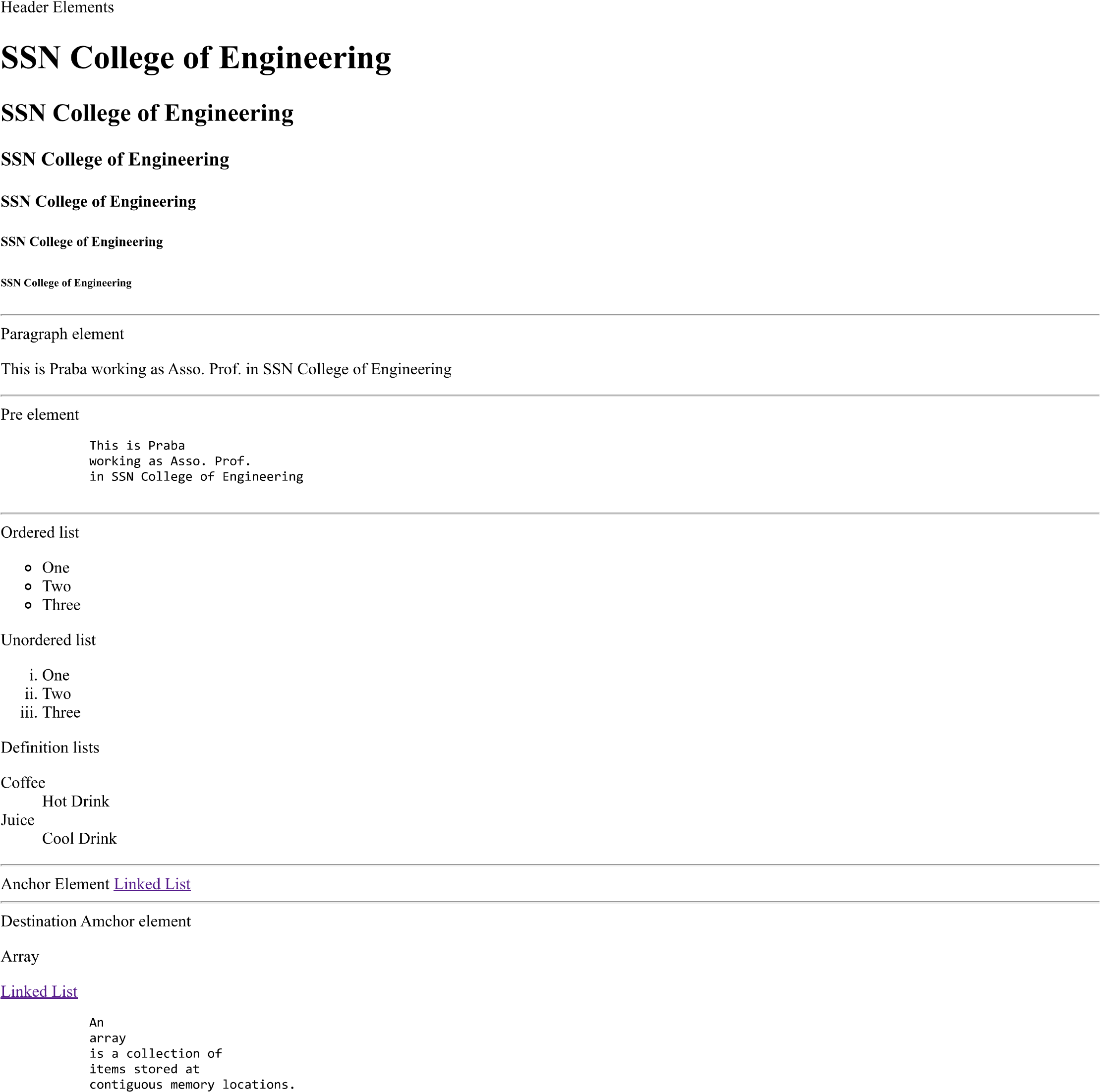
**UCS 2611 Internet Programming Lab**

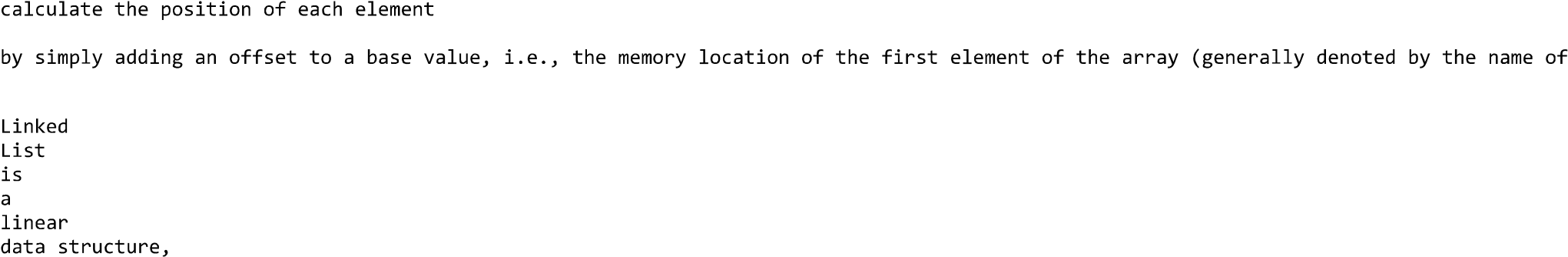
**Exercise 1: Implementation of Conference Website using HTML**

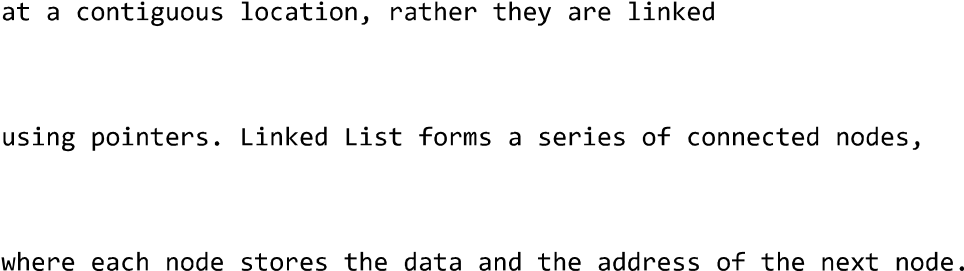
**Date of Exercise: 30.01.2024**

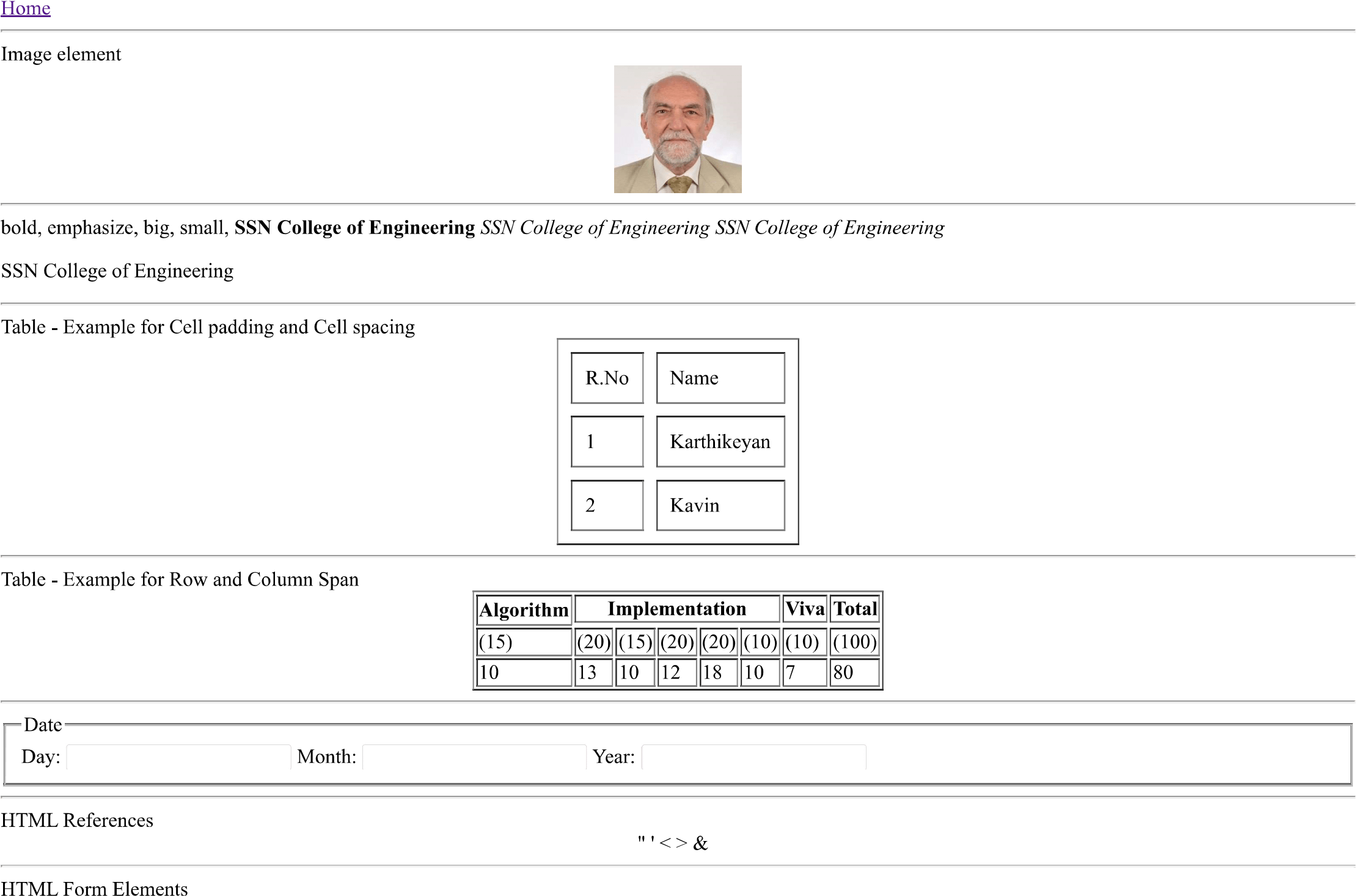
1. Create the following sample web page using necessary HTML elements [CO1, K3]

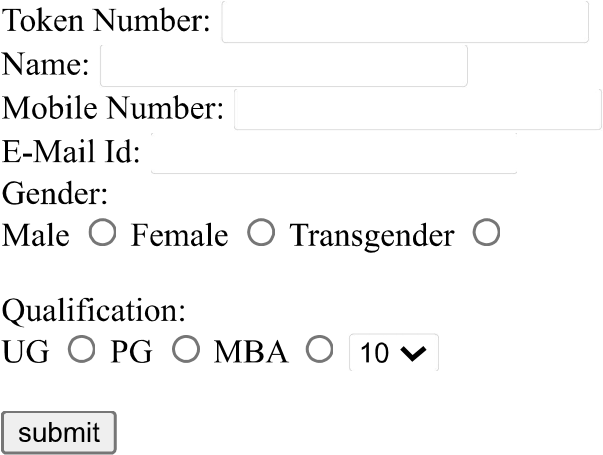
**Sample Webpage**





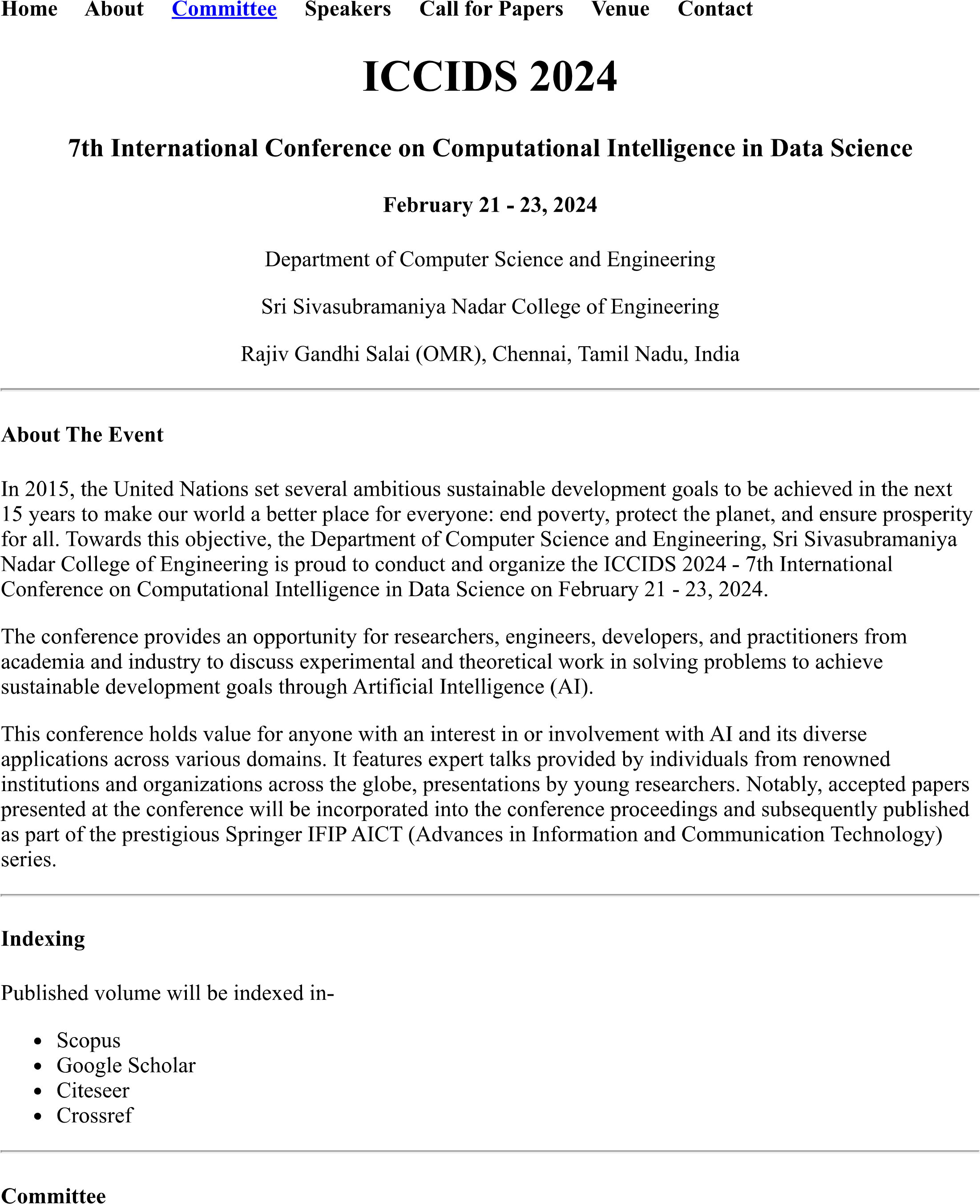


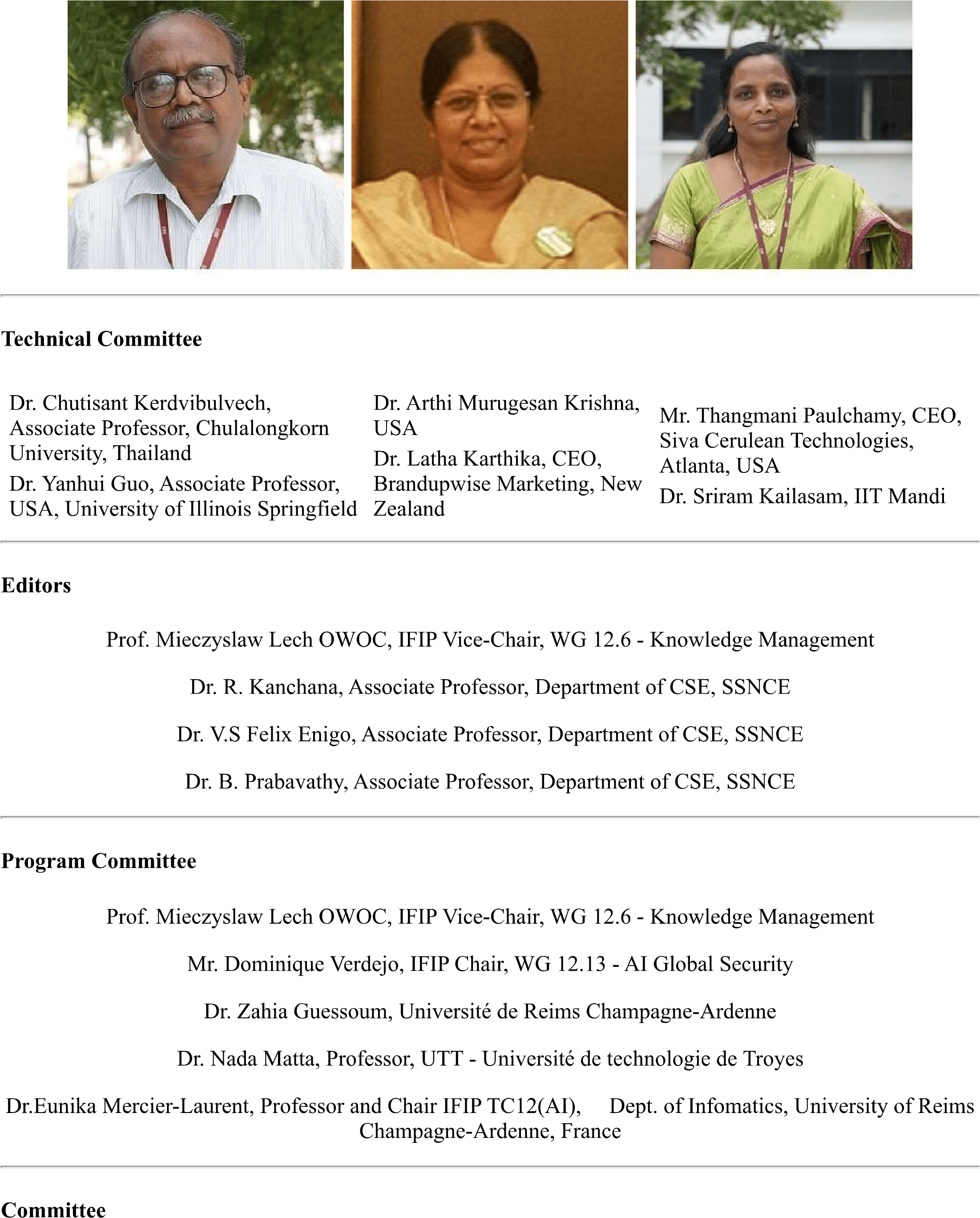




Create the following conference web site using necessary HTML elements

**Conference Website – ICCIDS**





Best Practices to be followed:

1. Design before coding
2. Incremental coding
3. Usage of proper naming convention
4. Usage of Comments to the code
5. Indentation of code

**Implementation of Conference Website using HTML**

**Design:-**

**Code:-**

1. Create the following sample web page using necessary HTML elements

<html>

    <head>

        <title>Implementation of website using HTML</title>

        <p>Header element</p>

        <h1> SSN College of Engineering</h1>

        <h2> SSN College of Engineering</h2>

        <h3> SSN College of Engineering</h3>

        <h4> SSN College of Engineering</h4>

        <h5> SSN College of Engineering</h5>

        <h6> SSN College of Engineering</h6>

    </head>

    <hr>

    <body>

        <p> Paragraph element</p>

        <p> This is Ashwin Ravi, prefinal year CSE undergrad in  SSN College of Engineering</p>

    </body>

    <hr>

    <p>Pre element</p>

    <pre>

        This is Ashwin Ravi,

        prefinal year CSE undergrad

        in  SSN College of Engineering

    </pre>

    <hr>

    <p> Unordered list</p>

    <ul type="circle">

        <li>One</li>

        <li>Two</li>

        <li>Three</li>

    </ul>

    <hr>

    <p>Ordered list</p>

    <ol type="I">

        <li>One</li>

        <li>Two</li>

        <li>Three</li>

    </ol>

    <hr>

    <p>Definitions</p>

    <dl>

        <dt>Coffee</dt>

        <dd>It is a hot drink</dd>

        <dt>Juice</dt>

        <dd>It is a cold drink</dd>

    </dl>

    <hr>

    <p>Anchor element<a href="https://www.programiz.com/dsa/linked-list"> Linked List Programiz </a></p>

    <hr>

    <p>Destination Anchor Element</p>

    <p>Array</p>

    <a href="#Array">Array</a>

    <pre id = "Array">

               An

               Array

               is a data structure

               consisting of

               a collection of elements (values or variables),

               of same memory size,

               each identified by

               at least one array index or key.

               An array is stored such that the

               position of each element can be computed from its

               index tuple by a mathematical formula">

    </pre>

    <hr>

    <a href="https://www.google.co.uk/">Home</a>

    <hr>

    <p>Image element</p>

    <center>

    <image src="Iron man with USA flag.jpg" alt="Iron Man" height="500px" width="540px"></image>

    </center>

    <hr>

    <emp>bold, emphasize,big, small,</emp>

    <b>SSN College of Engineering</b>

    <i>SSN College of Engineering SSN College of Engineering</i>

    <br>

    <emp>SSN College of Engineering</emp>

    <hr>

    <p>Table-Example for Cell padding and cell spacing</p>

    <table border="1" cellspacing="10" cellpadding="10">

        <tr><th>Roll No.</th><th>Name</th></tr>

        <tr><td>1.</td><td>Ashwin</td></tr>

        <tr><td>2.</td><td>Arjun</td></tr>

    </table>

    <hr>

    <p>Table-Example for row and column spanning</p>

    <table border="2">

        <tr>

            <th>Algorithm</th>

            <th colspan="5">Implementation</th>

            <th>Viva</th>

            <th>Total</th>

        </tr>

        <tr>

            <td>(15)</td>

            <td>(20)</td>

            <td>(15)</td>

            <td>(20)</td>

            <td>(20)</td>

            <td>(10)</td>

            <td>(10)</td>

            <td>(100)</td>

        </tr>

        <tr>

            <td>10</td>

            <td>15</td>

            <td>10</td>

            <td>20</td>

            <td>20</td>

            <td>10</td>

            <td>10</td>

            <td>95</td>

        </tr>

        </table>

    <hr>

    <fieldset>

        <legend>Date</legend>

        <label>Day:<input type="text"></label>

        <label>Month:<input type="text"></label>

        <label>Year:<input type="text"></label>

    </fieldset>

    <hr>

    <p>HTML References</p>

    <center>

        " ' < > &

    </center>

    <hr>

    <p>HTML Form Elements</p>

    <label>Token Number:<input type="text"></label><br><br>

    <label>Name:<input type="text"></label><br><br>

    <label>Mobile Number:<input type="text"></label><br><br>

    <label>E-Mail Id:<input type="text"></label><br><br>

    <label>Gender:<br><input type="radio">Male<input type="radio">Female<input type="radio">Transgender<input type="radio">Others</label><br><br>

    <label>Qualification:<br><input type="radio">UG<input type="radio">PG<input type="radio">MBA<input type="radio">Others<br><br>

    <label for="myDropDown">

        <select id="myDropDown">

        <option value ="op1">1</option>

        <option value ="op2">2</option>

        <option value ="op3">3</option>

        <option value ="op4">4</option>

        <option value ="op5">5</option>

        <option value ="op6">6</option>

        <option value ="op7">7</option>

        <option value ="op8">8</option>

        <option value ="op9">9</option>

        <option value ="op10">10</option>

        </select>

    </label>

    <br><br>

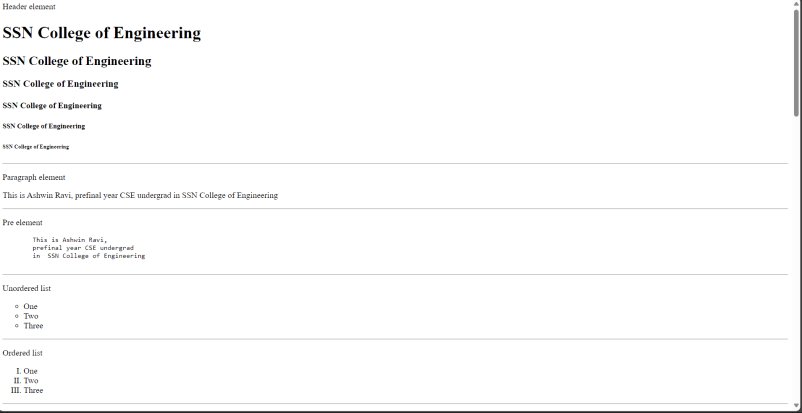
    <label>

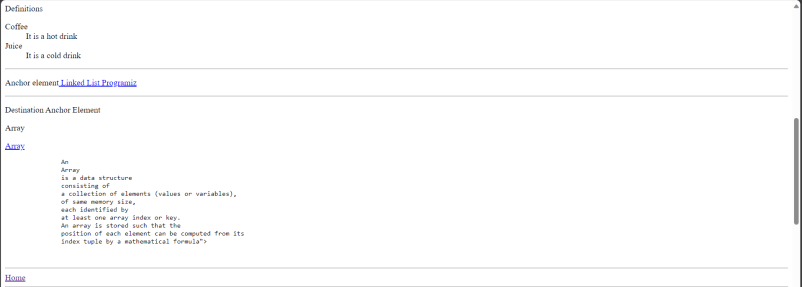
        <input type="submit"></label><br><br>

    </label>

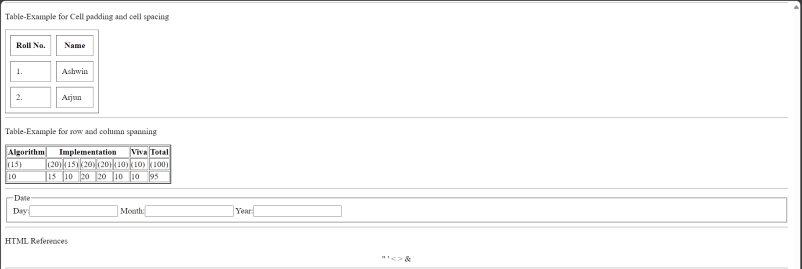
    <hr>

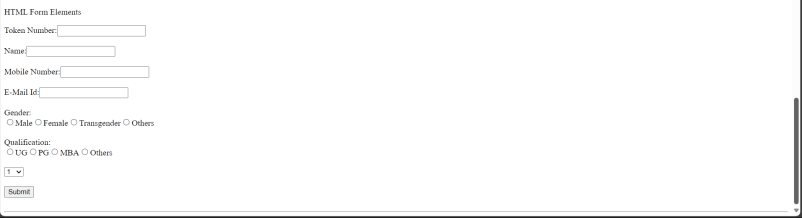
**Output:-**

****

****

****

****

****

2. Create the following conference web site using necessary HTML elements

<html>

    <head>

        <title> Conference website</title>

        <a href="#Home" style="font-weight: bold; color: black;">Home</a>

        <a href="#about" style="font-weight: bold; color: black;">About</a>

        <a href="#committee" style="font-weight: bold; color: black;">Committee</a>

        <a href="#speakers" style="font-weight: bold; color: black;">Speakers</a>

        <a href="#call-for-papers" style="font-weight: bold; color: black;">Call for Papers</a>

        <a href="#venue" style="font-weight: bold; color: black;">Venue</a>

        <a href="#contact" style="font-weight: bold; color: black;">Contact</a>

        <center>

            <h1>ICCIDS 2024</h1>

            <h2>7th International Conference on Computational Intelligence in Data Science</h2>

            <h2>February 21 - 23,2024</h2>

            <p> Department of Computer Science and Engineering</p>

            <p> Sri Sivasubramaniya Nadar College of Engineering</p>

            <p> Rajiv Gandhi Salai (OMR), Kalavakkam, Tamil Nadu-603110</p>

        </center>

        <hr>

        <h3>About the event</h3>

        <p>

            In 2015, the United Nations set several ambitious sustainable development goals to be achieved in the next

            15 years to make our world a better place for everyone: end poverty, protect the planet, and ensure prosperity

            for all. Towards this objective, the Department of Computer Science and Engineering, Sri Sivasubramaniya

            Nadar College of Engineering is proud to conduct and organize the ICCIDS 2024 - 7th International

            Conference on Computational Intelligence in Data Science on February 21 - 23, 2024.

        </p>

        <p>

            The conference provides an opportunity for researchers, engineers, developers, and practitioners from

            academia and industry to discuss experimental and theoretical work in solving problems to achieve

            sustainable development goals through Artificial Intelligence (AI).

        </p>

        <p>

            This conference holds value for anyone with an interest in or involvement with AI and its diverse

            applications across various domains. It features expert talks provided by individuals from renowned

            institutions and organizations across the globe, presentations by young researchers. Notably, accepted papers

            presented at the conference will be incorporated into the conference proceedings and subsequently published

            as part of the prestigious Springer IFIP AICT (Advances in Information and Communication Technology)

            series.

        </p>

        <hr>

        <h3>Indexing</h3>

        <p>Published volume will be indexed in-</p>

        <ul>

            <li>Scopus</li>

            <li>Google Scholar</li>

            <li>Citeseer</li>

            <li>Crossref</li>

        </ul>

        <hr>

        <h3>Commitee</h3>

        <center>

            <image src="annamalai.jpg" alt="Principal" height="250px" width="250px"></image> <image src="radha-1.jpg" alt="S.Radha" height="250px" width="250px"></image><image src="ttm.png" alt="HoD, CSE Dept." height="250px" width="250px"></image>

        </center>

        <hr>

        <h3>Technical Committee</h3>

        <p>

            <table cellspacing="20" cellpadding="1">

                <tr><th> </th><th> </th><th> </th></tr>

                <tr><td>Dr.Chutisant Kerdvibulvech, Associate Professor,Chulalongkorn University,Thailand</td><td>Dr.Arthi Murugesan Krishna,USA</td><td>Dr.Thangamani Paulchamy,CEO,Siva Cerulean Technologies,Atlanta,USA</td></tr>

               <tr><td>Dr.Yanhui Guo, Associate Professor,University of Illinois,Springfield, USA</td><td>Dr.Latha Karthika,CEO,Brandupwise Marketing,New Zealand</td><td>Dr.Dr.Sriram Kailasam, IIT Mandi</td></tr>

            </table>

        </p>

        <hr>

        <h3>Editors</h3>

        <center>

        <p>Prof. Mieczyslaw Lech OWOC,IFIP Vice Chair, WG 12.6-Knowledge Management</p>

        <p>Dr.R.Kanchana,Associate Professor,Department of CSE,SSNCE</p>

        <p>Dr.V.S Felix Enigo,Associate Professor,Department of CSE,SSNCE</p>

        <p>Dr.B.Prabavathy,Associate Professor,Department of CSE,SSNCE</p>

        </center>

        <hr>

        <h3>Program Committee</h3>

        <center>

        <p>Prof. Mieczyslaw Lech OWOC, IFIP Vice-Chair, WG 12.6 - Knowledge Management</p>

        <p>Mr. Dominique Verdejo, IFIP Chair, WG 12.13 - AI Global Security</p>

        <p>Dr. Zahia Guessoum, Université de Reims Champagne-Ardenne</p>

        <p>Dr. Nada Matta, Professor, UTT - Université de technologie de Troyes</p>

        <p>Dr.Eunika Mercier-Laurent, Professor and Chair IFIP TC12(AI), Dept. of Infomatics, University of Reims Champagne-Ardenne, France</p>

        </center>

        <hr>

        <h3>Committee</h3>

        <center>

            <image src="jpegphoto\_c2e4cdf509cfcca3d9e9b186388a39a5.jpg" alt="Prof. Mieczyslaw Lech Owoc" height="250px" width="250px"></image> <image src="image001.jpg" alt="Dr. Saroja P Kanchi" height="250px" width="250px"></image><image src="a4642881-a06a-409a-8f24-58a40d7fd123.jpg" alt="Dr. Zakaria Maamar" height="250px" width="250px"></image>

        </center>

        <hr>

        <h3>Event Date and Venue</h3>

        <table border="5">

            <tr><th>Date</th><th>Venue</th><th>Location(Google Maps)</th></tr>

            <tr><td>February 21 - 23, 2024</td><td>Department of Computer Science and Engineering, Sri Sivasubramaniya Nadar College of Engineering</td><td>https://maps.app.goo.gl/tP8rEEZzpB2BhNbg8</td></tr>

        </table>

        <hr>

        <h3>Entry fee</h3>

        <table border="5">

            <tr><th> </th><th>Indian</th><th>Indian</th><th>Foreigner</th><th>Non-Author</th></tr>

            <tr><th>Category</th><th>Early Bird</th><th>Normal</th><th>Normal</th><th>Normal</th></tr>

            <tr><td>Students</td><td>Rs.6000</td><td>Rs.6500</td><td>$150</td><td>Rs.3000</td></tr>

            <tr><td>Academicians</td><td>Rs.7000</td><td>Rs.7500</td><td>$250</td><td>Rs.3500</td></tr>

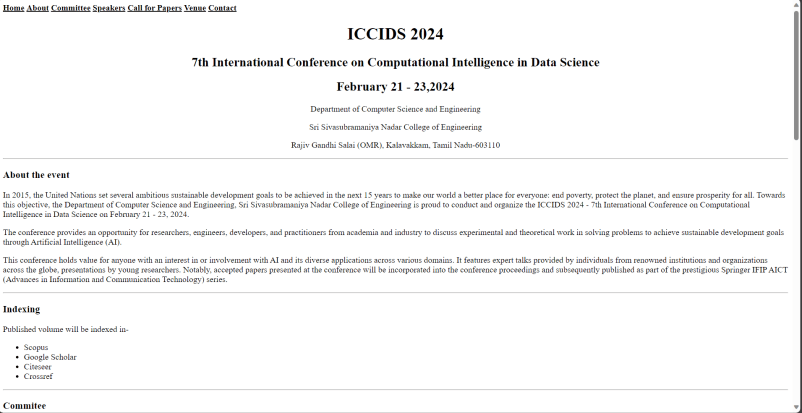
            <tr><td>Industry Designers</td><td>Rs.8000</td><td>Rs.8500</td><td>$350</td><td>Rs.4000</td></tr>

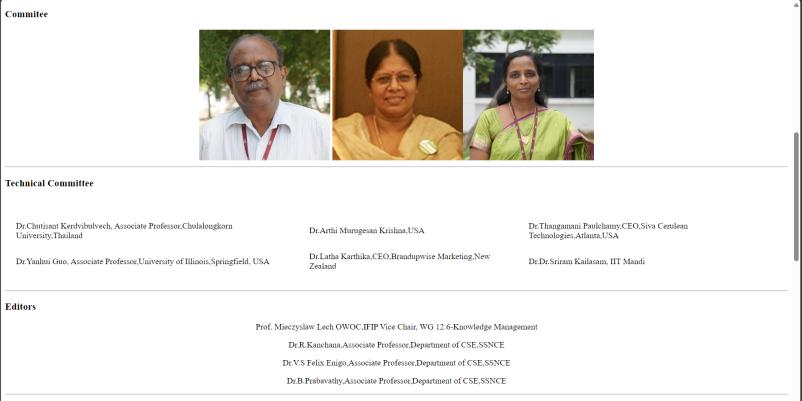
        </table>

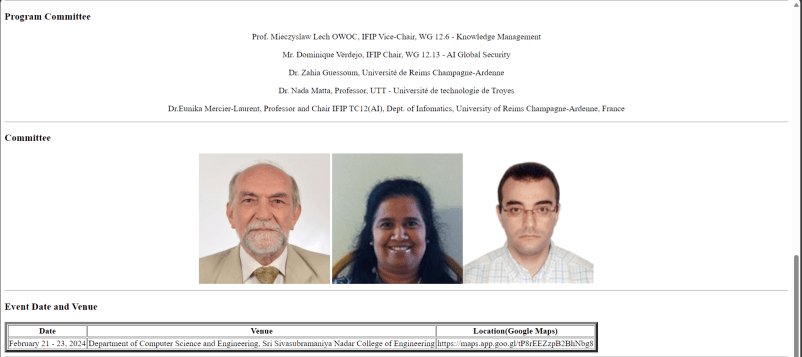
    </head>

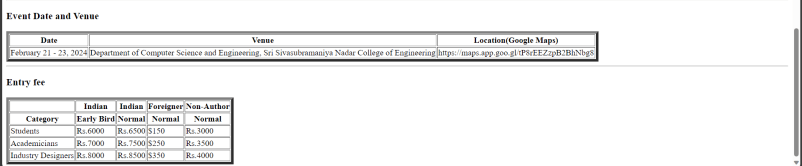
</html>

**Output:-**

****

****

****

****

Best Practices used:-

Learning Outcomes:-

**Webpage development in Native Language**

About yourself: (in 10 lines)

Name: XYZ

Gender:

Nationality:

Address: ---

City:

State:

Country:

Pin code:

Hobbies: (in 10 lines)

Read the following example and develop bio data about yourself in native language.

**Sample Example**

<!DOCTYPE html>

<html>

<head>

<title>Satcos-Tamil Web Pages</title>

**<meta http-equiv="content-type" content="text/html; charset=utf-8" />**

</head>

<body>

<p>தமிழ் வலைத்தளங்கள் உருவாக்குவது சுலபம்</p>

<p>தமிழ் வலைத்தளங்கள் உருவாக்குவது சுலபம்</p>

</body>

</html>

Follow these steps to create webpage using our native language

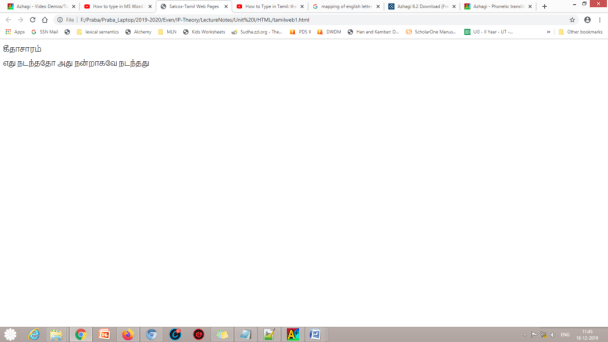
1. In order to use UTF encoding we need to specify character set meta tag in the “Head” section of the html.
2. Use [Azhagi+](http://www.azhagi.com/) or [Keyman](http://tamilkeyboard.com/#tam,Keyboard_ekwbamuni) to get the text in your desired language, copy past it in the html document.
3. Save the file as html with utf variant encoding

**Task**

Need to create the following content

கீதாசாரம்

எது நடந்ததோ அது நன்றாகவே நடந்தது



**How to develop a website in native language?**

**Steps**

1. Install free Azhaghi+ software using the link and install (setup file uploaded)
2. Install Notepad++
3. There will be something called existing hotkey in the azhaghi+ software. Note it. By default it is F10.

Open the word

Press F10 to type in tamil

Use keyboard mapping to type tamil letters.

If you want to change to English, type again F10

1. Copy the tamil sentences in the HTML tag as we type English sentences. Type the html code in Notepad++
2. Save as html file
3. Render in browser. You will be able to see a web page created in tamil font

**Website Development using Native Language**

**Design:-**

**Code:-**

<!DOCTYPE html>

<html>

<head>

<title>Satcos-Tamil Web Pages</title>

<meta http-equiv="content-type" content="text/html;charset=utf-8"

/>

</head>

<body>

<p>பெயர்: அஷ்வின் ரவி</p>

<p>பாலினம்: ஆண்</p>

<p>குடியுரிமை: இந்தியன்</p>

<p>முகவரி: டி-42&43, சாய் லட்சுமி என்கிளேவ், பிளாட் எண். 404, மதுரா நகர், ஹைதராபாத்-500038</p>

<p>நகரம்:ஹைதராபாத் </p>

<p>மாநிலம்: தெலுங்கானா</p>

<p>நாடு: இந்தியா</p>

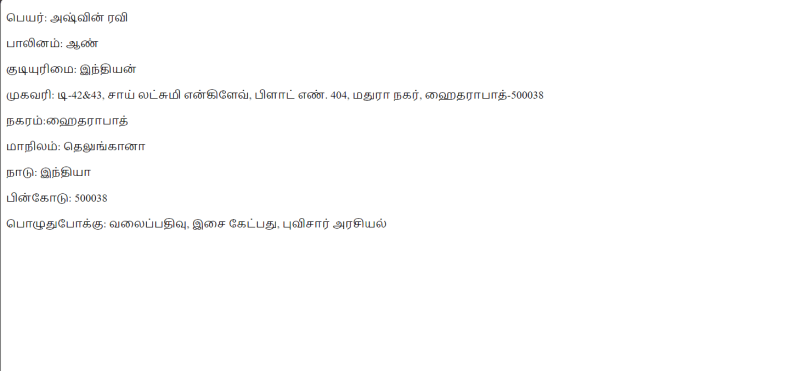
<p>பின்கோடு: 500038</p>

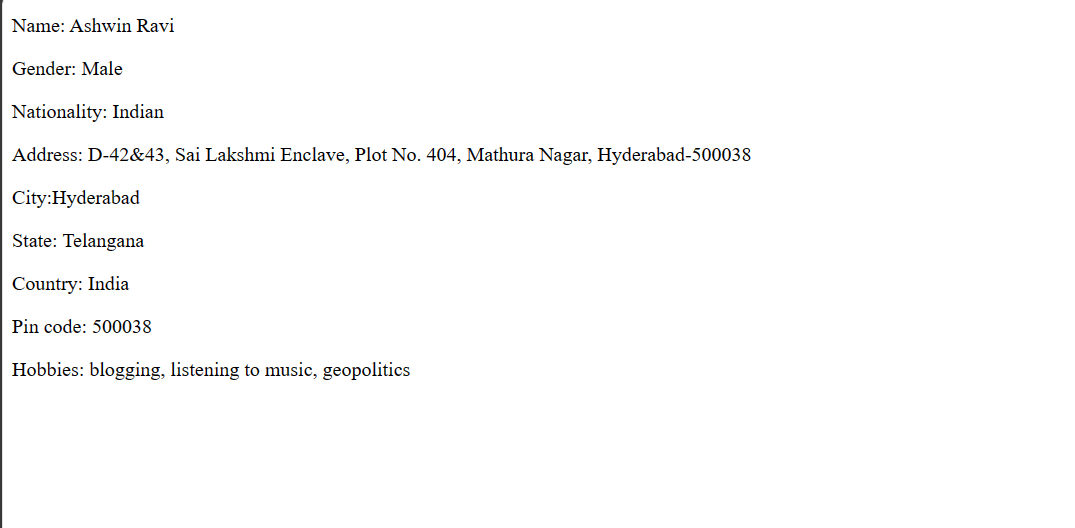
<p>பொழுதுபோக்கு: வலைப்பதிவு, இசை கேட்பது, புவிசார் அரசியல்</p>

</body>

</html>

Output:-





Best practices:-

Learning Outcomes:-

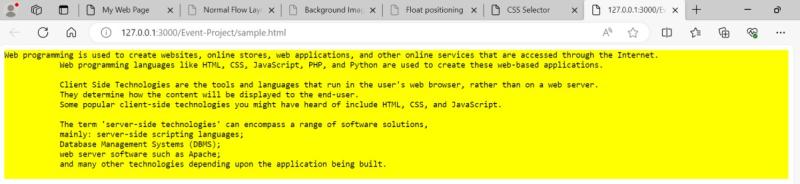
**UCS 2611 Internet Programming Lab**

**Exercise 2: Implementation of Conference Website using HTML & CSS**

**Date of Exercise: 03.01.2024**

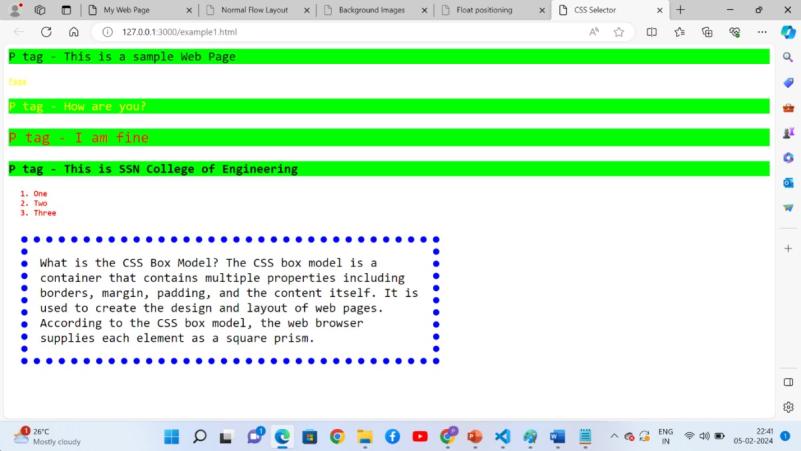
I. Practice the following examples using CSS [CO1, K3]

1. Write an HTML program to display the following using inline, embedded, and external stylesheets:

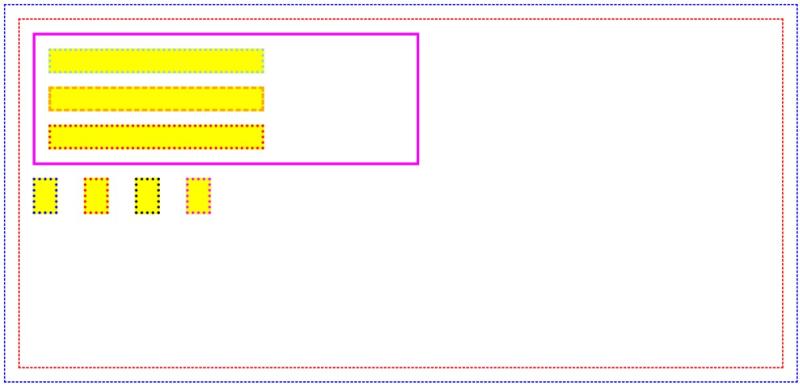


1. Given the following style file and the output snapshot, write HTML code recreate the output. Write down the types of selectors inheritance used in the code.

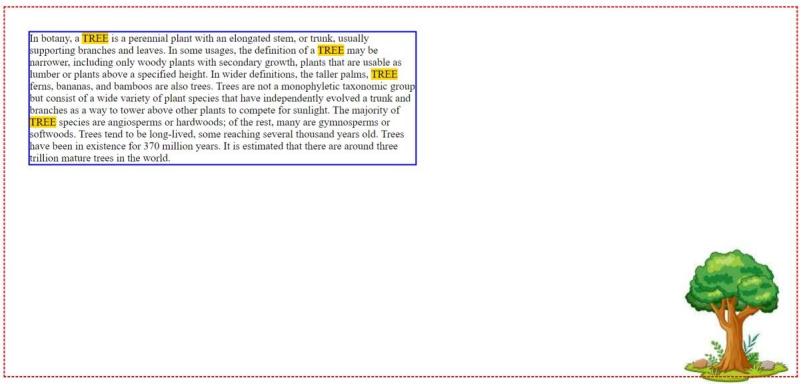
|  |
| --- |
| p { font-size:x-large; background-color: lime;  } \*{ font-family: monospace;  } a:link{color:black;} a:visited{color:yellow;} a:hover{color:blue;} a:active{color:red;}  #p2 { color:yellow  }  #box { background-color: white; border: 10px dotted blue; width:50%; padding: 20px; margin:30px 20px;  } |



1. Place html, body, div, span elements and highlight the boxes of these elements using different styles using border, margin, and padding properties of CSS box.



1. Place the background image and highlight the word *tree* using span element with the given style



1. Create the definition of fruits using background image and float the word *fruit* using the following style.



II. Following is the snapshot of international conference website. Create your version of the website with the same contents but it should never be the same as the original website [www.iccids.in](http://www.iccids.in).

Apply your thoughts to improve the website into an appealing way.

A screenshot of a web page

Description automatically generated

A screenshot of a website

Description automatically generated

A group of people in circles

Description automatically generated

A document with text on it

Description automatically generated

A group of people's heads

Description automatically generated

A screenshot of a document

Description automatically generated

A screenshot of a computer

Description automatically generated

Best Practices to be followed:

1. Design before coding
2. Incremental coding
3. Usage of proper naming convention
4. Usage of Comments to the code
5. Indentation of code

**Exercise 2: Implementation of Conference Website using HTML & CSS**

**Design:-**

1. Write an HTML program to display the following using inline, embedded and external stylesheets:

Code:-

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Sample text</title>

</head>

<body>

    <div class="main\_div" style="background-color:yellow">

        <p>Web programming is used to create websites, online stores, web applications, and other online services that are accessed through the Internet.</p>

        <p id="para" style="margin-left:100px">Web programming languages like HTML, CSS, JavaScript, PHP, and Python are used to create these web-based applications.</p>

        <div class="sub\_div" style="margin-left:100px">

            <p>Client Side Technologies are the tools and languages that run in the user's web browser, rather than on a web server.<br>

            They determine how the content will be displayed to the end-user.<br>

            Some popular client-side technologies you might have heard of include HTML, CSS, and JavaScript.</p>

        </div>

        <div class="sub\_div" style="margin-left:100px">

            <p>The term 'server-side technologies' can encompass a range of software solutions,<br>

                mainly: server-side scripting languages;<br>

                Database Management Systems (DBMS);<br>

                web server software such as Apache;<br>

                and many other technologies depending upon the application being built.</p>

        </div>

    </div>

</body>

</html>

Output:-



Code:-

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Just a sample</title>

    <style>

        .main\_div{

            background-color: yellow;

        }

        .main\_div #para{

            margin-left: 100px;

        }

        .sub\_div{

            margin-left: 100px;

        }

    </style>

</head>

<body>

    <div class="main\_div">

        <p>Web programming is used to create websites, online stores, web applications, and other online services that are accessed through the Internet.</p>

        <p id="para">Web programming languages like HTML, CSS, JavaScript, PHP, and Python are used to create these web-based applications.</p>

        <div class="sub\_div">

            <p>Client Side Technologies are the tools and languages that run in the user's web browser, rather than on a web server.<br>

            They determine how the content will be displayed to the end-user.<br>

            Some popular client-side technologies you might have heard of include HTML, CSS, and JavaScript.</p>

        </div>

        <div class="sub\_div">

            <p>The term 'server-side technologies' can encompass a range of software solutions,<br>

                mainly: server-side scripting languages;<br>

                Database Management Systems (DBMS);<br>

                web server software such as Apache;<br>

                and many other technologies depending upon the application being built.</p>

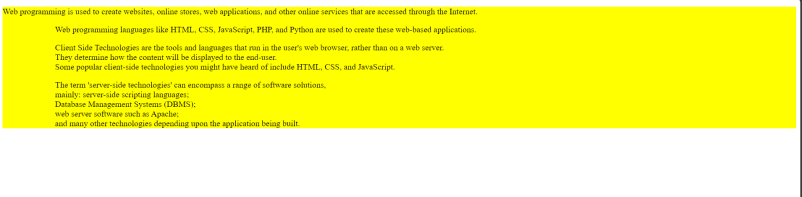
        </div>

    </div>

</body>

</html>

Output:-



Code:-

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <link rel="stylesheet" href="style.css">

    <title>External</title>

</head>

<body>

    <div class="main\_div">

        <p>Web programming is used to create websites, online stores, web applications, and other online services that are accessed through the Internet.</p>

        <p id="para">Web programming languages like HTML, CSS, JavaScript, PHP, and Python are used to create these web-based applications.</p>

        <div class="sub\_div">

            <p>Client Side Technologies are the tools and languages that run in the user's web browser, rather than on a web server.<br>

            They determine how the content will be displayed to the end-user.<br>

            Some popular client-side technologies you might have heard of include HTML, CSS, and JavaScript.</p>

        </div>

        <div class="sub\_div">

            <p>The term 'server-side technologies' can encompass a range of software solutions,<br>

                mainly: server-side scripting languages;<br>

                Database Management Systems (DBMS);<br>

                web server software such as Apache;<br>

                and many other technologies depending upon the application being built.</p>

        </div>

    </div>

</body>

</html>

CSS:-

.main\_div{

    background-color: yellow;

}

.main\_div #para{

    margin-left: 100px;

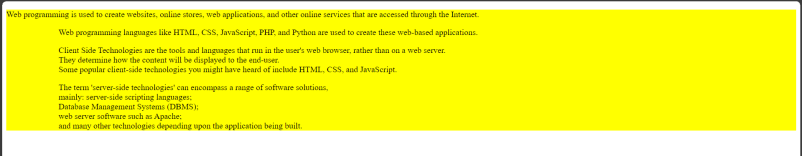
}

.sub\_div{

    margin-left: 100px;

}

Output:-



1. Given the following style file and the output snapshot, write HTML code recreate the output. Write down the types of selectors inheritance used in the code.

Code:-

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>HTML\_Styling</title>

    <link rel="stylesheet" href="style1.css">

</head>

<body>

    <p >P tag - This is a sample Web Page</p>

    <p id="p2">P tag - How are you?</p>

    <p><a>P tag - I am fine</a></p>

    <p>P tag - This is SSN College of Engineering</p>

    <a>

        <ol>

            <li>One</li>

            <li>Two</li>

            <li>Three</li>

        </ol>

    </a>

     <div>

        <p id="box">The CSS box model is a container that contains multiple properties including borders, margins, padding, and the content itself. It is used to create the design and layout of web pages. It can be used as a toolkit for customizing the layout of different elements. The web browser renders every element as a rectangular box according to the CSS box model. </p>

     </div>

</body>

</html>

CSS:-

p {

    font-size:x-large;

    background-color: lime;

}

\*{

    font-family: monospace;

}

a:link{color:black;}

a:visited{color:yellow;}

a:hover{color:blue;}

a:active{color:red;}

#p2 {

    color:yellow

}

#box {

    background-color: white;

    border: 10px dotted blue;

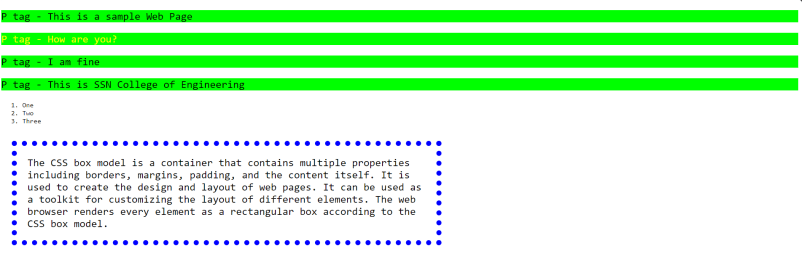
    width:50%;

    padding: 20px;

    margin:30px 20px;

}

Output:-



1. Place html, body, div, span elements and highlight the boxes of these elements using different styles using border, margin, and padding properties of CSS box.

Code:-

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <link rel="stylesheet" href="style2.css">

    <title>Border</title>

</head>

<body>

    <div class="outer\_div1">

        <div class="outer\_div2">

            <div class="outer\_div3">

                <div class="inner\_div1"></div>

                <div class="inner\_div2"></div>

                <div class="inner\_div3"></div>

            </div>

            <div class="inner\_div4"></div>

            <span class="inner\_div5"></span>

            <span class="inner\_div6"></span>

            <span class="inner\_div7"></span>

        </div>

    </div>

</body>

</html>

CSS:-

.outer\_div1{

    border: 2px solid blue;

    width:75%;

    height:600px;

    margin: auto;

    margin-top: 40px;

    border-style: dashed;

}

.outer\_div2{

    border: 2px solid red;

    height: 560px;

    margin: 20px;

    border-style: dashed;

}

.outer\_div3{

    border: 4px solid rgb(229, 22, 212);

    height: 200px;

    width: 50%;

    margin: 20px;

}

.inner\_div1{

    border: 4px solid rgb(11, 172, 83);

    height: 30px;

    width: 60%;

    margin: 20px;

    background-color: yellow;

    border-style: dotted;

}

.inner\_div2{

    border: 4px solid rgb(247, 184, 12);

    height: 30px;

    width: 60%;

    margin: 20px;

    background-color: yellow;

    border-style: dashed;

}

.inner\_div3{

    border: 4px solid rgb(230, 111, 14);

    height: 30px;

    width: 60%;

    margin: 20px;

    background-color: yellow;

    border-style: dotted;

}

.inner\_div4{

    border: 4px solid rgb(43, 9, 211);

    height: 70px;

    width: 4%;

    margin: 20px;

    background-color: yellow;

    border-style: dotted;

    display: inline-block;

}

.inner\_div5{

    border: 4px solid rgb(230, 111, 14);

    height: 70px;

    width: 4%;

    margin: 20px;

    background-color: yellow;

    border-style: dotted;

    display: inline-block;

}

.inner\_div6{

    border: 4px solid black;

    height: 70px;

    width: 4%;

    margin: 20px;

    background-color: yellow;

    border-style: dotted;

    display: inline-block;

}

.inner\_div7{

    border: 4px solid rgb(244, 18, 225);

    height: 70px;

    width: 4%;

    margin: 20px;

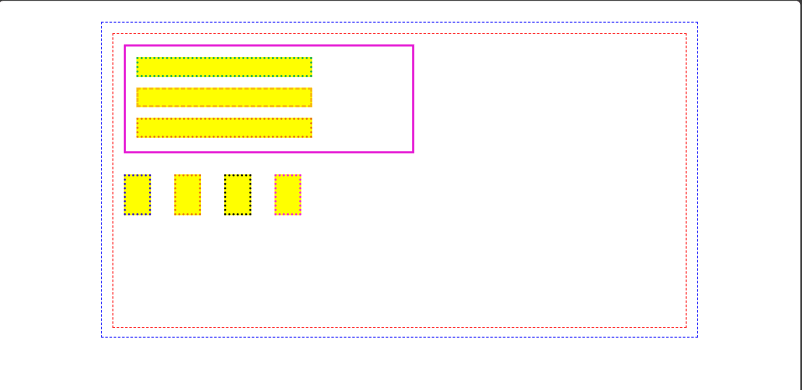
    background-color: yellow;

    border-style: dotted;

    display: inline-block;

}

Output:-



1. Place the background image and highlight the word *tree* using span element with the given style

Code:-

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <link rel="stylesheet" href="style3.css">

    <title>Tree</title>

</head>

<body>

    <div class="outer\_div">

        <p class="para\_div">In botany, a <span>TREE</span> is a perennial plant with an elongated stem, or trunk, usually supporting branches and leaves. In some usages, the definition of a <span>TREE</span> may be narrower, including only woody plants with secondary growth, plants that are usable as lumber or plants above a specified height. In wider definitions, the taller palms, <span>TREE</span> ferns, bananas, and bamboos are also trees.Trees are not a monophyletic taxonomic group but consist of a wide variety of plant species that have independently evolved a trunk and branches as a way to tower above other plants to compete for sunlight. The majority of <span>TREE</span> species are angiosperms or hardwoods; of the rest, many are gymnosperms or softwoods. <span>TREE</span> tend to be long-lived, some reaching several thousand years old. Trees have been in existence for 370 million years. It is estimated that there are around three trillion mature trees in the world.</p>

        <div class="img\_tree">

            <img src="tree.png">

        </div>

    </div>

</body>

</html>

CSS:-

.outer\_div{

    border: 2px solid red;

    width: 80%;

    margin: auto;

    height :500px;

    border-style: dashed;

}

.para\_div{

    border: 2px solid green;

    width: 50%;

    margin-left: 40px;

    margin-top: 40px;

}

span{

    background-color: yellow;

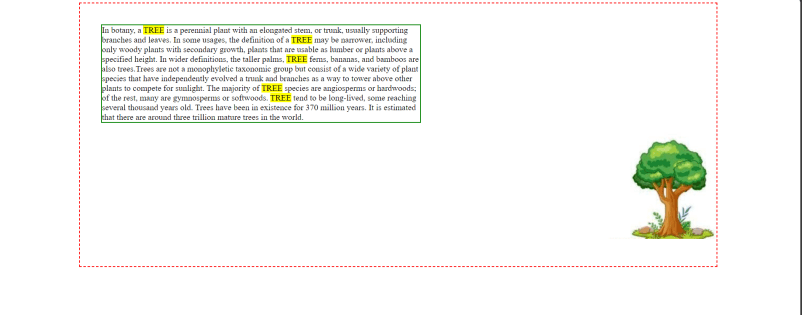
}

.img\_tree{

    float: right;

}

Output:-



1. Create the definition of fruits using background image and float the word *fruit* using the following style.

Code:-

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <link rel="stylesheet" href="style4.css">

    <title>Fruits</title>

</head>

<body>

    <div class="outer\_div">

        <p class="para\_div">In botany, a fruit is the seed-bearing structure in flowering plants that is formed from the ovary after flowering (see Fruit anatomy).<span>Fruits</span> are the means by which flowering plants (also known as angiosperms) disseminate their seeds. Edible fruits in particular have long propagated using the movements of humans and other animals in a symbiotic relationship that is the means for seed dispersal for the one group and nutrition for the other; in fact, humans and many other animals have become dependent on fruits as a source of food.[1] Consequently, fruits account for a substantial fraction of the world's agricultural output, and some (such as the apple and the pomegranate) have acquired extensive cultural and symbolic meanings.</p>

        <div class="img\_fruits">

            <img src="fruits.jpg">

        </div>

    </div>

</body>

</html>

CSS:-

.outer\_div{

    border: 2px solid blue;

    width: 80%;

    margin: auto;

    height :500px;

    border-style: dashed;

}

.para\_div{

    margin-left: 40px;

    margin-top: 40px;

}

span{

    font-size: larger;

    font-weight: bolder;

}

img{

    height: 200px;

    float:right;

    margin-top: 150px;

}

Output:-



Apply your thoughts to improve the website into an appealing way.

Following is the snapshot of international conference website. Create your version of the website with the same contents

* original website [www.iccids.in](http://www.iccids.in).

Code:-

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <link rel="stylesheet" href="style5.css">

    <title>Conference</title>

</head>

<body>

    <nav>

        <img src="my\_clg\_logo.png" id="my\_logo">

        <a href="index.html">Home</a>

        <a href="#event\_anchor">About</a>

        <a href="#index\_anchor">Indexing </a>

        <a href="#committee\_anchor">Committee</a>

        <a href="#speaker\_anchor">Speakers</a>

        <a href="#pc\_anchor">Past Conferences</a>

        <a href="#cop\_anchor">Call for Papers</a>

        <a>Venue</a>

        <a href="#contact\_anchor">Contact</a>

    </nav>

    <div class="cover">

        <div class="main\_heading"><p>ICCIDS 2024</p></div>

        <div class="heading\_7th"><h1>7<sup>th</sup> International Conference on Computational Intelligence in Data Science</h1></div>

    </div>

    <div class="department">

        <p>February 21 -23, 2024</p>

        <p>Department of Computer Science and Engineering</p>

        <p><span>Sri Sivasubramaniya Nadar College of Engineering</span></p>

    </div>

    <div class="location">

        Rajiv Gandhi Salai(OMR), Chennai, Tamil Nadu, India

    </div>

    <button class="Button\_About">

        <a href="#event\_anchor">About The Event</a>

    </button>

    <div class="about\_heading">

        <h2 id="event\_anchor">About The Event</h2>

    </div>

    <div class="event\_blink">

        <a href="https://link.springer.com/conference/iccids" target="blank"><span>ICCIDS 2023, ICCIDS 2022, ICCIDS 2021, ICCIDS 2020 - Proceedings</span></a>

    </div>

    <div class="img\_div">

        <div class="pic1"><img src="ifiplogo.jpg"></div>

        <div class="pic2"><img src="springerlogo.jpg"></div>

        <div class="pic3"><img src="jbsoftsystem.png"></div>

    </div>

    <div class="event\_para">

        <p><b>Theme: </b>AI based Knowledge Discovery and Management</p>

        <p>In 2015, the United Nations set several ambitious sustainable development goals to be achieved in the next 15 years to make our world a better place for everyone: end poverty, protect the planet, and ensure prosperity for all. Towards this objective, the Department of Computer Science and Engineering, Sri Sivasubramaniya Nadar College of Engineering is proud to conduct and organize the ICCIDS 2024 - 7th International Conference on Computational Intelligence in Data Science on February 21 - 23, 2024</p>

        <p>The conference provides an opportunity for researchers, engineers, developers, and practitioners from academia and industry to discuss experimental and theoretical work in solving problems to achieve sustainable development goals through Artificial Intelligence (AI).</p>

        <p>This conference holds value for anyone with an interest in or involvement with AI and its diverse applications across various domains. It features expert talks provided by individuals from renowned institutions and organizations across the globe, presentations by young researchers. Notably, accepted papers presented at the conference will be incorporated into the conference proceedings and subsequently published as part of the prestigious Springer IFIP AICT (Advances in Information and Communication Technology) series.</p>

    </div>

    <hr>

    <div class="indexing\_heading">

        <h2 id="index\_anchor">Indexing</h2>

    </div>

    <div class="indexing\_data">

        <p>Published volume will be indexed in-</p>

        <ul>

            <li>Scopus</li>

            <li>Google Scholar</li>

            <li>Citeseer</li>

            <li>Crossref</li>

        </ul>

    </div>

    <hr>

    <div class="committee\_container">

        <div class="box\_heading">

            <h2 id="committee\_anchor">Committee</h2>

        </div>

        <div class="box1">

            <div class="inner">

                <h3 class="patron">Chief Patron</h3>

                <img src="shivnadar.jpg" alt="shiv\_nadar">

                <h3>Dr. Shiv Nadar</h3>

                <p>Founder,SSN Institutions<br> Chairman,HCL Technologies</p>

            </div>

            <div class="inner">

                <h3 class="patron">Patron</h3>

                <img src="kala2.png" alt="kala">

                <h3>Dr. Kala Vijayakumar</h3>

                <p>President,SSN Institutions</p>

            </div>

            <div class="inner">

                <h3 class="patron">Advisory Committee</h3>

                <img src="P-Ramasamy.jpg" alt="PR">

                <h3>Dr. P. Ramasamy</h3>

                <p>Dean of Research, SSNCE</p>

            </div>

        </div>

        <div class="box\_heading">

            <h2>General Chairs</h2>

        </div>

        <div class="box1">

            <div class="inner">

                <img src="annamalai.jpg" alt="annamalai">

                <h3>Dr. V.E, Annamali</h3>

                <p>Principal,SSNCE</p>

            </div>

            <div class="inner">

                <img src="radha-1.jpg" alt="Radha">

                <h3>Dr. S. Radha</h3>

                <p>Vice Principal, SSNCE</p>

            </div>

            <div class="inner">

                <img src="ttm.png" alt="ttm">

                <h3>Dr. T.T. Mirnalinee</h3>

                <p>Head of Computer Science and <br>Engineering Department, SSNCE</p>

            </div>

        </div>

        <div class="box\_heading">

            <h2>Conference Chairs</h2>

        </div>

        <div class="box1">

        <div class="inner">

            <img src="Felix.png" alt="Felix">

            <h3>Dr. V.S. Felix Enigo</h3>

            <p>Associate Professor, CSE Dept, SSNCE</p>

        </div>

        <div class="inner">

            <img src="Praba.png" alt="Praba">

            <h3>Dr. B. Prabavathy</h3>

            <p>Associate Professor, CSE Dept, SSNCE</p>

        </div>

        <div class="inner">

            <img src="kanchana.png" alt="kanchana">

            <h3>Dr. R. Kanchana</h3>

            <p>Associate Professor, CSE Dept, SSNCE</p>

        </div>

        </div>

        <div class="box\_heading">

            <h2>Organising Committee</h2>

        </div>

        <div class="box1">

        <div class="inner">

            <img src="suresh.jpg" alt="suresh">

            <h3>Dr. J. Suresh</h3>

            <p>Associate Professor, CSE Dept, SSNCE</p>

        </div>

        <div class="inner">

            <img src="sakaliraghavendracse.jpg" alt="SR">

            <h3>Mr. Sakali Raghavendra Kumar</h3>

            <p>Assistant Professor, CSE Dept, SSNCE</p>

        </div>

        <div class="inner">

            <img src="Rajeswari.png" alt="Rajeswari">

            <h3>Ms. J. Rajeswari</h3>

            <p>Assistant Professor, CSE Dept, SSNCE</p>

        </div>

        </div>

    </div>

    <div class="tech-comitee\_heading">

        <h2>Technical Committee</h2>

    </div>

    <div class="tech-comitee">

        <p id="tc\_p1">Dr. Chutisant Kerdvibulvech, Associate Professor, Chulalongkorn University, Thailand<br>

            Dr. Yanhui Guo, Associate Professor, USA, University of Illinois Springfield<br>

            Dr. Ricardo Araujo Rios, Professor, Brazil, Ricardo Araújo Rios Institute of Computing, Federal University of Bahia<br>

            Dr. Leung, Yiu Wing, Professor, Hong Kong Baptist University, Hong Kong<br>

            Dr. Kaliappan Ravindran, Professor, City College of New York, USA<br>

            Dr. B. Vijayakumar, Professor, BITS Pilani, Dubai Campus<br>

            Dr. Kumaradevan Punithakumar, Associate Professor, University of Alberta<br>

            Dr. Sreenivasan Ramasamy Ramamurthy, Ph.D., Bowie State University, USA<br>

            Dr. Femilda Josephin Joseph Shobana Bai, Istinye University, Istanbul, Turkey<br>

            Dr. Elluri Lavanya, Texas A & M University, Central Texas, USA<br>

            Dr. Arun A, General Motors R&D, Michigan, USA<br>

            Mr. Senthilnathan Chidambaranathan, Associate Director, Virtusa, USA<br>

            Dr.V.Gomathi. Professor & Head, CSE Dept., National Engineering College</p>

        <p id="tc\_p2">Dr. Arthi Murugesan Krishna, USA<br>

            Dr. Latha Karthika, CEO, Brandupwise Marketing, New Zealand<br>

            Dr. Mittapalle Kiran, Aalto University, Finland<br>

            Ms. Lakshmi Divya JK, Microsoft, Pittsburgh, USA<br>

            Ms. R. Ishwarya, TD Bank, Ontario, Canada<br>

            Ms. Jyothi Chitra Thangaraj, CTS, USA<br>

            Mr. Chandrasekar, Software Engineer, Google LLC, Seattle, Washington, USA<br>

            Mr. Seshakumar, Doordash.inc Fremont, California<br>

            Ms. N. Malligeswari, Bank of America, North Carolina, USA<br>

            Ms. Nirmala Balakrishnan, Luminous Computing, Santa Clara, California, USA<br>

            Dr. C. S. Sridharan, Thiagarajar College of Engineering, Madurai<br>

            Ms. R. Uma, Cybersecurity Leader, Amazon, Virginia, United States<br>

            Dr.T. Bhaskaran, Professor, TCE, Madurai<br>

            Dr. S. Parthasarathy, Professor & Head, TCE, Madurai<br>

            Mr. Ravikumar Sadanandam, American Family Insurance, New Jersey, USA</p>

        <p id="tc\_p3">Mr. Thangmani Paulchamy, CEO, Siva Cerulean Technologies, Atlanta, USA<br>

            Dr. Sriram Kailasam, IIT Mandi<br>

            Dr. Rahul Raman, Assistant Professor, IIITDM, Kancheepuram<br>

            Dr. Umarani Jayaraman, Assistant Professor, IITDM<br>

            Dr. K. Bhoopathy Bagan, Professor, MIT, Anna University, Chennai<br>

            Dr. Mary Saira Banu, Professor & Head, NIT Trichy<br>

            Dr. S. Selvakumar, Professor, NIT, Trichy<br>

            Dr. S. Sheerazudeen, Assistant Professor, NIT Calicut<br>

            Dr. V. Vani, Assistant Professor, NIT Puducherry<br>

            Dr. R. Padmavathy, Professor, NIT Warangal<br>

            Dr. P. Varalakshmi, Professor, Anna University<br>

            Dr. R. Gunasundari, Professor, Puducherry Technological University<br>

            Dr. Latha Parthiban, Associate Professor, Pondicherry University<br>

            Dr. Kishore Balasubramanian, Associate Professor, Dr. Mahalingam College of Engineering & Technology<br>

            Dr. Mayuri Mehta, Sarvajanik College of Engineering & Technology, Surat

            </p>

    </div>

    <div class="editors\_heading">

        <h2>Editors</h2>

    </div>

    <div class="editors">

        <p>Prof. Mieczyslaw Lech OWOC, IFIP Vice-Chair, WG 12.6 - Knowledge Management</p>

        <p>Dr. R. Kanchana, Associate Professor, Department of CSE, SSNCE</p>

        <p>Dr. V.S Felix Enigo, Associate Professor, Department of CSE, SSNCE</p>

        <p>Dr. B. Prabavathy, Associate Professor, Department of CSE, SSNCE</p>

    </div>

    <div class="program\_comitee\_heading">

        <h2>Program Committee</h2>

    </div>

    <div class="program\_comitee">

        <p>Prof. Mieczyslaw Lech OWOC, IFIP Vice-Chair, WG 12.6 - Knowledge Management</p>

        <p>Mr. Dominique Verdejo, IFIP Chair, WG 12.13 - AI Global Security</p>

        <p>Dr. Zahia Guessoum, Université de Reims Champagne-Ardenne</p>

        <p>Dr. Nada Matta, Professor, UTT - Université de technologie de Troyes</p>

        <p>Dr.Eunika Mercier-Laurent, Professor and Chair IFIP TC12(AI)</p>

        <p>Dept. of Infomatics, University of Reims Champagne-Ardenne, France</p>

        <p>Dr. T.T. Mirnalinee, Professor & Head, Department of CSE, SSNCE</p>

        <p>Dr. R. Kanchana, Associate Professor, Department of CSE, SSNCE</p>

        <p>Dr. V.S Felix Enigo, Associate Professor, Department of CSE, SSNCE</p>

        <p>Dr. B. Prabavathy, Associate Professor, Department of CSE, SSNCE</p>

        <p>Dr. J. Suresh, Associate Professor, Department of CSE, SSNCE</p>

    </div>

    <hr>

    <div class="keynote\_container">

        <div class="box\_heading">

            <h2 id="speaker\_anchor">Keynote Talks</h2>

        </div>

        <div class="keybox">

            <div class="key\_inner">

                <h4 class="patron">Impact of Fakes on Sustainable Development Actions- Trusted Knowledge as Counteraction</h4>

                <img src="k1.jpg" alt="K1">

                <h3>Prof. Mieczyslaw Lech Owoc</h3>

                <p>Professor, Wroclaw University of Economics & Business Wroclaw, Poland</p>

                <a href="https://www.researchgate.net/profile/Mieczyslaw-Owoc" target="\_blank">Profile</a>

            </div>

            <div class="key\_inner">

                <h4 class="patron">Localization and Clustering in Wireless Sensor Networks</h4>

                <img src="k2.jpg" alt="k2">

                <h3>Dr. Saroja P Kanchi</h3>

                <p>Professor, Kettering University Flint, MI, USA</p>

                <a href="https://paws.kettering.edu/~skanchi/" target="\_blank">Profile</a>

            </div>

            <div class="key\_inner">

                <h4 class="patron">Cloud-Edge Competition: A win-win partnership</h4>

                <img src="k3.jpg" alt="k3" id="zm">

                <h3>Dr. Zakaria Maamar</h3>

                <p>Professor, Zayed University Dubai, UAE</p>

                <a href="https://ieeexplore.ieee.org/author/37282959400" target="\_blank">Profile</a>

            </div>

        </div>

        <div class="keybox">

            <div class="key\_inner">

                <h4 class="patron">Securing Tomorrow: Unveiling the Synergy of Computational Intelligence and Data Science in Network Security</h4>

                <img src="k4.jpg" alt="k4">

                <h3>Dr. Arun Raj Kumar P.</h3>

                <p>Assistant Professor National Institute of Technology Calicut, India</p>

                <a href="https://nitc.ac.in/department/computer-science-amp-engineering/faculty-and-staff/faculty/c7010e4a-dccb-4997-b1c1-919489cf7585" target="\_blank">Profile</a>

            </div>

            <div class="key\_inner">

                <h4 class="patron">Workshop on AI and XAI-based Use Cases in Healthcare using Python</h4>

                <img src="k5.jpg" alt="k5">

                <h3>Dr. Mayuri Mehta</h3>

                <p>Sarvajanik College of Engineering & Technology Surat, India</p>

                <a href="https://scet.ac.in/employee/prof-dr-mayuri-mehta/" target="\_blank">Profile</a>

            </div>

            <div class="key\_inner">

                <h4 class="patron">Gen-AI and the next wave of democratizatio of IT Speech</h4>

                <img src="K6.jpg" alt="k6">

                <h3>Kausikram Krishnasayee</h3>

                <p>Senior Director of Product Management Kissflow</p>

                <a href="https://www.linkedin.com/in/kausikram/?originalSubdomain=in" target="\_blank">Profile</a>

            </div>

        </div>

    </div>

    <hr>

    <div class="pc\_heading">

        <h2 id="pc\_anchor">Past Conferences</h2>

    </div>

    <div class="pc">

        <div class="dot"></div>

        <div class="dot"></div>

        <div class="dot"></div>

        <div class="dot"></div>

        <div class="dot"></div>

        <div class="dot"></div>

        <div class="line"></div>

    </div>

    <p id="icc1">ICCIDS<br>2017</p>

    <p id="icc2">ICCIDS<br>2019</p>

    <p id="icc3">ICCIDS<br>2020</p>

    <p id="icc4">ICCIDS<br>2021</p>

    <p id="icc5">ICCIDS<br>2022</p>

    <p id="icc6">ICCIDS<br>2023</p>

    <p id="dummy">Hi</p>

    <hr>

    <div class="call\_papers\_heading">

        <h2 id="cop\_anchor">Call for Papers</h2>

    </div>

    <div class="call\_papers\_para1">

        <p>We invite high-quality regular paper submissions with technical research or survey work describing original and unpublished work addressing the theme of the conference. The reviewed, accepted and presented papers will be published in Springer IFIP AICT. The paper should not exceed 15 pages including references. Article must include title, abstract, list of keywords, introduction, related work, proposed methodology, discussion of results, and conclusion with future work. Topics of interest for the conference include, but are not restricted to:</p>

        <ul>

            <li>AI-based Knowledge Discovery and Management</li>

            <li>AI-Enabled knowledge management, framework, and control</li>

            <li>AI-powered Knowledge-as-a-Service (KaaS)</li>

            <li>Predictive Intelligence for Knowledge Management</li>

            <li>Natural Language Processing techniques</li>

            <li>Applications of AI in computer vision</li>

            <li>Speech processing techniques</li>

            <li>ML/DL algorithms in intelligent autonomous systems</li>

            <li>Deep neural networks-based frameworks and algorithms for Energy Efficiency</li>

        </ul>

        <p>Authors must adhere to the formatting instructions, utilizing Latex or Word style as specified on the <span>Author&apos;s instruction page</span> for camera-ready papers. Authors are requested to make use of below templates for preparing the manuscript<br>

        <span>LaTex</span><br>

        <span>MS-Word</span><br>

        Submission Link:<a href="https://equinocs.springernature.com/service/ICCIDS2024" target="\_blank"><span>https://equinocs.springernature.com/service/ICCIDS2024</span></a></p>

    </div>

    <div class="call\_papers\_para2">

        <p>At least one author of an accepted paper is required to register for the conference.</p>

        <p>A distinguished Best Paper Award will be presented for each session, recognizing outstanding article quality.</p>

    </div>

    <hr>

    <div class="call\_papers\_heading">

        <h2>Important Dates</h2>

    </div>

    <table>

        <thead>

            <tr>

                <th>Event</th>

                <th>Date</th>

            </tr>

        </thead>

        <tbody>

            <tr>

                <td rowspan="4">Paper Submission Deadline</td>

                <td><span class="strike">December 10, 2023</span></td>

            </tr>

            <tr>

                <td><span class="strike">December 20, 2023</span></td>

            </tr>

            <tr>

                <td><span class="strike">December 31, 2023</span></td>

            </tr>

            <tr>

                <td id="closedBold">Closed</td>

            </tr>

            <tr>

                <td rowspan="2">Notification of Acceptance</td>

                <td><span class="strike">January 10, 2024</span></td>

            </tr>

            <tr>

                <td class="blink">January 31, 2024</td>

            </tr>

            <tr>

                <td rowspan="2">Camera Ready Copy</td>

                <td><span class="strike">January 30, 2024</span></td>

            </tr>

            <tr>

                <td class="blink">February 09, 2024</td>

            </tr>

            <tr>

                <td>Conference & Workshop</td>

                <td>February 21,22 &amp; 23, 2024</td>

            </tr>

        </tbody>

    </table>

    <hr>

    <div class="reg\_fee\_title">

        <h2>Registration Fees <span><sup>\*\*</sup></span></h2>

    </div>

    <div class="reg\_para">

        <p><span>Registration closes by February 13, 2024</span></p>

    </div>

    <table class="table2">

        <thead>

            <tr>

                <th></th>

                <th>Indian</th>

                <th>Indian</th>

                <th>Foreigners</th>

                <th>Non-Author</th>

            </tr>

        </thead>

        <tbody>

            <tr>

                <td rowspan="2">Category</td>

                <div class="table\_reg\_red">

                    <td><span>Early Bird</span></td>

                    <td rowspan="2"><span>Normal</span></td>

                    <td rowspan="2"><span>Normal</span></td>

                    <td rowspan="2"><span>Normal</span></td>

                </div>

            </tr>

            <tr>

                <td class="table\_reg\_red"><span>(on or before <span class="strike">January 20</span> February 09,2024)</span></td>

            </tr>

            <tr>

                <td>Students</td>

                <td>Rs.6000</td>

                <td>Rs.6500</td>

                <td>$150</td>

                <td>Rs.3000</td>

            </tr>

            <tr>

                <td>Academicians</td>

                <td>Rs.7000</td>

                <td>Rs.7500</td>

                <td>$250</td>

                <td>Rs.3500</td>

            </tr>

            <tr>

                <td>Industry Delegates</td>

                <td>Rs.8000</td>

                <td>Rs.8500</td>

                <td>$350</td>

                <td>Rs.4000</td>

            </tr>

        </tbody>

    </table>

    <div class="reg\_para2">

        <p><span>\*IFIP Technical Committees and Working Groups are entitled to a discount of at least 10% on all registration fees</span></p>

        <p><span>\*\* 18% GST extra</span></p>

    </div>

    <hr>

    <div class="contact\_heading">

        <h2 id="contact\_anchor">Contact Us</h2>

    </div>

    <div class="email">

        <p>E-mail: <span>ashwin\_ravi@hotmail.com</span></p>

    </div>

</body>

<footer>

    <div class="copyright">

        <p>&copy; Copyright AshRav 2024 .All Rights Reserved<br><br>Designed by Ashwin Ravi SSN CSE'25</p>

    </div>

</footer>

</html>

CSS:-

body{

    font-family: "Roboto", sans-serif;

    font-weight: 400;

}

body,html{

    height: 100%;

    margin: 0;

}

#my\_logo{

    height: 32px;

    margin-right: 35%;

    margin-top: 10px;

    margin-left: 19px;

}

nav{

    position: absolute;

    top:0%;

    width: 100%;

    height: 70px;

    align-items: center;

    position: fixed;

    color: aliceblue;

    background-color: #0e1b4d;

}

nav a{

    margin-right: 15px;

    color: aliceblue;

}

nav a:hover {

    color: aqua;

}

.cover{

    background-image: url("/Images/black.jpg");

    height:100%;

    background-position: center;

    background-repeat: no-repeat;

    background-size: cover;

    text-align: center;

}

.main\_heading{

    position: absolute;

    left:42%;

    top:25%;

    font-size: xxx-large;

    font-weight: bolder;

    color: white;

}

.heading\_7th{

    position: absolute;

    left:11%;

    top:40%;

    color: white;

}

.department{

    position: absolute;

    color: white;

    top: 53%;

    left: 40%;

    text-align: center;

}

.location{

    position: absolute;

    color: white;

    top: 72%;

    left: 40%;

    text-align: center;

}

.Button\_About a{

    position: absolute;

    top: 79%;

    left: 47%;

    cursor: pointer;

    border-radius: 50px;

    padding: 12px 32px;

    letter-spacing: 1px;

    font-size: 14px;

    border:2px solid #f82249;

    color:aliceblue;

    background-color:blueviolet;

}

.Button\_About a:hover{

    background-color: #f82249;

}

.tech-comitee\_heading,.editors\_heading,.program\_comitee\_heading,.call\_papers\_heading,.reg\_fee\_title,.contact\_heading,.about\_heading,.indexing\_heading, .box\_heading,.pc\_heading {

    color: #0e1b4d;

    text-align: center;

}

@keyframes blink\_event{

    50%{

        opacity: 0

    }

}

.event\_blink{

    text-align: center;

    font-size: 22px;

    animation: blink\_event 1.8s linear infinite;

}

.img\_div{

    display: grid;

    grid-template-columns: repeat(3,1fr);

    gap:20px;

    margin: 20px;

}

.img\_div .pic1,.pic2,.pic3{

     display: grid;

     justify-content: center;

     align-items: center;

}

.img\_div .pic2{

    grid-column-start: 3;

    grid-column-end: 4;

}

.img\_div .pic3{

    grid-column-start: 2;

}

.event\_para{

    width: 90%;

    margin-left: 100px;

    margin-right: 100px;

    text-align: justify;

    font-size: 20px;

    line-height: 1.5em;

}

.indexing\_data p{

    width: 90%;

    margin-left: 100px;

    margin-right: 100px;

    text-align: justify;

    font-size: 20px;

}

.indexing\_data ul{

    width: 90%;

    margin-left: 100px;

    margin-right: 100px;

    text-align: justify;

    font-size: 16px;

}

.committee\_container{

    width: 90%;

    margin-left: 100px;

    margin-right: 100px;

    display: grid;

    gap: 20px;

}

.box1{

    display: grid;

    grid-template-columns: repeat(3,1fr);

    gap:20px;

}

.inner{

    display: grid;

    justify-content: center;

    align-items: center;

}

.inner .patron{

    color:#0e1b4d;

    text-align: center;

}

.inner h3{

    color:red;

    text-align: center;

    margin-bottom: 5px;

}

.inner p{

    text-align: center;

    margin-top: 0;

}

.committee\_container img{

    border-radius: 50%;

}

.keynote\_container{

    width: 90%;

    margin-left: 100px;

    margin-right: 100px;

    display: grid;

    gap: 20px;

}

.keybox{

    display: grid;

    grid-template-columns: repeat(3,1fr);

    gap:20px;

}

.key\_inner{

    display: grid;

    justify-content: center;

    align-items: center;

}

.key\_inner .patron{

    color:#0e1b4d;

    text-align: center;

}

.key\_inner h3{

    color:red;

    text-align: center;

    margin-bottom: 5px;

}

.key\_inner p{

    text-align: center;

    margin-top: 0;

    margin-bottom: 0px;

}

.keynote\_container a{

    color:blue;

    cursor: pointer;

    text-align: center;

    margin-top: 0;

}

.keynote\_container img{

    border-radius: 50%;

    margin-left: 140px;

}

.keynote\_container #zm{

    margin-left: 110px;

}

.tech-comitee\_heading{

    margin-top: 50px;

}

.tech-comitee{

    display: flex;

    line-height: 1.5em;

}

#tc\_p1{

    flex:1;

    margin-left: 250px;

    text-align: justify;

}

#tc\_p2{

    flex:1;

    margin-left: 20px;

    margin-right: 20px;

    text-align: justify;

}

#tc\_p3{

    flex:1;

    margin-right: 250px;

    text-align: justify;

}

.editors{

    margin: auto;

    width: 50%;

}

.program\_comitee\_heading{

    margin-top: 50px;

}

.program\_comitee{

    margin: auto;

    width: 50%;

}

hr{

    margin-top: 60px;

}

.pc{

    width: 100%;

    height: 240px;

    margin: auto;

    display: flex;

    justify-content: center;

    align-items: center;

    position: absolute;

}

.pc .dot{

    height: 20px;

    width: 20px;

    background-color: #0e1b4d;

    border-radius: 50%;

    position: absolute;

}

.pc .dot:nth-child(1) {

    left: 25%;

}

.pc .dot:nth-child(2) {

    left: 35%;

}

.pc .dot:nth-child(3) {

    left: 45%;

}

.pc .dot:nth-child(4) {

    left: 55%;

}

.pc .dot:nth-child(5) {

    left: 65%;

}

.pc .dot:nth-child(6) {

    left: 75%;

}

.pc .line{

    background-color:rgb(60, 60, 223) ;

    height: 2px;

    width: 52%;

}

#icc1{

    margin-left:24.5%;

    margin-top: 11%;

}

#icc2{

    margin-left: 34.7%;

    margin-top: -9%;

}

#icc3{

    margin-left: 45%;

    margin-top: 2.8%;

}

#icc4{

    margin-left: 55%;

    margin-top: -9%;

}

#icc5{

    margin-left: 65%;

    margin-top: 2.8%;

}

#icc6{

    margin-left: 75%;

    margin-top: -8.9%;

}

#dummy{

    margin-left: 30%;

    margin-top: 6%;

    visibility: hidden;

}

.call\_papers\_para1{

    width: 90%;

    margin-left: 100px;

    margin-right: 100px;

    text-align: justify;

    font-size: 20px;

    line-height: 1.5em;

}

span{

    color: red;

}

a{

    text-decoration: none;

}

span:hover{

    cursor: pointer;

}

.call\_papers\_para2{

    width: 90%;

    margin-left: 100px;

    margin-right: 100px;

    text-align: justify;

    font-size: 24px;

    line-height: 1.5em;

}

table{

    margin: auto;

    border: 3px solid black;

    border-collapse: collapse;

    width:35%;

}

tr{

    height: 35px;

    font-size: 16px;

}

th{

    background-color: #0e1b4d;

    color : white;

}

td{

    border: 1px solid #CCCCB3;

    padding-left: 8px;

    font-size: 20px;

}

.strike{

    text-decoration: line-through;

}

#closedBold{

    font-weight: bold;

}

@keyframes blink{

    50%{

        opacity: 0

    }

}

.blink{

    font-weight: bold;

    animation: blink 1s linear infinite;

}

.reg\_para p{

    text-align: center;

    font-weight: bold;

    font-size: 20px;

}

.table2{

    margin: auto;

    border: 3px solid black;

    border-collapse: collapse;

    width:60%;

}

.table2 tr{

    height: 35px;

    font-size: 16px;

}

.table2 th{

    background-color: #0e1b4d;

    color : white;

}

.table2 td{

    border: 1px solid #CCCCB3;

    padding-left: 8px;

    font-size: 20px;

    text-align: center;

}

.reg\_para2 {

    margin-top: 45px;

    text-align: center;

    font-size: 20px;

}

.email{

    text-align: center;

    color:#0e1b4d;

}

.copyright{

    background-color: #0e1b4d;

    text-align: center;

    font-size: 14px;

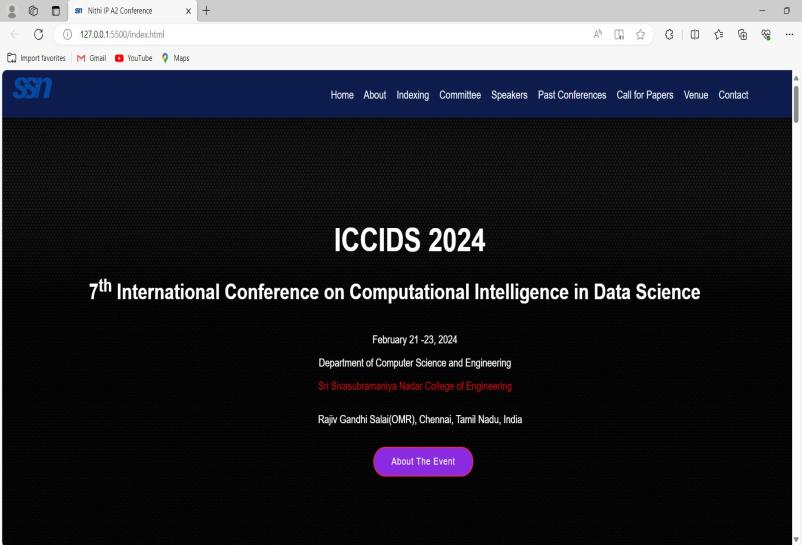
    padding-top: 15px;

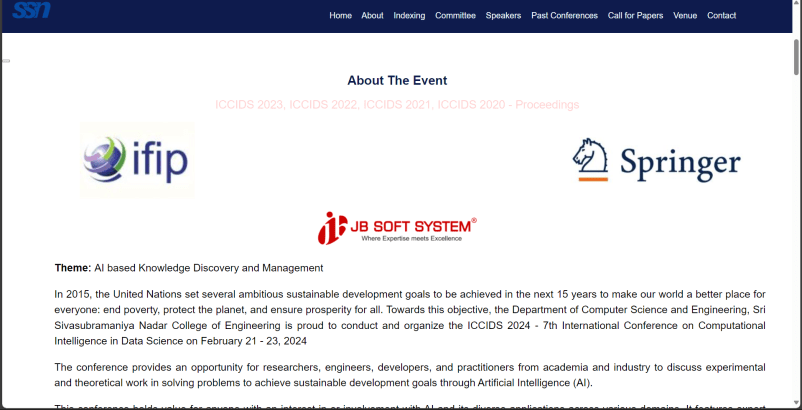
    color:white;

    height: 80px;

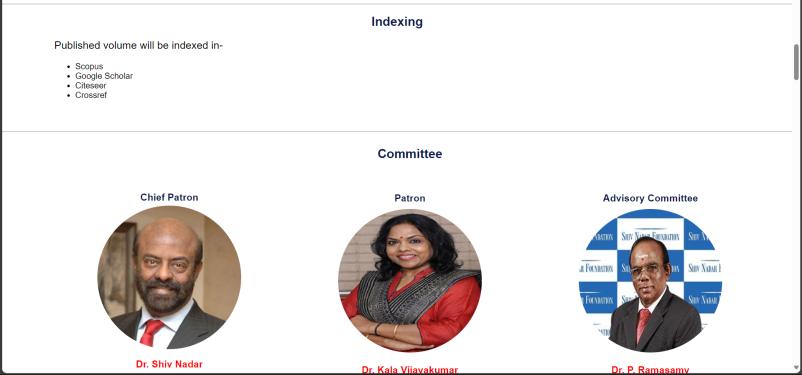
}

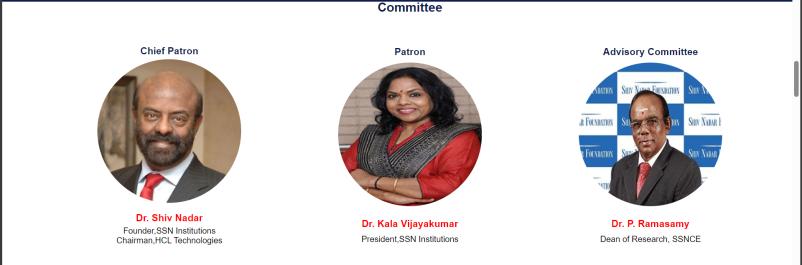
Output:-







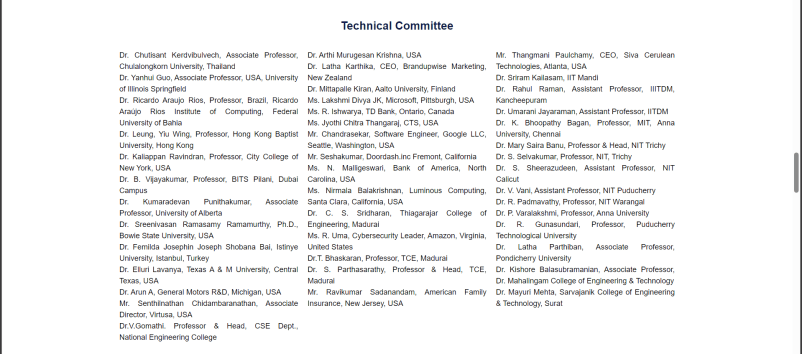




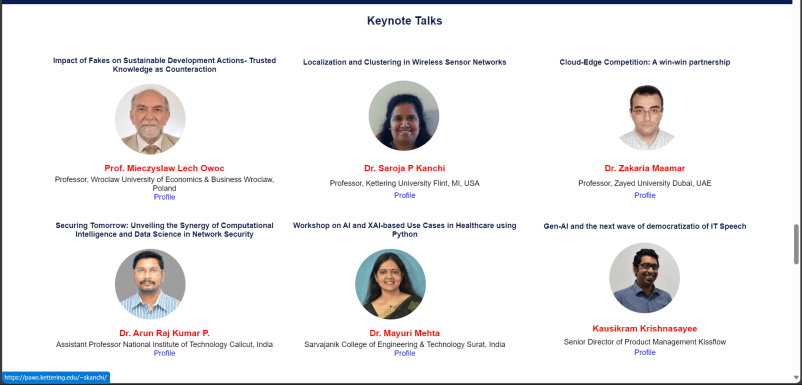


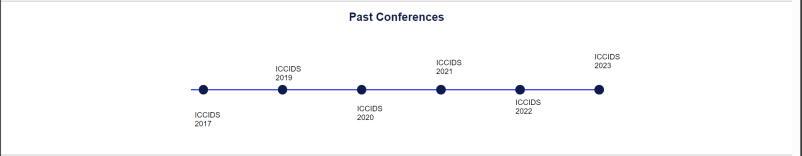


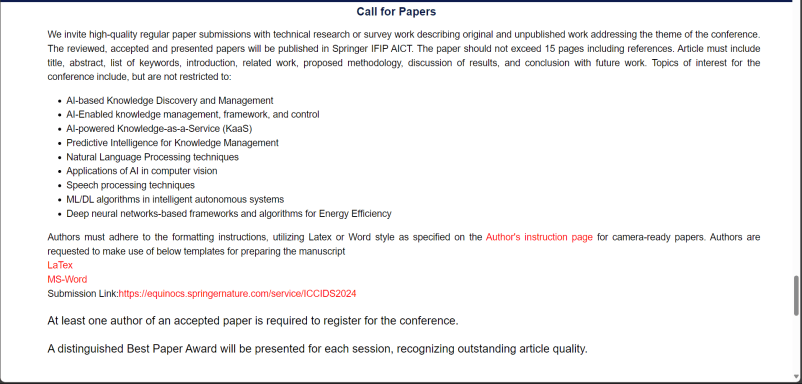




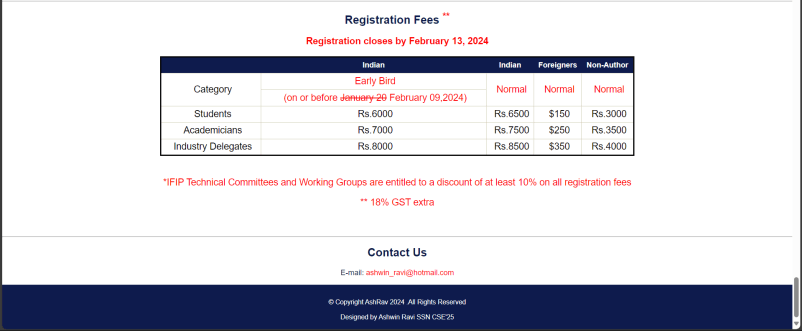












Best practices:-

Learning outcomes:-

**UCS 2611 Internet Programming Lab**

**Exercise 3: OMG: Online Memory Game for Kids**

**Date of Exercise: 10.02.2024**

**Goal**

To design an interactive online memory game for the kids to play with. [CO1, K3]

**Impact**

Memory games for kids build thinking skills, concentration, attention and persistence.

**Technologies to be Used**

HTML 5, CSS 3, Javascript

Kids can choose different boards with the options for choosing number, shape, picture or word. Various instances of OMG boards are as follows:

|  |  |
| --- | --- |
|  |  |
|  |  |

## How the Game Works?

The game uses a grid of images that all begin by showing question marks. The table must have an even number of squares.

|  |  |  |  |
| --- | --- | --- | --- |
| ? | ? | ? | ? |
| ? | ? | ? | ? |
| ? | ? | ? | ? |

When the user clicks on one of the squares marked with a ? the question mark is replaced with a word or image. Clicking on another square will reveal another word or image. If the images match they remain visible for the duration of the game. If they do not match, no further hidden squares can be seen unless at least one of the revealed images is turned over again by clicking on it. In other words, only two non-matching images can be seen at one time. In the following illustration

|  |  |  |  |
| --- | --- | --- | --- |
| ? | ? | the | ? |
| ? | Then | their | ? |
| ? | ? | ? | the |

clicking on additional ? squares has no effect because two non-matching squares (then and their) are showing. Clicking on either square with the word "the" in it has no effect because these squares are matched. Clicking on the "then" or the "their" square will "turn them over" so that only a ? is displayed. When all the squares are matched the game is over and the number of clicks on ? squares to reach the end is reported For Every new Game the image or word matching the square has to be randomly changed.

## Rules to be enforced for this game:

* No more than two unmatched squares can be turned over at any time;
* Matched squares cannot be turned over;
* Every click on a question mark that results in a square being turned over must be counted as a "turn";
* The game is over when all the squares are matched and therefore visible on the board.

Best Practices to be followed:

1. Design before coding
2. Incremental coding
3. Usage of proper naming convention
4. Usage of Comments to the code
5. Indentation of code

**Exercise 3: OMG: Online Memory Game for Kids**

**Design:-**

Code:-

Index.html

<html lang="en" dir="ltr">

<head>

<meta charset="utf-8">

<title>Memory game</title>

<link rel="stylesheet" href="style.css">

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

<link rel="stylesheet" href="https://fonts.googleapis.com/icon?family=Material+Icons">

</head>

<body>

<div class="wrapper">

<h3>Iron Man Memory Game</h3>

<ul class="cards">

<li class="card">

<div class="view front-view">

<span class="material-icons">question\_mark</span>

</div>

<div class="view back-view">

<img src="images/img1.jpg" alt="card-img">

</div>

</li>

<li class="card">

<div class="view front-view">

<span class="material-icons">question\_mark</span>

</div>

<div class="view back-view">

<img src="images/img2.jpg" alt="card-img">

</div>

</li>

<li class="card">

<div class="view front-view">

<span class="material-icons">question\_mark</span>

</div>

<div class="view back-view">

<img src="images/img3.jpg" alt="card-img">

</div>

</li>

<li class="card">

<div class="view front-view">

<span class="material-icons">question\_mark</span>

</div>

<div class="view back-view">

<img src="images/img4.jpg" alt="card-img">

</div>

</li>

<li class="card">

<div class="view front-view">

<span class="material-icons">question\_mark</span>

</div>

<div class="view back-view">

<img src="images/img5.jpg" alt="card-img">

</div>

</li>

<li class="card">

<div class="view front-view">

<span class="material-icons">question\_mark</span>

</div>

<div class="view back-view">

<img src="images/img6.jpg" alt="card-img">

</div>

</li>

<li class="card">

<div class="view front-view">

<span class="material-icons">question\_mark</span>

</div>

<div class="view back-view">

<img src="images/img7.jpg" alt="card-img">

</div>

</li>

<li class="card">

<div class="view front-view">

<span class="material-icons">question\_mark</span>

</div>

<div class="view back-view">

<img src="images/img8.jpg" alt="card-img">

</div>

</li>

<li class="card">

<div class="view front-view">

<span class="material-icons">question\_mark</span>

</div>

<div class="view back-view">

<img src="images/img1.jpg" alt="card-img">

</div>

</li>

<li class="card">

<div class="view front-view">

<span class="material-icons">question\_mark</span>

</div>

<div class="view back-view">

<img src="images/img2.jpg" alt="card-img">

</div>

</li>

<li class="card">

<div class="view front-view">

<span class="material-icons">question\_mark</span>

</div>

<div class="view back-view">

<img src="images/img3.jpg" alt="card-img">

</div>

</li>

<li class="card">

<div class="view front-view">

<span class="material-icons">question\_mark</span>

</div>

<div class="view back-view">

<img src="images/img4.jpg" alt="card-img">

</div>

</li>

<li class="card">

<div class="view front-view">

<span class="material-icons">question\_mark</span>

</div>

<div class="view back-view">

<img src="images/img5.jpg" alt="card-img">

</div>

</li>

<li class="card">

<div class="view front-view">

<span class="material-icons">question\_mark</span>

</div>

<div class="view back-view">

<img src="images/img6.jpg" alt="card-img">

</div>

</li>

<li class="card">

<div class="view front-view">

<span class="material-icons">question\_mark</span>

</div>

<div class="view back-view">

<img src="images/img7.jpg" alt="card-img">

</div>

</li>

<li class="card">

<div class="view front-view">

<span class="material-icons">question\_mark</span>

</div>

<div class="view back-view">

<img src="images/img8.jpg" alt="card-img">

</div>

</li>

</ul>

</div>

<div class="popup\_overlay"></div>

<div class="popup\_box">

<h1>Congratulations!</h1>

<h3>You finally reached the end</h3>

<img src="images/ironmansnap.jpg">

<p id="flip\_text"></p>

<button id="ok\_button">OK</button>

</div>

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

</body>

</html>

script.js

const cards = document.querySelectorAll(".card");

let matched = 0;

let cardOne, cardTwo;

let disableDeck = false;

let flips = 0;

function flipCard({target: clickedCard}) {

if(cardOne !== clickedCard && !disableDeck) {

clickedCard.classList.add("flip");

flips++;

if(!cardOne) {

return cardOne = clickedCard;

}

cardTwo = clickedCard;

disableDeck = true;

let cardOneImg = cardOne.querySelector(".back-view img").src,

cardTwoImg = cardTwo.querySelector(".back-view img").src;

matchCards(cardOneImg, cardTwoImg);

}

}

let popup\_bg = document.querySelector('.popup\_overlay');

let popup\_display = document.querySelector('.popup\_box');

function matchCards(img1, img2) {

if(img1 === img2) {

matched++;

if(matched == 8) {

setTimeout(() => {

popup\_display.style.display = "block";

popup\_bg.style.display = "block";

let no\_of\_flips = `You've completed the game in ${flips} flips.`;

document.getElementById('flip\_text').innerText = no\_of\_flips; // Assuming you have an element with id 'popup\_text'

return shuffleCard();

}, 1000);

}

cardOne.removeEventListener("click", flipCard);

cardTwo.removeEventListener("click", flipCard);

cardOne = cardTwo = "";

return disableDeck = false;

}

setTimeout(() => {

cardOne.classList.add("shake");

cardTwo.classList.add("shake");

}, 400);

setTimeout(() => {

cardOne.classList.remove("shake", "flip");

cardTwo.classList.remove("shake", "flip");

cardOne = cardTwo = "";

disableDeck = false;

}, 1200);

}

let btn\_close = document.getElementById('ok\_button');

btn\_close.addEventListener('click',function(){

popup\_display.style.display = "none";

popup\_bg.style.display = "none";

})

function shuffleCard() {

matched = 0;

disableDeck = false;

cardOne = cardTwo = "";

let arr = [1, 2, 3, 4, 5, 6, 7, 8, 1, 2, 3, 4, 5, 6, 7, 8];

arr.sort(() => Math.random() > 0.5 ? 1 : -1);

cards.forEach((card, i) => {

card.classList.remove("flip");

let imgTag = card.querySelector(".back-view img");

imgTag.src = `images/img${arr[i]}.jpg`;

card.addEventListener("click", flipCard);

});

}

shuffleCard();

cards.forEach(card => {

card.addEventListener("click", flipCard);

});

style.css

@import url('https://fonts.googleapis.com/css2?family=Poppins:wght@400;500;600;700&display=swap');

\* {

    margin: 0;

    padding: 0;

    box-sizing: border-box;

    font-family: 'Poppins', sans-serif;

}

body {

    display: flex;

    align-items: center;

    justify-content: center;

    min-height: 100vh;

    background-color:  #4b0908;

}

h3 {

    margin-bottom: 20px;

    text-align: center;

}

.wrapper {

    padding: 25px;

    border-radius: 10px;

    background: #b97d10;

    box-shadow: 0 10px 30px rgba(0, 0, 0, 0.1);

}

.cards,

.card,

.view {

    display: flex;

    align-items: center;

    justify-content: center;

}

.cards {

    height: 400px;

    width: 400px;

    flex-wrap: wrap;

    justify-content: space-between;

}

.cards .card {

    cursor: pointer;

    list-style: none;

    user-select: none;

    position: relative;

    perspective: 1000px;

    transform-style: preserve-3d;

    height: calc(100% / 4 - 10px);

    width: calc(100% / 4 - 10px);

}

.card.shake {

    animation: shake 0.35s ease-in-out;

}

@keyframes shake {

    0%, 100% {

        transform: translateX(0);

    }

    20% {

        transform: translateX(-13px);

    }

    40% {

        transform: translateX(13px);

    }

    60% {

        transform: translateX(-8px);

    }

    80% {

        transform: translateX(8px);

    }

}

.card .view {

    width: 100%;

    height: 100%;

    position: absolute;

    border-radius: 7px;

    background: #fbca03;

    pointer-events: none;

    backface-visibility: hidden;

    box-shadow: 0 3px 10px rgba(0, 0, 0, 0.1);

    transition: transform 0.25s linear;

}

.card .front-view img {

    width: 19px;

}

.card .back-view img {

    max-width: 90px;

}

.card .back-view {

    transform: rotateY(-180deg);

}

.card.flip .back-view {

    transform: rotateY(0);

}

.card.flip .front-view {

    transform: rotateY(180deg);

}

.popup\_overlay {

    background-color: black;

    opacity: 0.8;

    position: absolute;

    width: 100%;

    height: 100%;

    top: 0;

    left: 0;

    z-index: 1;

    display: none;

}

.popup\_box {

    background-color: #6a060b;

    width: 30%;

    height: 360px;

    padding-top: 45px;

    padding-left: 40px;

    border-radius: 10px;

    position: absolute;

    top: 20%;

    left: 35%;

    z-index: 2;

    display: none;

}

@keyframes blinker {

    50% {

        opacity: 0;

    }

}

.popup\_box h1 {

    animation: blinker 1s linear infinite;

    text-align: center;

    color: red;

}

.popup\_box h3 {

    text-align: center;

}

.popup\_box img {

    display: center;

    width: 140px;

    height: 140px;

}

.popup\_box p {

    text-align: center;

}

.popup\_box button {

    background-color: black;

    color: white;

    padding-left: 15px;

    padding-right: 15px;

    border: none;

    padding-top: 5px;

    padding-bottom: 5px;

    margin: 20px;

    margin-left: 42%;

}

@media screen and (max-width: 700px) {

    .cards {

        height: 350px;

        width: 350px;

    }

    .card .front-view img {

        width: 17px;

    }

    .card .back-view img {

        max-width: 40px;

    }

}

@media screen and (max-width: 530px) {

    .cards {

        height: 300px;

        width: 300px;

    }

    .card .front-view img {

        width: 15px;

    }

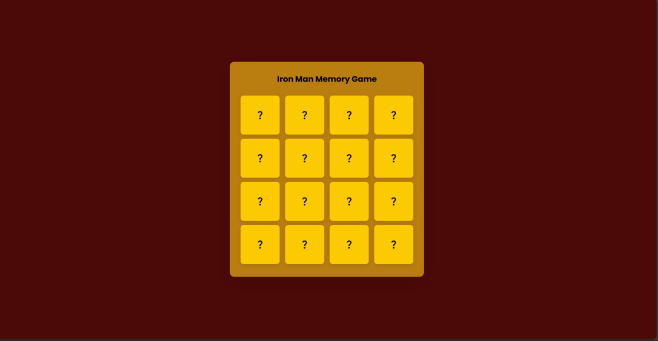
    .card .back-view img {

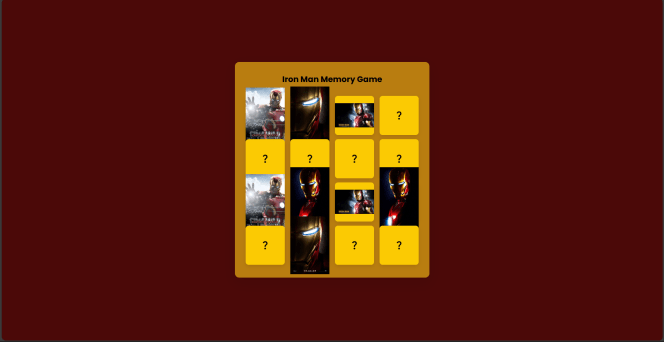
        max-width: 35px;

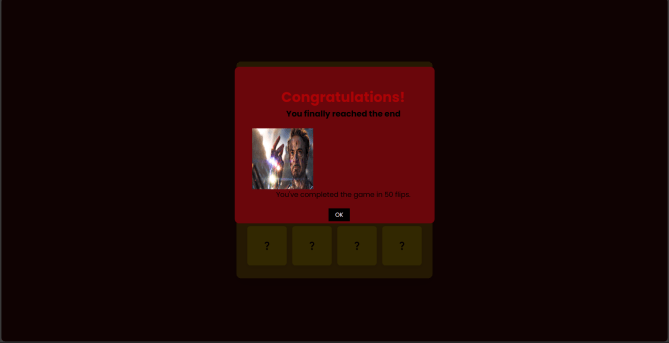
    }

}

Output:-







**UCS 2611 Internet Programming Lab**

**Exercise 4: HellServlet Application using Tomcat Server**

**Date of Exercise: 14.02.2024**

Write index.html to display form with a submit button. Set the method attribute of the form to “get” to generate GET request to the server. Write a java servlet to display welcome message when the button is clicked. Also write the configuration file, namely, web.xml which maps the url mapped to the servlet class file. [CO1,K3]

Follow the directory structure for creating HelloServlet Web application and follow the guidelines below to configure and deploy web application.



Assuming java is working on your laptop/desktop, To check, go to command prompt and type >javac

Download tomcat from tomcat official website Extract in to say C drive

**How to install tomcat? (For Windows)**

You need to create an *environment variable* called "JAVA\_HOME" and set it to your JDK installed

directory.

1. First, take note of your JDK installed directory. The default is "c:\Program

Files\Java\jdk1.8.0\_{*xx*}", where {*xx*} is the latest upgrade number. It is **important** to verify

your JDK installed directory, via the "Computer", before you proceed further.

2. Start a CMD shell, and issue the command "set JAVA\_HOME" to check if variable JAVA\_HOME

has been set:

3. > **set JAVA\_HOME**

Environment variable JAVA\_HOME not defined

If JAVA\_HOME is set, check if it is set to your JDK installed directory correctly. Otherwise,

goto next step.

4. To set the environment variable JAVA\_HOME in Windows 2000/XP/Vista/7/8: Push "Start"

button ⇒ Control Panel ⇒ System ⇒ (Vista/7/8) Advanced system settings ⇒ Switch to

"Advanced" tab ⇒ Environment Variables ⇒ System Variables ⇒ "New" (or "Edit" for

modification) ⇒ In "Variable Name", enter "JAVA\_HOME" ⇒ In "Variable Value", enter your

JDK installed directory (e.g., " c:\Program Files\Java\jdk1.8.0\_{xx}").

5. To verify, **RE-START** a CMD shell (need to refresh the environment) and issue:

6. > **set JAVA\_HOME**

JAVA\_HOME=c:\Program Files\Java\jdk1.8.0\_{*xx*} <== Verify that this is YOUR JDK installed

directory

**How to set class path?**

1. Go to tomcat home directory and browse to …/lib/

2. Copy the path

3. Go to environment variables and under system variables create a new entry with

Variable name = CLASSPATH

Variable value = <the url you have just copied>

4. Save and exit.

**How to install tomcat? (For Ubuntu)**

* In Google, Type Apache Tomcat Installation Ubuntu 16.04 and find the liquid website
* Download tar file of Apache Tomcat 8.5.50 from Liquidweb site
* Extract and place it in /usr/local
* Check the version of java and its availability
* Usually we find in /usr/lib/jvm
* Need to set JAVA\_HOME, PATH & CATALINA\_HOME in bashrc file
* bashrc is a hidden file in home folder (~)
* Open the bashrc file using nano editor
* sudo nano ~/.bashrc
* Add the following
  + JAVA\_HOME=“/usr/lib/jvm/java-1.8-openjdk-amd64”
  + PATH=$PATH:$JAVA\_HOME
  + CATALINA\_HOME=“/usr/local/apache-tomcat=8.5.50
* Update bashrc file using any one of the commands
  + source ~/.bashrc
  + . ~/.bashrc
* Check the environment variables by the following commands
  + $JAVA\_HOME
  + $PATH
  + $CATALINA\_HOME
* Start the server
  + $CATALINA\_HOME/startup.sh
* Type the following in browser to check the administration page of Tomcat
  + http://localhost:8080
* Take demo Servlet from the following link
  + <https://www.javatpoint.com/steps-to-create-a-servlet-using-tomcat-server>
* Extract the folder, you will get demo servlet
* Place the demo servlet in webapps folder in apache-tomcat-8.5.50
* Type the following in browser to check demo servlet
  + <http://localhost:8080/demo>
* Shutdown the server
  + $CATALINA\_HOME/bin/shutdown.sh

**How to run the web application? (In windows)**

1. Copy the HelloServlet application in webapps folder

2. Navigate to tomcat directory, go to bin

3. Execute startup.bat

4. Wait for the server to start

5. Open browser and navigate to the url : localhost:8080

6. A tomcat administartion page will be displayed.

7. To run firstservlet navigate to the url : <http://localhost:8080/HelloServlet>

**How to run the web application? (In Ubuntu)**

1. Copy the HelloServlet application in webapps folder

2. In terminal, type $CATALINA\_HOME/bin/startup.sh3.

3. Wait for the server to start

4. Open browser and navigate to the url : localhost:8080

5. A tomcat administration page will be displayed.

6. To run firstservlet navigate to the url : <http://localhost:8080/HelloServlet>

Best Practices to be followed:

1. Design before coding
2. Incremental coding
3. Usage of proper naming convention
4. Usage of Comments to the code
5. Indentation of code

**A4:- HelloServlet Web Application using Tomcat Server**

**Code:-**

**index.html**

<html>

<head>

<title>My First Web Application</title>

</head>

<body>

<form action="hello" method="get">

<button type="submit">Click Me</button>

</form>

</body>

</html>

<web-app>

**web.xml**

<servlet>

<servlet-name>HelloServlet</servlet-name>

<servlet-class>HelloServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>HelloServlet</servlet-name>

<url-pattern>/hello</url-pattern>

</servlet-mapping>

<welcome-file-list>

<welcome-file>index.html</welcome-file>

</welcome-file-list>

</web-app>

**HelloServlet.java**

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

public class HelloServlet extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws IOException, ServletException

{

response.setContentType("text/html");

PrintWriter out = response.getWriter();

out.println("<html>");

out.println("<head>");

out.println("<title>Hello World!</title>");

out.println("</head>");

out.println("<body>");

out.println("<h1>Hello World!</h1>");

out.println("</body>");

out.println("</html>");

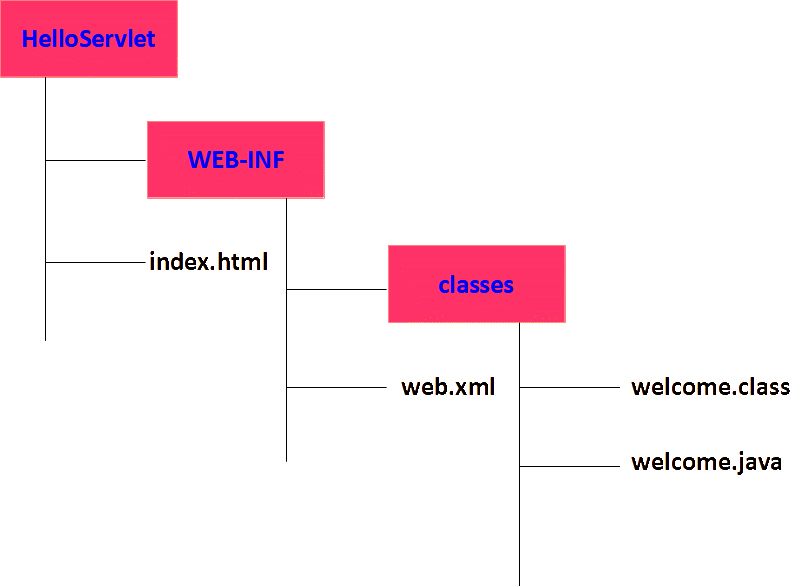
}

}

**Output:-**

****

**Things to be taken care of:-**



1. First, take note of your JDK installed directory. The default is "c:\Program

Files\Java\jdk1.8.0\_{*xx*}", where {*xx*} is the latest upgrade number. It is **important** to verify

your JDK installed directory, via the "Computer", before you proceed further.

2. Start a CMD shell, and issue the command "set JAVA\_HOME" to check if variable JAVA\_HOME

has been set:

3. > **set JAVA\_HOME**

Environment variable JAVA\_HOME not defined

If JAVA\_HOME is set, check if it is set to your JDK installed directory correctly. Otherwise,

goto next step.

4. To set the environment variable JAVA\_HOME in Windows 2000/XP/Vista/7/8: Push "Start"

button ⇒ Control Panel ⇒ System ⇒ (Vista/7/8) Advanced system settings ⇒ Switch to

"Advanced" tab ⇒ Environment Variables ⇒ System Variables ⇒ "New" (or "Edit" for

modification) ⇒ In "Variable Name", enter "JAVA\_HOME" ⇒ In "Variable Value", enter your

JDK installed directory (e.g., " c:\Program Files\Java\jdk1.8.0\_{xx}").

5. To verify, **RE-START** a CMD shell (need to refresh the environment) and issue:

6. > **set JAVA\_HOME**

JAVA\_HOME=c:\Program Files\Java\jdk1.8.0\_{*xx*} <== Verify that this is YOUR JDK installed

directory

**How to set class path?**

1. Go to tomcat home directory and browse to …/lib/

2. Copy the path

3. Go to environment variables and under system variables create a new entry with

Variable name = CLASSPATH

Variable value = <the url you have just copied>

4. Save and exit.

**How to run the web application?**

1. Copy the HelloServlet application in webapps folder

2. Navigate to tomcat directory, go to bin

3. Execute startup.bat

4. Wait for the server to start

5. Open browser and navigate to the url : localhost:8080

6. A tomcat administartion page will be displayed.

7. To run firstservlet navigate to the url : <http://localhost:8080/HelloServlet>

Best Practices:-

Learning outcomes:-

**UCS 2611 Internet Programming Lab**

**Exercise 5: DatabaseServlet Application using Tomcat Server**

**Date of Exercise: 14.02.2024**

Write index.html to display form with a submit button. Set the method attribute of the form to “get” to generate GET request to the server. Write a java servlet to display the records of the table when the button is clicked. Also write the configuration file, namely, web.xml which maps the url mapped to the servlet class file. [CO1,K3]

Follow the directory structure for creating DatabaseServlet Web application and follow the guidelines below to configure and deploy web application.



**To install mysql server in ubuntu 16.04 environment**

To install mysql server

sudo apt-get update

sudo apt-get install mysql-server

mysql\_secure\_installation

To check the mysql service

systemctl status mysql.service

To start and stop the service

service mysql start

service mysql stop

To get into the mysql prompt

mysql –u root –p

To clear the screen in mysql prompt

system clear

To exit from the mysql prompt

Exit

In /usr/share/java, there won’t be any mysql connector files available. In order to run the servlet code with the database, we need to have connector files in tomcat/lib folder

Execute the following statement to get the connector files

apt-get install Libmysql-java

You will have mysql.jar, mysql-connector-java.jar, mysql-connector-java-5.1.38.jar in /usr/share/java folder. Copy them and put it in tomcat/lib folder.

Start the server using $CATALINA\_HOME/bin/startup.sh

To run DatabaseServlet navigate to the url : <http://localhost:8080/DatabaseServlet>

Stop the server using $CATALINA\_HOME/bin/shutdown.sh

**To install mysql server in windows environment**

Use **mysql-5.5.30-win32** to install mysql

Set the path of mysql in environment variables

While installing, provide user name and password

Copy the **mysql-connector-java-3.1.14-bin** jar file to “C:\Program Files\Apache Software Foundation\Tomcat 7.0\lib”

To run mysql go to Start🡪All Program🡪 MySqL🡪MySql5.5 Command Line Client and double

click it

mysql> use test;

mysql> create table Employess (id int, age int, first varchar(20), last varchar(20));

mysql> insert into Employess (101, 20, “Zara”,”Ali”);

mysql> select \* from Employees;

**How to run the Web application?**

1. Copy the DatabaseServlet application in webapps folder

2. Navigate to tomcat directory, go to bin

3. Execute startup.bat

4. Wait for the server to start

5. Open browser and navigate to the url : localhost:8080

6. A tomcat demo page will be displayed.

7. To run firstservlet navigate to the url : <http://localhost:8080/DatabaseServlet>

Best Practices to be followed:

1. Design before coding
2. Incremental coding
3. Usage of proper naming convention
4. Usage of Comments to the code
5. Indentation of code

**A5:- DatabaseServlet Web Application using Tomcat Server**

**Design:-**

**Code:-**

# index.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Student Web App</title>

<style>

body{

height: 100%;

width: 100%; display: flex;

align-items: center; justify-content: center;

background-color: lightcoral;

}

.content{ height: 70%; width: 400px;

background-color: red; padding: 5%;

}

h1{

text-align: center;

}

label{

padding: 10px; display: block;

}

button{

margin-left: 35%; padding: 4px 30px; margin-top: 20px;

}

</style>

</head>

<body>

<section class="content">

<h1>LOGIN</h1>

<form action="login" method="get">

<label for="name">Username: <input type="text" name="username"></label>

<label for="age">Password: <input type="text" name="password"></label>

<button type="submit">Login</button>

</form>

</section>

</body>

</html>

# web.xml

<web-app>

<servlet>

<servlet-name>Main</servlet-name>

<servlet-class>Main</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>Main</servlet-name>

<url-pattern>/login</url-pattern>

</servlet-mapping>

<welcome-file-list>

<welcome-file>index.html</welcome-file>

</welcome-file-list>

</web-app>

# Main.java

import java.io.\*;

import java.sql.Connection; import java.sql.DriverManager;

import java.sql.PreparedStatement; import java.sql.ResultSet;

import java.sql.SQLException; import javax.servlet.\*;

import javax.servlet.http.\*;

public class Main extends HttpServlet {

private static final String URL = "jdbc:mysql://localhost:3306/authwebapp"; private static final String USERNAME = "user1";

private static final String PASSWORD = "10Red@pple";

static {

try {

Class.forName("com.mysql.cj.jdbc.Driver");

} catch (ClassNotFoundException e) {

throw new RuntimeException("Error loading MySQL JDBC driver", e);

}

}

public void doGet(HttpServletRequest request, HttpServletResponse response) throws IOException, ServletException {

response.setContentType("text/html"); PrintWriter out = response.getWriter();

String username = request.getParameter("username"); String password = request.getParameter("password");

try (Connection connection = DriverManager.getConnection(URL, USERNAME, PASSWORD)) {

String sql = "SELECT \* FROM myusers WHERE username = ? AND passwd = ?"; try (PreparedStatement statement = connection.prepareStatement(sql)) {

statement.setString(1, username); statement.setString(2, password);

try (ResultSet resultSet = statement.executeQuery()) { if (resultSet.next()) {

out.println("<html><head><title>Login Success</title></head><body>"); out.println("<h2>Login Success</h2>");

out.println("<h2>Hello " + username + "!</h2>"); out.println("</body></html>");

} else {

out.println("<html><head><title>User Not Found</title></head><body>"); out.println("<h2>User Not Found</h2>");

out.println("</body></html>");

}

}

}

} catch (SQLException e) {

out.println("<html><head><title>Database Error</title></head><body>"); out.println("<h2>Database Error</h2>");

out.println("<p>" + e.getMessage() + "</p>"); out.println("</body></html>");

}

}

}

# SQL Query:

CREATE TABLE myusers( username VARCHAR(255), passwd VARCHAR(255)

);

SELECT \* FROM myusers;

INSERT INTO myusers VALUES('karthikeyan','password'); INSERT INTO myusers VALUES('Testing','testing');

SELECT \* FROM myusers;

**To install mysql server in windows environment**

Use **mysql-5.5.30-win32** to install mysql

Set the path of mysql in environment variables

While installing, provide user name and password

Copy the **mysql-connector-java-3.1.14-bin** jar file to “C:\Program Files\Apache Software Foundation\Tomcat 7.0\lib”

To run mysql go to Start🡪All Program🡪 MySqL🡪MySql5.5 Command Line Client and double

click it

mysql> use test;

mysql> create table Employess (id int, age int, first varchar(20), last varchar(20));

mysql> insert into Employess (101, 20, “Zara”,”Ali”);

mysql> select \* from Employees;

**How to run the Web application?**

1. Copy the DatabaseServlet application in webapps folder

2. Navigate to tomcat directory, go to bin

3. Execute startup.bat

4. Wait for the server to start

5. Open browser and navigate to the url : localhost:8080

6. A tomcat demo page will be displayed.

7. To run firstservlet navigate to the url : <http://localhost:8080/DatabaseServlet>

Best practices:-

Learning outcomes:-

**UCS 2611 Internet Programming Lab**

**Exercise 6: Session Tracking using Servlet**

Write a java servlet to store and retrieve the value of the variable using the following session tracking mechanisms: [CO1, K3]

* HttpSession object
* Cookies
* Hidden form field
* URL Rewriting

Best Practices to be followed:

1. Design before coding
2. Incremental coding
3. Usage of proper naming convention
4. Usage of Comments to the code
5. Indentation of code

**A6**- **Session Tracking using Servlet**

**Design:-**

HttpSession object

<form action="servlet1" method="post">

Name:<input type="text" name="userName"/><br/>

<input type="submit" value="go"/>

</form>

    <web-app>

    <servlet>

    <servlet-name>s1</servlet-name>

    <servlet-class>FirstServlet</servlet-class>

    </servlet>

    <servlet-mapping>

    <servlet-name>s1</servlet-name>

    <url-pattern>/servlet1</url-pattern>

    </servlet-mapping>

    <servlet>

    <servlet-name>s2</servlet-name>

    <servlet-class>SecondServlet</servlet-class>

    </servlet>

    <servlet-mapping>

    <servlet-name>s2</servlet-name>

    <url-pattern>/servlet2</url-pattern>

    </servlet-mapping>

    </web-app>

    import java.io.\*;

    import javax.servlet.\*;

    import javax.servlet.http.\*;

    public class FirstServlet extends HttpServlet {

      public void doPost(HttpServletRequest request, HttpServletResponse response){

        try{

        response.setContentType("text/html");

        PrintWriter out = response.getWriter();

        String n=request.getParameter("userName");

        out.print("Welcome "+n);

        Cookie ck=new Cookie("uname",n);//creating cookie object

        response.addCookie(ck);//adding cookie in the response

        //creating submit button

        out.print("<form action='servlet2'>");

        out.print("<input type='submit' value='go'>");

        out.print("</form>");

        out.close();

            }catch(Exception e){System.out.println(e);}

      }

    }

    import java.io.\*;

    import javax.servlet.\*;

    import javax.servlet.http.\*;

    public class SecondServlet extends HttpServlet {

    public void doPost(HttpServletRequest request, HttpServletResponse response){

        try{

        response.setContentType("text/html");

        PrintWriter out = response.getWriter();

        Cookie ck[]=request.getCookies();

        out.print("Hello "+ck[0].getValue());

        out.close();

             }catch(Exception e){System.out.println(e);}

        }

    }

Output:-



URL Servlet

    <form action="servlet1">

    Name:<input type="text" name="userName"/><br/>

    <input type="submit" value="go"/>

    </form>

<web-app>

<servlet>

<servlet-name>s1</servlet-name>

<servlet-class>FirstServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>s1</servlet-name>

<url-pattern>/servlet1</url-pattern>

</servlet-mapping>

<servlet>

<servlet-name>s2</servlet-name>

<servlet-class>SecondServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>s2</servlet-name>

<url-pattern>/servlet2</url-pattern>

</servlet-mapping>

</web-app>

    import java.io.\*;

    import javax.servlet.\*;

    import javax.servlet.http.\*;

    public class FirstServlet extends HttpServlet {

    public void doGet(HttpServletRequest request, HttpServletResponse response){

            try{

            response.setContentType("text/html");

            PrintWriter out = response.getWriter();

            String n=request.getParameter("userName");

            out.print("Welcome "+n);

            //appending the username in the query string

            out.print("<a href='servlet2?uname="+n+"'>visit</a>");

            out.close();

                    }catch(Exception e){System.out.println(e);}

        }

    }

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

public class SecondServlet extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response){

        try{

          response.setContentType("text/html");

          PrintWriter out = response.getWriter();

         //getting value from the query string

          String n=request.getParameter("uname");

          out.print("Hello "+n);

          out.close();

                  }catch(Exception e){System.out.println(e);}

    }

}

Output:-



Session Servlet

    <form action="servlet1">

    Name:<input type="text" name="userName"/><br/>

    <input type="submit" value="go"/>

    </form>

    <web-app>

    <servlet>

    <servlet-name>s1</servlet-name>

    <servlet-class>FirstServlet</servlet-class>

    </servlet>

    <servlet-mapping>

    <servlet-name>s1</servlet-name>

    <url-pattern>/servlet1</url-pattern>

    </servlet-mapping>

    <servlet>

    <servlet-name>s2</servlet-name>

    <servlet-class>SecondServlet</servlet-class>

    </servlet>

    <servlet-mapping>

    <servlet-name>s2</servlet-name>

    <url-pattern>/servlet2</url-pattern>

    </servlet-mapping>

    </web-app>

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

public class FirstServlet extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response){

        try{

        response.setContentType("text/html");

        PrintWriter out = response.getWriter();

        String n=request.getParameter("userName");

        out.print("Welcome "+n);

        HttpSession session=request.getSession();

        session.setAttribute("uname",n);

        out.print("<a href='servlet2'>visit</a>");

        out.close();

                }catch(Exception e){System.out.println(e);}

    }

}

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

public class SecondServlet extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response){

        try{

        response.setContentType("text/html");

        PrintWriter out = response.getWriter();

        HttpSession session=request.getSession(false);

        String n=(String)session.getAttribute("uname");

        out.print("Hello "+n);

        out.close();

                }catch(Exception e){System.out.println(e);}

    }

}

Output:-



HiddenForm

    <form action="servlet1">

    Name:<input type="text" name="userName"/><br/>

    <input type="submit" value="go"/>

    </form>

    <web-app>

    <servlet>

    <servlet-name>s1</servlet-name>

    <servlet-class>FirstServlet</servlet-class>

    </servlet>

    <servlet-mapping>

    <servlet-name>s1</servlet-name>

    <url-pattern>/servlet1</url-pattern>

    </servlet-mapping>

    <servlet>

    <servlet-name>s2</servlet-name>

    <servlet-class>SecondServlet</servlet-class>

    </servlet>

    <servlet-mapping>

    <servlet-name>s2</servlet-name>

    <url-pattern>/servlet2</url-pattern>

    </servlet-mapping>

    </web-app>

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

public class FirstServlet extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response){

        try{

        response.setContentType("text/html");

        PrintWriter out = response.getWriter();

        String n=request.getParameter("userName");

        out.print("Welcome "+n);

        //creating form that have invisible textfield

        out.print("<form action='servlet2'>");

        out.print("<input type='hidden' name='uname' value='"+n+"'>");

        out.print("<input type='submit' value='go'>");

        out.print("</form>");

        out.close();

                }catch(Exception e){System.out.println(e);}

    }

}

    import java.io.\*;

    import javax.servlet.\*;

    import javax.servlet.http.\*;

    public class SecondServlet extends HttpServlet {

    public void doGet(HttpServletRequest request, HttpServletResponse response) {

            try{

            response.setContentType("text/html");

            PrintWriter out = response.getWriter();

            //Getting the value from the hidden field

            String n=request.getParameter("uname");

            out.print("Hello "+n);

            out.close();

                    }catch(Exception e){System.out.println(e);

                    }

        }

    }

Output:-





Best practices:-

Learning Outcomes:-

**UCS 2611 Internet Programming Lab**

**Exercise 7: Hello AJAX Application using Servlet**

Write index.html to retrieve the name of the user. Write a Ajax to refresh the index.html with the welcome message along with the name when the submit button is clicked. [CO1, K3]

Best Practices to be followed:

1. Design before coding
2. Incremental coding
3. Usage of proper naming convention
4. Usage of Comments to the code
5. Indentation of code

**A7:- Hello Ajax Application**

**Design:-**

Index.html

<html>

    <head>

        <script>

            function loadData() {

                var xhttp = new XMLHttpRequest();

                var n = document.getElementById("username").value;

                xhttp.onreadystatechange = function()

                {

                    if (this.readyState == 4 && this.status == 200)

                        document.getElementById("content").innerHTML=this.responseText;

                };

                xhttp.open("GET","hello?username="+n,true);

                xhttp.send();

            }

        </script>

    </head>

    <body>

        <h1 id="content"></h1>

        Name : <input type="text" id="username" oninput = "loadData()">

    </body>

</html>

Web.xml

<web-app>

    <servlet>

        <servlet-name>HelloServlet</servlet-name>

        <servlet-class>HelloServlet</servlet-class>

    </servlet>

    <servlet-mapping>

        <servlet-name>HelloServlet</servlet-name>

        <url-pattern>/hello</url-pattern>

    </servlet-mapping>

    <welcome-file-list>

    <welcome-file>index.html</welcome-file>

    </welcome-file-list>

</web-app>

HelloServlet.java

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

public class HelloServlet extends HttpServlet

{

    public void doGet(HttpServletRequest request, HttpServletResponse response) throws IOException, ServletException

    {

        response.setContentType("text/html");

        PrintWriter out = response.getWriter();

        String n = request.getParameter("username");

        out.println("<h4>" + n + " </h4>");

        out.close();

    }

}

Output:-



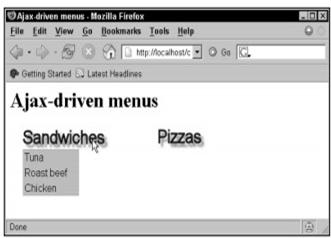
Best practices:-

Learning outcomes:-

**UCS 2611 Internet Programming Lab**

**Exercise 8: Popup Menu Application using AJAX and Servlet**

Develop an Application in AJAX to implement pop-up menu. These pop-up menus appear when the mouse is hovered. Display text retrieved from the server as shown in Figure below using Ajax techniques. For example, when the mouse is hovered over Sandwiches, various types of sandwiches are displayed which helps the user to quicky see the menu items. [CO1, K3]



Best Practices to be followed:

1. Design before coding
2. Incremental coding
3. Usage of proper naming convention
4. Usage of Comments to the code
5. Indentation of code

**A8:- Ajax Popup Menu Application**

Design:-

Index.html

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Popup Menu App</title>

    <script>

        function showMenu(position) {

            var xhttp = new XMLHttpRequest();

            xhttp.onreadystatechange = function () {

                if (this.readyState == 4 && this.status == 200) {

                    document.getElementById(position).innerHTML = this.responseText;

                }

            };

            xhttp.open("GET", "/PopupMenu/" + position, true);

            xhttp.send();

        }

        function resetMenu(position){

            document.getElementById(position).innerHTML = "";

        }

    </script>

</head>

<body>

    <h1>Popup Menu</h1>

    <section style="display: flex;">

        <h1 onclick="showMenu('programeMenu')" onmouseout="resetMenu('programeMenu')">Programming <div id="programeMenu" style="font-size: medium;">.</div>

        </h1>

        <div style="width: 100px;"></div>

        <h1 onclick="showMenu('softwareMenu')" onmouseout="resetMenu('softwareMenu')">Software <div id="softwareMenu" style="font-size: medium;">.</div>

        </h1>

    </section>

</body>

</html>

Web.xml

<web-app>

    <servlet>

        <servlet-name>ProgrameMenu</servlet-name>

        <servlet-class>ProgrameMenu</servlet-class>

    </servlet>

    <servlet-mapping>

        <servlet-name>ProgrameMenu</servlet-name>

        <url-pattern>/programeMenu</url-pattern>

    </servlet-mapping>

    <servlet>

        <servlet-name>SoftwareMenu</servlet-name>

        <servlet-class>SoftwareMenu</servlet-class>

    </servlet>

    <servlet-mapping>

        <servlet-name>SoftwareMenu</servlet-name>

        <url-pattern>/softwareMenu</url-pattern>

    </servlet-mapping>

    <welcome-file-list>

        <welcome-file>index.html</welcome-file>

    </welcome-file-list>

</web-app>

ProgrameMenu.java

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

import java.sql.\*;

public class ProgrameMenu extends HttpServlet {

    static {

        try {

            Class.forName("com.mysql.cj.jdbc.Driver");

        } catch (ClassNotFoundException e) {

            throw new RuntimeException("Error loading MySQL JDBC driver", e);

        }

    }

    public void doGet(HttpServletRequest request, HttpServletResponse response) throws IOException, ServletException {

        response.setContentType("text/html");

        PrintWriter out = response.getWriter();

        try (

                Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/courses", "root",

                        "password");

                Statement stmt = conn.createStatement();) {

            String sql = "SELECT toolname FROM development WHERE type='programming'";

            ResultSet result = stmt.executeQuery(sql);

            int rowCount = 0;

            while (result.next()) {

                String toolName = result.getString("toolname");

                out.println("<p>" + toolName + " </p>");

            }

        } catch (Exception e) {

            // TODO: handle exception

        }

        out.close();

    }

}

SoftwareMenu.java

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

import java.sql.\*;

public class SoftwareMenu extends HttpServlet {

    static {

        try {

            Class.forName("com.mysql.cj.jdbc.Driver");

        } catch (ClassNotFoundException e) {

            throw new RuntimeException("Error loading MySQL JDBC driver", e);

        }

    }

    public void doGet(HttpServletRequest request, HttpServletResponse response) throws IOException, ServletException {

        response.setContentType("text/html");

        PrintWriter out = response.getWriter();

        try (

                Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/courses", "root",

                        "password");

                Statement stmt = conn.createStatement();) {

            String sql = "SELECT toolname FROM development WHERE type='software'";

            ResultSet result = stmt.executeQuery(sql);

            int rowCount = 0;

            while (result.next()) {

                String toolName = result.getString("toolname");

                out.println("<p>" + toolName + " </p>");

            }

        } catch (Exception e) {

            // TODO: handle exception

        }

        out.close();

    }

}

SQL Queries:-

CREATE TABLE development(toolname VARCHAR(255), type VARCHAR(255));

INSERT INTO development VALUES('JAVA','programming');

INSERT INTO development VALUES('JavaScript','programming');

INSERT INTO development VALUES('C/Cpp','programming');

INSERT INTO development VALUES('Python','programming');

INSERT INTO development VALUES('VScode','software');

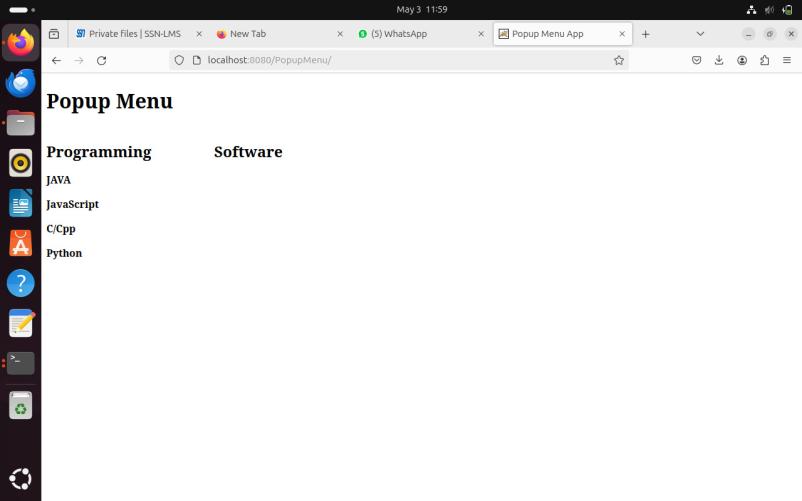
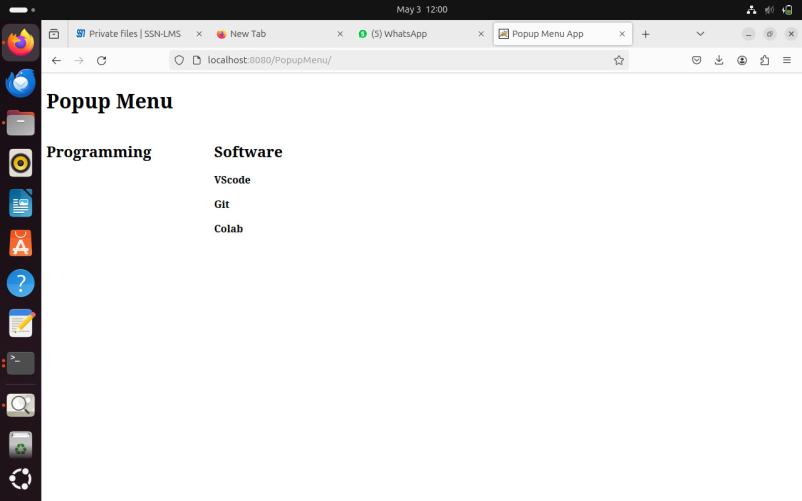
INSERT INTO development VALUES('Git','software');

INSERT INTO development VALUES('Colab','software');

SELECT \* FROM development;

SELECT toolname FROM development WHERE type='programming'

Output:-



**UCS 2611 Internet Programming Lab**

**Exercise 9: Two Tier Authentication Application with ReactJS and Node Server**

Develop an Application to Authenticate the credentials of the user. [CO2, K3]

Application must contain two endpoints namely, login and signup. The *Login* ensures the credentials of the user to enter into the application. Whereas the *Signup* endpoint enrols the new user.

Application must possess the following:

* Provide the HTTP request from front end using ReactJS
* Maintain the credentials of the users in the NodeJS as array of objects
* Define end points in the node server to handle the HTTP requests and generate necessary HTTP response message

Best Practices to be followed:

1. Design before coding
2. Incremental coding
3. Usage of proper naming convention
4. Usage of Comments to the code
5. Indentation of code

**A9:- Two Tier Authentication Application with ReactJS and Node Server**

Design:-

App.js

import { useState } from 'react';

function App() {

const [user, setUser] = useState('');

const [password, setPassword] = useState('');

function updateUser(event) {

setUser(event.target.value);

}

function updatePass(event) {

setPassword(event.target.value);

}

function callLogin() {

const postData = {

username: user,

password: password

};

fetch('http://localhost:5000/login', {

method: 'POST',

headers: {

'Content-Type': 'application/json'

},

body: JSON.stringify(postData)

})

.then(res => res.json())

.then(data => console.log(data))

.catch(error => console.error('Error:', error));

}

function callSignup() {

const postData = {

username: user,

password: password

};

fetch('http://localhost:5000/signup', {

method: 'POST',

headers: {

'Content-Type': 'application/json'

},

body: JSON.stringify(postData)

})

.then(response => {

if (!response.ok) {

throw new Error('Network response was not ok');

}

return response.json();

})

.then(data => {

console.log('Signup successful:', data);

})

.catch(error => {

console.error('Error:', error);

});

}

const formStyle = {

padding: '50px',

backgroundColor: 'red',

}

const AppStyle = {

textAlign: "center",

display: "flex",

justifyContent: "center",

alignContent: "center",

marginTop:"5%",

}

return (

<div style={AppStyle}>

<table style={formStyle} >

<tr>

<th><label>UserName </label></th>

<th> <input type="text" placeholder="username" value={user}

onChange={updateUser}></input> </th>

</tr>

<tr>

<th><label>Password </label></th>

<th><input type="password" placeholder="password" value={password}

onChange={updatePass}></input></th>

</tr>

<tr>

<td><button onClick={callLogin}>Login</button></td>

<td><button onClick={callSignup}>Register New User</button></td>

</tr>

</table>

</div>

);

}

export default App;

twotier.js

const express = require('express');

const bodyParser = require('body-parser');

const cors = require('cors'); // Import the cors middleware

const app = express();

app.use(cors()); // Enable CORS

const PORT = process.env.PORT || 5000;

// Middleware

app.use(bodyParser.json());

// Dummy user data (Replace this with data from your database)

let users = [

{ id: 1, username: 'user1', password: 'password1' },

{ id: 2, username: 'user2', password: 'password2' }

];

// Login endpoint

app.post('/login', (req, res) => {

const { username, password } = req.body;

// Find user by username

const user = users.find(user => user.username === username);

if (!user) {

return res.status(404).json({ message: 'User not found' });

}

// Check password

if (user.password !== password) {

return res.status(401).json({ message: 'Invalid password' });

}

// Return a success message or token if needed

res.json({ message: 'Login successful' });

});

// Signup endpoint

app.post('/signup', (req, res) => {

const { username, password } = req.body;

// Check if the username already exists

const existingUser = users.find(user => user.username === username);

if (existingUser) {

return res.status(400).json({ message: 'Username already exists' });

}

// Create a new user object

const newUser = {

id: users.length + 1,

username,

password

};

// Add the new user to the users array (this would typically be stored in a database)

users.push(newUser);

res.status(201).json({ message: 'Signup successful', user: newUser });

console.log(users);

});

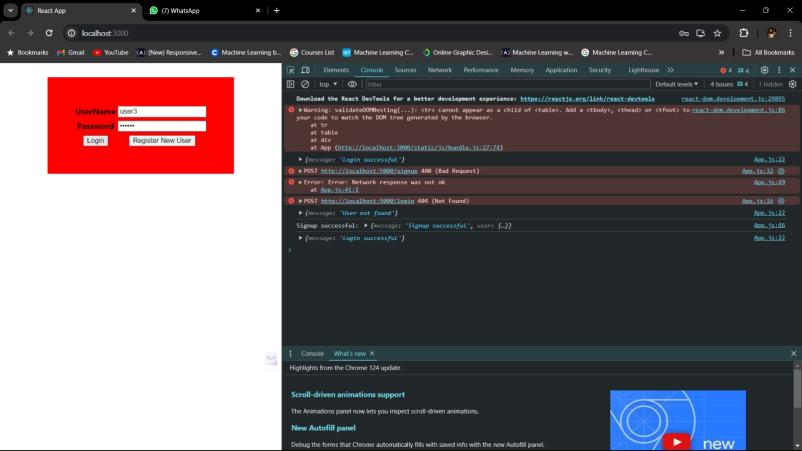
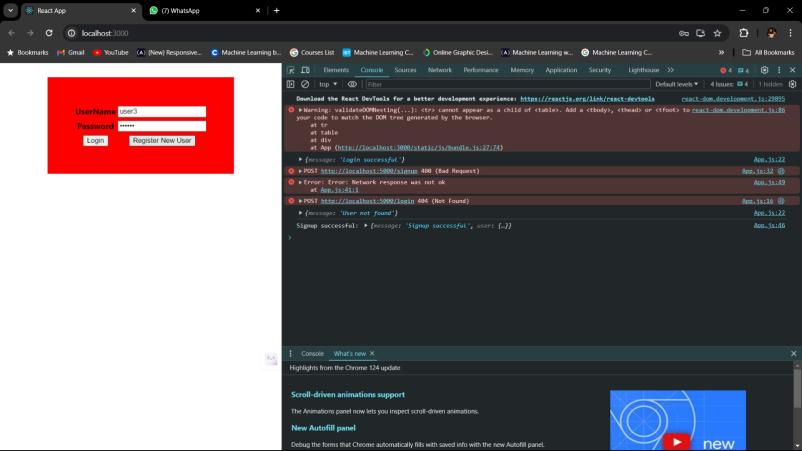
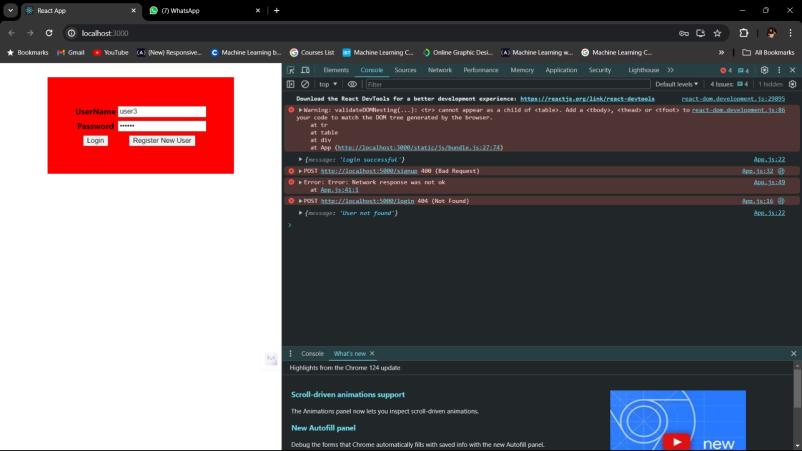
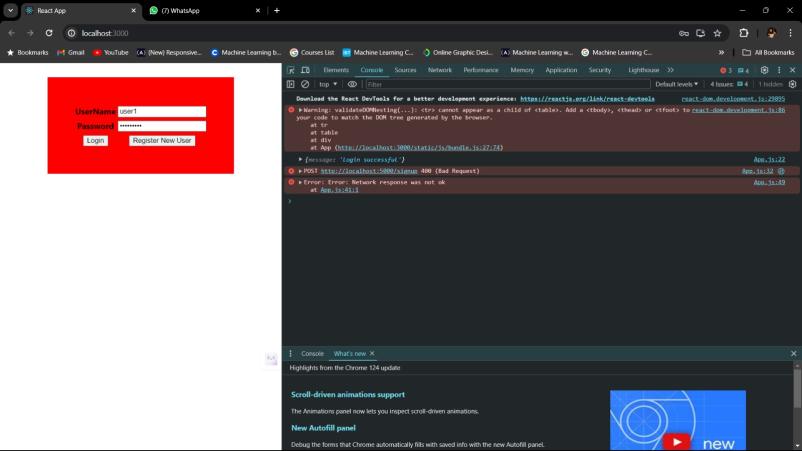
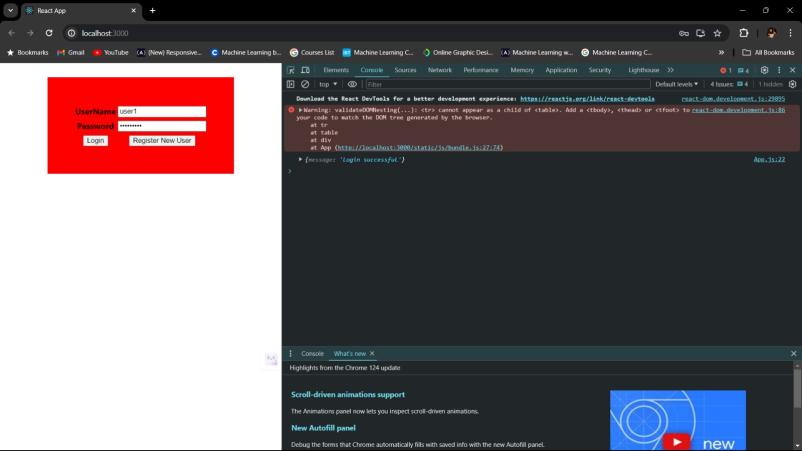
// Start server

app.listen(PORT, () => {

console.log(`Server is running on port ${PORT}`);

});

Output:-



Best practices:-

Learning practices:-

**UCS 2611 Internet Programming Lab**

**Exercise 10. Three Tier Authentication Application with ReactJS, Node and MongoDB**

Develop an Application to Authenticate the credentials of the user. [CO2, K3]

Application must contain two endpoints namely, login and signup. The *Login* ensures the credentials of the user to enter into the application. Whereas the *Signup* endpoint enrols the new user.

Application must possess the following:

* Provide the HTTP request from front end using ReactJS
* Maintain the credentials of the users in the MongoDB Collection
* Define end points in the node server to handle the HTTP requests and generate necessary HTTP response message

Best Practices to be followed:

1. Design before coding
2. Incremental coding
3. Usage of proper naming convention
4. Usage of Comments to the code
5. Indentation of code

**A10:- Three Tier Authentication Application with ReactJS, Node and MongoDB**

Design:-

App.js

import { useState } from 'react';

function App() {

const [user, setUser] = useState('');

const [password, setPassword] = useState('');

function updateUser(event) {

setUser(event.target.value);

}

function updatePass(event) {

setPassword(event.target.value);

}

function callLogin() {

const postData = {

username: user,

password: password

};

fetch('http://localhost:5000/login', {

method: 'POST',

headers: {

'Content-Type': 'application/json'

},

body: JSON.stringify(postData)

})

.then(res => res.json())

.then(data => console.log(data))

.catch(error => console.error('Error:', error));

}

function callSignup() {

const postData = {

username: user,

password: password

};

fetch('http://localhost:5000/signup', {

method: 'POST',

headers: {

'Content-Type': 'application/json'

},

body: JSON.stringify(postData)

})

.then(response => {

if (!response.ok) {

throw new Error('Network response was not ok');

}

return response.json();

})

.then(data => {

console.log('Signup successful:', data);

})

.catch(error => {

console.error('Error:', error);

});

}

const formStyle = {

padding: '50px',

backgroundColor: 'red',

}

const AppStyle = {

textAlign: "center",

display: "flex",

justifyContent: "center",

alignContent: "center",

marginTop:"5%",

}

return (

<div style={AppStyle}>

<table style={formStyle} >

<tr>

<th><label>UserName </label></th>

<th> <input type="text" placeholder="username" value={user}

onChange={updateUser}></input> </th>

</tr>

<tr>

<th><label>Password </label></th>

<th><input type="password" placeholder="password" value={password}

onChange={updatePass}></input></th>

</tr>

<tr>

<td><button onClick={callLogin}>Login</button></td>

<td><button onClick={callSignup}>Register New User</button></td>

</tr>

</table>

</div>

);

}

export default App;

threetier.js

const express = require('express');

const bodyParser = require('body-parser');

const cors = require('cors');

const { MongoClient } = require('mongodb');

const app = express();

const PORT = process.env.PORT || 5000;

// Connection URI

const uri = "mongodb://localhost:27017";

const dbName = "ipLab";

// Middleware

app.use(cors()); // Enable CORS

app.use(bodyParser.json());

// Connect to MongoDB

async function connect() {

try {

// Create a new MongoClient instance

const client = new MongoClient(uri);

// Connect to MongoDB

await client.connect();

console.log("Connected to MongoDB server");

// Access the database

const db = client.db(dbName);

// Define collection

const usersCollection = db.collection("credential");

// Signup endpoint

app.post('/signup', async (req, res) => {

const { username, password } = req.body;

try {

// Check if the username already exists

const existingUser = await usersCollection.findOne({ username });

if (existingUser) {

return res.status(400).json({ message: 'Username already exists' });

}

// Insert the new user into the collection

const newUser = await usersCollection.insertOne({ username, password });

res.status(201).json({ message: 'Signup successful'});

} catch (err) {

console.error("Error signing up:");

res.status(500).json({ message: 'Internal server error' });

}

});

// Login endpoint

app.post('/login', async (req, res) => {

const { username, password } = req.body;

try {

// Find user by username

const user = await usersCollection.findOne({ username });

if (!user) {

return res.status(404).json({ message: 'User not found' });

}

// Check password

if (user.password !== password) {

return res.status(401).json({ message: 'Invalid password' });

}

// Return a success message or token if needed

res.json({ message: 'Login successful', user });

} catch (err) {

console.error("Error logging in:");

res.status(500).json({ message: 'Internal server error' });

}

});

// Start server

app.listen(PORT, () => {

console.log(`Server is running on port ${PORT}`);

});

} catch (err) {

console.error("Error connecting to MongoDB:", err);

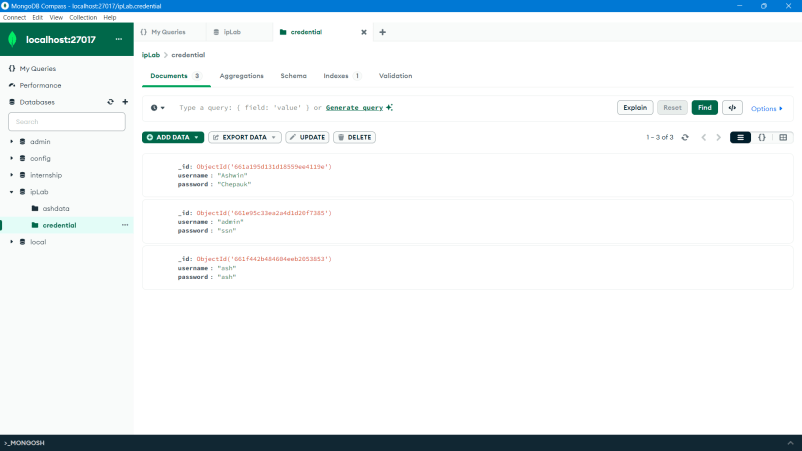
}

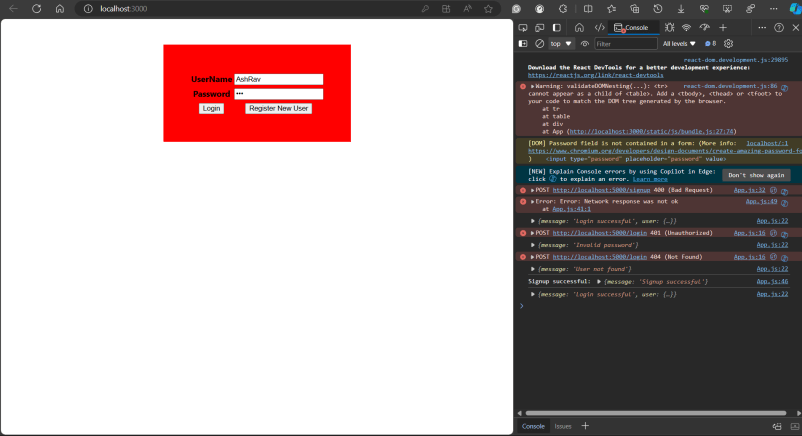
}

// Call the connect function

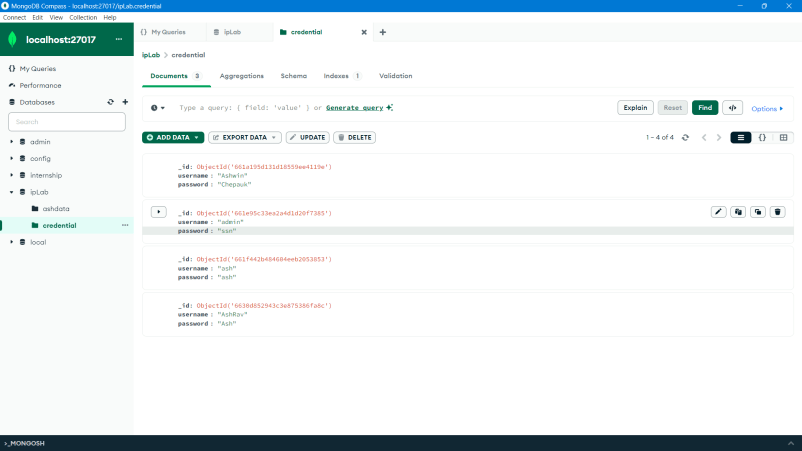
connect();

Output:-





Updated:-



Best Practices:-

Learning Outcomes:-

**UCS 2611 Internet Programming Lab**

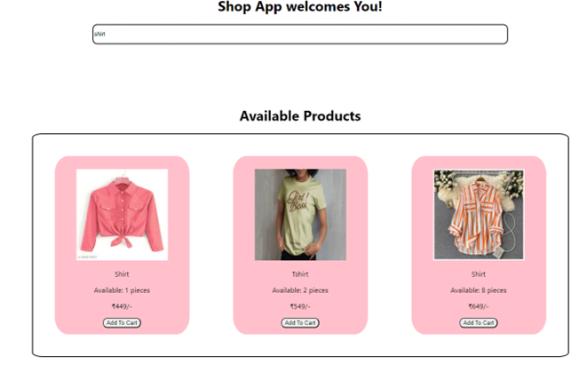
**Exercise 11. Online Shopping Application using ReactJS**

Develop an Online Shopping Application to enable the users to order the products. [CO2, K3]

This Application contains necessary components namely, *App, Show, and Cart*. The *App* Component ensures maintaining the data related to products and the necessary functions. *Show* Component helps to show the details of the products to the users based on the input. *Cart* component is responsible for maintaining the data the user wishes to purchase.

Application has the following:

* The parent component App hosts the other components.
* Maintain the details of the products in the App component
* Define necessary functionalities in the required components to display the shopping cart application looks like below:





Best Practices to be followed:

1. Design before coding
2. Incremental coding
3. Usage of proper naming convention
4. Usage of Comments to the code
5. Indentation of code

**A11:-Shopping Application using ReactJS**

**Design:-**

**App.css**

body {

    font-family: Arial, sans-serif;

    margin: 0;

    padding: 0;

    background-color: #ffffff;

}

.App-header {

    background-color: #6cc24a;

    padding: 20px;

    color: white;

    text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.2);

    text-align: center;

}

.App-main {

    display: flex;

    flex-wrap: wrap;

    flex-direction: column;

    align-items: center;

    justify-content: center;

    padding: 20px;

}

/\* Search Component \*/

.search-bar {

    width: 100%;

    max-width: 400px;

    margin: 0 auto;

    display: flex;

    align-items: center;

    background-color: white;

    border-radius: 20px;

    box-shadow: 0px 4px 8px rgba(0, 0, 0, 0.1);

    padding: 5px 10px;

}

.search-bar input {

    flex: 1;

    padding: 10px;

    border: none;

    border-radius: 5px;

    font-size: 16px;

    background-color: #ffffff;

    transition: background-color 0.3s ease-in-out;

}

.search-bar input:focus {

    outline: none;

    background-color: #ffffff;

}

.search-icon {

    font-size: 1.2rem;

    margin-right: 10px;

    color: #6cc24a;

}

/\* Product Display \*/

.product-list {

    flex: 2;

    display: flex;

    gap: 20px;

}

.product {

    background-color: rgb(255, 245, 245);

    border: 1px solid #1b4e1f;

    border-radius: 10%;

    padding: 10px 60px;

    text-align: center;

    width: 33%;

    transition: transform 0.2s ease-in-out;

    cursor: pointer;

    overflow: hidden;

    position: relative;

}

.product:hover {

    transform: translateY(-5px) scale(1.03);

}

.product img {

    max-width: 150px;

    /\* Increased image size \*/

    height: auto;

    margin-bottom: 10px;

    border-radius: 50%;

    box-shadow: 0px 6px 12px rgba(0, 0, 0, 0.2);

    /\* Enhanced box shadow \*/

    transition: transform 0.3s ease-in-out, box-shadow 0.3s ease-in-out;

    /\* Added box shadow transition \*/

}

.product:hover img {

    transform: scale(1.1);

    z-index: 1;

    box-shadow: 0px 8px 16px rgba(0, 0, 0, 0.3);

    /\* Enhanced box shadow on hover \*/

}

.product h2 {

    font-size: 1.5rem;

    margin: 10px 0;

    color: #323754;

}

.product p {

    font-size: 1.1rem;

    color: #777;

    margin: 5px 0;

}

/\* Cart Checkout \*/

.cart {

    flex: 1;

    min-width: 300px;

    margin-top: 3%;

    background-color: #fff9e6;

    border: 2px solid #193d10;

    border-radius: 20px;

    padding: 10px 20px;

    box-shadow: 0px 10px 20px rgba(0, 0, 0, 0.1);

    display: none;

    position: sticky;

    top: 20px;

}

.cart.active {

    display: block;

}

.cart h2 {

    font-size: 1.8rem;

    color: #323754;

    margin-top: 0;

    text-align: center;

}

.cart ul {

    list-style: none;

    padding: 0;

    margin: 0;

}

.cart-item {

    display: flex;

    flex-direction: column;

    justify-content: space-between;

    align-items: center;

    margin: 15px 0;

    padding: 10px;

    border-bottom: 1px solid #e0e0e0;

}

.cart-item .item-image img {

    max-width: 90px;

    height: auto;

    margin-right: 20px;

    border-radius: 50%;

    box-shadow: 0px 4px 8px rgba(0, 0, 0, 0.1);

    transition: transform 0.3s ease-in-out;

}

.cart-item .item-image img:hover {

    transform: scale(1.1);

}

.cart-item .item-details {

    flex: 1;

    display: inline;

}

.cart-item h3 {

    display: inline;

    font-size: 1.4rem;

    margin: 0;

    color: #323754;

}

.cart-item p {

    font-size: 1.1rem;

    color: #777;

    margin: 5px 0;

}

.cart-item .item-actions {

    display: flex;

    flex-direction: row;

    align-items: center;

}

.cart-item .item-actions button {

    background-color: #6cc24a;

    border: none;

    color: white;

    padding: 8px 15px;

    border-radius: 20px;

    cursor: pointer;

    transition: background-color 0.3s ease-in-out;

    font-size: 1rem;

}

.cart-item .item-actions button:hover {

    background-color: #449e30;

}

.cart-item .quantity {

    display: flex;

    flex-direction: row;

    margin-left: 15px;

    font-size: 1rem;

    color: #323754;

}

.cart .total {

    font-size: 1.4rem;

    margin: 20px 0;

    text-align: right;

    color: #323754;

}

.cart .checkout-button {

    background-color: #6cc24a;

    border: none;

    color: white;

    padding: 10px 15px;

    border-radius: 20px;

    cursor: pointer;

    transition: background-color 0.3s ease-in-out;

    font-size: 1.2rem;

    float: left;

}

.cart .checkout-button:hover {

    background-color: #449e30;

}

.checkout-message {

    font-size: 1.4rem;

    margin-top: 30px;

    color: #6cc24a;

    text-align: center;

}

.no-results,

.empty-cart {

    text-align: center;

    font-size: 1.4rem;

    color: #777;

    margin-top: 20px;

}

@media screen and (max-width: 768px) {

    .App-main {

        flex-direction: column;

        align-items: center;

    }

    .product-list {

        width: 100%;

        margin-bottom: 5px;

    }

    .product {

        width: 50%;

    }

    .cart {

        min-width: unset;

        margin-top: 30px;

    }

}

.add-to-cart-button {

    background-color: #6cc24a;

    border: none;

    color: white;

    padding: 8px 15px;

    border-radius: 20px;

    cursor: pointer;

    transition: background-color 0.3s ease-in-out;

    font-size: 1rem;

}

.add-to-cart-button:hover {

    background-color: #449e30;

}

.item-info {

    display: flex;

    flex-direction: row;

}

.item-details {

    margin-top: 4%;

}

.item-actions .quantity p {

    margin: 10% 10%;

}

.product-list {

    padding-bottom: 5%;

    border-bottom: 2px solid green;

}

.checkout-section {

    float: right;

    margin: 0;

}

.checkout-section .checkout-button {

    background-color: #323754;

}

.checkout-section .checkout-button:hover {

    background-color: #4c5cb6;

}

**App.js**

import React, { useState } from 'react';

import './App.css';

import Search from './Search';

import Show from './Show';

import Cart from './Cart';

function App() {

    const [courses, setCourses] = useState([

        {

            id: 1,

            name: 'Samsung Galaxy Watch 6',

            price: 32990,

            image: 'https://images.samsung.com/is/image/samsung/p6pim/in/2307/gallery/in-galaxy-watch6-r945-sm-r940nzsains-537406468?$650\_519\_PNG'

        },

        {

            id: 2,

            name: 'Samsung Galaxy Buds 2 Pro',

            price: 18000,

            image: 'https://dailydeals365.in/wp-content/uploads/2023/05/samsung-buds-2-pro-black.webp'

        },

        {

            id: 3,

            name: 'Samsung Galaxy Book4 Pro 360',

            price: 181990,

            image: 'https://images.samsung.com/is/image/samsung/assets/in/computers/galaxy-book4-pro/buy/GB4\_Pro-360\_Product-KV\_UK\_PC.jpg?imbypass=true'

        },

        {

            id: 4,

            name: 'Samsung Galaxy S22 Ultra',

            price: 124999,

            image: 'https://images.samsung.com/is/image/samsung/in-galaxy-s22-g998/200143591-320x320?$320\_320\_PNG'

        },

        {

            id: 5,

            name: 'Samsung Galaxy Tab S8',

            price: 61999,

            image: 'https://images.samsung.com/is/image/samsung/assets/in/tablets/galaxy-tab-s8/buy/S8\_TB.png?imwidth=1600'

        },

        {

            id: 6,

            name: 'Samsung Galaxy Watch 4 Classic',

            price: 27999,

            image: 'https://images.samsung.com/is/image/samsung/p6pim/in/galaxy-watch4-classic/gallery/in-galaxy-watch4-classic-r86a-sm-r860nzkiins-425056306?$320\_320\_PNG$'

        },

        {

            id: 7,

            name: 'Samsung Galaxy Z Fold 3',

            price: 149999,

            image: 'https://images.samsung.com/is/image/samsung/in-galaxy-z-fold3-f926/200143571-320x320?$320\_320\_PNG'

        },

        {

            id: 8,

            name: 'Samsung Galaxy Note 20 Ultra',

            price: 104999,

            image: 'https://images.samsung.com/is/image/samsung/p6pim/in/sm-n985fzkdins/gallery/in-note20ultra-mystic-black-317597882?$320\_320\_PNG$'

        },

        {

            id: 9,

            name: 'Samsung Galaxy A52s 5G',

            price: 35999,

            image: 'https://images.samsung.com/is/image/samsung/p6pim/in/galaxy-a52s-5g/gallery/in-galaxy-a52s-5g-a528b-375524471?$320\_320\_PNG$'

        },

        {

            id: 10,

            name: 'Samsung Galaxy M52 5G',

            price: 24999,

            image: 'https://images.samsung.com/is/image/samsung/assets/in/smartphones/galaxy-m52/m52\_buy\_buy\_now\_260718225.jpg?$320\_320\_PNG$'

        },

        {

            id: 11,

            name: 'Samsung Galaxy S21 FE 5G',

            price: 51999,

            image: 'https://images.samsung.com/is/image/samsung/assets/in/smartphones/galaxy-s21-fe/gallery/in-galaxy-s21-fe-fan-edition-g991b-443349978?$320\_320\_PNG$'

        },

        {

            id: 12,

            name: 'Samsung Galaxy A22 5G',

            price: 19999,

            image: 'https://images.samsung.com/is/image/samsung/assets/in/smartphones/galaxy-a22/gallery/a22\_haze\_black\_4G\_130221015?$320\_320\_PNG$'

        },

        {

            id: 13,

            name: 'Samsung Galaxy S20 FE 5G',

            price: 47499,

            image: 'https://images.samsung.com/is/image/samsung/assets/in/smartphones/galaxy-s20-fe/gallery/in-galaxy-s20-fe-fan-edition-5g-g781b-444152453?$320\_320\_PNG$'

        },

        {

            id: 14,

            name: 'Samsung Galaxy A12',

            price: 12999,

            image: 'https://images.samsung.com/is/image/samsung/assets/in/smartphones/galaxy-a12/gallery/a12\_white\_1\_260621275?$320\_320\_PNG$'

        },

        {

            id: 15,

            name: 'Samsung Galaxy A32',

            price: 21999,

            image: 'https://images.samsung.com/is/image/samsung/assets/in/smartphones/galaxy-a32/gallery/a32\_awesome\_black\_1\_241121020?$320\_320\_PNG$'

        },

        {

            id: 16,

            name: 'Samsung Galaxy Fit 2',

            price: 3999,

            image: 'https://images.samsung.com/is/image/samsung/assets/in/wearables/galaxy-fit-2/gallery/in-galaxy-fit-2-fitness-sm-r220-sm-r220nzka1ins-452359596?$320\_320\_PNG$'

        },

        {

            id: 17,

            name: 'Samsung Galaxy A03s',

            price: 11499,

            image: 'https://images.samsung.com/is/image/samsung/assets/in/smartphones/galaxy-a03s/gallery/in-a03s-b-black-370915622?$320\_320\_PNG$'

        },

        {

            id: 18,

            name: 'Samsung Galaxy Book Go',

            price: 37880,

            image: 'https://images.samsung.com/is/image/samsung/p6pim/in/galaxy-book-go/gallery/in-galaxy-book-go-n4500-n4020-n5000-n5100-364013701?$320\_320\_PNG$'

        },

        {

            id: 19,

            name: 'Samsung Galaxy M32',

            price: 18999,

            image: 'https://images.samsung.com/is/image/samsung/assets/in/smartphones/galaxy-m32/gallery/in-galaxy-m32-128gb-6gb-blue-395555773?$320\_320\_PNG$'

        },

        {

            id: 20,

            name: 'Samsung Galaxy Watch 4',

            price: 24999,

            image: 'https://images.samsung.com/is/image/samsung/p6pim/in/galaxy-watch4/gallery/in-galaxy-watch4-44mm-bluetooth-black-sm-r870nzkainu-377389696?$320\_320\_PNG$'

        },

        {

            id: 21,

            name: 'Samsung Galaxy M21 2021 Edition',

            price: 12999,

            image: 'https://images.samsung.com/is/image/samsung/assets/in/smartphones/galaxy-m21-2021/gallery/in-galaxy-m21-2021-sm-m215glbgins-474042023?$320\_320\_PNG$'

        },

        {

            id: 22,

            name: 'Samsung Galaxy M02s',

            price: 10499,

            image: 'https://images.samsung.com/is/image/samsung/assets/in/smartphones/galaxy-m02s/gallery/in-galaxy-m02s-m027fzbdins-333646157?$320\_320\_PNG$'

        },

        {

            id: 23,

            name: 'Samsung Galaxy F41',

            price: 16999,

            image: 'https://images.samsung.com/is/image/samsung/assets/in/smartphones/galaxy-f41/gallery/in-galaxy-f41-sm-f415fzb2ins-274118319?$320\_320\_PNG$'

        },

        {

            id: 24,

            name: 'Samsung Galaxy M02',

            price: 8199,

            image: 'https://images.samsung.com/is/image/samsung/assets/in/smartphones/galaxy-m02/gallery/in-galaxy-m02-m022-m022fzkdins-358828623?$320\_320\_PNG$'

        },

        {

            id: 25,

            name: 'Samsung Galaxy Fit e',

            price: 2490,

            image: 'https://images.samsung.com/is/image/samsung/assets/in/wearables/galaxy-fit-e/gallery/in-galaxy-fit-e-r375-r375nzbbbtl-159130825?$320\_320\_PNG$'

        },

        {

            id: 26,

            name: 'Samsung Type-C to Type-C PD Charging Cable',

            price: 1499,

            image: 'https://images.samsung.com/is/image/samsung/assets/in/mobile-accessories/charger/c-to-c-ep-ta800cbe-original-474085782?$320\_320\_PNG$'

        },

        {

            id: 27,

            name: 'Samsung 45W USB-C Fast Charging Wall Charger',

            price: 2999,

            image: 'https://images.samsung.com/is/image/samsung/p6pim/in/charger/usb-c-ep-ta845xbeg-original-478146226?$320\_320\_PNG$'

        },

        {

            id: 28,

            name: 'Samsung 6.5 kg Fully Automatic Top Load Washing Machine',

            price: 17490,

            image: 'https://images.samsung.com/is/image/samsung/in-fh622qdxbt/20200810-323209272?$320\_320\_PNG$'

        },

        {

            id: 29,

            name: 'Samsung 253 L 2 Star Inverter Frost-Free Double Door Refrigerator',

            price: 21190,

            image: 'https://images.samsung.com/is/image/samsung/in-rt28a3052sp-hl/20210202-323394926?$320\_320\_PNG$'

        },

        {

            id: 30,

            name: 'Samsung 1 Ton 3 Star Inverter Split AC',

            price: 31990,

            image: 'https://images.samsung.com/is/image/samsung/in-ar12aylyatbtl-2019/home-air-conditioners/5-star/in-samsung-ar12aylyatbtl-r32-526563032?$320\_320\_PNG$'

        },

    ]);

    const [cartCourses, setCartCourses] = useState([]);

    const [searchCourse, setSearchCourse] = useState('');

    const addCourseToCartFunction = (SamsungProduct) => {

        const alreadyCourses = cartCourses.find(item => item.product.id === SamsungProduct.id);

        if (alreadyCourses) {

            const latestCartUpdate = cartCourses.map(item =>

                item.product.id === SamsungProduct.id ? { ...item, quantity: item.quantity + 1 } : item

            );

            setCartCourses(latestCartUpdate);

        } else {

            setCartCourses([...cartCourses, { product: SamsungProduct, quantity: 1 }]);

        }

    };

    const deleteCourseFromCartFunction = (SamsungProduct) => {

        const updatedCart = cartCourses.filter(item => item.product.id !== SamsungProduct.id);

        setCartCourses(updatedCart);

    };

    const totalAmountCalculationFunction = () => {

        return cartCourses.reduce((total, item) => total + item.product.price \* item.quantity, 0);

    };

    const courseSearchUserFunction = (event) => {

        setSearchCourse(event.target.value);

    };

    const filterCourseFunction = courses.filter((course) => {

        const searchQuery = searchCourse.toLowerCase();

        const courseName = course.name.toLowerCase();

        const isMatch = courseName.includes(searchQuery);

        console.log(`Search query: ${searchQuery}, Course name: ${courseName}, Match: ${isMatch}`);

        return isMatch;

    });

    return (

        <div className="App">

            <Search searchCourse={searchCourse} courseSearchUserFunction={courseSearchUserFunction} />

            <main className="App-main">

                <Show

                    courses={courses}

                    filterCourseFunction={filterCourseFunction}

                    addCourseToCartFunction={addCourseToCartFunction}

                />

                <Cart

                    cartCourses={cartCourses}

                    deleteCourseFromCartFunction={deleteCourseFromCartFunction}

                    totalAmountCalculationFunction={totalAmountCalculationFunction}

                    setCartCourses={setCartCourses}

                />

            </main>

        </div>

    );

}

export default App;

**Search.js**

import React from 'react';

function Search({ searchCourse, courseSearchUserFunction }) {

    return (

        <header className="App-header">

            <h1> Samsung Shopping Cart</h1>

            <div className="search-bar">

                <input

                    type="text"

                    placeholder="Search for Samsung accessories..."

                    value={searchCourse}

                    onChange={courseSearchUserFunction}

                />

            </div>

        </header>

    );

}

export default Search;

**Show.js**

import React from 'react';

function Show({ filterCourseFunction, addCourseToCartFunction }) {

    // Split filtered courses into rows with three products each

    const rows = [];

    for (let i = 0; i < filterCourseFunction.length; i += 3) {

        const row = filterCourseFunction.slice(i, i + 3);

        rows.push(row);

    }

    return (

        <div className="product-list">

            {rows.map((row, rowIndex) => (

                <div className="row" key={rowIndex}>

                    {row.map(product => (

                        <div className="product" key={product.id}>

                            <img src={product.image} alt={product.name} />

                            <h2>{product.name}</h2>

                            <p>Price: ₹{product.price}</p>

                            <button

                                className="add-to-cart-button"

                                onClick={() => addCourseToCartFunction(product)}

                            >

                                Add to Shopping Cart

                            </button>

                        </div>

                    ))}

                    {/\* Add empty divs to ensure each row contains three products \*/}

                    {[...Array(3 - row.length)].map((\_, index) => (

                        <div className="empty-product" key={index}></div>

                    ))}

                </div>

            ))}

        </div>

    );

}

export default Show;

**Cart.js**

import React from 'react';

function Cart({

    cartCourses,

    deleteCourseFromCartFunction,

    totalAmountCalculationFunction,

    setCartCourses,

}) {

return (

<div className={`cart ${cartCourses.length > 0 ? 'active' : ''}`}>

    <h2>My Cart</h2>

    {cartCourses.length === 0 ? (

    <p className="empty-cart">Your cart is empty.</p>

    ) : (

<div>

    <ul>

        {cartCourses.map((item) => (

            <li key={item.product.id} className="cart-item">

                <div>

                    <div className="item-info">

                        <div className="item-image">

                            <img src={item.product.image}

                                alt={item.product.name} />

                        </div>

                        <div className="item-details">

                            <h3>{item.product.name}</h3>

                            <p>Price: ₹{item.product.price}</p>

                        </div>

                    </div>

                    <div>

                        <div className="item-actions">

                            <button

                                className="remove-button"

                                onClick={() =>

                                deleteCourseFromCartFunction(item.product)}>

                                Remove Product

                            </button>

                            <div className="quantity">

                                <button style={{ margin: "1%" }}

                                    onClick={(e) => {

                                    setCartCourses((prevCartCourses) => {

                                        const updatedCart = prevCartCourses.map(

                                        (prevItem) =>

                                        prevItem.product.id === item.product.id

                                                ? { ...prevItem, quantity:

                                                item.quantity + 1 }

                                                : prevItem

                                        );

                                        return updatedCart;

                                    })

                                }}>+</button>

                                <p className='quant'>{item.quantity} </p>

                                <button

                                    onClick={(e) => {

                                    setCartCourses((prevCartCourses) => {

                                        const updatedCart = prevCartCourses.map(

                                        (prevItem) =>

                                        prevItem.product.id === item.product.id

                                                ? { ...prevItem, quantity:

                                                Math.max(item.quantity - 1, 0) }

                                                : prevItem

                                        );

                                        return updatedCart;

                                    })

                                }}>-</button>

                            </div>

                        </div>

                    </div>

                </div>

            </li>

        ))}

    </ul>

    <div className="checkout-section">

        <div className="checkout-total">

            <p className="total">Total Amount:

                ₹{totalAmountCalculationFunction()}

            </p>

        </div>

        <button

            className="checkout-button"

            disabled={cartCourses.length === 0 ||

            totalAmountCalculationFunction() === 0}

        >

            Proceed to Payment

        </button>

    </div>

</div>

            )}

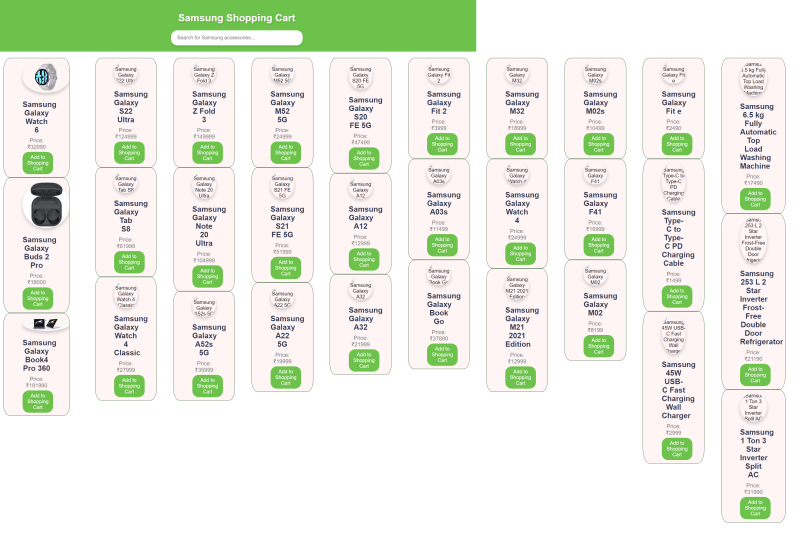
</div>

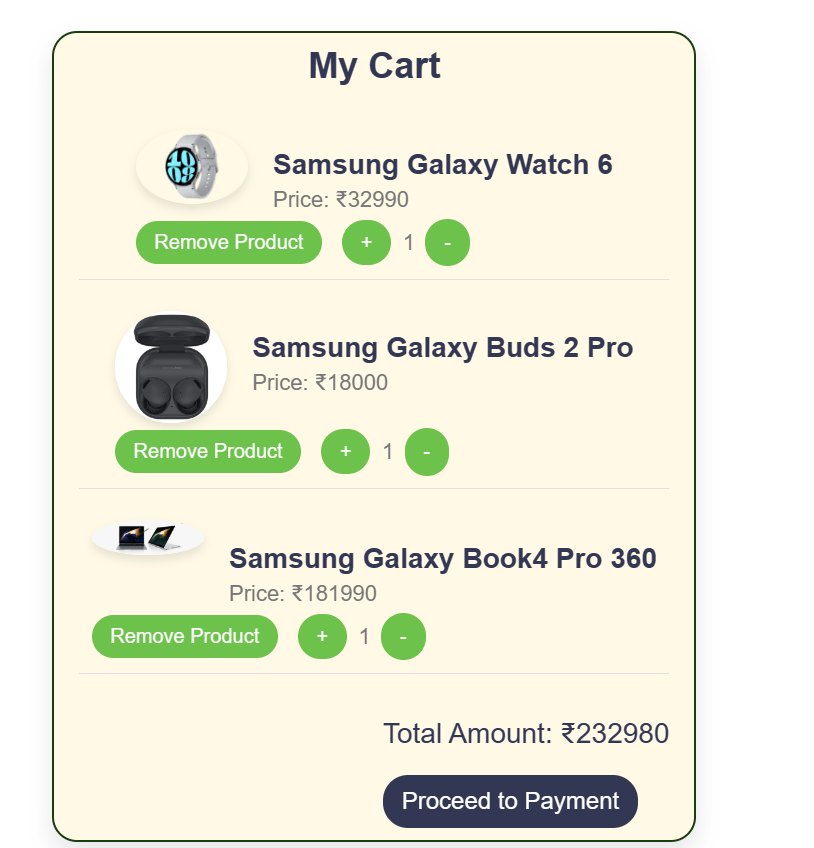
    );

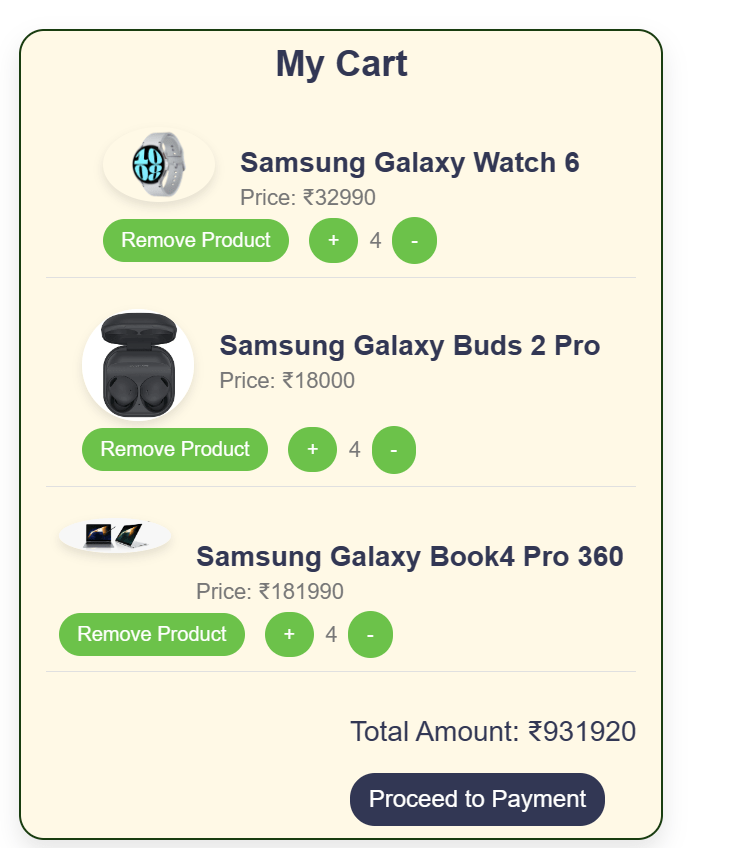
}

export default Cart;

**Output:-**

****

****

****

****

Best practices:-

Learning outcomes:-

**UCS 2611 Internet Programming Lab**

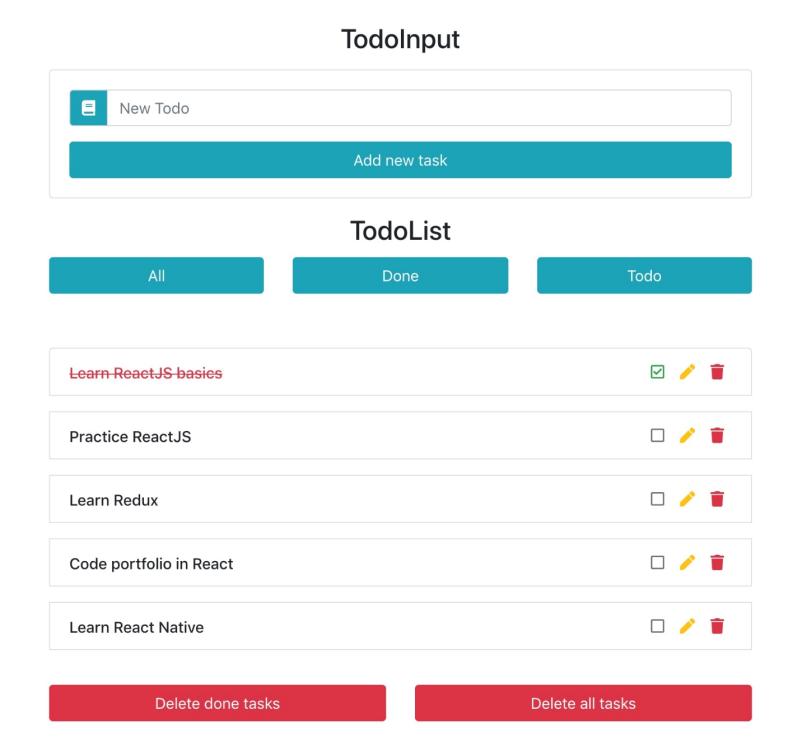
**Exercise 12. Full Stack Todo Web Application using ReactJS, Node and MongoDB**

Develop a ***Todo Web Application*** that enables the user to maintain and keep track of their Todo list [CO2, K3]

Draw a diagram depicting the design flow of your project with necessary components, endpoints and collections

Application has the following:

* Front end to provide necessary requests using ReactJS
* Maintain the details in MongoDB
* Define necessary functionalities in the required components and endpoints in node server to display the Todo application like below:

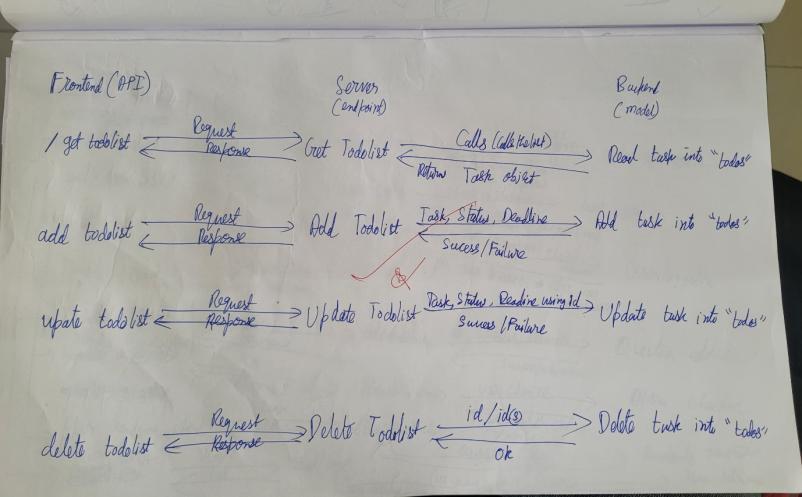


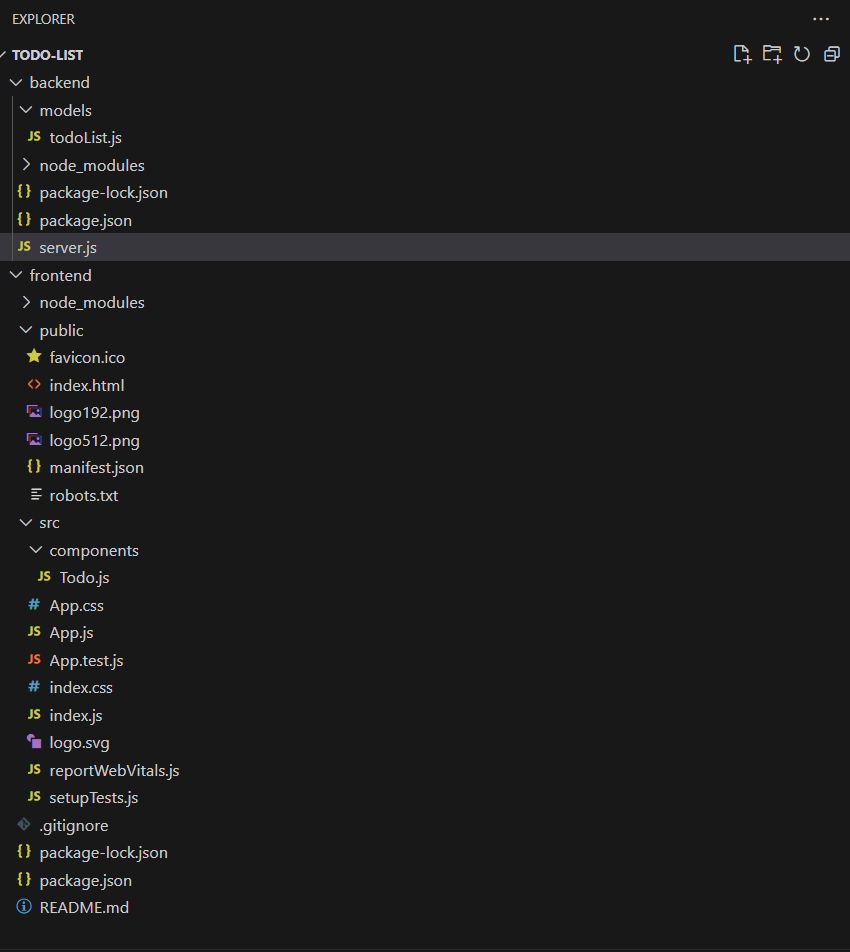
Best Practices to be followed:

1. Design before coding
2. Incremental coding
3. Usage of proper naming convention
4. Usage of Comments to the code
5. Indentation of code

**A12:- Full Stack ToDo Web Application using ReactJS, Node and MongoDB**

**Design:-**



****

Todolist.js

const mongoose = require('mongoose');

const todoSchema = new mongoose.Schema({

    task: {

        type: String,

        required: true,

    },

    status: {

        type: String,

        required: true,

    },

    deadline: {

        type: Date,

    },

});

const todoList = mongoose.model("todo", todoSchema);

module.exports = todoList;

server.js

const express = require('express')

const mongoose = require('mongoose')

const cors = require('cors')

const TodoModel = require("./models/todoList")

var app = express();

app.use(cors());

app.use(express.json());

// Connect to your MongoDB database (replace with your database URL)

mongoose.connect("mongodb://localhost:27017/todo");

// Check for database connection errors

mongoose.connection.on("error", (error) => {

    console.error("MongoDB connection error:", error);

});

// Get saved tasks from the database

app.get("/getTodoList", (req, res) => {

    TodoModel.find({})

        .then((todoList) => res.json(todoList))

        .catch((err) => res.json(err))

});

// Add new task to the database

app.post("/addTodoList", (req, res) => {

    TodoModel.create({

        task: req.body.task,

        status: req.body.status,

        deadline: req.body.deadline,

    })

        .then((todo) => res.json(todo))

        .catch((err) => res.json(err));

});

// Update task fields (including deadline)

app.post("/updateTodoList/:id", (req, res) => {

    const id = req.params.id;

    const updateData = {

        task: req.body.task,

        status: req.body.status,

        deadline: req.body.deadline,

    };

    TodoModel.findByIdAndUpdate(id, updateData)

        .then((todo) => res.json(todo))

        .catch((err) => res.json(err));

});

// Delete task from the database

app.delete("/deleteTodoList/:id", (req, res) => {

    const id = req.params.id;

    TodoModel.findByIdAndDelete({ \_id: id })

        .then((todo) => res.json(todo))

        .catch((err) => res.json(err));

});

app.listen(3001, () => {

    console.log('Server running on 3001');

});

Todo.js

import axios from "axios";

import React from "react";

import { useEffect, useState } from "react";

function Todo() {

    const [todoList, setTodoList] = useState([]);

    const [editableId, setEditableId] = useState(null);

    const [editedTask, setEditedTask] = useState("");

    const [editedStatus, setEditedStatus] = useState("");

    const [newTask, setNewTask] = useState("");

    const [newStatus, setNewStatus] = useState("");

    const [newDeadline, setNewDeadline] = useState("");

    const [editedDeadline, setEditedDeadline] = useState("");

    // Fetch tasks from database

    useEffect(() => {

        axios.get('http://127.0.0.1:3001/getTodoList')

            .then(result => {

                setTodoList(result.data)

            })

            .catch(err => console.log(err))

    }, [])

    // Function to toggle the editable state for a specific row

    const toggleEditable = (id) => {

        const rowData = todoList.find((data) => data.\_id === id);

        if (rowData) {

            setEditableId(id);

            setEditedTask(rowData.task);

            setEditedStatus(rowData.status);

            setEditedDeadline(rowData.deadline || "");

        } else {

            setEditableId(null);

            setEditedTask("");

            setEditedStatus("");

            setEditedDeadline("");

        }

    };

    // Function to add task to the database

    const addTask = (e) => {

        e.preventDefault();

        if (!newTask || !newStatus || !newDeadline) {

            alert("All fields must be filled out.");

            return;

        }

        axios.post('http://127.0.0.1:3001/addTodoList', { task: newTask, status: newStatus, deadline: newDeadline })

            .then(res => {

                console.log(res);

                window.location.reload();

            })

            .catch(err => console.log(err));

    }

    // Function to save edited data to the database

    const saveEditedTask = (id) => {

        const editedData = {

            task: editedTask,

            status: editedStatus,

            deadline: editedDeadline,

        };

        // If the fields are empty

        if (!editedTask || !editedStatus || !editedDeadline) {

            alert("All fields must be filled out.");

            return;

        }

        // Updating edited data to the database through updateById API

        axios.post('http://127.0.0.1:3001/updateTodoList/' + id, editedData)

            .then(result => {

                console.log(result);

                setEditableId(null);

                setEditedTask("");

                setEditedStatus("");

                setEditedDeadline(""); // Clear the edited deadline

                window.location.reload();

            })

            .catch(err => console.log(err));

    }

    // Delete task from database

    const deleteTask = (id) => {

        axios.delete('http://127.0.0.1:3001/deleteTodoList/' + id)

            .then(result => {

                console.log(result);

                window.location.reload();

            })

            .catch(err =>

                console.log(err)

            )

    }

    return (

        <div className="container mt-5">

            <div className="row">

                <div className="col-md-7">

                    <h2 className="text-center">Todo List</h2>

                    <div className="table-responsive">

                        <table className="table table-bordered">

                            <thead className="table-primary">

                                <tr>

                                    <th>Task</th>

                                    <th>Status</th>

                                    <th>Deadline</th>

                                    <th>Actions</th>

                                </tr>

                            </thead>

                            {Array.isArray(todoList) ? (

                                <tbody>

                                    {todoList.map((data) => (

                                        <tr key={data.\_id}>

                                            <td>

                                                {editableId === data.\_id ? (

                                                    <input

                                                        type="text"

                                                        className="form-control"

                                                        value={editedTask}

                                                        onChange={(e) => setEditedTask(e.target.value)}

                                                    />

                                                ) : (

                                                    data.task

                                                )}

                                            </td>

                                            <td>

                                                {editableId === data.\_id ? (

                                                    <input

                                                        type="text"

                                                        className="form-control"

                                                        value={editedStatus}

                                                        onChange={(e) => setEditedStatus(e.target.value)}

                                                    />

                                                ) : (

                                                    data.status

                                                )}

                                            </td>

                                            <td>

                                                {editableId === data.\_id ? (

                                                    <input

                                                        type="datetime-local"

                                                        className="form-control"

                                                        value={editedDeadline}

                                                        onChange={(e) => setEditedDeadline(e.target.value)}

                                                    />

                                                ) : (

                                                    data.deadline ? new Date(data.deadline).toLocaleString() : ''

                                                )}

                                            </td>

                                            <td>

                                                {editableId === data.\_id ? (

                                                    <button className="btn btn-success btn-sm" onClick={() => saveEditedTask(data.\_id)}>

                                                        Save

                                                    </button>

                                                ) : (

                                                    <button className="btn btn-primary btn-sm" onClick={() => toggleEditable(data.\_id)}>

                                                        Edit

                                                    </button>

                                                )}

                                                <button className="btn btn-danger btn-sm ml-1" onClick={() => deleteTask(data.\_id)}>

                                                    Delete

                                                </button>

                                            </td>

                                        </tr>

                                    ))}

                                </tbody>

                            ) : (

                                <tbody>

                                    <tr>

                                        <td colSpan="4">Loading products...</td>

                                    </tr>

                                </tbody>

                            )}

                        </table>

                    </div>

                </div>

                <div className="col-md-5">

                    <h2 className="text-center">Add Task</h2>

                    <form className="bg-light p-4">

                        <div className="mb-3">

                            <label>Task</label>

                            <input

                                className="form-control"

                                type="text"

                                placeholder="Enter Task"

                                onChange={(e) => setNewTask(e.target.value)}

                            />

                        </div>

                        <div className="mb-3">

                            <label>Status</label>

                            <input

                                className="form-control"

                                type="text"

                                placeholder="Enter Status"

                                onChange={(e) => setNewStatus(e.target.value)}

                            />

                        </div>

                        <div className="mb-3">

                            <label>Deadline</label>

                            <input

                                className="form-control"

                                type="datetime-local"

                                onChange={(e) => setNewDeadline(e.target.value)}

                            />

                        </div>

                        <button onClick={addTask} className="btn btn-success btn-sm">

                            Add Task

                        </button>

                    </form>

                </div>

            </div>

        </div>

    )

}

export default Todo;

index.html

<!-- index.html -->

<!DOCTYPE html>

<html lang="en">

  <head>

    <title>Todo List</title>

  </head>

  <body>

    <noscript>You need to enable JavaScript to run this app.</noscript>

    <div id="root"></div>

  </body>

</html>

App.js

//App.js

import React from 'react';

import { BrowserRouter, Routes, Route } from 'react-router-dom';

import 'bootstrap/dist/css/bootstrap.min.css';

import Todo from './components/Todo';

function App() {

  const headStyle = {

    textAlign: "center",

  }

  return (

    <div>

      <h1 style={headStyle}>Ashwin Todo List</h1>

      <BrowserRouter>

        <Routes>

          <Route path='/' element={<Todo/>}></Route>

        </Routes>

      </BrowserRouter>

    </div>

  );

}

export default App;

index.js

import React from 'react';

import ReactDOM from 'react-dom/client';

import './index.css';

import App from './App';

import reportWebVitals from './reportWebVitals';

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(

  <React.StrictMode>

    <App />

  </React.StrictMode>

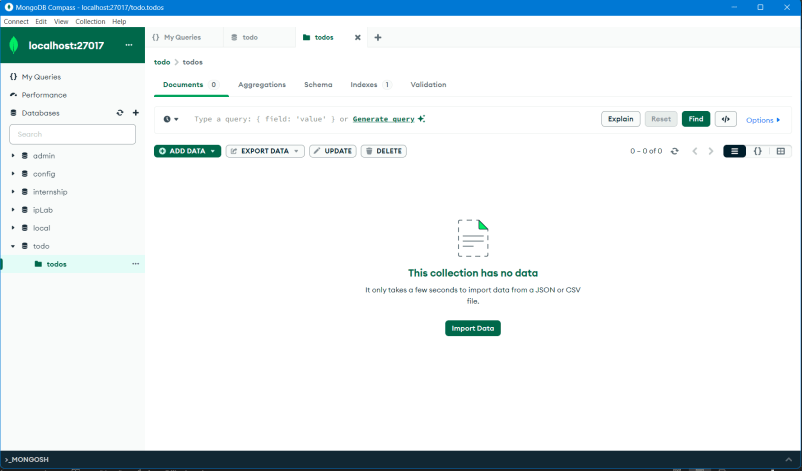
);

// If you want to start measuring performance in your app, pass a function

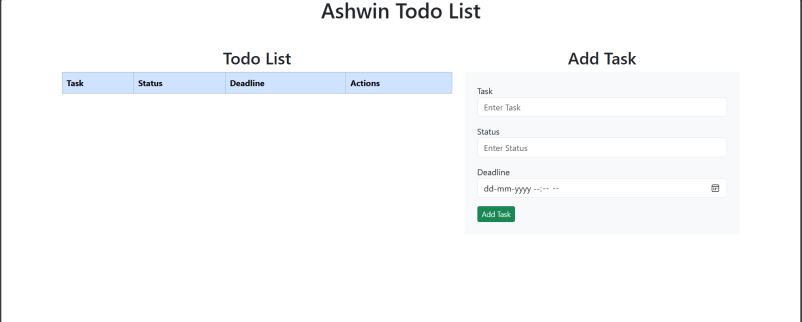
// to log results (for example: reportWebVitals(console.log))

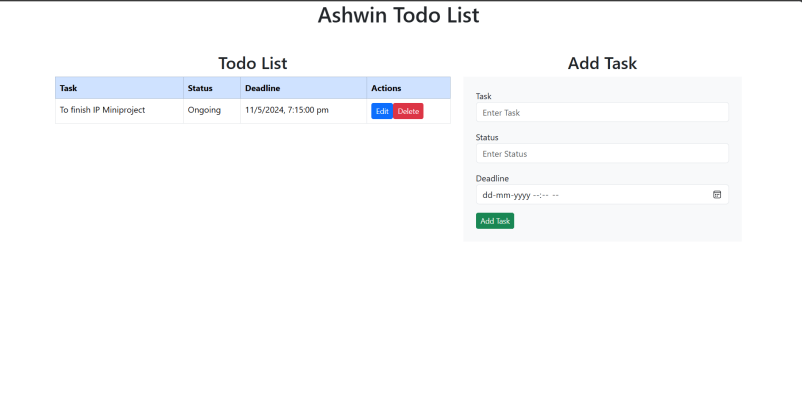
// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals

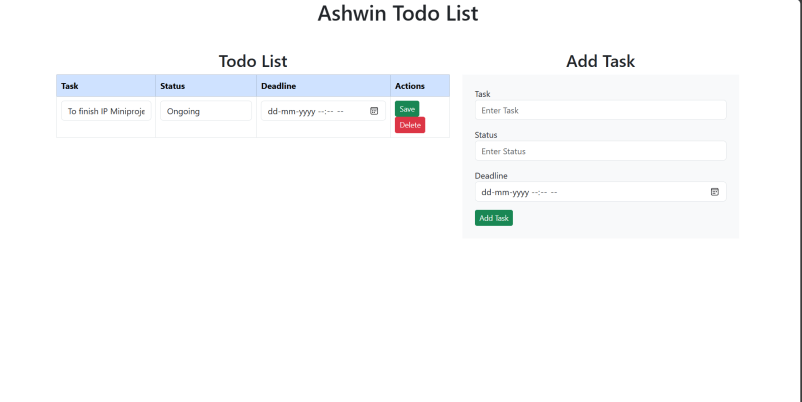
reportWebVitals();

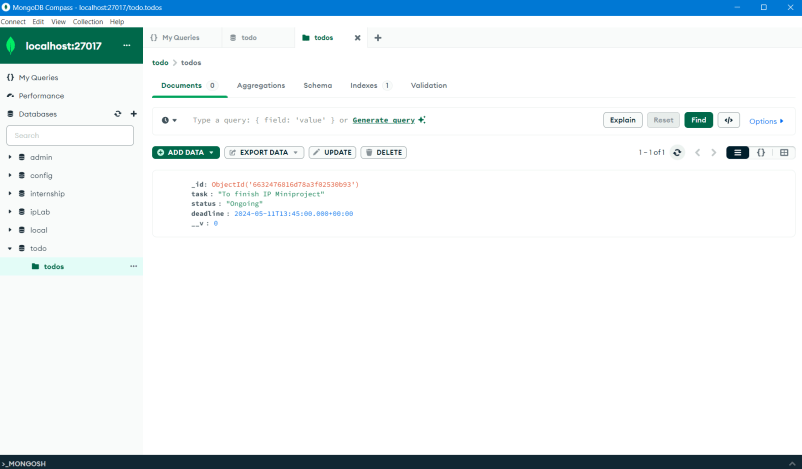


Output:-









Best practices:-

Learning Outcomes:-

UCS2611 – Internet Programming Lab

------------------------------------------------------------------------------------

**Mini project Description**

Develop a full stack web application for conducting *On-line quiz using MVC architecture*. The application should facilitate the normal and admin users to access it. To the normal user, instructions, questions with options and the score needs to be provided as the following snapshots. Admin user can view the registered users and their scores.

Design the following:

* Schema of the necessary MongoDB collections. One of which is shown below
  + Create Questions collection in MongoDB as follows to store the information about the students and Questions with choices.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Question | Choice\_A | Choice\_B | Choice\_C | Choice\_D | Answer |

* Design the application with the necessary endpoints, controllers, collections, and components as a sequence diagram

Do the following operations:

* Authenticate the user and provide necessary messages
* Keep a timer for the Quiz
* Sending appropriate GET http requests from front end, to the endpoint in the node server.
* Perform necessary operations in the Mongo Collection at the endpoints
* Create a suitable interface in ReactJS to display the results





Use the following Best Practices:

* Design before coding
* Incremental coding
* Usage of proper naming convention
* Usage of Comments to the code
* Indentation of code

**SSN COLLEGE OF ENGINEERING KALAVAKKAM -603110**

**Department of Computer Science and Engineering**

**Mini Project**

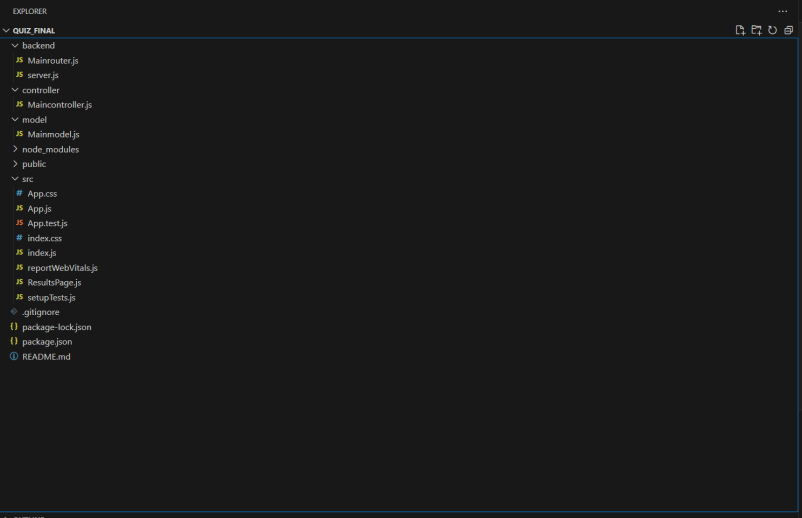
**UCS2611 – Internet Programming Lab**

**Full stack Quiz app using MERN**

**TEAM MEMBERS:**

1. **Ashwin R - 3122 21 5001 014**
2. **Jaanus Sri K.G:- 3122 21 5001 037**
3. **Karthik Vijayakumar:- 3122 21 5001 040**

**App directory:-**

****

**MongoDB database directory inside the database named “IpLab”**

****

**quiz:- The questions is added and stored by the admin for the users to play the game.**

**QuizAdmin:- Admin’s login credential.**

**quizUsers:- User’s login credentials(created by signing up in the app).**

**quizResults:- User’s results are stored for the admin to view the results of the users who all have played and their score.**

**In the React App,**

* MainRouter.js communicates with the database and server.js
* Server.js communicates between MainRouter and the Frontend(App.js)
* MainModel.js is to connect to the database and the collections inside it.
* MainController.js validates the login info for User.
* ResultsPage.js retrieves the results from the quizResults collection and displays it to the admin.
* App.js is the frontend responsible for showing the UI along with the CSS designs done.

**User can,**

* Signup for playing the game
* Login to play the game(if already signed up)
* View questions and answer the questions by selecting a choice
* Switch to the next question
* See the results of the test taken

**Admin can,**

* Login to the admin dashboard
* Add questions to the game by giving the question and the choices along with the right answer
* See user results with the username and score in a tabular form

MainRouter.js

const express = require('express');

const router = express.Router();

const { MongoClient } = require('mongodb');

const { generateMessage } = require('../controller/Maincontroller');

// MongoDB connection setup

const uri = "mongodb://localhost:27017";

const dbName = "ipLab";

const client = new MongoClient(uri);

// Connect to MongoDB

async function connectMongoDB() {

    try {

        await client.connect();

        console.log("Connected to MongoDB");

    } catch (error) {

        console.error("Error connecting to MongoDB:", error);

        throw error;

    }

}

// Connect to MongoDB when the server starts

connectMongoDB();

// Routes

const adminCollection = client.db(dbName).collection("quizAdmin");

const userCollection = client.db(dbName).collection("quizUsers");

const quizCollection = client.db(dbName).collection("quiz");

const resultsCollection = client.db(dbName).collection("quizResults");

// Admin login route

router.post("/admin/login", async (req, res) => {

    const { username, password } = req.body;

    try {

        const data = await generateMessage(username, password, 'quizAdmin');

        res.json({ message: data });

    } catch (error) {

        console.error("Error generating message:", error);

        res.status(500).json({ message: 'Internal server error' });

    }

});

// User login route

router.post("/login", async (req, res) => {

    const { username, password } = req.body;

    try {

        const data = await generateMessage(username, password, 'quizUsers');

        res.json({ message: "Login Successful" });

    } catch (error) {

        console.error("Error generating message:", error);

        res.status(500).json({ message: 'Internal server error' });

    }

});

// User signup route

router.post('/signup', async (req, res) => {

    const { username, password } = req.body;

    try {

        const existingUser = await userCollection.findOne({ username });

        if (existingUser) {

            return res.status(400).json({ message: 'Username already exists' });

        } else {

            const newUser = await userCollection.insertOne({ username, password });

            res.status(201).json({ message: 'User signup successful' });

        }

    } catch (error) {

        console.error("Error signing up:", error);

        res.status(500).json({ message: 'Internal server error' });

    }

});

// Route to add a question to the quiz

router.post('/admin/addQuestion', async (req, res) => {

    try {

        const { question, answer, choice1, choice2, choice3, choice4 } = req.body;

        const newQuestion = await quizCollection.insertOne({ question, answer, choice1, choice2, choice3, choice4 });

        res.status(201).json({ message: 'Question added to quiz' });

    } catch (error) {

        console.error("Error adding question:", error);

        res.status(500).json({ message: 'Internal server error' });

    }

});

// Route to fetch random quiz questions

router.get('/quiz/random', async (req, res) => {

    try {

        const randomQuestions = await quizCollection.aggregate([{ $sample: { size: 10 } }]).toArray();

        res.json(randomQuestions);

    } catch (error) {

        console.error("Error fetching random questions:", error);

        res.status(500).json({ message: 'Internal server error' });

    }

});

// Route to update user score

router.post('/user/updateScore', async (req, res) => {

    const { username, password, score } = req.body;

    try {

        // Update user's score

        const result = await resultsCollection.insertOne({ username, password , score  });

        res.status(201).json({ message: 'User score updated successfully', result });

    } catch (error) {

        console.error("Error updating user score:", error);

        res.status(500).json({ message: 'Internal server error' });

    }

});

// Route to fetch quiz results

router.get('/quiz/results', async (req, res) => {

    try {

        const quizResults = await resultsCollection.find({}).toArray();

        res.json(quizResults);

    } catch (error) {

        console.error("Error fetching quiz results:", error);

        res.status(500).json({ message: 'Internal server error' });

    }

});

module.exports = router;

Server.js

const express = require('express');

const bodyParser = require('body-parser');

const cors = require('cors');

const mainRoute = require('./Mainrouter');

const app = express();

const PORT = process.env.PORT || 5000;

app.use(bodyParser.json());

app.use(cors());

app.use('/', mainRoute);

app.listen(PORT, () => {

    console.log("Server Listening on port", PORT);

});

MainController.js

const { connect } = require('../model/Mainmodel');

async function generateMessage(username, password, collection) {

    try {

        const db = await connect();

        const usersCollection = db.collection(collection);

        const user = await usersCollection.findOne({ username });

        if (!user) {

            return 'incorrect username or password'; // Return error message if user not found

        }

        if (user.password !== password) {

            return 'incorrect username or password'; // Return error message if password is incorrect

        }

        console.log("Success. ", user.username + " " + user.password);

        return user.username;

    } catch (error) {

        console.log("Error logging in:", error);

        throw error;

    }

}

module.exports = { generateMessage };

MainModel.js

const { MongoClient } = require('mongodb');

const uri = "mongodb://localhost:27017";

const dbName = "ipLab";

async function connect() {

    try {

        const client = new MongoClient(uri);

        await client.connect();

        const db = client.db(dbName);

        return db;

    } catch (error) {

        console.error("Error connecting to MongoDB:", error);

        throw error;

    }

}

module.exports = { connect };

App.js

// App.js

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

import './App.css'

import ResultsPage  from './ResultsPage';

export default function App() {

    const [currentPage, setCurrentPage] = useState("home");

    const [message, setMessage] = useState("");

    const [username, setUsername] = useState("");

    const [password, setPassword] = useState("");

    const [question, setQuestion] = useState("");

    const [answer, setAnswer] = useState("");

    const [choice1, setChoice1] = useState("");

    const [choice2, setChoice2] = useState("");

    const [choice3, setChoice3] = useState("");

    const [choice4, setChoice4] = useState("");

    const [isAdminLoggedIn, setIsAdminLoggedIn] = useState(false);

    const [isUserLoggedIn, setIsUserLoggedIn] = useState(false);

    const [questionData, setQuestionData] = useState([]);

    const [quizStarted, setQuizStarted] = useState(false);

    const [currentQuestionIndex, setCurrentQuestionIndex] = useState(0);

    const [score, setScore] = useState(0);

    const [timer, setTimer] = useState(10);

    const [selectedAnswerIndex, setSelectedAnswerIndex] = useState(null);

    // Timer logic

    useEffect(() => {

        let interval;

        if (quizStarted && timer > 0) {

            interval = setInterval(() => {

                setTimer(prevTime => prevTime - 1);

            }, 1000);

        } else if (quizStarted && timer === 0) {

            handleTimeout(); // Call function to handle timeout

        }

        return () => clearInterval(interval);

    }, [quizStarted, timer]);

    useEffect(() => {

        fetchRandomQuestions();

    }, []);

    const handleAdminClick = () => {

        setCurrentPage("adminLogin");

    };

    const handleUserClick = () => {

        setCurrentPage("userOptions");

    };

    // Handle timeout

    const handleTimeout = () => {

        handleAnswer(null, null); // Move to the next question with no selected answer

    };

    const handleLogin = () => {

        if (currentPage === "adminLogin") {

            adminLogin();

        } else if (currentPage === "userLogin") {

            userLogin();

        }

    };

    const handleSignup = () => {

        fetch('http://localhost:5000/signup', {

            method: 'POST',

            headers: {

                'Content-Type': 'application/json'

            },

            body: JSON.stringify({ username, password }),

        })

            .then(res => res.json())

            .then(data => {

                setMessage(data.message);

                console.log(data.message);

                if (data.message === 'User signup successful') {

                    setIsUserLoggedIn(true);

                    setCurrentPage("userDashboard");

                }

            })

            .catch(error => console.error('Error:', error));

    };

    const handleAddQuestion = () => {

        fetch('http://localhost:5000/admin/addQuestion', {

            method: 'POST',

            headers: {

                'Content-Type': 'application/json'

            },

            body: JSON.stringify({ question, answer, choice1, choice2, choice3, choice4 }),

        })

            .then(res => res.json())

            .then(data => { setMessage(data.message); console.log(data.message);alert("Quiz added to the database")})

            .catch(error => console.error('Error:', error));

    };

    const adminLogin = () => {

        fetch('http://localhost:5000/admin/login', {

            method: 'POST',

            headers: {

                'Content-Type': 'application/json'

            },

            body: JSON.stringify({ username, password }),

        })

            .then(res => res.json())

            .then(data => {

                setMessage(data.message);

                console.log(data.message);

                if(data.message === 'Admin') {

                    setIsAdminLoggedIn(true);

                    setCurrentPage("renderAdminDashboard");

                }

            })

            .catch(error => console.error('Error:', error));

    };

    const userLogin = () => {

        fetch('http://localhost:5000/login', {

            method: 'POST',

            headers: {

                'Content-Type': 'application/json'

            },

            body: JSON.stringify({ username, password }),

        })

            .then(res => res.json())

            .then(data => {

                setMessage(data.message);

                console.log(data.message);

                if (data.message !== 'incorrect username or password') {

                    setIsUserLoggedIn(true);

                    setCurrentPage("userDashboard");

                }

            })

            .catch(error => console.error('Error:', error));

    };

    const fetchRandomQuestions = () => {

        fetch('http://localhost:5000/quiz/random')

            .then(res => res.json())

            .then(data => {

                setQuestionData(data);

            })

            .catch(error => console.error('Error fetching questions:', error));

    };

    const handleStartQuiz = () => {

        setQuizStarted(true);

        setTimer(10);

        setScore(0);

        setCurrentQuestionIndex(0);

    };

    const renderHomePage = () => {

        return (

            <center>

                <div className="d1">

                    <h2>Welcome to Quiz</h2>

                    <button onClick={handleAdminClick}>Admin</button>

                    <button onClick={handleUserClick}>User</button>

                </div>

            </center>

        );

    };

    const renderUserOptions = () => {

        return (

            <center>

                <div className="d1">

                    <h2>User Options</h2>

                    <button onClick={() => setCurrentPage("userLogin")}>Login</button>

                    <button onClick={() => setCurrentPage("userSignup")}>Signup</button>

                    <button onClick={() => setCurrentPage("home")}>Back</button>

                </div>

            </center>

        );

    };

    const renderAdminLoginPage = () => {

        return (

            <center>

                <div className="d1">

                    <h2>Login</h2>

                    <label>Username: <input type="text" value={username} onChange={(e) => setUsername(e.target.value)} /></label><br /><br />

                    <label>Password: <input type="password" value={password} onChange={(e) => setPassword(e.target.value)} /></label><br /><br />

                    <button onClick={() => setCurrentPage("home")}>Back</button>

                    <button onClick={handleLogin}>Login</button>

                </div>

            </center>

        );

    };

    const renderUserLoginPage = () => {

        return (

            <center>

                <div className="d1">

                    <h2>Login</h2>

                    <label>Username: <input type="text" value={username} onChange={(e) => setUsername(e.target.value)} /></label><br /><br />

                    <label>Password: <input type="password" value={password} onChange={(e) => setPassword(e.target.value)} /></label><br /><br />

                    <button onClick={() => setCurrentPage("userOptions")}>Back</button>

                    <button onClick={handleLogin}>Login</button>

                </div>

            </center>

        );

    };

    const renderSignupPage = () => {

        return (

            <center>

                <div className="d1">

                    <h2>Signup</h2>

                    <label>Username: <input type="text" value={username} onChange={(e) => setUsername(e.target.value)} /></label><br /><br />

                    <label>Password: <input type="password" value={password} onChange={(e) => setPassword(e.target.value)} /></label><br /><br />

                    <button onClick={() => setCurrentPage("userOptions")}>Back</button>

                    <button onClick={handleSignup}>Signup</button>

                </div>

            </center>

        );

    };

    const renderAddQuestionPage = () => {

        return (

            <center>

                <div className="addQuiz">

                    <h2>Add Quiz</h2>

                    <label>Question: <input type="text" value={question} onChange={(e) => setQuestion(e.target.value)} /></label><br /><br />

                    <label>Answer: <input type="text" value={answer} onChange={(e) => setAnswer(e.target.value)} /></label><br /><br />

                    <label>Choice 1: <input type="text" value={choice1} onChange={(e) => setChoice1(e.target.value)} /></label><br /><br />

                    <label>Choice 2: <input type="text" value={choice2} onChange={(e) => setChoice2(e.target.value)} /></label><br /><br />

                    <label>Choice 3: <input type="text" value={choice3} onChange={(e) => setChoice3(e.target.value)} /></label><br /><br />

                    <label>Choice 4: <input type="text" value={choice4} onChange={(e) => setChoice4(e.target.value)} /></label><br /><br />

                    <button onClick={() => setCurrentPage("adminLogin")}>Back</button>

                    <button onClick={handleAddQuestion}>Add Question</button>

                </div>

            </center>

        );

    };

    // const renderQuizQuestion = () => {

    //     const currentQuestion = questionData[currentQuestionIndex];

    //     if (!currentQuestion) {

    //         return null;

    //     }

    //     const choices = [currentQuestion.choice1, currentQuestion.choice2, currentQuestion.choice3, currentQuestion.choice4];

    //     const correctAnswer = currentQuestion.answer;

    //     return (

    //         <div>

    //             <h3 id ="qno">Question No: {currentQuestionIndex + 1}/10</h3>

    //             <h3>{currentQuestion.question}</h3>

    //             {choices.map((choice, index) => {

    //                 const isCorrect = choice === correctAnswer;

    //                 const buttonStyle = selectedAnswerIndex === index ? (isCorrect ? 'correct' : 'incorrect') : 'default';

    //                 return (

    //                     <button key={index} className={buttonStyle} onClick={() => handleAnswer(choice, index)}>{choice}</button>

    //                 );

    //             })}

    //         </div>

    //     );

    // };

    const renderQuizQuestion = () => {

        const currentQuestion = questionData[currentQuestionIndex];

        if (!currentQuestion) {

            return null;

        }

        const choices = [currentQuestion.choice1, currentQuestion.choice2, currentQuestion.choice3, currentQuestion.choice4];

        const correctAnswer = currentQuestion.answer;

        return (

            <div>

                <h3 id="qno">Question No: {currentQuestionIndex + 1}/10</h3>

                <h3>{currentQuestion.question}</h3>

                {choices.map((choice, index) => {

                    const isCorrect = choice === correctAnswer;

                    const buttonStyle = selectedAnswerIndex === index ? (isCorrect ? 'correct' : 'incorrect') : 'default';

                    return (

                        <button key={index} className={buttonStyle} onClick={() => handleAnswer(choice, index)}>{choice}</button>

                    );

                })}

            </div>

        );

    };

    // const handleAnswer = (selectedAnswer, selectedIndex) => {

    //     const currentQuestion = questionData[currentQuestionIndex];

    //     let isCorrect = false;

    //     if (selectedAnswer === currentQuestion.answer) {

    //         // Increment score if answer is correct

    //         setScore(prevScore => prevScore + 1);

    //         isCorrect = true;

    //     }

    //     // Update the UI to show the correct and incorrect answers

    //     setQuestionData(prevQuestionData => {

    //         const updatedQuestionData = [...prevQuestionData];

    //         updatedQuestionData[currentQuestionIndex] = { ...currentQuestion, selectedAnswer, isCorrect };

    //         return updatedQuestionData;

    //     });

    //     // Set the selected answer index

    //     setSelectedAnswerIndex(selectedIndex);

    //     // Delay the transition to the next question

    //     setTimeout(() => {

    //         if (currentQuestionIndex === 9) {

    //             // Stop the quiz

    //             setQuizStarted(false);

    //             // Display the score

    //             setCurrentPage("quizResult");

    //         } else {

    //             // Move to the next question

    //             setCurrentQuestionIndex(prevIndex => prevIndex + 1);

    //             // Reset selected answer index

    //             setSelectedAnswerIndex(null);

    //         }

    //     }, 1000);

    // };

    const handleAnswer = (selectedAnswer, selectedIndex) => {

        const currentQuestion = questionData[currentQuestionIndex];

        let isCorrect = false;

        if (selectedAnswer === currentQuestion.answer) {

            // Increment score if answer is correct

            setScore(prevScore => prevScore + 1);

            isCorrect = true;

        }

        // Update the UI to show the correct and incorrect answers

        setQuestionData(prevQuestionData => {

            const updatedQuestionData = [...prevQuestionData];

            updatedQuestionData[currentQuestionIndex] = { ...currentQuestion, selectedAnswer, isCorrect };

            return updatedQuestionData;

        });

        // Set the selected answer index

        setSelectedAnswerIndex(selectedIndex);

        // Delay the transition to the next question

        setTimeout(() => {

            if (currentQuestionIndex === 9) {

                // Stop the quiz

                setQuizStarted(false);

                // Display the score

                setCurrentPage("quizResult");

                // Update user data with score

                updateScore();

            } else {

                // Move to the next question

                setCurrentQuestionIndex(prevIndex => prevIndex + 1);

                // Reset selected answer index

                setSelectedAnswerIndex(null);

                // Reset timer for the next question

                setTimer(10);

            }

        }, 1000);

    };

    // Function to update user score

    const updateScore = () => {

        // Fetch endpoint to update user score

        fetch('http://localhost:5000/user/updateScore', {

            method: 'POST',

            headers: {

                'Content-Type': 'application/json'

            },

            body: JSON.stringify({ username, password, score }), // Pass username, password, and score

        })

            .then(res => res.json())

            .then(data => {

                console.log(data.message);

            })

            .catch(error => console.error('Error updating user score:', error));

    };

    const renderAdminDashboard = () => {

        return(

            <center>

                <h3>Admin Dashboard</h3>

                {isAdminLoggedIn && ( // Conditionally render the "Add Question" button

                    <div>

                        <button onClick={() => setCurrentPage("addQuestion")}>Add Question</button>

                        <button onClick={() => setCurrentPage("viewResults")}>View Results</button>

                    </div>

                )}

                <button className="adminBack" onClick={() => setCurrentPage("adminLogin")} style={{ position: 'absolute', bottom: 10, left: 10 }}>Back</button> {/\* Back button \*/}

            </center>

        )

    }

    const handleGoBack = () => {

        setCurrentPage("renderAdminDashboard");

    };

    const renderUserDashboard = () => {

        return (

            <center>

                <div className='quizPage'>

                    <h2>QUIZ TIME</h2>

                    <button className="quizBack" onClick={() => setCurrentPage("home")}>Back</button>

                    {!quizStarted && (

                        <button className="quizStart" onClick={handleStartQuiz}>Start Quiz</button>

                    )}

                    {quizStarted && <h3 id="timer">Timer : {timer}</h3>}

                    {quizStarted && renderQuizQuestion()}

                </div>

            </center>

        );

    };

    const QuizResultPage = () => {

        return (

            <center>

                <div>

                    <h2>You Scored {score} out of 10</h2>

                    <button onClick={() => setCurrentPage("home")}>Back to Home</button>

                </div>

            </center>

        );

    };

    return (

        <>

            {currentPage === "home" && renderHomePage()}

            {currentPage === "adminLogin" && renderAdminLoginPage()}

            {currentPage === "userOptions" && renderUserOptions()}

            {currentPage === "userLogin" && renderUserLoginPage()}

            {currentPage === "userSignup" && renderSignupPage()}

            {isAdminLoggedIn && currentPage === "renderAdminDashboard" && renderAdminDashboard()} {/\* Updated this line \*/}

            {isAdminLoggedIn && currentPage === "addQuestion" && renderAddQuestionPage()}

            {isUserLoggedIn && currentPage === "userDashboard" && renderUserDashboard()}

            {currentPage === "quizResult" && <QuizResultPage/>}

            {isAdminLoggedIn && currentPage === "viewResults" && <ResultsPage goBack={handleGoBack}/>}

        </>

    );

}

**App.css**

.d1{

  margin-top: 15%;

  background-color: rgb(50, 50, 255,0.3);

  border: solid 2px rgb(12, 1, 10);

  padding: 80px;

  width: 450px;

}

button{

  margin: 30px;

  padding: 10px;

  border: solid 1px ;

  background: rgba(142, 26, 138, 0.396);

  border-radius: 10%;

  color: aliceblue;

  font-size: large;

}

.addQuiz{

  background-color: rgb(50, 50, 255,0.3);

  border: solid 2px rgb(12, 1, 10);

  padding: 30px;

  background-size: 100vh 100vm;

  height: 493px;

}

#timer{

  display: flex;

  margin-left: 2%;

  color: red;

  top:2px;

}

#qno{

  display: flex;

  position: absolute;

  right:2px;

  top:50px;

  margin-right: 2%;

  color: rgb(90, 205, 247);

}

.quizBack{

  display: flex;

  bottom: 0%;

  position: absolute;

  background: rgba(92, 87, 92, 0.396);

  color: black;

}

.quizStart{

  margin: 50px;

  padding: 10px;

  border: solid 2px;

  color: aliceblue;

  background: rgba(142, 26, 138, 0.396);

  border-radius: 10px;

  font-size: large;

  size: 50px;

  display: flex;

  margin-top: 20%;

}

.correct {

  background-color: green;

}

.incorrect {

  background-color: red;

}

#scoretable{

  font: 25px;

  color: white;

}

**index.css**

body {

  background-image: url('https://miro.medium.com/v2/resize:fit:1400/format:webp/1\*xMuIOwjliGUPjkzukeWKfw.jpeg');

  background-repeat: no-repeat;

  background-size: cover;

  margin: 0;

  padding: 0;

  font-family: Arial, sans-serif; /\* Example font family \*/

  color: white;

  font-size: 30px;

}

code {

  font-family: source-code-pro, Menlo, Monaco, Consolas, 'Courier New',

    monospace;

}

**ResultsPage.js**

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

const ResultsPage = ({ goBack }) => {

    const [quizResults, setQuizResults] = useState([]);

    const [error, setError] = useState(null);

    useEffect(() => {

        fetchQuizResults();

    }, []);

    const fetchQuizResults = () => {

        fetch('http://localhost:5000/quiz/results')

            .then(res => res.json())

            .then(data => {

                setQuizResults(data);

            })

            .catch(error => {

                console.error('Error fetching quiz results:', error);

                setError('Failed to fetch quiz results. Please try again later.');

            });

    };

    return (

        <div id ="scoretable">

            {error && <p>Error: {error}</p>}

            <h2>Quiz Results</h2>

            <table border="1">

                <thead>

                    <tr>

                        <th>Username</th>

                        <th>Password</th>

                        <th>Score</th>

                    </tr>

                </thead>

                <tbody>

                    {quizResults.map(result => (

                        <tr key={result.\_id}>

                            <td>{result.username}</td>

                            <td>{result.password}</td>

                            <td>{result.score}</td>

                        </tr>

                    ))}

                </tbody>

            </table>

            <button onClick={goBack} style={{ position: 'absolute', bottom: 10, left: 10 }}>Back</button>

        </div>

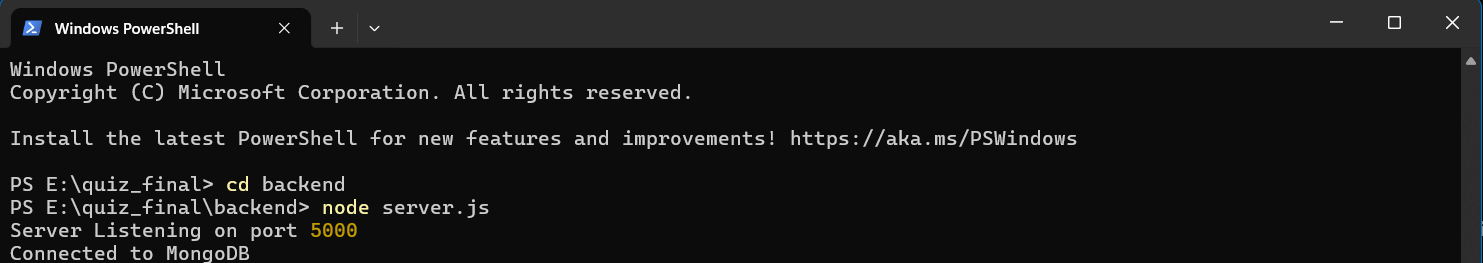
    );

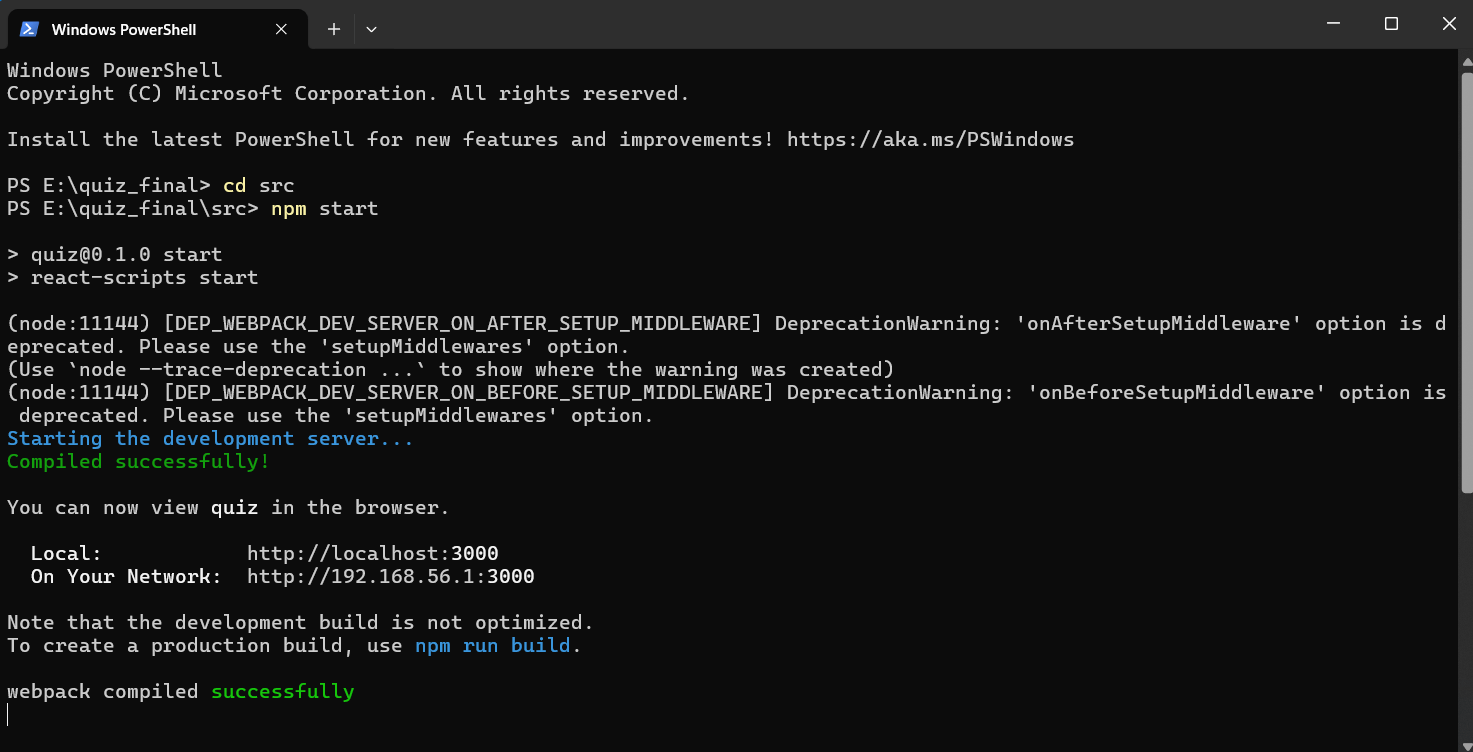
};

export default ResultsPage;

Run the server.js using “node server.js “ in it’s directory.

Run the App.js using “npm start” in it’s directory.

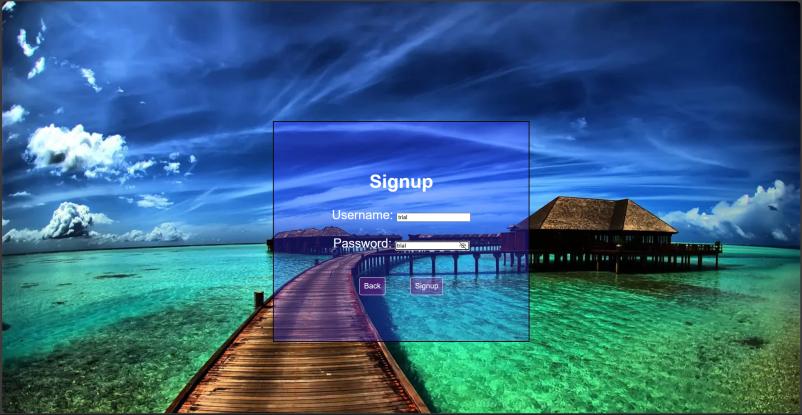


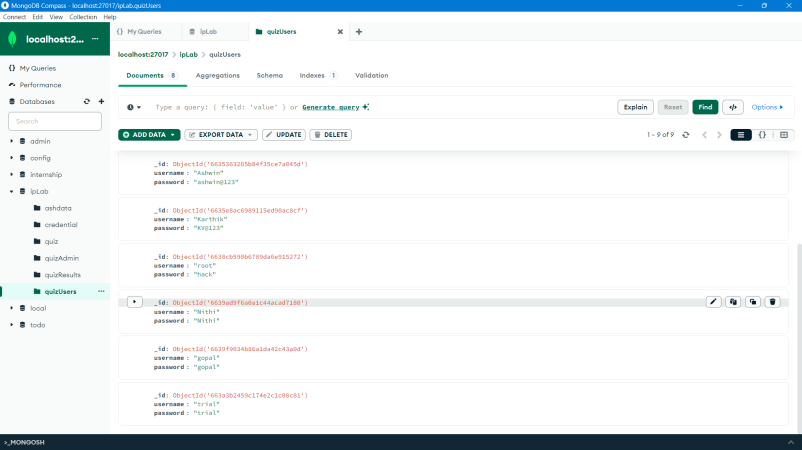


Now, the app gets deployed in the browser using the URL “http://localhost:3000”



**1. For user:-**



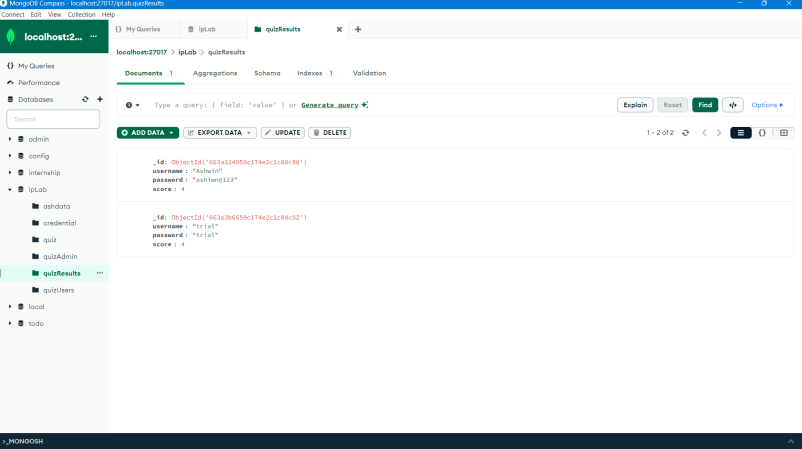












User after signing up, can start playing the game. If he/she has already played the game earlier, they can directly go for log in.

After signup/login, the user can start the test.

The timer will keep running and the page will automatically move to the next question after 10 seconds and if no option is selected.

If selected, the user will get a red/green box surrounding the option. If it is green the selected choice is correct and goes to the next question immediately. Else if it is red, the selected choice is wrong and goes to the next question immediately.

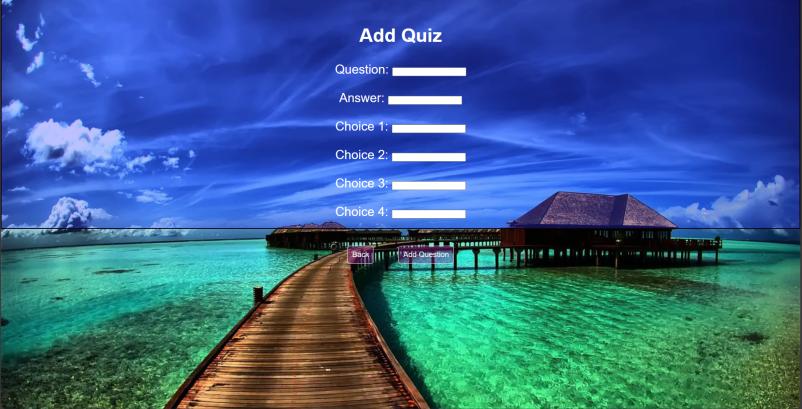
At the end of the test, the user can see the test score. The score is added to the quizResults collection of the ipLab database with the “username” , “password” and “score” attained. This information is visible to admin only and we will see below how the admin can retrieve and see the details.

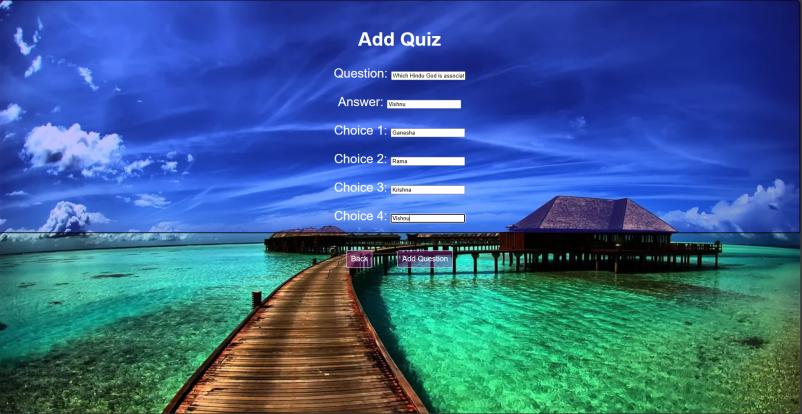
2. **For Admin:-**

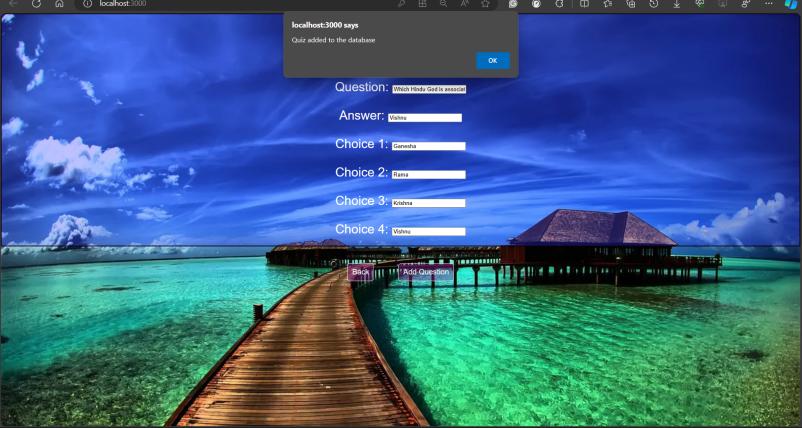
****

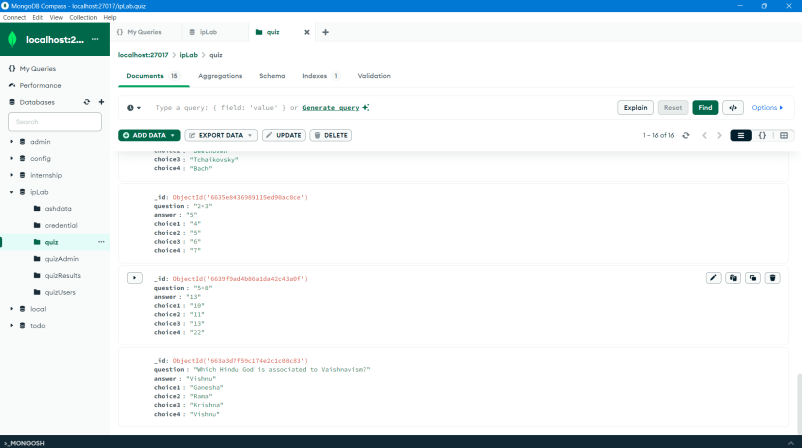
****











****



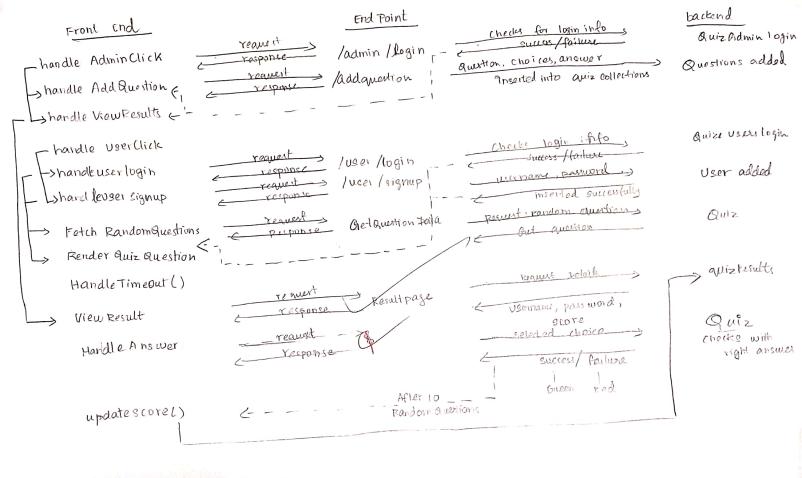


The admin logs in using the credential allotted. The there are two tabs in the Admin Dashboard. One is “Add Question” and other is “View Results”.

If the Admin selects Add Question, admin gives the question, answer and the choices. After adding, there is a window.alert() prompt that the question is added to the database. This is verified by checking the “quiz” collection of ipLab database and we see the question is added.

If the Admin selects View Results, the admin can see the Quiz results of the user in a tabular column. Each row consists of the username, password and the score attained by the user(s).

**Diagram**



**GitHub Link:-**

<https://github.com/SolitudeAsh/Quiz-app-using-MERN>

Learning Outcome:-

Best Practices:-