Final Assignment

Summer 2024

Course Title: Web Technology

Course Code: CSE480

Submitted by:

Name: Shakil Ahmed Shawon

ID: 201014081

Department of CSE

University of Liberal Arts Bangladesh (ULAB)

Ans. To The Question No. 01

A. The main differences between '$_GET' and '$_POST' are:
   1. '$_GET' is used to collect data sent through the URL parameters, while '$_POST' is used to collect data sent through HTTP POST method.
   2. '$_GET' has a limitation on the amount of data that can be sent (usually 2048 characters), while '$_POST' has a higher limit.
   3. '$_GET' requests are visible in the URL, making them less secure for sensitive data. while '$_POST' requests are not visible in the URL, making them more secure.

   In case of a form submit, using '$_POST' is more secure because:

   1. '$_POST' requests are not visible in the URL, so sensitive data like passwords is not exposed.
   2. '$_POST' requests are more reliable for sending large amounts of data.

B. Most developers prefer using JSON over XML for data transmission due to the following reasons:
   1. JSON is lighter and more compact compared to XML, resulting in faster transmission and parsing
   2. JSON uses a simple syntax with key-value pairs and arrays, making it easier to read and write compared to XML.
   3. JSON is natively supported by JavaScript, the primary language for web development, allowing for easy parsing and manipulation of data.
   4. JSON is less verbose than XML, requiring fewer characters to represent the same data.

C. A loosely typed language, also known as a dynamically typed language, is a programming language that does not require explicit declaration of variable types. Some example of loosely typed language include:-
   1. JavaScript: Variables ca hold values of any data type without explicit declaration.
   2. Python: Variables are dynamically typed, and their type is determined at runtime.
   3. PHP: Variables can be used to store values of different data types without explicit declaration.

D. Synchronous calls are blocking, meaning the program execution is halted until the called ffunction returns a result.
Asynchronous calls are non-blocking, allowing the program to continue executing other tasks while waiting for the called function to return.

Scenarios where synchronous calls fit better:

1. Simple operations that complete as the blocking behavior is not noticeable.
2. When the order of execution is crucial, and the program must wait for a result before proceeding.
   Example: User authentication, where the application needs to verify for credentials before allowing access.

Scenarios where asynchronous calls fir better:

1. Long-running operations like network requests, file I/O, or database queries, to avoid blocking the main thread.
2. When the order of execution is not critical, and the program can continue with other tasks while waiting for the result.
3. In event-driven architectures and user interfaces to provide a responsive experience.
   Example: Loading data from a server without blocking the UI.

Ans. To The Question No. 02

Screenshots of code:


CSE480
Projects
tower_of_Hanoi.html
OUTLINE
Projects > tower_of_Hanoi.html > html > body > script > audio
1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5 <meta charset="UTF-8">
6 <meta http-equiv="X-UA-Compatible" content="IE=edge">
7 <meta name="viewport" content="width=device-width, initial-scale=1.0">
8 <title>Tower of Hanoi</title>
9 <style>
10 .div1,
11 .div2,
12 .div3 {
13 height: 50vh;
14 width: 32.9%;
15
16 float: left;
17 border: 2px solid orange;
18 display: flex;
19 flex-direction: column-reverse;
20 }
21
22 .div4 {
23
24 clear: left;
25
26 }
27
28 .btn-primary {
29 background-color: blue;
30 width: 100%;
31 height: 3rem;
32 border-radius: 16px;
33 opacity: 0.6;
34
35 display: inline-block;
36 text-decoration: none;
37
38 }

```
40
41
42      .btn-default {
43
44          width: 33%;
45          height: 2rem;
46          background-color: teal;
47          margin-top: 30px;
48          border-radius: 80px;
49
50          opacity: 0.6;
51
52          display: inline-block;
53          text-decoration: none;
54      }
55
56      .flex-parent {
57          display: flex;
58          margin-top:30px ;
59       }
60
61       .jc-center {
62          justify-content: center;
63       }
64
65       button.margin-right {
66          margin-right: 70px;
67          background-color: cornflowerblue;
68          height: 4rem;
69          width: 8rem;
70          opacity: 0.6;
71
72          display: inline-block;
73          text-decoration: none;
74          border-radius: 80px;
```

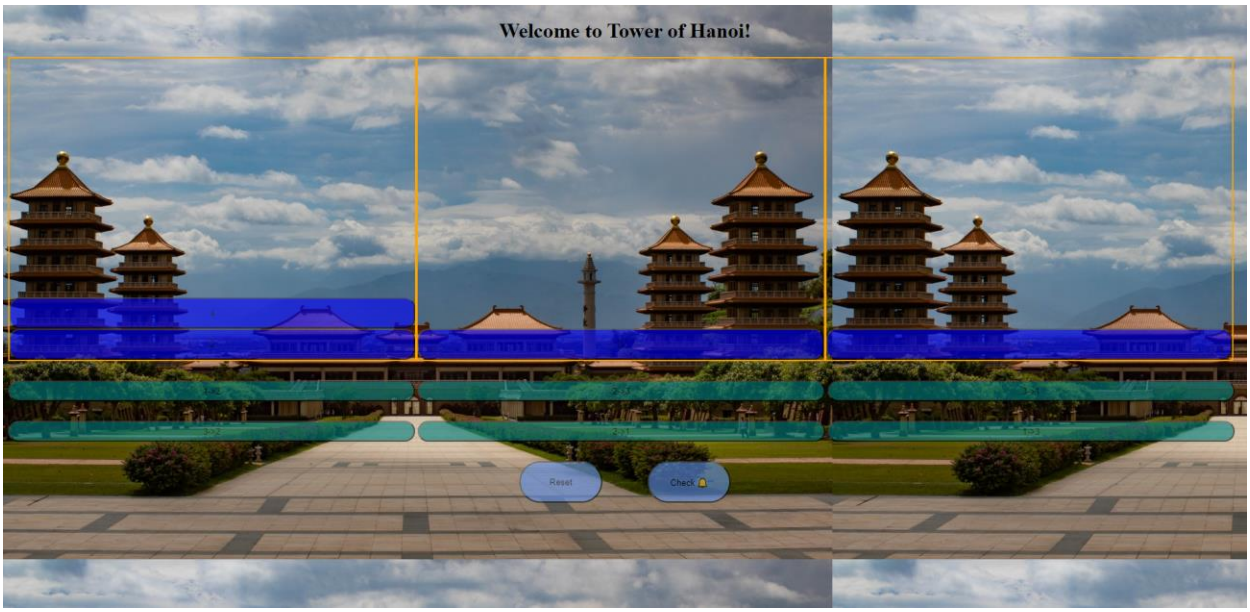
```
            button.button2{
              background-color: cornflowerblue;
              height: 4rem;
              width: 8rem;
              opacity: 0.6;
              transition: 0.3s;
              display: inline-block;
              text-decoration: none;
              border-radius: 80px;
            }

            body {
              background-image: url("https://cdn-media-2.freecodecamp.org/w1280/
              5f9ca6e9740569d1a4ca739a.jpg");

            }


    </style>
</head>

<!-- <body style="background-color: aquamarine;"> -->
    <body>

        <h1 style="text-align: center;">Welcome to Tower of Hanoi! </h1>
        <div class="div1" id="d1">

            <button type="button"  class="btn btn-large btn-block btn-primary"
            id="0"onclick="">2</button>
            <button type="button" class="btn btn-large btn-block btn-primary"
            id="1"onclick="">4</button>
            <button type="button" class="btn btn-large btn-block btn-primary"
            id="2"onclick="">3</button>
            <button type="button" class="btn btn-large btn-block btn-primary"
            id="3"onclick="">1</button>


        </div>
        <div class="div2" id="d2"></div>
        <div class="div3" id="d3"></div>
        <div class="div4"  >

            <button type="button" class="btn btn-large btn-block btn-default" onclick="oneto2
            ()">1→2</button>
            <button type="button" class="btn btn-large btn-block btn-default" onclick="twoto3
            ()">2→3</button>
            <button type="button" class="btn btn-large btn-block btn-default"
            onclick="threeto1()">3→1</button>
            <button type="button" class="btn btn-large btn-block btn-default"
            onclick="threeto2()">3→2</button>
            <button type="button" class="btn btn-large btn-block btn-default" onclick="twoto1
            ()">2→1</button>
            <button type="button" class="btn btn-large btn-block btn-default" onclick="oneto3
            ()">1→3</button>
            </div>

        <div class="flex-parent jc-center">

```

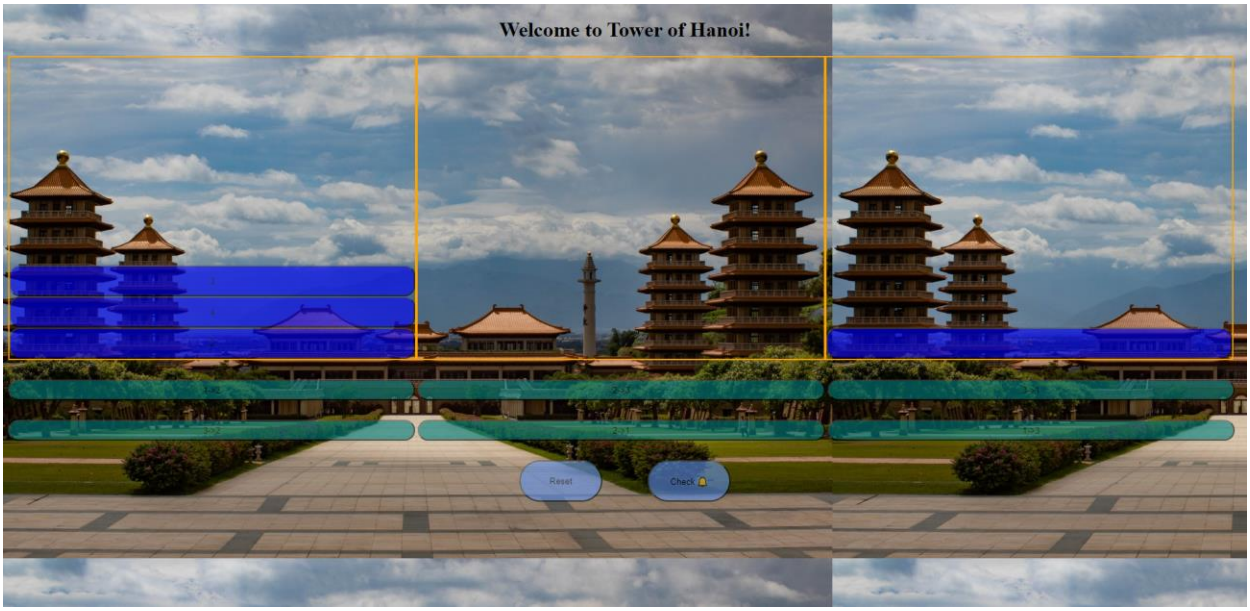
```
123
124             <button type="button" class="green margin-right" id="reset"  onclick="reset()
                ">Reset</button>
125             <button type="button" class="button2" id="check" onclick="check()">Check 🔔</
                button>
126         </div>
127
128     <script>
129
130         const audio = new Audio("https://www.fesliyanstudios.com/play-mp3/387");
131         const buttons = document.querySelectorAll("button");
132
133         buttons.forEach(button => {
134             button.addEventListener("click", () => {
135               audio.play();
136             });
137           });
138
139
140     // Select all the tower divs
141     const tower1 = document.getElementById("d1");
142     const tower2 = document.getElementById("d2");
143     const tower3 = document.getElementById("d3");
144
145     let selectedDisk = null;
146
147     // Helper function to move the disk from one tower to another
148     function moveDisk(fromTower, toTower) {
149         // Get the topmost disk in each tower
150         const fromDisk = fromTower.querySelector("button:last-child");
151         const toDisk = toTower.querySelector("button:last-child");
152
153         if (!fromDisk) {
154             alert("No disk to move!");
155             return;
156         }
```

```
        if (!toDisk || fromDisk.innerText < toDisk.innerText) {
            toTower.appendChild(fromDisk);
        } else {
            alert("Invalid move! A larger disk cannot be placed on a smaller disk.");
        }
    }

    // Functions for buttons to move disks between towers
    function oneto2() {
        moveDisk(tower1, tower2);
    }

    function oneto3() {
        moveDisk(tower1, tower3);
    }

    function twoto1() {
        moveDisk(tower2, tower1);
    }

    function twoto3() {
        moveDisk(tower2, tower3);
    }

    function threeto1() {
        moveDisk(tower3, tower1);
    }

    function threeto2() {
        moveDisk(tower3, tower2);
    }

    // Reset function to restart the game
    function reset() {
```

```
        // Move all disks back to tower 1
        while (tower2.firstChild) {
            tower1.appendChild(tower2.firstChild);
        }
        while (tower3.firstChild) {
            tower1.appendChild(tower3.firstChild);
        }
    }

    // Check if the game is won
    function check() {
        if (tower3.children.length === 4) {
            alert("Congratulations! You have won the Tower of Hanoi!");
        } else {
            alert("Keep trying!");
        }
    }
</script>

</body>
</html>
```

Implementation Screenshots:





Ans. To The Question No. 03

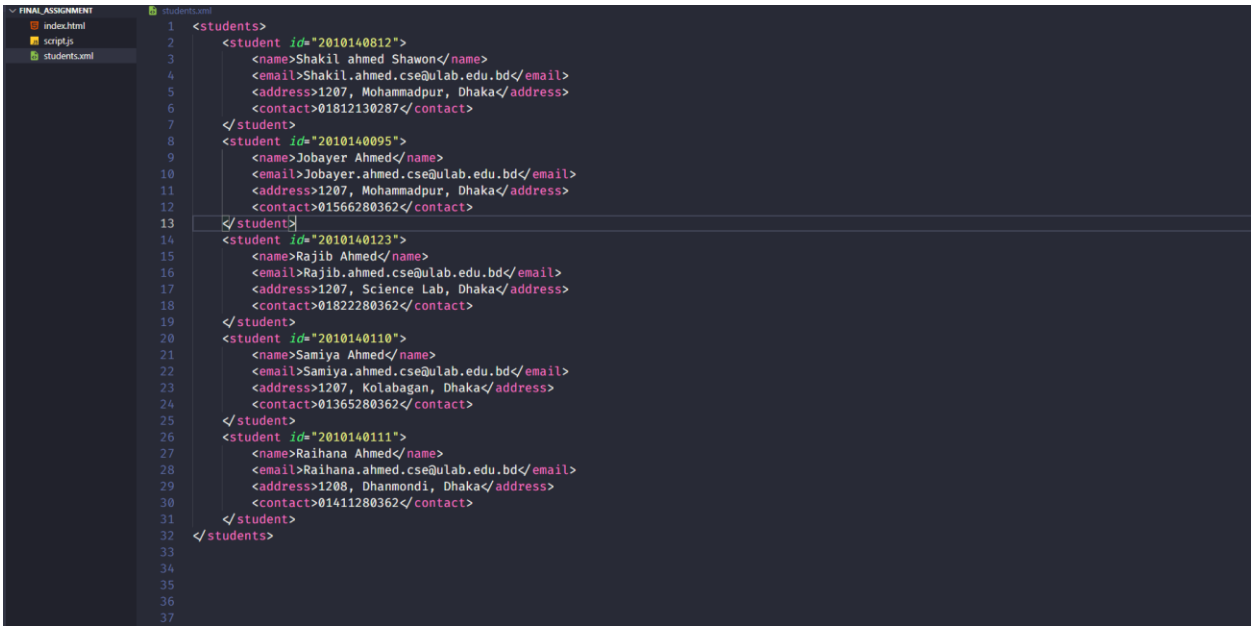
Index.html code:

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Student Information</title>
    <script src="script.js" defer></script>
</head>
<body>
    <header>
        <link rel="stylesheet" href="https://matcha.mizu.sh/matcha.css">

    </header>
    <h1>Student Information</h1>
    <input type="text" id="studentId" placeholder="Enter 10-digit Student ID">
    <button onclick="fetchStudentInfo()">Get Student Info</button>

    <table id="studentTable" border="1">
    <tr>
        <th>Name</th>
        <th>Email</th>
        <th>Address</th>
        <th>Contact Number</th>
    </tr>
    <tr id="studentData">
        <td colspan="4">No Data Available</td>
    </tr>
</table>

</body>
</html>
```

Script.js code:

```javascript
function fetchStudentInfo() {
    const studentId = document.getElementById('studentId').value;

    if (studentId.length !== 10) {
        alert('Please enter a 10-digit Student ID.');
        return;
    }

    const xhr = new XMLHttpRequest();
    xhr.open('GET', 'students.xml', true);
    xhr.onload = function () {
        if (xhr.status === 200) {
            const xml = xhr.responseXML;
            const student = xml.querySelector(`student[id="${studentId}"]`);

            if (student) {
                const name = student.querySelector('name').textContent;
                const email = student.querySelector('email').textContent;
                const address = student.querySelector('address').textContent;
                const contact = student.querySelector('contact').textContent; // Fetch the contact number

                document.getElementById('studentData').innerHTML = `
                    <td>${name}</td>
                    <td>${email}</td>
                    <td>${address}</td>
                    <td>${contact}</td>
                `;
            } else {
                document.getElementById('studentData').innerHTML = '<td colspan="4">Student Not Found</td>';
            }
        }
    };
    xhr.send();
}
```

Students.xml:



Outputs:

Student Information

2010140123

Get Student Info

Name	Email	Address	Contact Number
Rajib Ahmed	Rajib.ahmed.cse@ulab.edu.bd	1207, Science Lab, Dhaka	01822280362

Student Information

2010140110

Get Student Info

Name	Email	Address	Contact Number
Samiya Ahmed	Samiya.ahmed.cse@ulab.edu.bd	1207, Kolabagan, Dhaka	01365280362

Student Information

2010140812

Get Student Info

Name	Email	Address	Contact Number
Shakil ahmed Shawon	Shakil.ahmed.cse@ulab.edu.bd	1207, Mohammadpur, Dhaka	01812130287