
            } else {
                document.getElementById("nameMsg").innerHTML = "";
                return true;
            }
        }

        function validateEmail() {
            var email = document.getElementById("email").value.trim();
            if (email === "") {
                document.getElementById("emailMsg").innerHTML = "Email cannot be empty";
                return false;
            } else if (!email.includes("@")) {
                document.getElementById("emailMsg").innerHTML = "Email must contain @";
                return false;
            } else {
                document.getElementById("emailMsg").innerHTML = "";
                return true;
            }
        }

        function validateAge() {
            var age = document.getElementById("age").value;
            if (age === "") {
                document.getElementById("ageMsg").innerHTML = "Age cannot be empty";
                return false;
            } else if (age <= 18) {
                document.getElementById("ageMsg").innerHTML = "Age must be greater than 18";
                return false;
            } else {
                document.getElementById("ageMsg").innerHTML = "";
                return true;
            }
        }

        function submitForm(event) {
            event.preventDefault();

            var isNameValid  = validateName();
            var isEmailValid = validateEmail();
            var isAgeValid   = validateAge();
```



```
                var email = document.getElementById("email").value;
                var age   = document.getElementById("age").value;

                sessionStorage.setItem("userName", name);
                sessionStorage.setItem("userEmail", email);
                sessionStorage.setItem("userAge", age);

                document.getElementById("successMsg").innerHTML = "Registration Successful!";
                document.getElementById("wrongMsg").innerHTML = "";

            } else {
                document.getElementById("wrongMsg").innerHTML =
                    "Please enter the correct details before submitting.";
                document.getElementById("successMsg").innerHTML = "";
            }
        }
    </script>

</body>
```

Output:

Output File: index.html

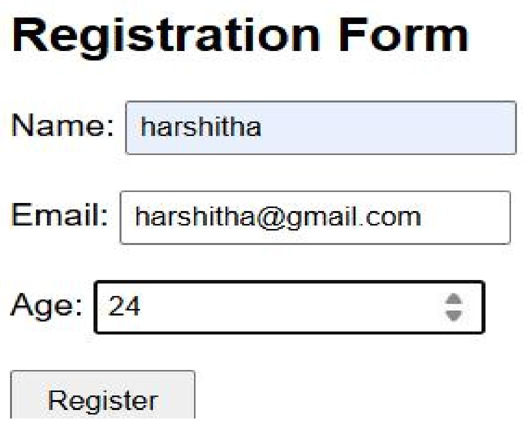

Registration Form

Name:

Email:

Age:

Register

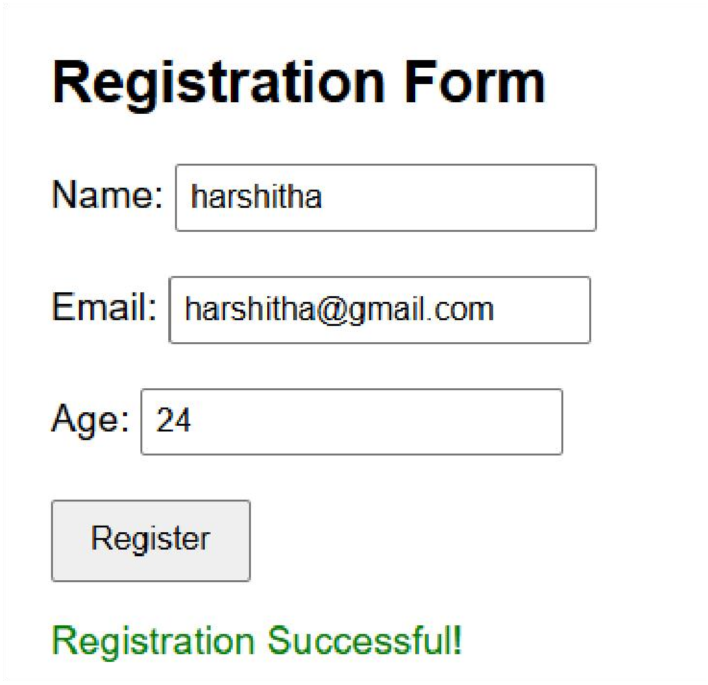

Registration Form

Name: harshitha

Email: harshitha@gmail.com

Age: 24

Register


Registration Form

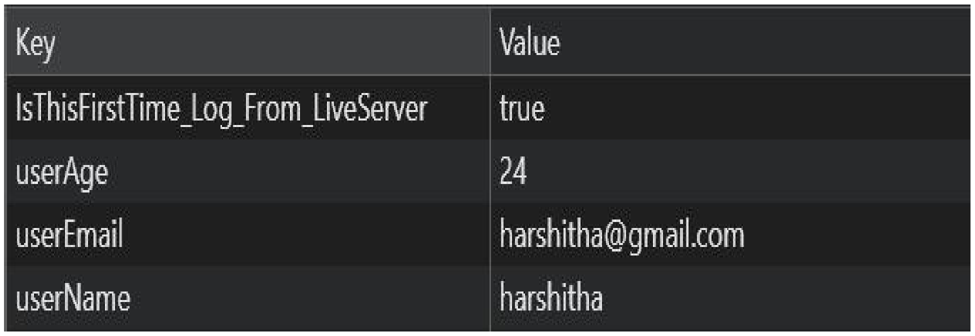
Name: harshitha

Email: harshitha@gmail.com

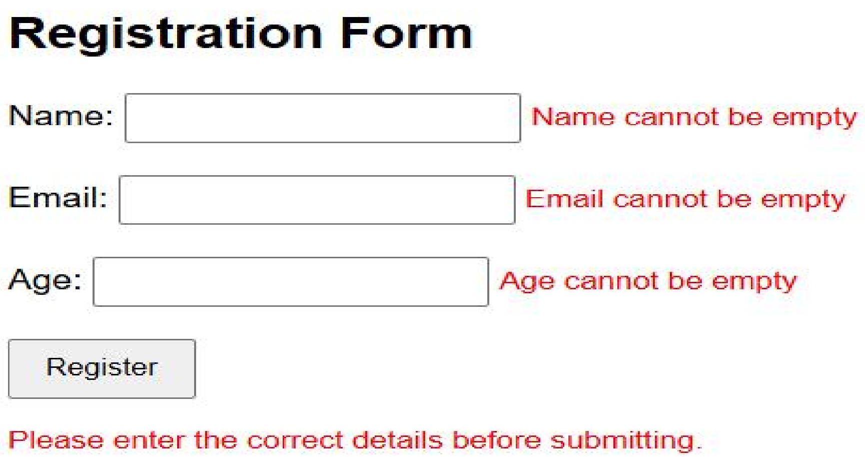
Age: 24

Register

Registration Successful!



Key	Value
IsThisFirstTime_Log_From_LiveServer	true
userAge	24
userEmail	harshitha@gmail.com
userName	harshitha



Registration Form

Name: Name cannot be empty

Email: Email cannot be empty

Age: Age cannot be empty

Please enter the correct details before submitting.

Explanation:

This code creates a **Registration Form** with validation using JavaScript. It checks if the name, email, and age are filled correctly (email must contain "@" and age must be above 18). If everything is correct, it shows "Registration Successful!" and stores the details in sessionStorage. If not, it displays error messages in red and asks the user to enter valid details.

Problem 3

Problem Statement:

Location-Based Weather Logger (Level-2)

Scenario
Create a web application that fetches the user's geographic location and stores location history in localStorage.

Requirements

- Button: **Get My Location**
- Use:

  navigator.geolocation.getCurrentPosition()

- Display:

  Latitude

  Longitude

- Handle:

  Permission denied

  Timeout

Location unavailable

- Save last 5 location entries in localStorage.
- Display location history on page load.

Technical Constraints

- Must handle:

  Success callback

  Error callback

- Use browser permission handling.
- Store data as JSON using:

  JSON.stringify()

  JSON.parse()
- Use inline event (onclick).

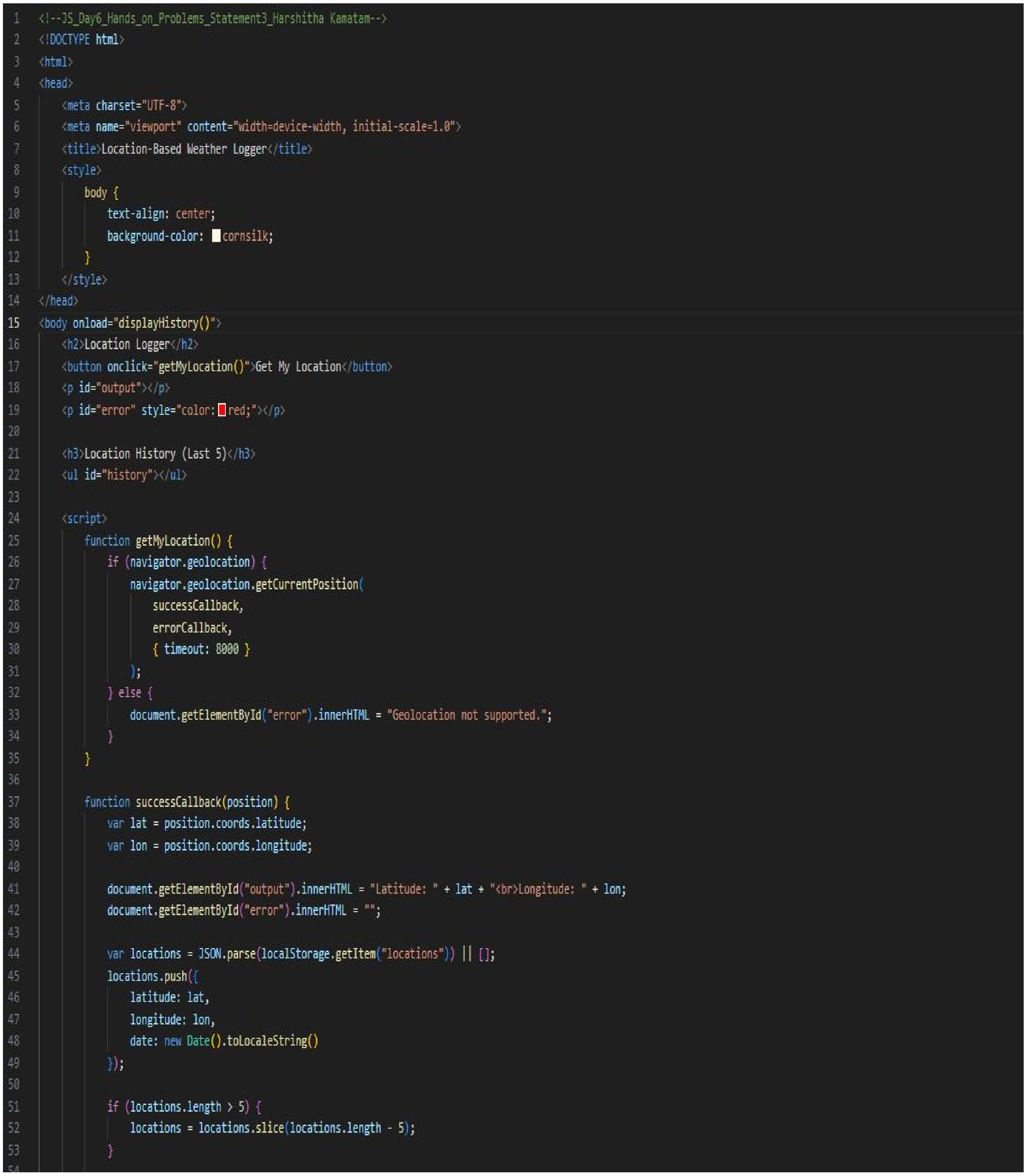
Learning Outcome

Learners should be able to:

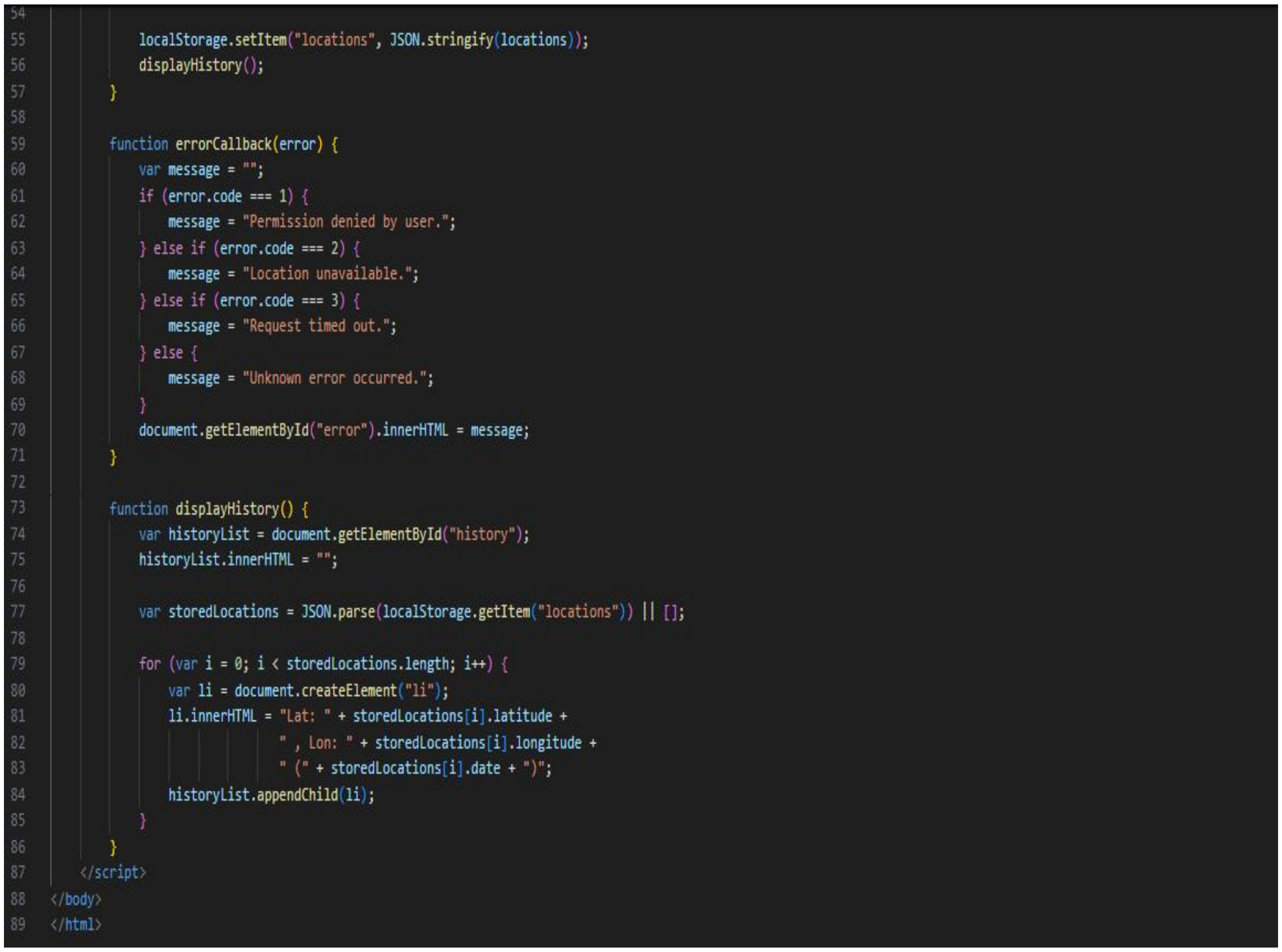
- Geolocation API
- Handling browser permissions
- Error handling in APIs
- Managing structured data in localStorage
- JSON parsing and stringifying

Source Code:

File Name: index.html



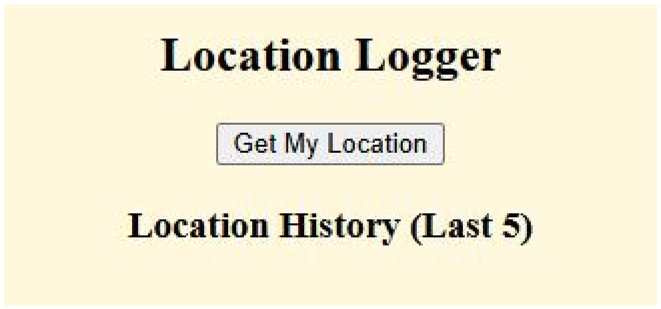
```html
<!--JS_Day6_Hands_on_Problems_Statement3_Harshitha Kamatam-->
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Location-Based Weather Logger</title>
    <style>
        body {
            text-align: center;
            background-color: cornsilk;
        }
    </style>
</head>
<body onload="displayHistory()">
    <h2>Location Logger</h2>
    <button onclick="getMyLocation()">Get My Location</button>
    <p id="output"></p>
    <p id="error" style="color: red;"></p>

    <h3>Location History (Last 5)</h3>
    <ul id="history"></ul>

    <script>
        function getMyLocation() {
            if (navigator.geolocation) {
                navigator.geolocation.getCurrentPosition(
                    successCallback,
                    errorCallback,
                    { timeout: 8000 }
                );
            } else {
                document.getElementById("error").innerHTML = "Geolocation not supported.";
            }
        }

        function successCallback(position) {
            var lat = position.coords.latitude;
            var lon = position.coords.longitude;

            document.getElementById("output").innerHTML = "Latitude: " + lat + "<br>Longitude: " + lon;
            document.getElementById("error").innerHTML = "";

            var locations = JSON.parse(localStorage.getItem("locations")) || [];
            locations.push({
                latitude: lat,
                longitude: lon,
                date: new Date().toLocaleString()
            });

            if (locations.length > 5) {
                locations = locations.slice(locations.length - 5);
            }
```



```
            localStorage.setItem("locations", JSON.stringify(locations));
            displayHistory();
        }

        function errorCallback(error) {
            var message = "";
            if (error.code === 1) {
                message = "Permission denied by user.";
            } else if (error.code === 2) {
                message = "Location unavailable.";
            } else if (error.code === 3) {
                message = "Request timed out.";
            } else {
                message = "Unknown error occurred.";
            }
            document.getElementById("error").innerHTML = message;
        }

        function displayHistory() {
            var historyList = document.getElementById("history");
            historyList.innerHTML = "";

            var storedLocations = JSON.parse(localStorage.getItem("locations")) || [];

            for (var i = 0; i < storedLocations.length; i++) {
                var li = document.createElement("li");
                li.innerHTML = "Lat: " + storedLocations[i].latitude +
                               " , Lon: " + storedLocations[i].longitude +
                               " (" + storedLocations[i].date + ")";
                historyList.appendChild(li);
            }
        }
    </script>
</body>
</html>
```

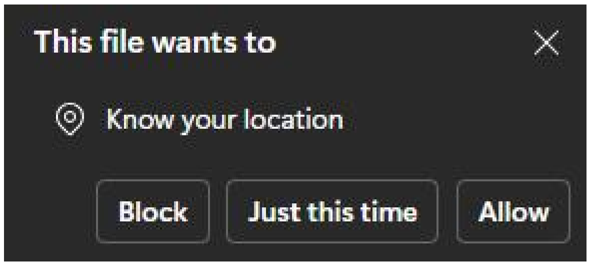
Output:

Output File: index.html



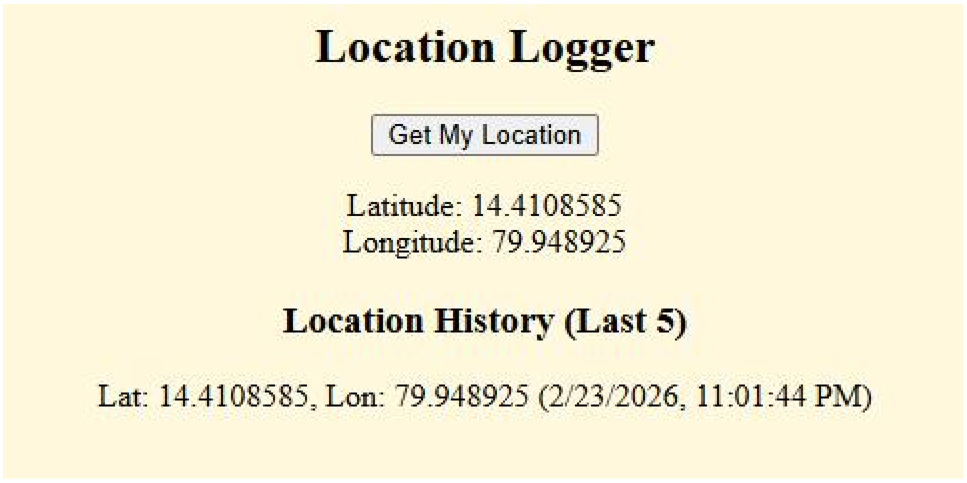
Location Logger

Location History (Last 5)



This file wants to

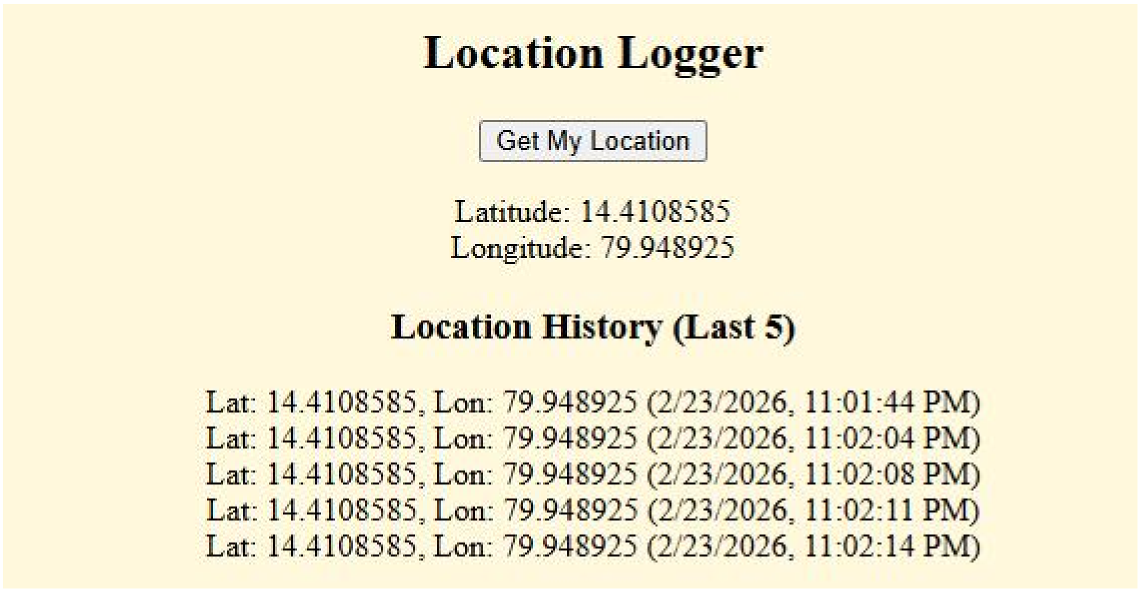
Know your location


Location Logger

Latitude: 14.4108585
Longitude: 79.948925

Location History (Last 5)

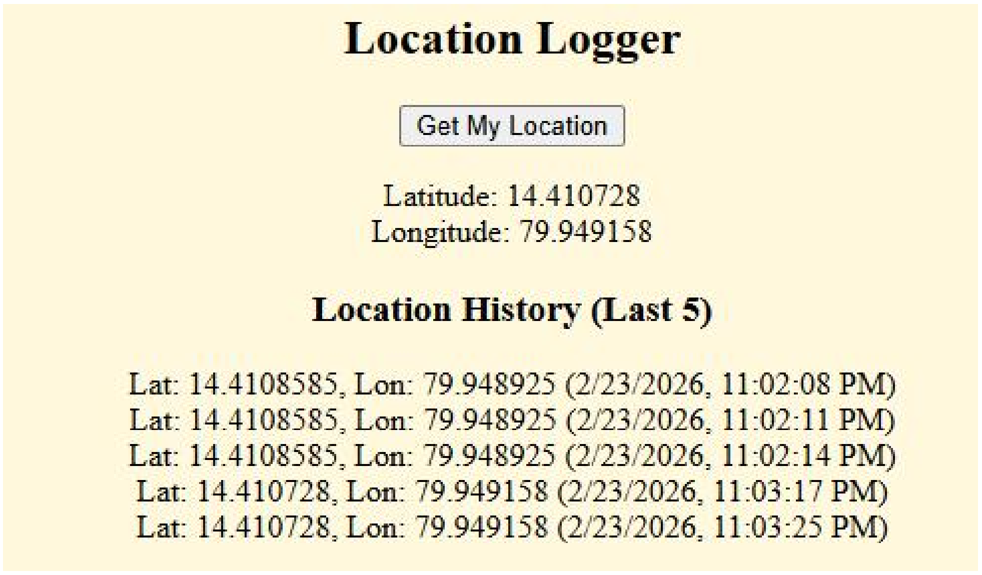
Lat: 14.4108585, Lon: 79.948925 (2/23/2026, 11:01:44 PM)


Location Logger

Latitude: 14.4108585
Longitude: 79.948925

Location History (Last 5)

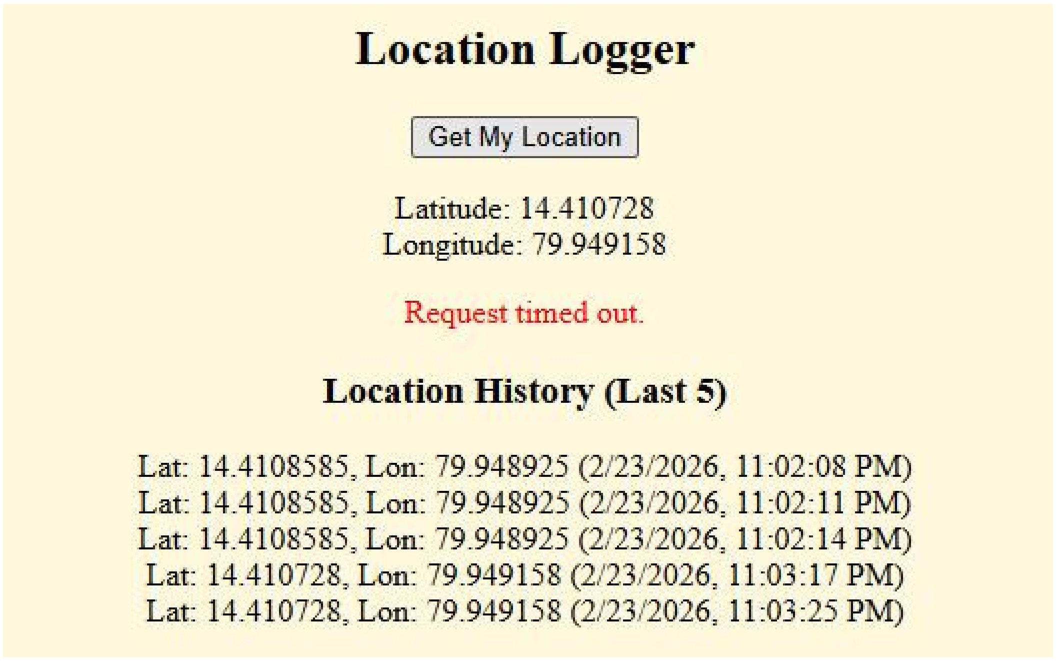
Lat: 14.4108585, Lon: 79.948925 (2/23/2026, 11:01:44 PM)
Lat: 14.4108585, Lon: 79.948925 (2/23/2026, 11:02:04 PM)
Lat: 14.4108585, Lon: 79.948925 (2/23/2026, 11:02:08 PM)
Lat: 14.4108585, Lon: 79.948925 (2/23/2026, 11:02:11 PM)
Lat: 14.4108585, Lon: 79.948925 (2/23/2026, 11:02:14 PM)


Location Logger

Latitude: 14.410728
Longitude: 79.949158

Location History (Last 5)

Lat: 14.4108585, Lon: 79.948925 (2/23/2026, 11:02:08 PM)
Lat: 14.4108585, Lon: 79.948925 (2/23/2026, 11:02:11 PM)
Lat: 14.4108585, Lon: 79.948925 (2/23/2026, 11:02:14 PM)
Lat: 14.410728, Lon: 79.949158 (2/23/2026, 11:03:17 PM)
Lat: 14.410728, Lon: 79.949158 (2/23/2026, 11:03:25 PM)



```
Location Logger

Get My Location

Latitude: 14.410728
Longitude: 79.949158

Request timed out.

Location History (Last 5)

Lat: 14.4108585, Lon: 79.948925 (2/23/2026, 11:02:08 PM)
Lat: 14.4108585, Lon: 79.948925 (2/23/2026, 11:02:11 PM)
Lat: 14.4108585, Lon: 79.948925 (2/23/2026, 11:02:14 PM)
Lat: 14.410728, Lon: 79.949158 (2/23/2026, 11:03:17 PM)
Lat: 14.410728, Lon: 79.949158 (2/23/2026, 11:03:25 PM)
```

Explanation:

This code creates a **Location Logger** that uses the browser's geolocation feature to get your current latitude and longitude when you click "Get My Location." It asks for permission, shows your location on the screen, and handles errors like timeout or permission denied. It also saves the last 5 locations in localStorage and displays them as history, so you can see your recent location logs even after refreshing the page.

Problem 4

Problem Statement:

Mini Expense Tracker using Client-Side Database (Level-2)

Scenario
Develop a client-side expense tracker where users can add, view, and delete expenses using a browser-supported database.

Requirements

1. **Fields:**
   - **Expense Title**
   - **Amount**
   - **Date**
2. **Buttons:**
   - **Add Expense**
   - **View Expenses**
   - **Delete Expense**
3. **Use:**
   - **Client-side database (Web SQL or IndexedDB)**
4. **Execute SQL-like queries:**
   - **CREATE TABLE**
   - **INSERT**
   - **SELECT**
   - **DELETE**
5. **Maintain transaction handling.**
6. **Display expense list dynamically.**

Technical Constraints

- Must use:
    - Client-side DB transactions
    - SQL execution methods
- Must handle:
    - Transaction errors
    - Query errors
- No server/database allowed.
- Pure JavaScript + HTML.

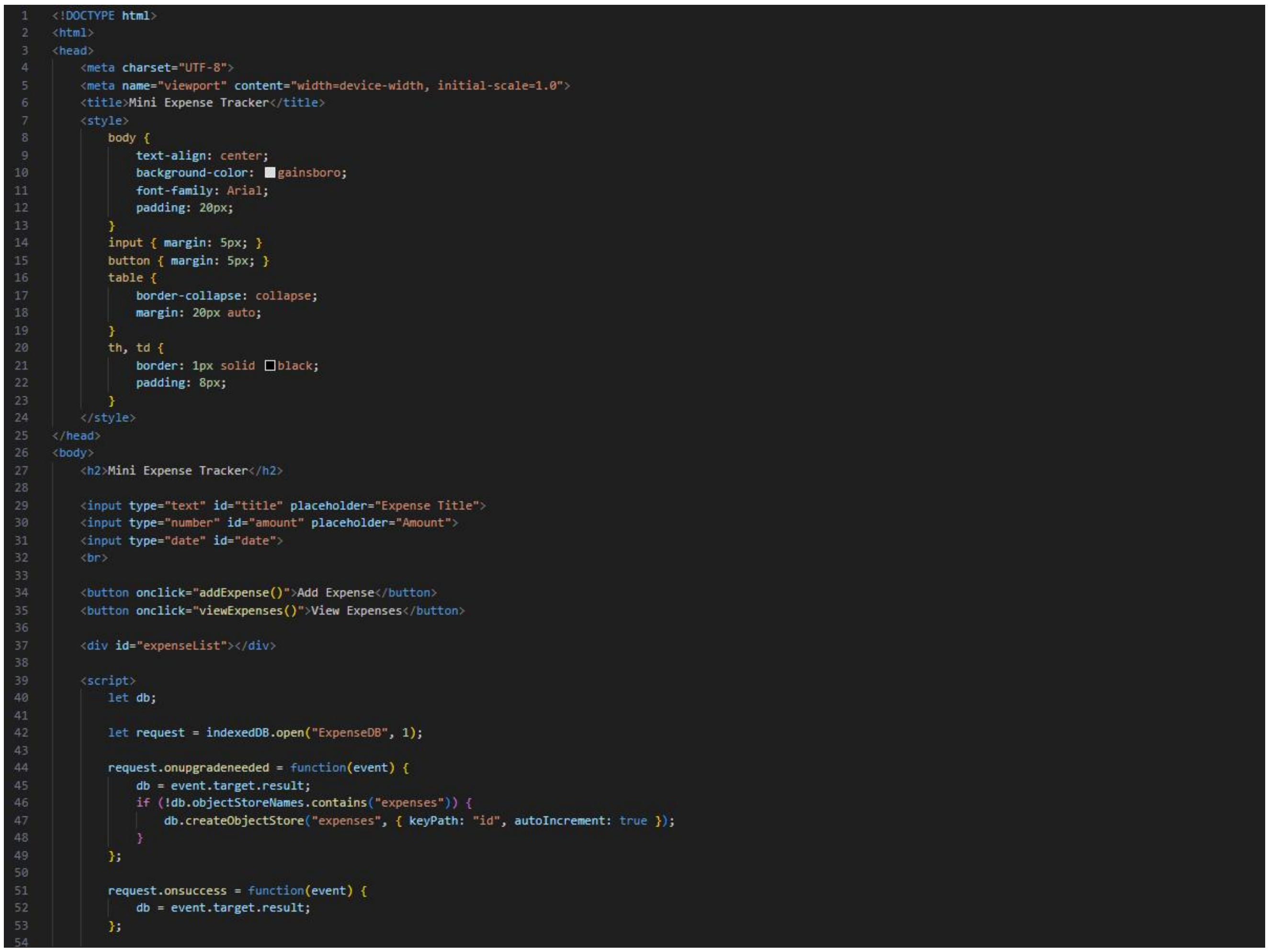
Learning Outcome

You will be able to:

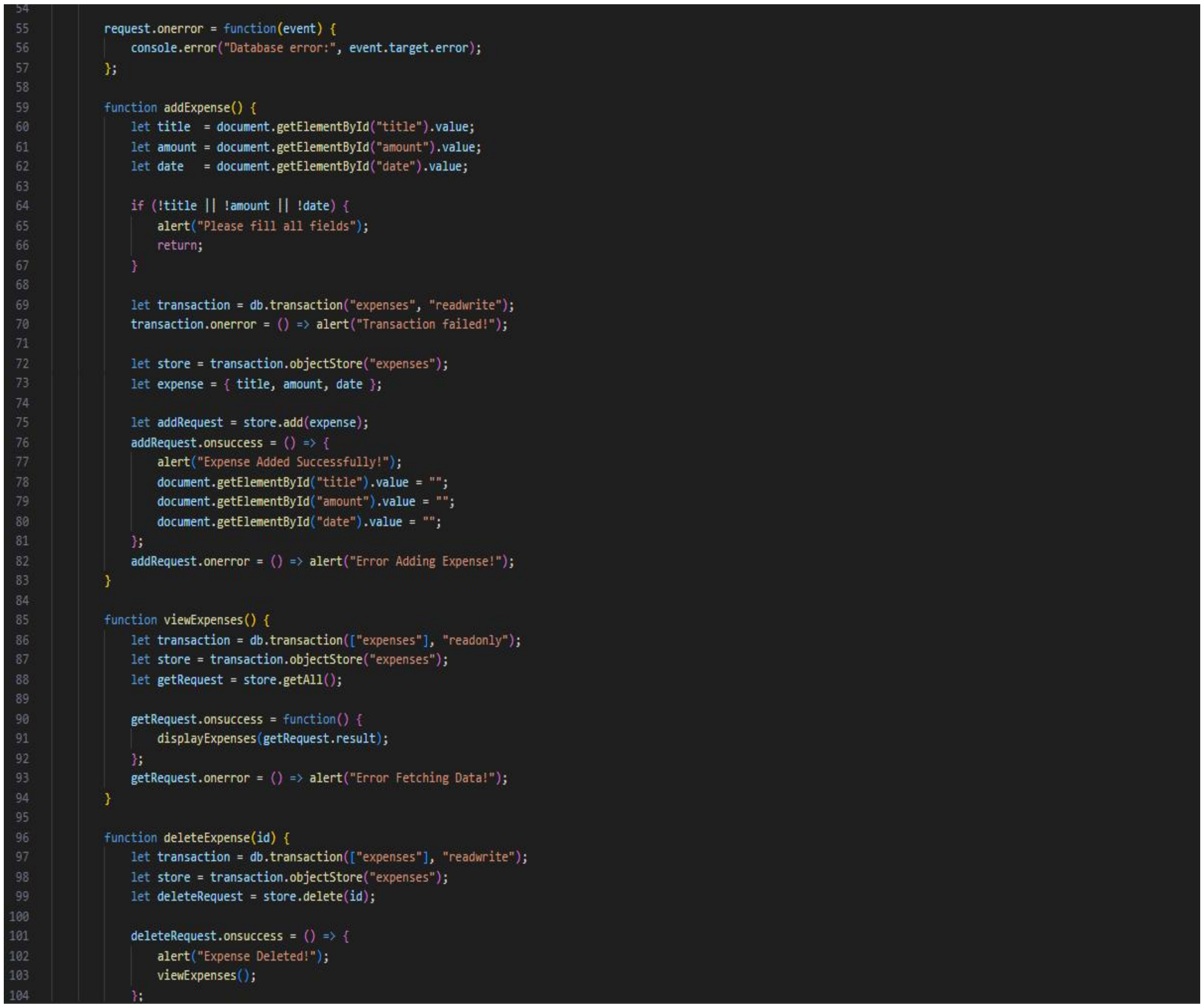
- Client-side database concepts
- Executing SQL queries in browser
- Managing transactions
- Advanced DOM rendering
- Persistent structured storage

Source Code:

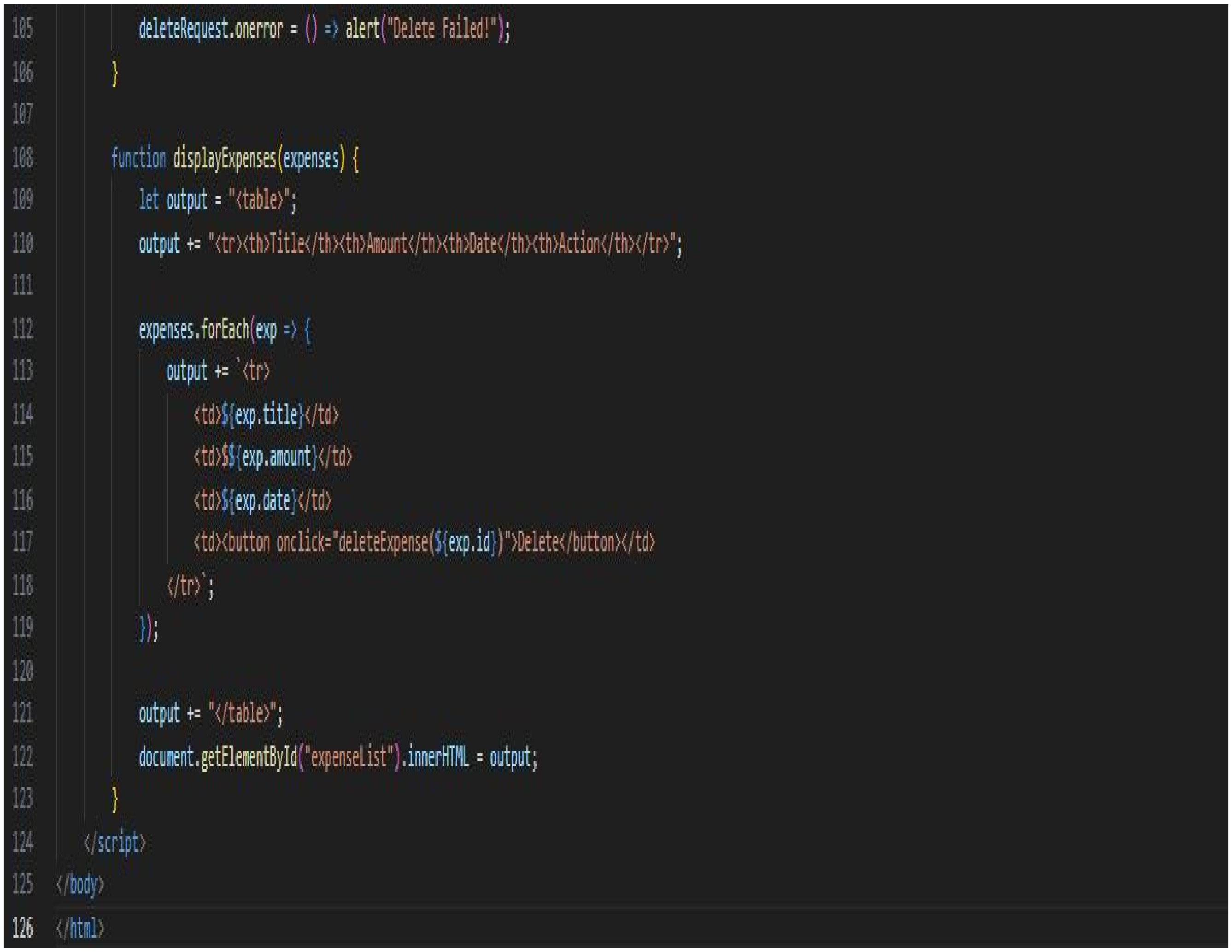
File Name: index.html



```html
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Mini Expense Tracker</title>
    <style>
        body {
            text-align: center;
            background-color: gainsboro;
            font-family: Arial;
            padding: 20px;
        }
        input { margin: 5px; }
        button { margin: 5px; }
        table {
            border-collapse: collapse;
            margin: 20px auto;
        }
        th, td {
            border: 1px solid black;
            padding: 8px;
        }
    </style>
</head>
<body>
    <h2>Mini Expense Tracker</h2>

    <input type="text" id="title" placeholder="Expense Title">
    <input type="number" id="amount" placeholder="Amount">
    <input type="date" id="date">
    <br>

    <button onclick="addExpense()">Add Expense</button>
    <button onclick="viewExpenses()">View Expenses</button>

    <div id="expenseList"></div>

    <script>
        let db;

        let request = indexedDB.open("ExpenseDB", 1);

        request.onupgradeneeded = function(event) {
            db = event.target.result;
            if (!db.objectStoreNames.contains("expenses")) {
                db.createObjectStore("expenses", { keyPath: "id", autoIncrement: true });
            }
        };

        request.onsuccess = function(event) {
            db = event.target.result;
        };

```



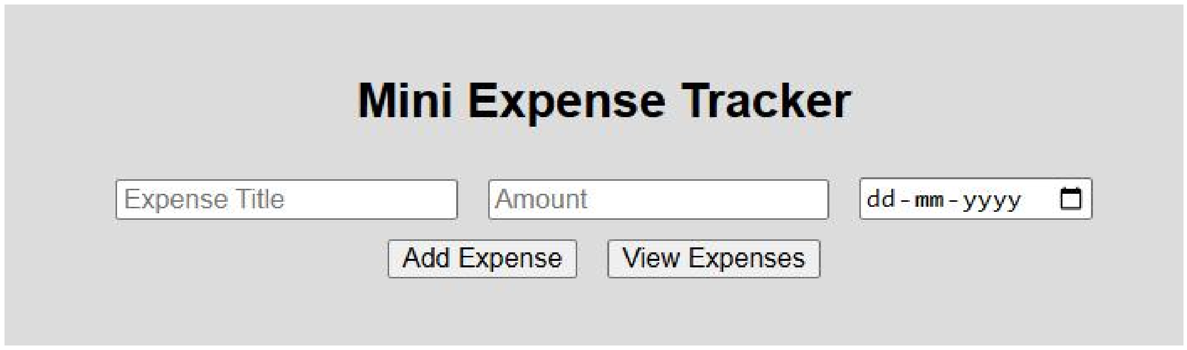
```
        request.onerror = function(event) {
            console.error("Database error:", event.target.error);
        };

        function addExpense() {
            let title  = document.getElementById("title").value;
            let amount = document.getElementById("amount").value;
            let date   = document.getElementById("date").value;

            if (!title || !amount || !date) {
                alert("Please fill all fields");
                return;
            }

            let transaction = db.transaction("expenses", "readwrite");
            transaction.onerror = () => alert("Transaction failed!");

            let store = transaction.objectStore("expenses");
            let expense = { title, amount, date };

            let addRequest = store.add(expense);
            addRequest.onsuccess = () => {
                alert("Expense Added Successfully!");
                document.getElementById("title").value = "";
                document.getElementById("amount").value = "";
                document.getElementById("date").value = "";
            };
            addRequest.onerror = () => alert("Error Adding Expense!");
        }

        function viewExpenses() {
            let transaction = db.transaction(["expenses"], "readonly");
            let store = transaction.objectStore("expenses");
            let getRequest = store.getAll();

            getRequest.onsuccess = function() {
                displayExpenses(getRequest.result);
            };
            getRequest.onerror = () => alert("Error Fetching Data!");
        }

        function deleteExpense(id) {
            let transaction = db.transaction(["expenses"], "readwrite");
            let store = transaction.objectStore("expenses");
            let deleteRequest = store.delete(id);

            deleteRequest.onsuccess = () => {
                alert("Expense Deleted!");
                viewExpenses();
            };
```

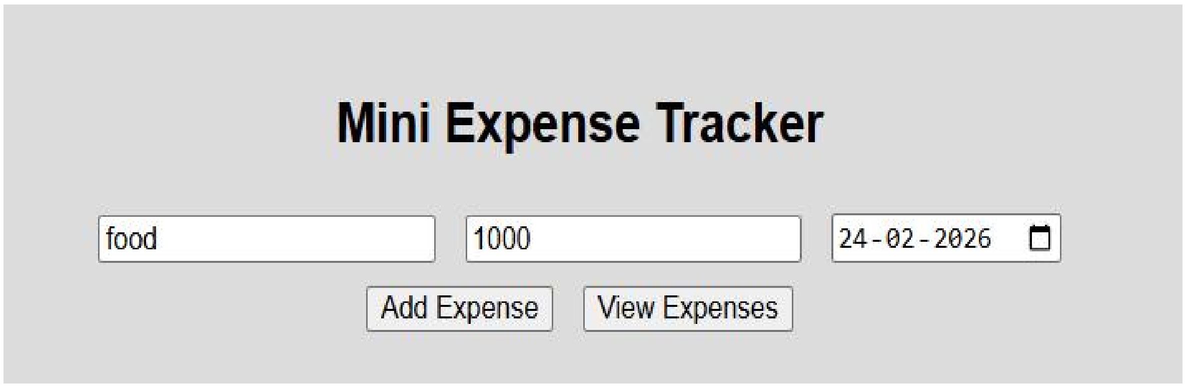


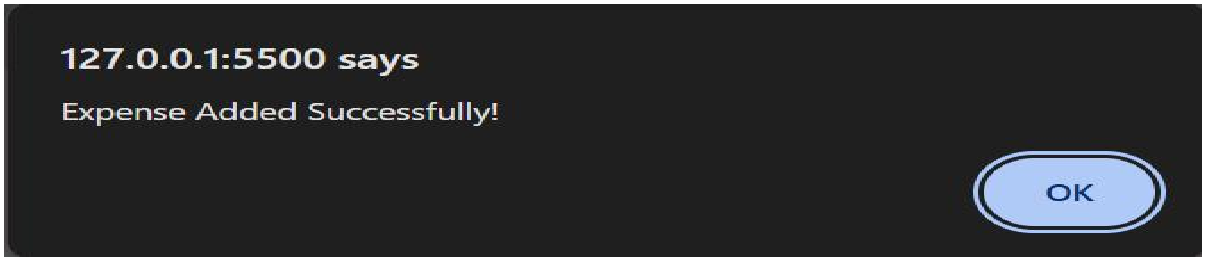
```
            deleteRequest.onerror = () => alert("Delete Failed!");
        }

        function displayExpenses(expenses) {
            let output = "<table>";
            output += "<tr><th>Title</th><th>Amount</th><th>Date</th><th>Action</th></tr>";

            expenses.forEach(exp => {
                output += `<tr>
                    <td>${exp.title}</td>
                    <td>$${exp.amount}</td>
                    <td>${exp.date}</td>
                    <td><button onclick="deleteExpense(${exp.id})">Delete</button></td>
                </tr>`;
            });

            output += "</table>";
            document.getElementById("expenseList").innerHTML = output;
        }
    </script>
</body>
</html>
```

Output:

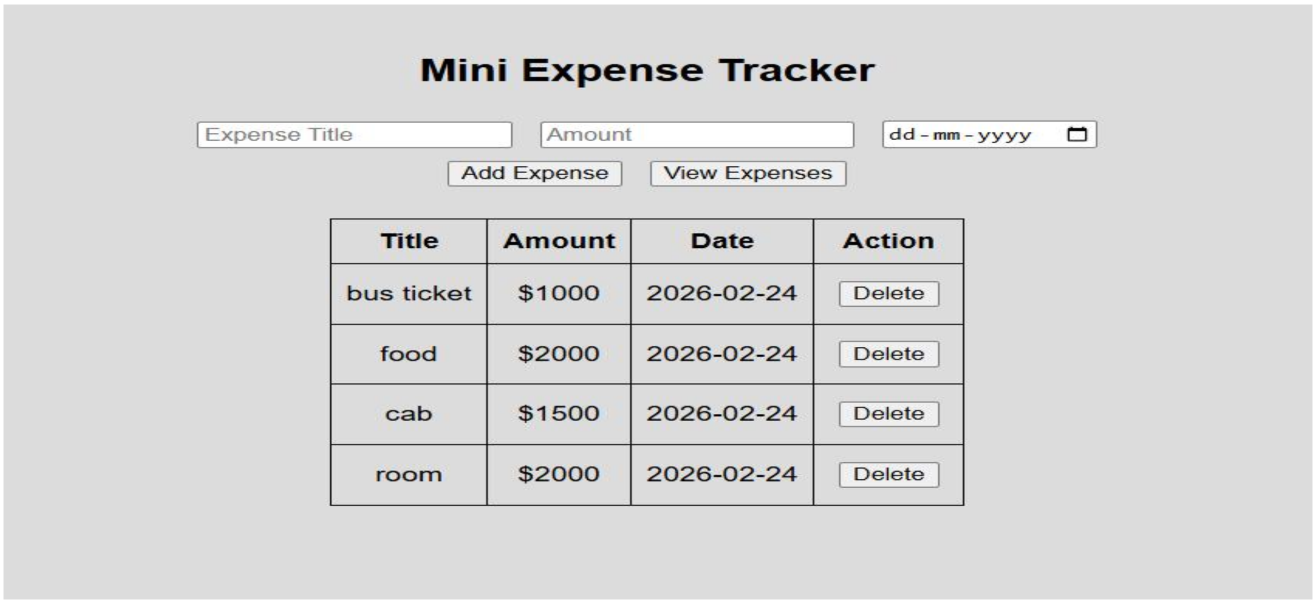
Output File: index.html





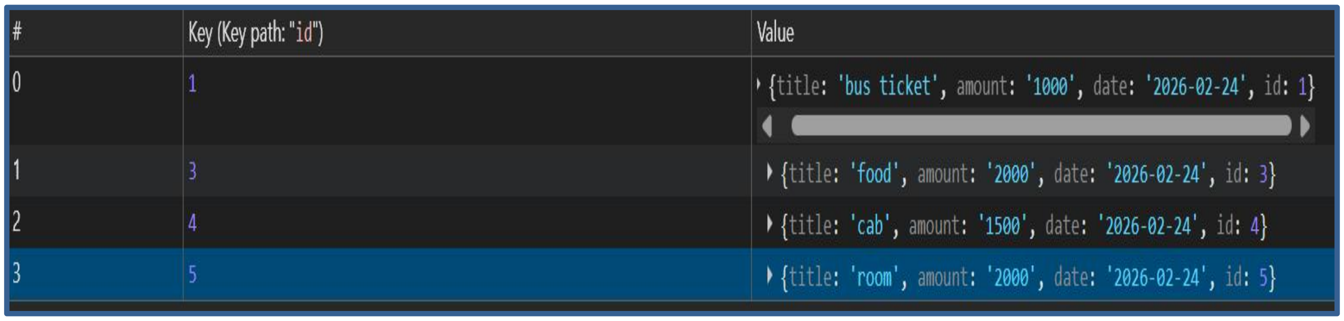

127.0.0.1:5500 says

Expense Added Successfully!



Mini Expense Tracker

Title	Amount	Date	Action
bus ticket	$1000	2026-02-24	Delete
food	$2000	2026-02-24	Delete
cab	$1500	2026-02-24	Delete
room	$2000	2026-02-24	Delete



#	Key (Key path: "id")	Value
0	1	{title: 'bus ticket', amount: '1000', date: '2026-02-24', id: 1}
1	3	{title: 'food', amount: '2000', date: '2026-02-24', id: 3}
2	4	{title: 'cab', amount: '1500', date: '2026-02-24', id: 4}
3	5	{title: 'room', amount: '2000', date: '2026-02-24', id: 5}

Explanation:

This code creates a **Mini Expense Tracker** using IndexedDB in the browser. You can add an expense (title, amount, date), store it in a database, view all expenses in a table, and delete any record. The output shows your entered expenses neatly in a table, and the data is saved in the browser’s IndexedDB, so it stays even after refreshing the page.