Internship Report – Frontend Dev

Week 5: JavaScript Advanced Topics

Name: Zainab
Father Name: Assad Qayyum
Date: 22nd July, 2025
Internship Domain: Front-end Intern
Task: JS - Arrays Methods (map, filter, reduce), JSON & LocalStorage

Task Overview: (Day2)

On Day 2 of Week 5, my focus was on exploring advanced JavaScript concepts — specifically the usage of Array methods (map, filter, and reduce), working with JSON data, and storing information in the browser using LocalStorage.

Content Covered:

- JavaScript Array Methods:
  - map()
  - filter()
  - reduce()
- JSON (JavaScript Object Notation)
- localStorage (Web Storage API)

1. Array Methods:

JavaScript arrays come with **built-in methods** that allow us to **manipulate data efficiently**. These methods don't modify the original array; they return new results.
Think of an array as a **list of items**, and these methods are tools to **process or transform** that list.

Common Types:

a) map(): Transform Each Item

Use it when you want to **change every item** in an array.

Syntax:

```
array.map(function(element, index, array) {
  // return new value for each item
}  );
```

Example:

```
let numbers = [1, 2, 3];
let squared = numbers.map((num) => num * num);

console.log(squared);     // [1, 4, 9]
```

- Returns a new array
- Original array is not modified
- Great for transformations

b) filter(): Keep Specific Items

Use it when you want to **select some items** based on a **condition**.

Syntax:

```
array.filter(function(element, index, array) {
  return condition; // true to keep the item
}  );
```

Example:

```
let scores = [45, 85, 70, 30];
let passed = scores.filter(score => score >= 50);
```

```
console.log(passed); // [85, 70]
```

- Returns a new array
- Only includes elements where the condition is true

c) reduce(): Get One Final Result:

Use it when you want to **accumulate all array values** into a single result; like a sum, average, product, or object.

Syntax:

```
array.reduce(function(accumulator, currentValue, index, array) {
  return newAccumulator;
}, initialValue);
```

Example:

```
let prices = [100, 200, 300];

let total = prices.reduce((sum, price) => sum + price, 0);

console.log(total); // 600
```

- Returns a single value
- Often used for math (sum, product) or combining data

2. JSON (JavaScript Object Notation):

JSON is a lightweight data format used for exchanging data. It looks like JavaScript objects but is stored as text (a string).

It is used:

- To send data to a server
- To store objects in localStorage
- To read/write data from a file or database

Syntax:

```
let user = { name: "Zainab", age: 21 };
let jsonData = JSON.stringify(user);          // Convert object to JSON string
let parsedData = JSON.parse(jsonData);    // Convert JSON string back to object
```

- Use **JSON.stringify()** to convert **object/array → string**
- Use **JSON.parse()** to convert **string → object/array**

3. LocalStorage:

A place in the browser to **store small pieces of data**, like user info or app settings.

- Built-in browser storage
- Stores key-value pairs in the browser
- Data persists even after refreshing or closing the browser tab

When to Use LocalStorage?

Save user preferences (theme: light/dark)

Store form data temporarily

Save login status or tokens

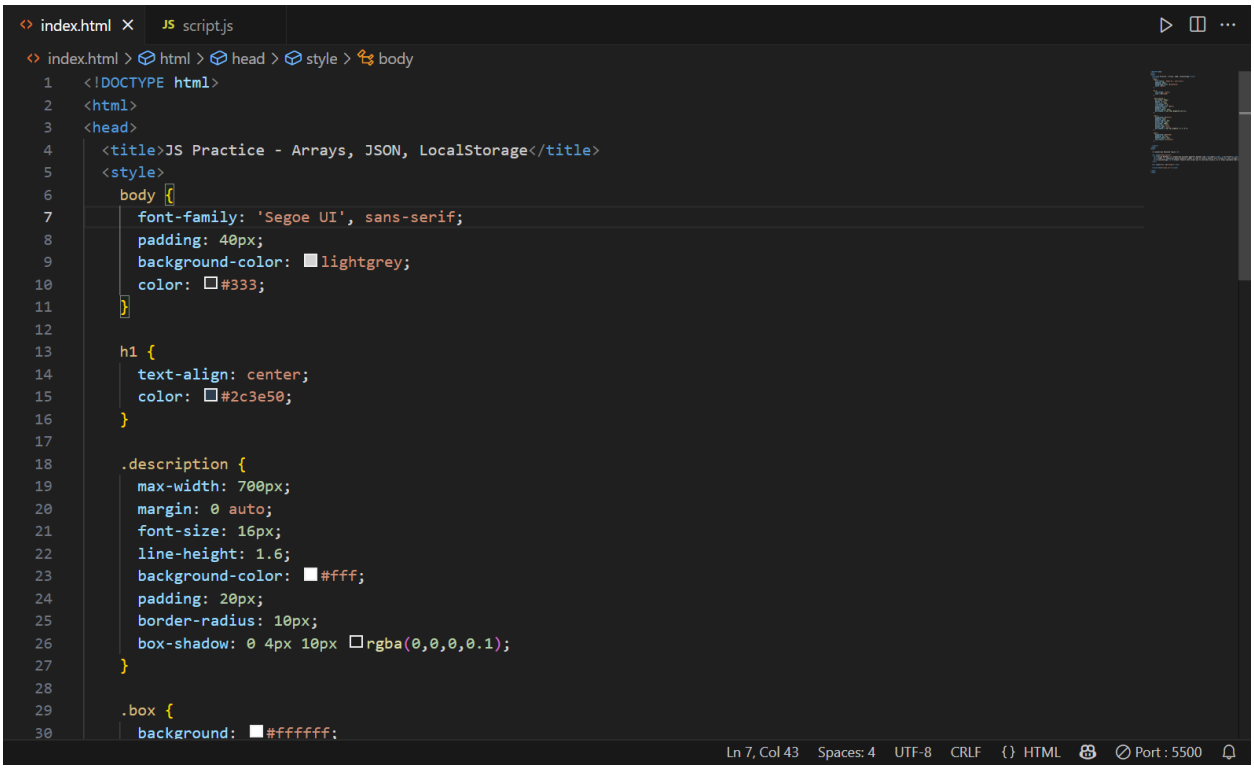
Avoid hitting the server repeatedly for the same data

Example:

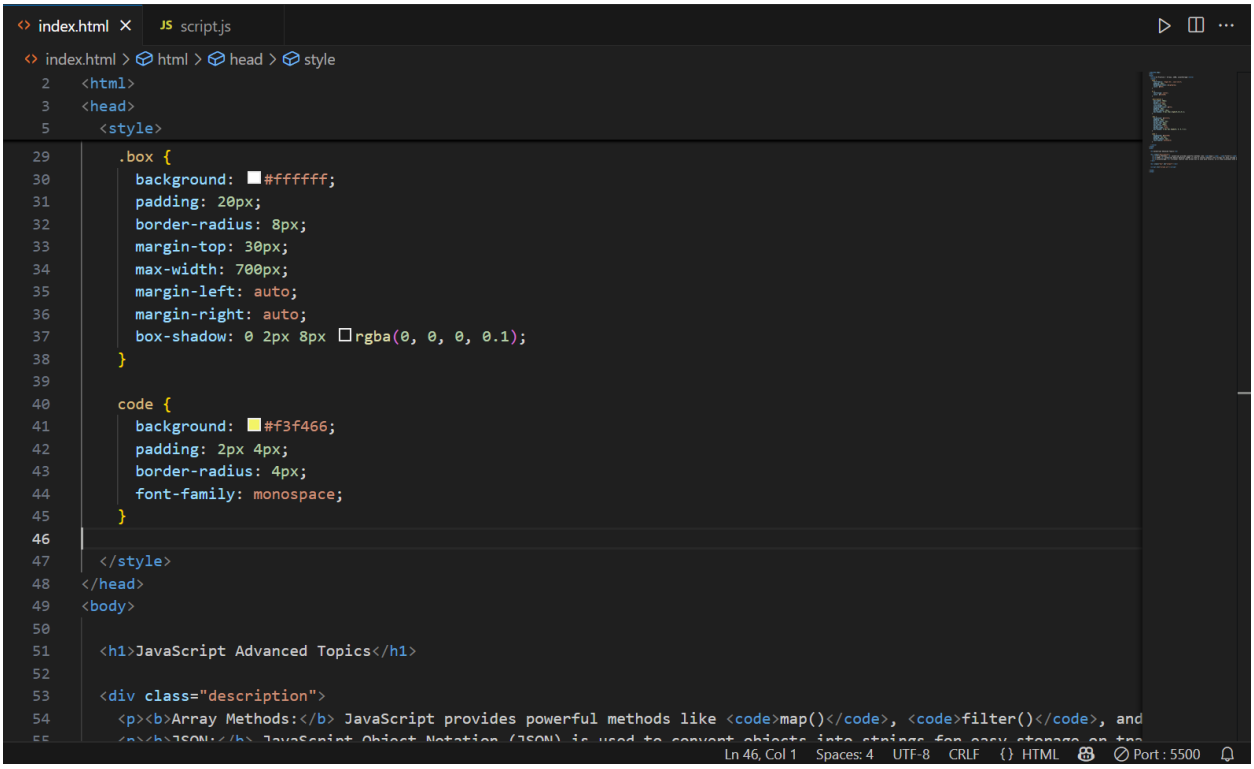
```
// Save data
localStorage.setItem("name", "Zainab");

// Retrieve data
let name = localStorage.getItem("name");
console.log(name); // Zainab

// Remove data
localStorage.removeItem("name");

// Clear everything
localStorage.clear();
```

Practice Code:

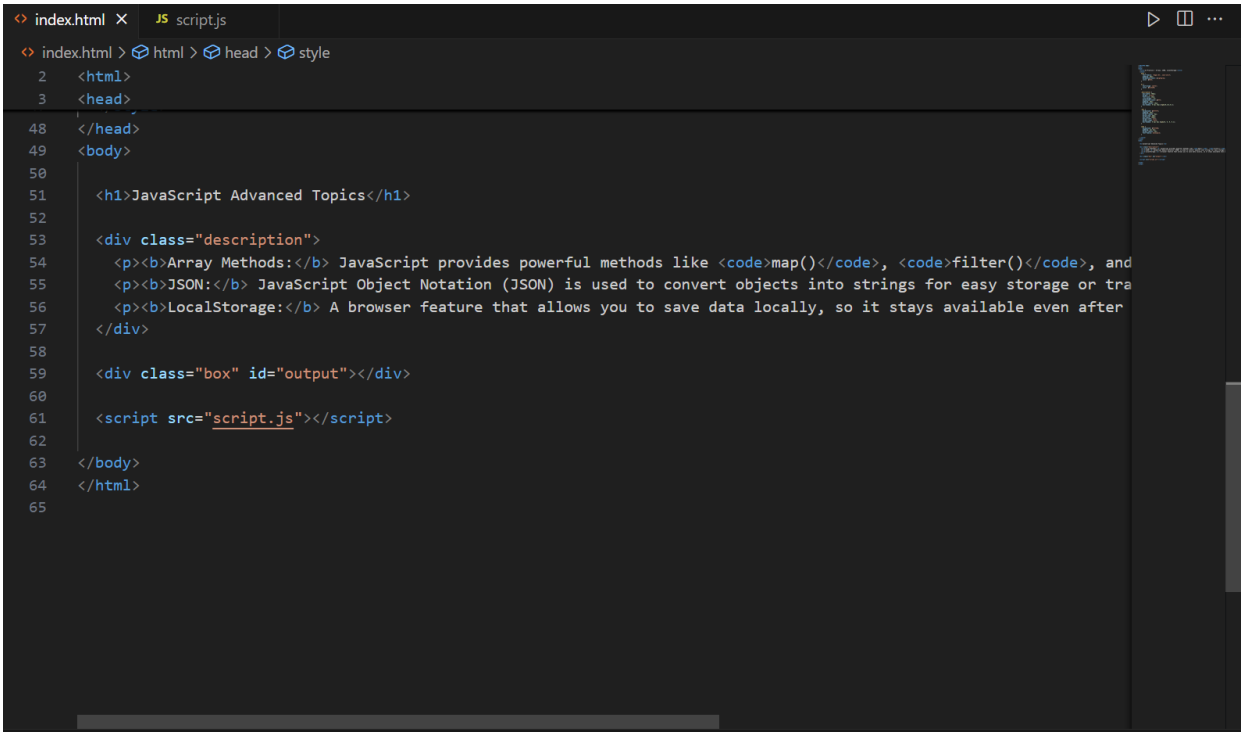
Html and CSS:



```
index.html  script.js
index.html > html > head > style > body
1  <!DOCTYPE html>
2  <html>
3  <head>
4    <title>JS Practice - Arrays, JSON, LocalStorage</title>
5    <style>
6      body {
7        font-family: 'Segoe UI', sans-serif;
8        padding: 40px;
9        background-color: lightgrey;
10       color: #333;
11     }
12
13     h1 {
14       text-align: center;
15       color: #2c3e50;
16     }
17
18     .description {
19       max-width: 700px;
20       margin: 0 auto;
21       font-size: 16px;
22       line-height: 1.6;
23       background-color: #fff;
24       padding: 20px;
25       border-radius: 10px;
26       box-shadow: 0 4px 10px rgba(0,0,0,0.1);
27     }
28
29     .box {
30       background: #ffffff;
Ln 7, Col 43  Spaces: 4  UTF-8  CRLF  {} HTML  Port : 5500
```



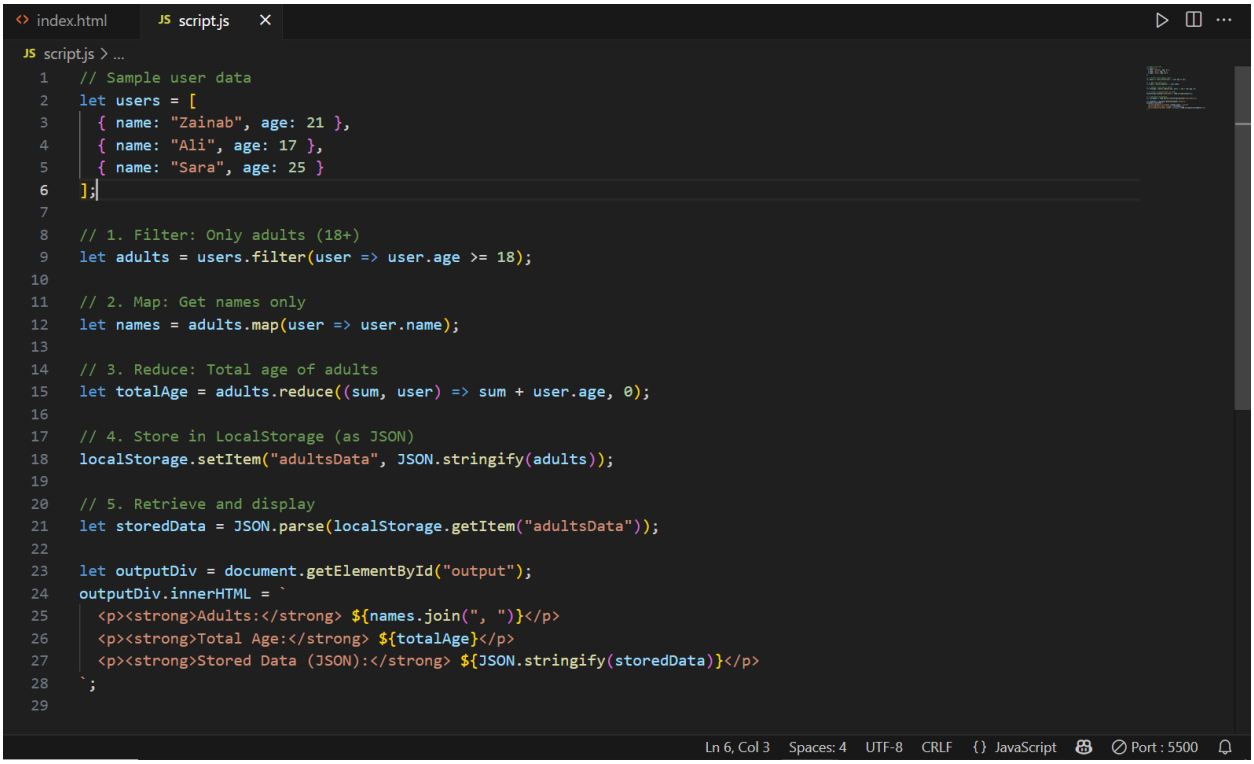
```
index.html  script.js
index.html > html > head > style
2  <html>
3  <head>
5    <style>
29     .box {
30       background: #ffffff;
31       padding: 20px;
32       border-radius: 8px;
33       margin-top: 30px;
34       max-width: 700px;
35       margin-left: auto;
36       margin-right: auto;
37       box-shadow: 0 2px 8px rgba(0, 0, 0, 0.1);
38     }
39
40     code {
41       background: #f3f466;
42       padding: 2px 4px;
43       border-radius: 4px;
44       font-family: monospace;
45     }
46
47   </style>
48 </head>
49 <body>
50
51   <h1>JavaScript Advanced Topics</h1>
52
53   <div class="description">
54     <p><b>Array Methods:</b> JavaScript provides powerful methods like <code>map()</code>, <code>filter()</code>, and
Ln 46, Col 1  Spaces: 4  UTF-8  CRLF  {} HTML  Port : 5500
```

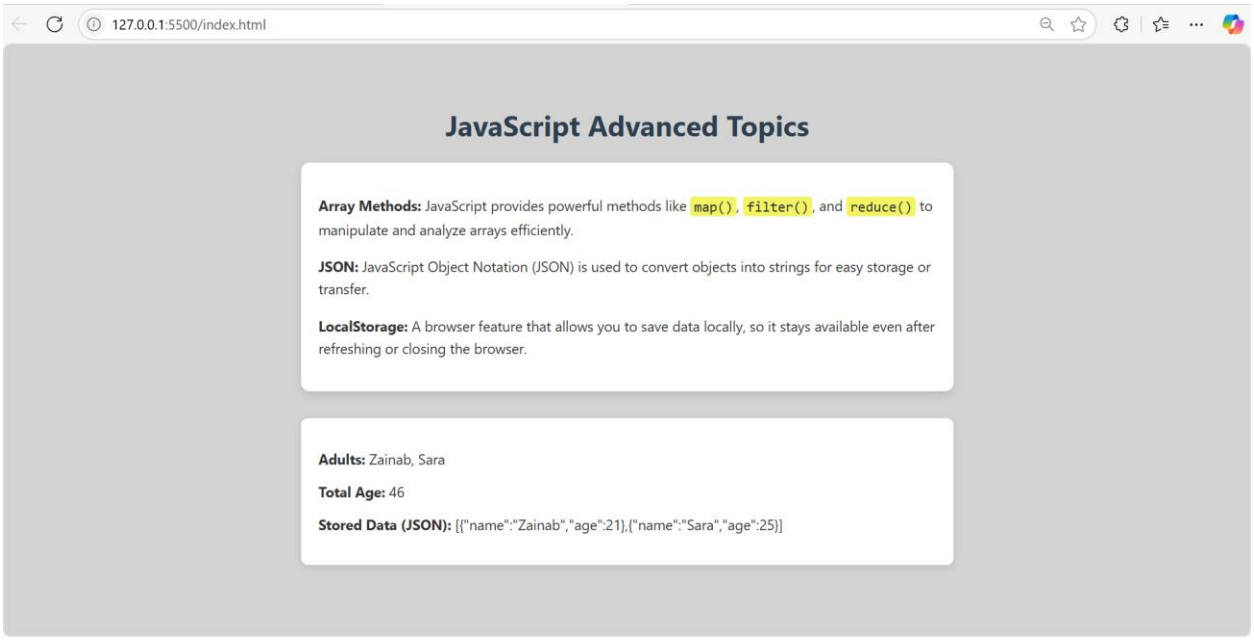

index.html
script.js
index.html > html > head > style
<html>
<head>
</head>
<body>
<h1>JavaScript Advanced Topics</h1>

Array Methods: JavaScript provides powerful methods like <code>map()</code>, <code>filter()</code>, and
JSON: JavaScript Object Notation (JSON) is used to convert objects into strings for easy storage or tra
LocalStorage: A browser feature that allows you to save data locally, so it stays available even after

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

JavaScript:


index.html
script.js
script.js > ...
// Sample user data
let users = [
{ name: "Zainab", age: 21 },
{ name: "Ali", age: 17 },
{ name: "Sara", age: 25 }
];
// 1. Filter: Only adults (18+)
let adults = users.filter(user => user.age >= 18);
// 2. Map: Get names only
let names = adults.map(user => user.name);
// 3. Reduce: Total age of adults
let totalAge = adults.reduce((sum, user) => sum + user.age, 0);
// 4. Store in LocalStorage (as JSON)
localStorage.setItem("adultsData", JSON.stringify(adults));
// 5. Retrieve and display
let storedData = JSON.parse(localStorage.getItem("adultsData"));
let outputDiv = document.getElementById("output");
outputDiv.innerHTML = `
Adults: ${names.join(", ")}
Total Age: ${totalAge}
Stored Data (JSON): ${JSON.stringify(storedData)}
`;
Ln 6, Col 3
Spaces: 4
UTF-8
CRLF
{} JavaScript
Port : 5500



Conclusion:

Today's task deepened my understanding of how data is handled in JavaScript through array methods like map, filter, and reduce. I also learned how to format and store data using JSON and how to persist it using LocalStorage.