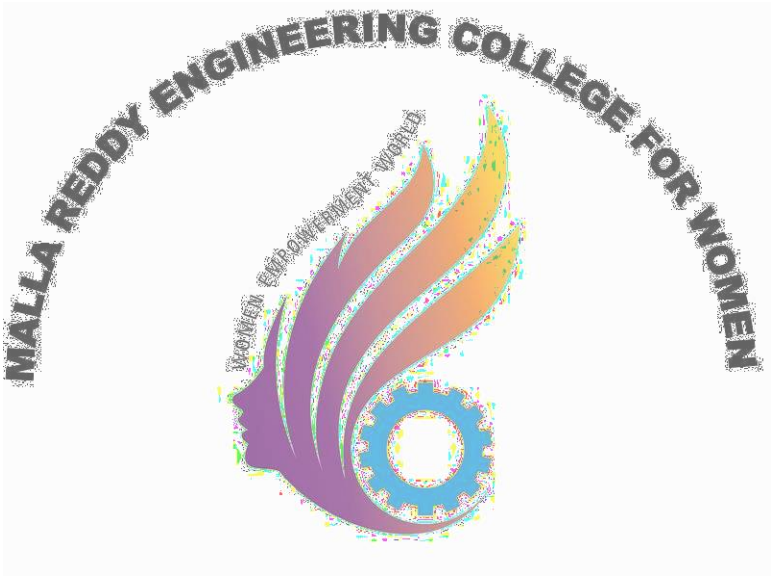
LABORATORY MANUAL

FULL STACK DEVELOPMENT (2212PC61)

III YEAR II SEM

Prepared by

Mrs. P V Aparanjini priyadarshini
Asst.Professor



DEPARTMENT OF CSE -AI&ML

MALLAREDDY ENGINEERING COLLEGE FOR WOMEN

(Autonomous Institution-UGC, Govt. of India)
Accredited by NAAC with 'A+' Grade,
Affiliated to JNTUH, Approved by AICTE, ISO 9001:2015 Certified Institute
Maisammaguda, Dhulapally, Secunderabad, Kompally-500100
2024 – 2025

COURSE NAME: FULL STACK DEVELOPMENT LAB
LAB COURSE CODE: 2212PC61

Course Objectives:

1. Usage of various front and backend Tools.
2. Understand and create applications.
3. Demonstrate and Designing of Websites can be carried out.
4. Develop web based application using suitable client side and server side code.
5. Implement web based application using effective database access.

Course outcomes:

1. Understand and apply the foundational principles of web design to create static, user-friendly interfaces. Develop skills in structuring content using HTML.
2. Gain hands-on experience in using CSS to enhance the aesthetics and responsiveness of a webpage. Learn to apply styles dynamically to improve user experience.
3. Acquire knowledge of DOM manipulation and event handling in JavaScript. Enhance interactive capabilities of web pages by responding to user actions.
4. Learn how to use jQuery for client-side validation. Develop a system that evaluates password strength to improve security awareness.
5. Understand asynchronous communication between the client and server. Learn to fetch and display data without refreshing the webpage, improving user experience.
6. Gain proficiency in building dynamic single-page applications (SPAs).

PROGRAM OUTCOMES

PO 1	**Engineering knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
PO 2	**Problem analysis**: Identify, formulate, review research literature, and analyze complex engineeringproblems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
PO 3	**Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
PO 4	**Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
PO 5	**Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
PO 6	**The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
PO 7	**Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
PO 8	**Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
PO 9	**Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
PO10	**Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
PO11	**Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
PO12	**Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

COURSE OUTCOME'S MAPPING WITH PROGRAM OUTCOME'S

CS506P C	PO1	PO2	PO3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12
CO 1	✓	✓	✓		✓						✓	
CO 2	✓		✓	✓		✓		✓				
CO 3		✓	✓		✓	✓					✓	
CO 4	✓	✓			✓						✓	

INDEX

WEEK-1 Designing of following static web pages required for an online book store web site.

1) **HOME PAGE**
2) **LOGIN PAGE**
3) **CATOLOGUE PAGE:**
   The catalogue page should contain the details of all the books available in the web site in a table.
   The details contain the following:
   a. Snap shot of Cover Page.
   b. Author Name.
   c. Publisher.
   d. Price.
   e. Add to cart button.
4) **CART PAGE**:
   a. The cart page contains the details about the books which are added to the cart.
   b. The cart page should look like this:
5) REGISTRATION PAGE:
   "*registration form* "with the following fields

1) Name (Text field)
2) Password (password field)
3) E-mail id (text field)
4) Phone number (text field)
5) Sex (radio button)
6) Date of birth (3 select boxes)
7) Languages known (check boxes – English, Telugu, Hindi, Tamil)
8) Address (text area)

- Create a file in notepad and save it as "book.html"

```
<html>
<head><title>Book</title></head>
<frameset rows="25%,75%">
<frame src="top.html" name="top" framespacing="0" scrolling="no"
frameborder="0"noresize>
<frameset cols="15%,85%">
<frame src="left.html" name="left"  framespacing="0" scrolling="auto"
frameborder="0"noresize>
<frame src="right.html" name="right" framespacing="0" scrolling="auto"
frameborder="0"noresize>
</frameset>
</frameset>
</html>
```

- Create a file in notepad and save with “top.html”

```
<html>
<head>
<title>top</title>
<style>
a:link{text-decoration:none}
a:visited {text-decoration:none;color:red}
a:hover {text-decoration:underline;color:green}
a:active {text-decoration:none;color:blue}
</style>
</head>
<body bgcolor="#fedcba" >
<p align="center">
<table height="100%" width="100%" border="0" align="center"
height="100%">
<colgroup span="5" width="20%"></colgroup>
<tr align="center"><td><img src="logo.gif" width="85%" name="logo"
alt="sitelogo"></td><td colspan="4"><h1>AMAZON BOOKS
WORLD</h1></td></tr>
<tr align="center"><td>
<a name="home" href="home.html" target="right">HOME</a></td>
<td><a name="login" href="login.html" target="right">LOGIN</a></td>
<td>
<a name="registration" href="registration.html"
target="right">REGISTRATION</a></td>
<td><a name="catalogue" href="catalogue.html"
target="right">CATALOGUE</a></td>
<td><a name="cart" href="cart.html" target="right">CART</a></td>
</tr>
</table>
</p>
</body>
</html>
```

- Create a file in notepad and save it as “left.html”

```
<html>
<head>
<style>
a:link{text-decoration:none}
```

```
a:visited {text-decoration:none;color:red}
a:hover {text-decoration:underline;color:green}
a:active {text-decoration:none;color:blue}
</style>
<title>Left</title>
</head>
<body bgcolor="#fabecd">
<p align="center"><a href="ece.html" target="right">ECE</a></p >
<p align="center"><a href="eee.html" target="right">EEE</a></p >
<p align="center"><a href="mech.html" target="right">MECH</a></p >
<p align="center"><a href="csit.html" target="right">CSE</a>
</body>
</html>
```

- Create a file in notepad and save it as “right.html”

```
<html>
<head>
<title>right frame
</title>
</head>
<body bgcolor="#abcdef">
<font color='#123123' size='+3'>welcome to amazon books world</font>
<br>
This site provides the books information related to various categories.
</body>
</html>
```

- Create a file in notepad and save it as “home.html”

```
<html>
<head>
<title>home page
</title>
</head>
<body bgcolor="#abcdef">
<font color='#123123' size='+3'>welcome to amazon books world</font>
<br>
This site provides the books information related to various categories.
</body>
</html>
```

- Create a in notepad and save it as “login.html”

```
<html>
<head>
<title>login form</title>
</head>
<body bgcolor="#abcdef">
<h3 align="center">login into the site</h3>
<form  method="post">
<table align="center">
<tr><td>name</td><td><input type="text" name="uname"></td></tr>
<tr><td>password</td><td><input type="password" name="pass"></td></tr>
   <tr align="center"><td><input type="submit" value=" login "></td><td><input
type="reset"  value="  reset  "></td></tr>
</table>
</form>
</body>
</html>
```

- Create a in notepad and save it as “registration.html”

```
<html>
<head>
<title>Registration form</title>
</head>
<body bgcolor="orange">
<h3 align="center" color="pink">Registration form</h3>
<center>
<form name="regform"  method="post">
<table cellspacing="15">
<tr><td>NAME</td><td><input type="text" name="name"  size="25"></td></tr>
<tr><td>PASSWORD</td><td><input type="password" name="pass" size="25">
</td></tr>
<tr><td>PHONE NUMBER</td><td><input type="text" maxLength="10"
name="phno"size="25"></td></tr>
<tr><td>E-MAIL</td><td><input type="text" name="emai" size="25">
</td></tr>

<tr><td>GENDER</td><td>male <input type="radio" name="gender"
value="male"checked="checked"> female <input type="radio" name="gender"
value="female"></td></tr>
```

```
<tr><td>DATE OF BIRTH</td><td>day<select name="day">
<option value="1">1</option>
<option value="2">2</option>
</select>  month  <select name="month">
<option value="jan">jan</option>
<option value="feb">feb</option>
</select>  year<select name="year">
<option value="1990">1990</option>
<option value="1991">1991</option></select> </td></tr>

<tr><td>LANGUAGES KNOWN</td>
<td>TELUGU  <input type="checkbox" name="telugu" value="telugu"><br>
ENGLISH <input type="checkbox" name="english" value="english"><br>
HINDI  <input type="checkbox"  name="hindi" value="hindi">
</td>
</tr>
<tr><td>ADDRESS</td><td><textarea rows="3" cols="25" name="address"
wrap="soft"></textarea></td>
</tr>
</table>
<br><br>
<input type="submit" align="center" value="send"> <input
type="reset"align="center" value="cancel">
</form>
</center>
</body>
</html>
```

- Create a in notepad and save it as “catalogue.html”

```
<html>
<head>
<title>catalogue page</title>
</head>
<body bgcolor="#abcdef">

<table width="100%">
<tr><td><img src="xml.jpg"></td>
    <td>Book: XML Bible<br>
Author: Winston<br>
    Publication: Wiely
  </td>
  <td>$ 40.5</td>
```

```
    <td><img src="cartbutton.jpg"></td>
</tr>
<tr><td><img src="html.jpg"></td>
        <td>Book : HTML in 24 hours<br>
        Author : Sam Peter<br>
        Publication : Sam publication
    </td>
    <td>$ 50</td>
    <td><img src="cartbutton.jpg"></td>
</tr>
<tr><td><img src="ai.jpg"></td>
        <td>
        Book :  AI<br>
        Author : S.Russel<br>
        Publication : Princeton hall
    </td>
    <td>$ 63</td>
    <td><img src="cartbutton.jpg"></td>
</tr>
<tr><td><img src="java.jpg"></td>
        <td>Book : Java 2<br>
        Author : Watson<br>
        Publication : BPB publications
    </td>
    <td>$ 35.5</td>
    <td><img src="cartbutton.jpg"></td>
</tr>
</table>
</body>
</html>
```

- Create a in notepad and save it as “cart.html”

```
<html>
<head>
<title>cart page</title>
</head>
<body bgcolor="#abcdef">
<table align="center" width="75%">
<tr align="center"><th>Book Name</th>
    <th>Price</th>
    <th>Quantity</th>
```

```
    <th>Amount</th>
</tr>
<tr align="center"><td>Java2</td>
    <td>$35.5</td>
    <td>2</td>
    <td>$70</td>
</tr>
<tr align="center"><td>XML Bible</td>
    <td>$40.5</td>
    <td>1</td>
    <td>$40.5</td>
</tr>
<tr align="right"><th colspan="4">Total amount -$110</th>
</table>
</body>
</html>
```

- Create a in notepad and save it as “csit.html”

```
<html>
<head>
<title>csit books</title>
</head>
<body bgcolor="skyblue">
<h3 align="center">Computer Science & IT Books</h3>
<table width="100%">
<tr><td><img src="xml.jpg"></td>
        <td>Book: Mobile Computing<br>
  Author: Winston<br>
         Publication: Wiely
    </td>
    <td>$ 40.5</td>
    <td><img src="cartbutton.jpg"></td>
</tr>
<tr><td><img src="html.jpg"></td>
          <td>
          Book : Computer Networks<br>
          Author : Sam Peter<br>
                Publication : Sam publication
         </td>
         <td>$ 50</td>
         <td><img src="cartbutton.jpg"></td>
         </tr>
```

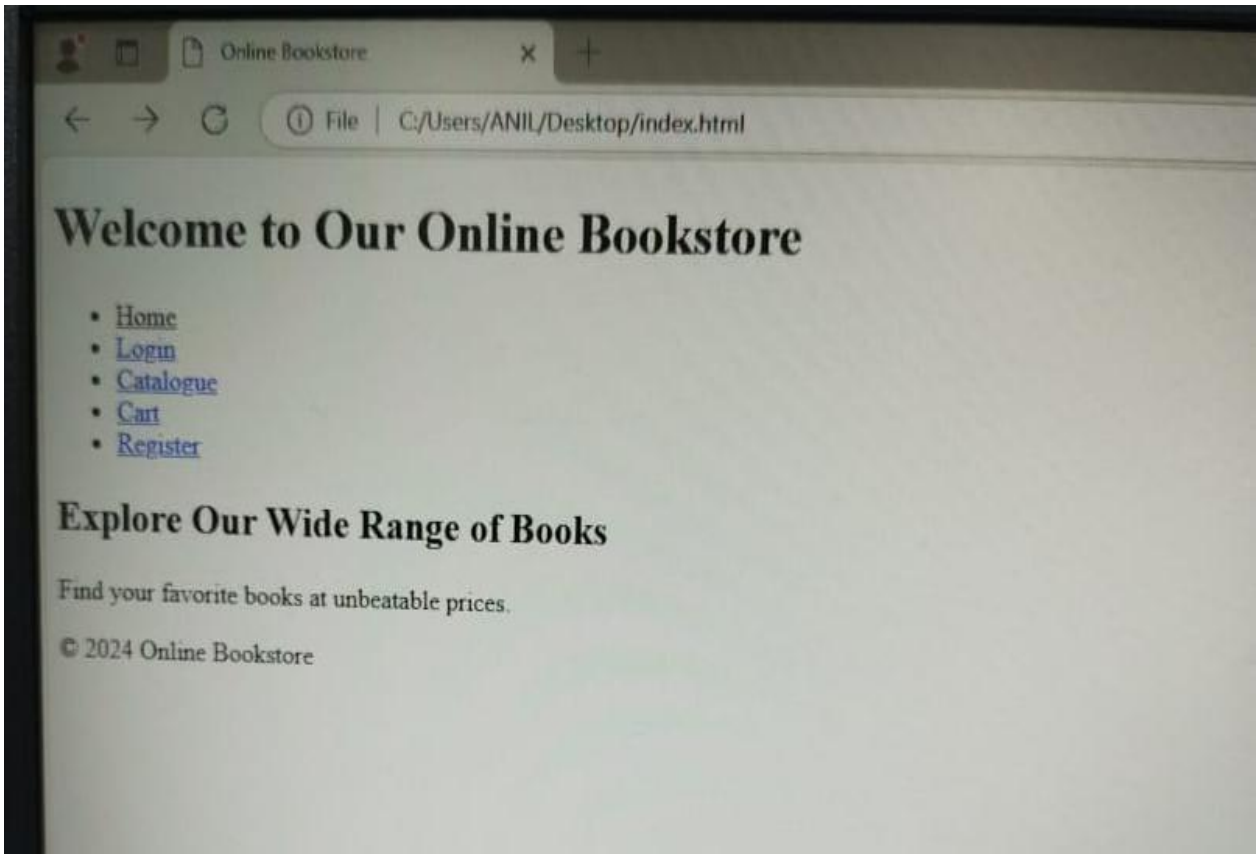
```
<tr><td><img src="ai.jpg"></td>
<td>Book : Computer
Communications<br>Author :
S.Russel<br>
Publication : Princeton hall
</td>
<td>$ 63</td>
<td><img src="cartbutton.jpg"></td>
</tr>
<tr><td><img src="java.jpg"></td>
<td>Book : Web
Design<br> Author :
Watson<br>
Publication : BPB
publications
</td>
<td>$ 35.5</td>
<td><img src="cartbutton.jpg"></td>
</tr>
</body>
</html>
```

- Create a in notepad and save it as “eee.html”

```
<html>
<head>
<title>eee books</title>
</head>
<body bgcolor="skyblue">
<h3 align="center">Electrical and Electronics Eng. Books</h3>
<table width="100%">
<tr><td><img src="xml.jpg"></td>
<td>Book: Machines<br>Author: Winston<br>
Publication: Wiely
</td>
<td>$ 40.5</td>
<td><img src="cartbutton.jpg"></td>
</tr>
<tr><td><img src="html.jpg"></td>
<td>Book : Power
Electronics<br>Author
: Sam Peter<br>
```

```
            Publication : Sam
            publication
</td>

                <td>$ 50</td>
                <td><img src="cartbutton.jpg"></td>
                </tr>
                <tr><td><img src="ai.jpg"></td>
            <td>Book : Transmision
                Systems<br>Author :
                S.Russel<br>
                Publication : Princeton hall
                </td>
                <td>$ 63</td>
                <td><img src="cartbutton.jpg"></td>
                </tr>
                <tr><td><img src="ai.jpg"></td>
            <td>Book : Digital Logic Design<br>Author : S.Russel<br> Publication :
                Princeton hall
                </td>
                <td>$ 63</td>
                <td><img src="cartbutton.jpg"></td>
                </tr>
                <tr><td><img src="java.jpg"></td>
            <td>Book : Signal
                Processing<br>
                Author :
                Watson<br>
                Publication : BPB
                publications
                </td>
                <td>$ 35.5</td>
                <td><img src="cartbutton.jpg"></td>
                </tr>
                </body>
                </html>
```

OUTPUT:0


Online Bookstore
File | C:/Users/ANIL/Desktop/index.html
Welcome to Our Online Bookstore
Home
Login
Catalogue
Cart
Register
Explore Our Wide Range of Books
Find your favorite books at unbeatable prices.
© 2024 Online Bookstore

WEEK-2: Designing a webpage using CSS Which includes different styles.

a) **Use different font, styles Create file in notepad and save with "fonts.html"**

```
<html>
<head>
<link rel="stylesheet" type="text/css" href="test1.css">
</head>
<body>
<h1>This header is red</h1>
<h2>This header is blue</h2>
<p> This text is normal</p>
</body>
</html>
```

Create file in notepad and save with "test1.css"

```
h1{color:red;
font-size:22px; font-
family:arial;
text-decoration:underline}
h2{color:blue;font-size:16px} p{font-
family:arial;font-size:30px}
```

b) **Setting a background image for both the page and single elements on the page**

Create file in notepad and save with "backgrounds.html"

```
<html>
<head>
<style type="text/css">
body{ background-image:url("flower.jpg");
background-repeat:no-repeat; }
</style>
</head>
<body>
<center>
<h1>Life is beautiful!!!</h1>
<h2>Strength is live</h2>
<h2>weakness is death</h2>
</center>
</body>
</html>
```

c) **Setting a background image for both the page and single elements on the page**

Create file in notepad and save with “backgrounds.html”

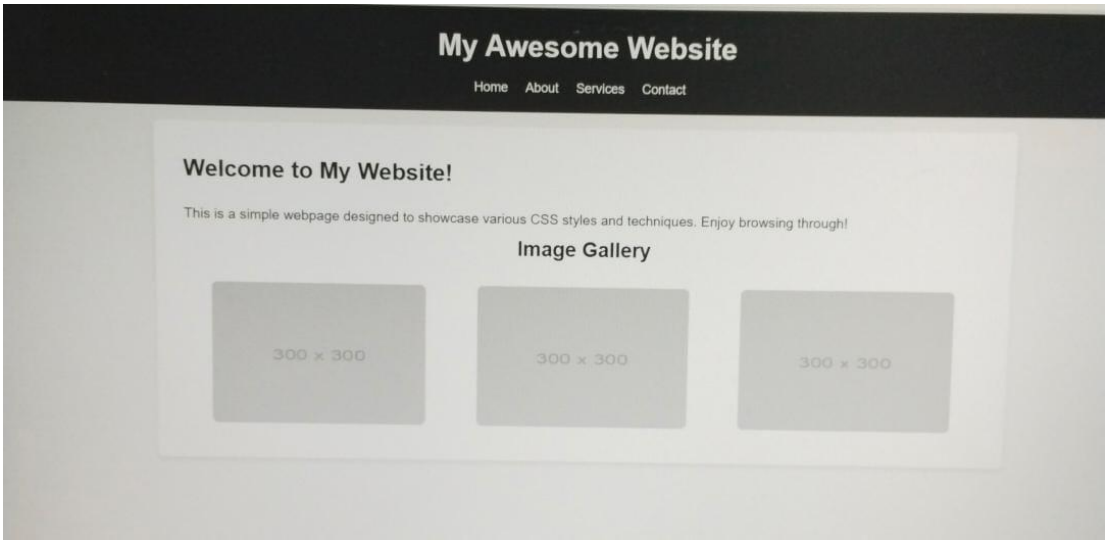
```
<html>
<head>
<style type="text/css">
body{ background-image:url("flower.jpg");
background-repeat:no-repeat; }
</style>
</head>
<body>
<center>
<h1>Life is beautiful!!!</h1>
<h2>Strength is live</h2>
<h2>weakness is death</h2>
</center>
</body>
</html>
```

d) **Controlling the repetition of the image with the background-repeat property.**

Create file in notepad and save with “bg_property.html”

```
<html>
<head>
<style type="text/css">
body{ background-image:url("flower1.jpg");
background-repeat:repeat;
}
h1{ color:green;
font-size:35px;
}
</style>
</head>
<body>
<center>
<h1>Life is beautiful!!!</h1>
<h2>Strength is live</h2>
<h2>weakness is death</h2>
</center>
</body>
</html>
```

OUTPUT:



Week-3. Write a JavaScript to implement the following various events.

a)Mouse

b)Keyboard

c)Form

d)Window

click()

```
<html><head>  <title>mouse events</title>
<script language="javascript">

function fun1()

{
alert("hai");
}
function fun2()
{
alert("bye");
}
</script>
</head>
<body>
<button onclick="fun1()">onClickme</button>
<button ondblclick="fun2()">ondblClickme</button>
</body></html>
```

Mouseover()

```
<html>
<head>
<h1> Javascript Events </h1>
<script >
        function mouseoverevent()
        {
                alert("This is mouseover event");
        }
</script>
</head>
<body>
          <p onmouseover="mouseoverevent()"> Keep cursor over me</p>
</body>
</html>
```

Mouseout()

```
<html>
<head>
<title>mouse events</title>
<script language="javascript">

function fun2()

{
alert("bye");
}
</script>
</head>
<body>
<button  onmouseout="fun2()" >onClickme</button>
</body>
</html>
```

Mousedown()

```
<html>
  <head>
    <script>
                function sayHello()
            {
              alert("You clicked here.")
            }
    </script>
  </head>
  <body>
    <p onmousedown = "sayHello()">The onmousedown event triggers when a
```

```
mouse button is pressed</p>
  </body>
</html>
```

Mouseup()

```
<html>
  <head>
    <script>

      { function sayHello()
      alert("Mouse Up")
    }

    </script>
  </head>
  <body>
    <p onmouseup = "sayHello()">This is demo text for mouseup event.</p>
  </body>
</html>
```

Keydown()

```
<html>
<script>
function myFunction()
{
  document.getElementById("demo").style.backgroundColor = "red";
}
</script>
<body>
<input type="text" id="demo" onkeydown="myFunction()">
</body>
</html>
```

Keyup()

```
<html>
<p>Type something: </p>
<script>
function handleKeyup()
{
document.write('onkeyup event occurred.');
}
</script>
<body>
<input type="text" onkeyup="handleKeyup()">
</body>
</html>
```

Focus()

```
<html>
<head> Javascript Events</head>
<body>
<h2> Enter something here</h2>
<input type="text" id="input" onfocus="focusevent()"/>
<script>
        function focusevent()
        {
                document.getElementById("input").style.background="blue";
        }
</script>
</body>
</html>
```

Blur()

```
<html>
<script>
function focusFunction()
 {
document.getElementById("myInput").style.background = "yellow";
}
function blurFunction()
{
  document.getElementById("myInput").style.background = “white";
}
</script>
<body>
<p>When you enter the input field, a function is triggered which sets the
```

background color to yellow. When you leave the input field, a function is triggered which sets the background color to white.</p>

Enter your name: <input type="text" id="myInput" onfocus="focusFunction()" onblur="blurFunction()">

</body>
</html>

Onsubmit()

```
<html>
<body>
<form onsubmit="myFunction()">
<input type="text" placeholder="UserName“ required=“”><br>
<input type="text"><br>
<input type="text"><br>
<input type=“text"><br>
<input type="submit"  value="SUBMIT">
</form>
</body>
<script>
  function myFunction()
    {
       alert("The form is succesfully submitted");
    }
</script>
</html>
```

Load() and unload()

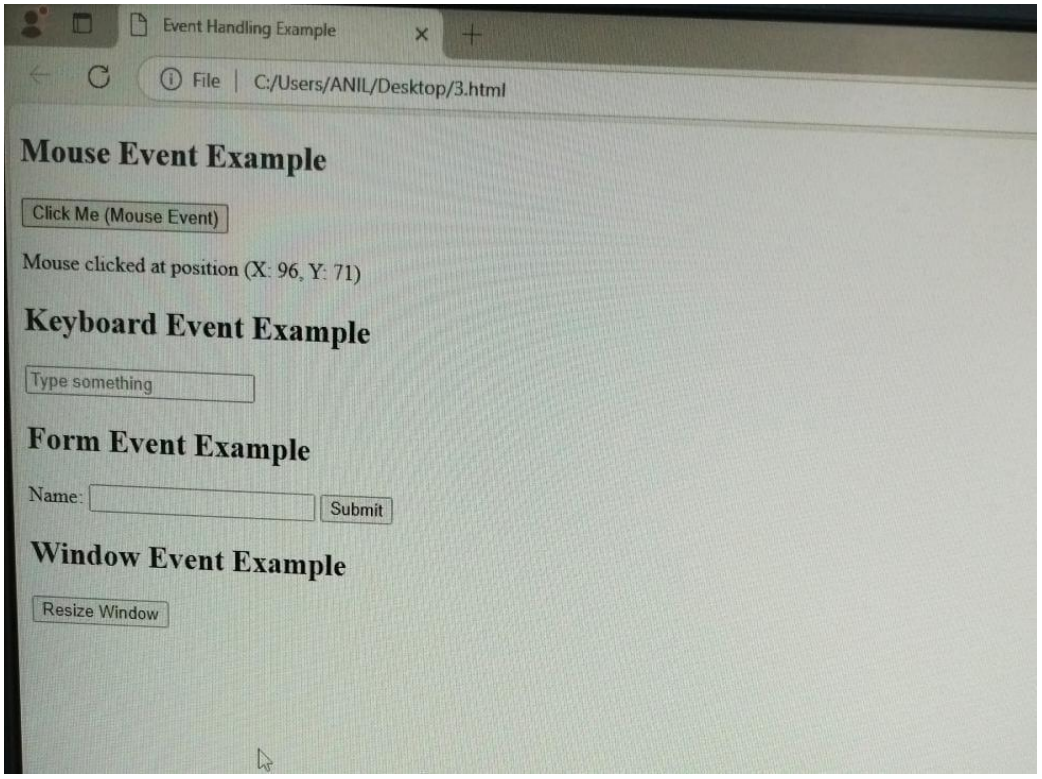
```
<html>
<body onload="alert('welcome')" onunload="alert('thankyou')">

hello

</body>
</html>
```

Resize()

```
<html>
<body onresize="myFunction()">
<script>
function myFunction() {
alert("You have changed the size of the browser window!");
```

```
}
</script>
</body>
</html>
```

OUTPUT:



Week-4. Write a program to create and Build a Password Strength Check using JavaScript

HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Password Strength Checker</title>
  <link rel="stylesheet" href="styles.css">
</head>
<body>
  <div class="container">
    <h1>Password Strength Checker</h1>
    <input
      type="password"
      id="password"
      placeholder="Enter your password"
      oninput="checkPasswordStrength()">
    <div id="strength-indicator" class="indicator">
      <span id="weak"></span>
      <span id="medium"></span>
      <span id="strong"></span>
    </div>
    <p id="strength-text"></p>
  </div>
  <script src="script.js"></script>
</body>
</html>
```

CSS **(styles.css)**

```
body {
  font-family: Arial, sans-serif;
  display: flex;
  justify-content: center;
  align-items: center;
  height: 100vh;
  margin: 0;
  background-color: #f5f5f5;
}

.container {
  text-align: center;
  background: #ffffff;
  padding: 20px;
  border-radius: 8px;
  box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);
}

input {
  width: 100%;
  padding: 10px;
  margin-bottom: 15px;
  border: 1px solid #ccc;
  border-radius: 5px;
  font-size: 16px;
}
.indicator {
  display: flex;
  justify-content: space-between;
  margin-bottom: 10px;
}

.indicator span {
width: 30%;
height: 10px;
background: #ccc;
  border-radius: 5px;
  transition: background 0.3s;
}
```

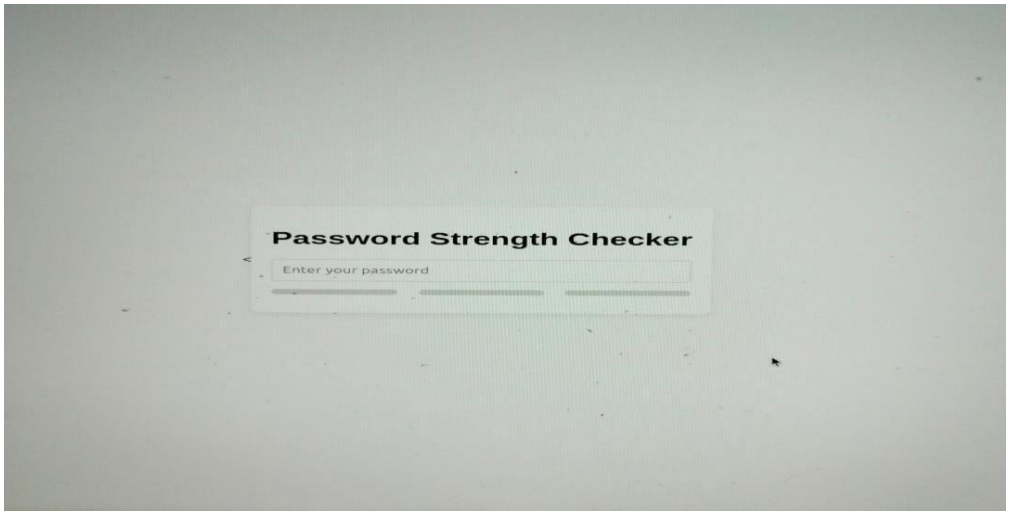
```
#strength-text {
  font-size: 14px;
  font-weight: bold;
  color: #666;
}

.weak {
  background: #ff4d4d !important;
}

.medium {
  background: #ffc107 !important;
}

.strong {
  background: #4caf50 !important;
}
```

JavaScript **(script.js)**

```
function checkPasswordStrength() {
  const password = document.getElementById('password').value;
  const strengthIndicator = document.getElementById('strength-indicator');
  const strengthText = document.getElementById('strength-text');
  const [weak, medium, strong] = strengthIndicator.children;

  // Reset styles
  weak.classList.remove('weak');
  medium.classList.remove('medium');
  strong.classList.remove('strong');
  strengthText.textContent = '';

// Define password strength rules
  const hasUpperCase = /[A-Z]/.test(password);
  const hasLowerCase = /[a-z]/.test(password);
  const hasNumbers = /[0-9]/.test(password);
  const hasSpecialChars = /[@$!%*?&]/.test(password);
  const isLongEnough = password.length >= 8;

let strength = 0;
```

```
if (hasUpperCase) strength++;
  if (hasLowerCase) strength++;
  if (hasNumbers) strength++;
  if (hasSpecialChars) strength++;
  if (isLongEnough) strength++;

  // Update UI based on strength
  if (strength <= 2) {
  weak.classList.add('weak');
    strengthText.textContent = 'Weak';
    strengthText.style.color = '#ff4d4d';
  } else if (strength <= 4) {
    weak.classList.add('weak');
    medium.classList.add('medium');
    strengthText.textContent = 'Medium';
    strengthText.style.color = '#ffc107';
  } else {
    weak.classList.add('weak');
    medium.classList.add('medium');
    strong.classList.add('strong');
    strengthText.textContent = 'Strong';
    strengthText.style.color = '#4caf50';
  }
}
```

OUTPUT:



Week-5. Write a program using string function and operators in JavaScript..

Program: String Manipulation and Operations

```
// Function to demonstrate string functions and operators
function stringOperations() {
  const str1 = "Hello, ";
  const str2 = "World!";
  const str3 = "  JavaScript is fun.  ";

  // 1. String concatenation using '+'
  const combined = str1 + str2;
  console.log("1. Concatenated String:", combined);

  // 2. String length property
  console.log("2. Length of String:", combined.length);

  // 3. Convert to uppercase
  console.log("3. Uppercase:", combined.toUpperCase());

  // 4. Convert to lowercase
  console.log("4. Lowercase:", combined.toLowerCase());

  // 5. Trim whitespaces
  console.log("5. Trimmed String:", str3.trim());

// 6. Substring extraction
  console.log("6. Substring (0 to 5):", combined.substring(0, 5));

  // 7. Check if string includes a substring
  console.log("7. Includes 'World':", combined.includes("World"));

  // 8. Find index of a character
  console.log("8. Index of 'o':", combined.indexOf("o"));
```

```
// 9. Replace a substring
  const replaced = combined.replace("World", "JavaScript");
  console.log("9. Replaced String:", replaced);

  // 10. Split the string into an array
  const splitString = str3.trim().split(" ");
  console.log("10. Split String into Array:", splitString);

  // 11. Repeat a string
  console.log("11. Repeated String:", str1.repeat(3));

  // 12. Compare strings
  const comparison = str1 > str2 ? "str1 is greater" : "str2 is greater or equal";
  console.log("12. String Comparison:", comparison);

  // 13. Concatenation using template literals
  const greeting = `${str1.trim()} JavaScript Enthusiast!`;
  console.log("13. Greeting with Template Literals:", greeting);
}

// Call the function
stringOperations();
```

OUTPUT:

```
1. Concatenated String: Hello, World!
2. Length of String: 13
3. Uppercase: HELLO, WORLD!
4. Lowercase: hello, world!
5. Trimmed String: JavaScript is fun.
6. Substring (0 to 5): Hello
7. Includes 'World': true
8. Index of 'o': 4
9. Replaced String: Hello, JavaScript!
10. Split String into Array: [ 'JavaScript', 'is', 'fun.' ]
11. Repeated String: Hello, Hello, Hello,
12. String Comparison: str2 is greater or equal
13. Greeting with Template Literals: Hello, JavaScript Enthusiast!
```

Week-6 : Write a program for sending request to a server by using Angular.

Steps to Send a Request to a Server

1. **Install HttpClientModule**: Ensure you import and set up `HttpClientModule` in your Angular app.
2. **Create a Service**: Use Angular's dependency injection to encapsulate API calls.
3. **Use the Service in a Component**: Call the service from a component to interact with the server.

`Code Implementation`

app.module.ts

First, import the `HttpClientModule` in your app's main module.

```
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import { AppComponent } from './app.component';
import { HttpClientModule } from '@angular/common/http';

@NgModule({
  declarations: [
  AppComponent
  ],
  imports: [
    BrowserModule,
    HttpClientModule // Import HttpClientModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

api.service.ts

Create a service to handle the HTTP requests.

```
import { Injectable } from '@angular/core';
import { HttpClient } from '@angular/common/http';
import { Observable } from 'rxjs';
@Injectable({
```

```
providedIn: 'root',
})
export class ApiService {
  private apiUrl = 'https://jsonplaceholder.typicode.com/posts'; // Example API endpoint

  constructor(private http: HttpClient) {}

  // Method to send a GET request
  getPosts(): Observable<any> {
  return this.http.get(this.apiUrl);
  }
  // Method to send a POST request
  createPost(data: any): Observable<any> {
  return this.http.post(this.apiUrl, data);
  }
}
```

app.component.ts

Use the service to send a request and display the response.

```
import { Component, OnInit } from '@angular/core';
import { ApiService } from './api.service';

@Component({
  selector: 'app-root',
  template: `
    <div>
      <h1>Posts from Server:</h1>
      <ul>
        <li *ngFor="let post of posts">{{ post.title }}</li>
      </ul>

      <h2>Create a New Post</h2>
      <button (click)="addPost()">Send POST Request</button>
    </div>`,
styleUrls: ['./app.component.css'],
})
export class AppComponent implements OnInit {
  posts: any[] = [];
```

```
constructor(private apiService: ApiService) {}

  ngOnInit() {
    // Fetch posts on component initialization
    this.apiService.getPosts().subscribe((data) => {
    this.posts = data;
      console.log('GET Response:', data);
    });
  }

  addPost() {
    const newPost = {
      title: 'New Post',
      body: 'This is a new post.',
      userId: 1,
    };

    // Send a POST request
    this.apiService.createPost(newPost).subscribe((response) => {
    console.log('POST Response:', response);
    });
  }
}
```

OUTPUT

```
Posts from Server:
- sunt aut facere repellat provident occaecati excepturi optio reprehenderit
- qui est esse
- eum et est occaecati
- nesciunt quas odio
- dolorem eum magni eos aperiam quia
- magnam facilis autem
- dolorem dolore est ipsam
- nesciunt iure omnis dolorem tempora et accusantium
- optio molestias id quia eum
- temporibus molestiae aut atque
```

Week-7: Develop an Angular JS application that displays a list of shopping items. Allow users to add and remove items from the list using directives and controllers.
Note: The default values of items may be included in the program.

Step 1: HTML list based on the items of an array

```
<script>
var app = angular.module("myShoppingList", []);
app.controller("myCtrl", function($scope) {
  $scope.products = ["Milk", "Bread", "Cheese"];
});
</script>

<div ng-app="myShoppingList" ng-controller="myCtrl">
  <ul>
    <li ng-repeat="x in products">{{x}}</li>
  </ul>
</div>
```

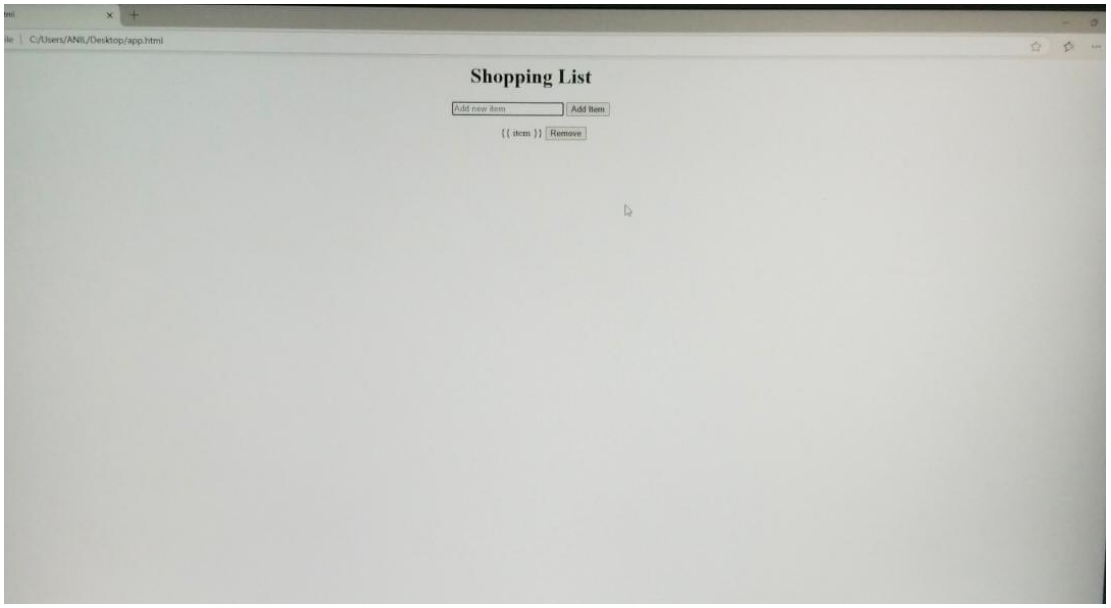
Step 2: Adding Items

```
<script>
var app = angular.module("myShoppingList", []);
app.controller("myCtrl", function($scope) {
  $scope.products = ["Milk", "Bread", "Cheese"];
  $scope.addItem = function () {
    $scope.products.push($scope.addMe);
  }
});
</script>

<div ng-app="myShoppingList" ng-controller="myCtrl">
  <ul>
    <li ng-repeat="x in products">{{x}}</li>
  </ul>
  <input ng-model="addMe">
  <button ng-click="addItem()">Add</button>
</div>
```

Step 3: Removing Items

```
<script>
var app = angular.module("myShoppingList", []);
app.controller("myCtrl", function($scope) {
  $scope.products = ["Milk", "Bread", "Cheese"];
  $scope.addItem = function () {
    $scope.products.push($scope.addMe);
  }
  $scope.removeItem = function (x) {
    $scope.products.splice(x, 1);
  }
});

</script>

<div ng-app="myShoppingList" ng-controller="myCtrl">
  <ul>
    <li ng-repeat="x in products">
      {{x}}<span ng-click="removeItem($index)">&times;</span>
    </li>
  </ul>
  <input ng-model="addMe">
  <button ng-click="addItem()">Add</button>
</div>
```

OUTPUT:



Week-8. Write a program to create a simple calculator Application using React JS.

- **Create a calculatorTitle.js file for showing the title of the calculator and paste the code given below for this file.**

```
//calculatorTitle.js File

import React from "react"; // Import React (Mandatory Step)

// Create Functional Component.
// Takes title as props.value.
const CalculatorTitle = (props) => {
        return (
        <div className="calculator-title">{props.value}</div>
        );
};
export default CalculatorTitle; // Export Calculator Title
```

- **Now create a file outputScreenRow.js for taking input and showing the output of the calculation.**

```
// outputScreenRow.js File
import React from "react"; // Import React (Mandatory Step)

// Functional Component.
// Used to show Question/Answer.
const OutputScreenRow = () => {
    return (
        <div className="screen-row">
            <input type="text" readOnly />
        </div>
    );
};
export default OutputScreenRow; // Export Output Screen Row
```

- **Create an outputScreen.js file and import the outputScreenRow.js file.**

```
// outputScreen.js File
import React from "react"; // Import React (Mandatory Step).
import OutputScreenRow from "./outputScreenRow.js"; // Import Output Screen Row.

// Functional Component.
```

```
// Use to hold two Screen Rows.
const OutputScreen = () => {
        return (
                <div className="screen">
                        <OutputScreenRow />
                        <OutputScreenRow />
                </div>
        );
};
export default OutputScreen; // Export Output Screen.
```

- **Create a button.js file and paste the code given below.**

```
// button.js File
import React from "react"; // Import React (Mandatory Step)

// Create our Button component as a functional component.
const Button = (props) => {
        return (
        <input type="button" value={props.label} />
        );
};
export default Button; // Export our button component
```

- **Now create a calculator.js file and import calculatorTitle.js, outputScreen.js, and button.js files.**

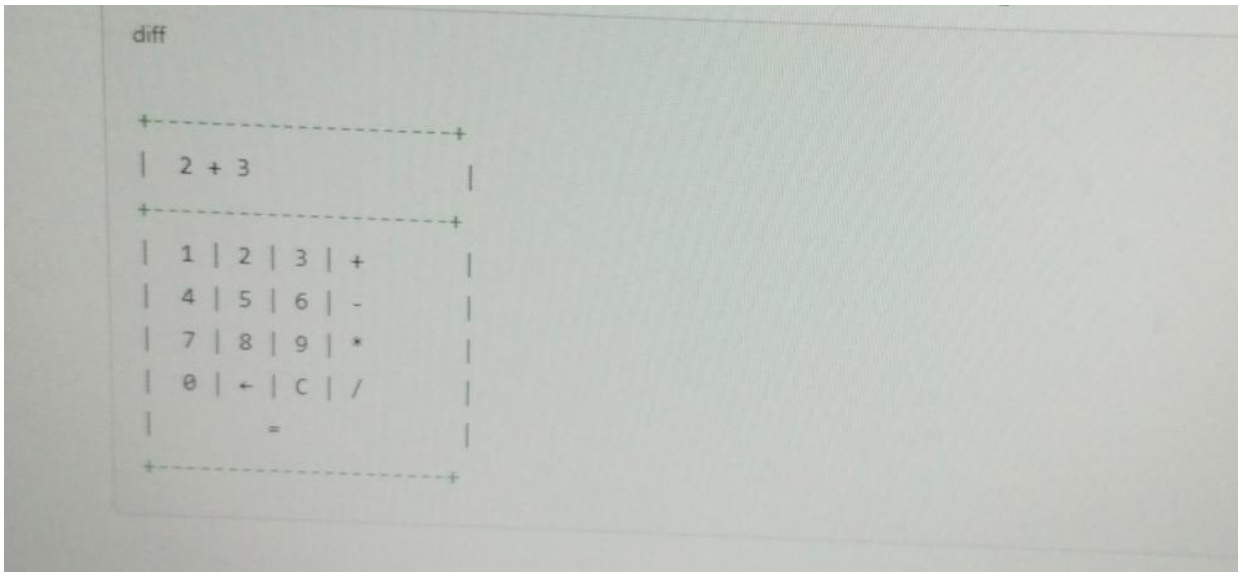
```
// calculator.js File
// Imports.
import React from "react";
import CalculatorTitle from "./calculatorTitle.js";
import OutputScreen from "./outputScreen.js";
import Button from "./button.js";

class Calculator extends React.Component {
        render() {
                return (
                        <div className="frame">
                                <CalculatorTitle value="MRCET Calculator" />
                                <div class="mainCalc">
                                        <OutputScreen />
                                        <div className="button-row">
                                                <Button label={"Clear"} />
                                                <Button label={"Delete"} />
```

```
                                        <Button label={"."} />
                                        <Button label={"/"} />
                                    </div>
                                    <div className="button-row">
                                        <Button label={"7"} />
                                        <Button label={"8"} />
                                        <Button label={"9"} />
                                        <Button label={"*"} />
                                    </div>
                                    <div className="button-row">
                                        <Button label={"4"} />
                                        <Button label={"5"} />
                                        <Button label={"6"} />
                                        <Button label={"-"} />
                                    </div>
                                    <div className="button-row">
                                        <Button label={"1"} />
                                        <Button label={"2"} />
                                        <Button label={"3"} />
                                        <Button label={"+"} />
                                    </div>
                                    <div className="button-row">
                                        <Button label={"0"} />
                                        <Button label={"="} />
                                    </div>
                                </div>
                            </div>
                        );
                    }
                }
                export default Calculator; // Export Calculator Component
```

- Inside the **index.js** file import, the calculator.js file.

```
//index.js File
import React from "react";
import ReactDOM from "react-dom";
import Calculator from "./components/calculator.js";

// Render the Calculator to the Web page.
ReactDOM.render(<Calculator />, document.getElementById("root"));
```

OUTPUT



Week-9. Write a program to create a voting application using React JS.

Src/App.js

```
import React,{Component} from 'react';
import './App.css';

class App extends Component{
    constructor(props){
        super(props);
        this.state = {
            languages : [
                {name: "Php", votes: 0},
                {name: "Python", votes: 0},
                {name: "Go", votes: 0},
                {name: "Java", votes: 0}
            ]
        }
    }

    vote (i) {
        let newLanguages = [...this.state.languages];
        newLanguages[i].votes++;
        function swap(array, i, j) {
            var temp = array[i];
            array[i] = array[j];
            array[j] = temp;
        }
        this.setState({languages: newLanguages});

    }
```

```
render(){
    return(
        <>
            <h1>Vote Your Language!</h1>
            <div className="languages">
                {
                    this.state.languages.map((lang, i) =>
                        <div key={i} className="languageName">
                            {lang.name}
                        </div>
```

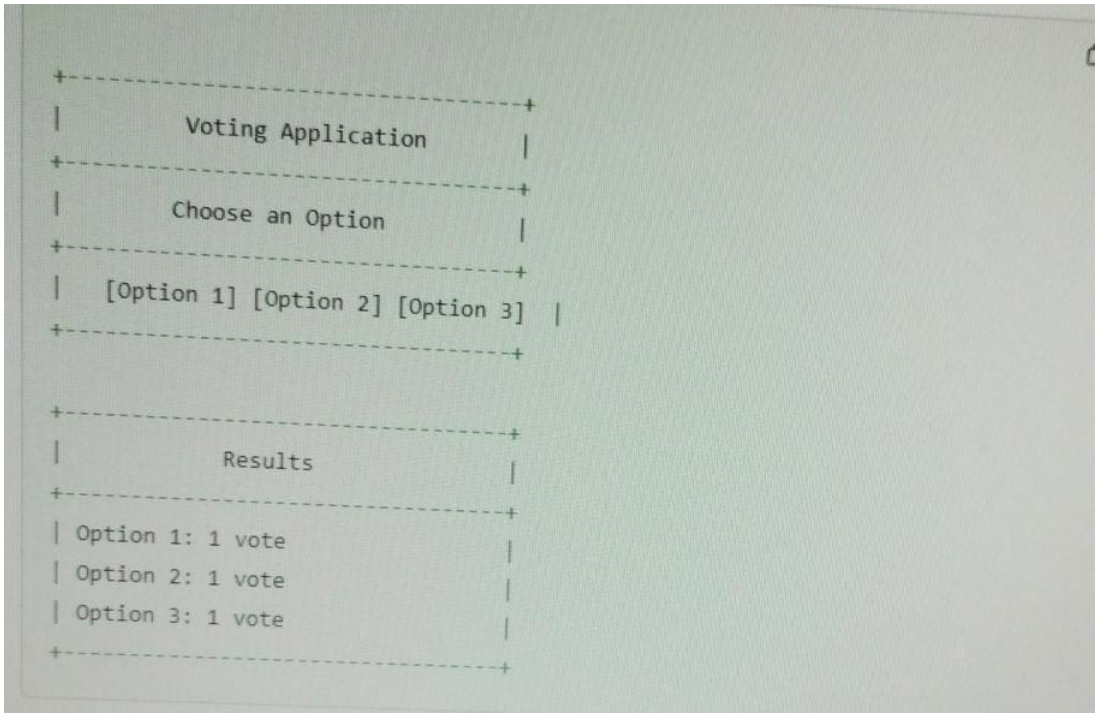
```
        <div className="voteCount">                    </div>
              {lang.votes}
        <button onClick={this.vote.bind(this, i)}>Click Here</button>
        </div>
                                              )
                                        }
                                  </div>
                            </>
                  );
        }
  }

  export default App;
```

Src/App.css

```
  *{
    margin: 0;
    padding: 0;
  }

  body {
    text-align: center;
    color: #222;
    font-size: 24px;
    font-family: sans-serif;
  }

  h1 {
    margin: 30px;
  }

  .languages {
    height: 400px;
    width: 400px;
    margin: 10px auto;

    display: flex;
    flex-direction: column;
  }

  .language {
    flex: 1;
    display: flex;
```

```
  justify-content: space-between;
  align-items: center;
  padding: 10px 40px;
  background-color: blanchedalmond;
  border: 1px solid #222;
  margin: 2px;
 }

 .voteCount {
  border-radius: 50%;
  display: flex;
  justify-content: center;
  align-items: center;
 }

 .language button {
  color: green;
  background-color: #0000;
  border: none;
  font-size: 30px;
  outline: none;
  cursor: pointer;
 }
```

OUTPUT:



Week-10. Write a server side program for Accessing MongoDB from Node.js.

Step 1: download mongodb from the below link

https://www.mongodb.com/try/download/community

Step 1: Create a database called "mydb":

```
var MongoClient = require('mongodb').MongoClient;
var url = "mongodb://localhost:27017/mydb";

MongoClient.connect(url, function(err, db) {
  if (err) throw err;
  console.log("Database created!");
  db.close();
});
```

Save the code above in a file called "demo_create_mongo_db.js" and run the file:

Run "demo_create_mongo_db.js"

C:\Users*Your Name*>node demo_create_mongo_db.js

OUTPUT:

Step 2:Creating a Collection

To create a collection in MongoDB, use the createCollection() method:

Create a collection called "customers":

```
var MongoClient = require('mongodb').MongoClient;
var url = "mongodb://localhost:27017/";

  MongoClient.connect(url, function(err, db) {
    if (err) throw err;
var dbo = db.db("mydb");
dbo.createCollection("customers", function(err, res) { if
(err) throw err;
      console.log("Collection created!");
      db.close();
    });
  });
```

Note: Save the code above in a file called "demo_mongodb_createcollection.js" and run the file.

Step 3: Insert into Collection

```
var MongoClient = require('mongodb').MongoClient;
var url = "mongodb://localhost:27017/";

MongoClient.connect(url, function(err, db) {
  if (err) throw err;
  var dbo = db.db("mydb");
  var myobj = [
    { name: 'John', address: 'Highway 71'},
    { name: 'Peter', address: 'Lowstreet 4'},
    { name: 'Amy', address: 'Apple st 652'},
    { name: 'Hannah', address: 'Mountain 21'},
    { name: 'Michael', address: 'Valley 345'},
    { name: 'Sandy', address: 'Ocean blvd 2'},
    { name: 'Betty', address: 'Green Grass 1'},
    { name: 'Richard', address: 'Sky st 331'},
    { name: 'Susan', address: 'One way 98'},
    { name: 'Vicky', address: 'Yellow Garden 2'},
    { name: 'Ben', address: 'Park Lane 38'},
    { name: 'William', address: 'Central st 954'},
    { name: 'Chuck', address: 'Main Road 989'},
    { name: 'Viola', address: 'Sideway 1633'}
  ];
  dbo.collection("customers").insertMany(myobj, function(err, res) {
    if (err) throw err;
    console.log("Number of documents inserted: " + res.insertedCount);
    db.close();
  });
});
```

Note: Save the code above in a file called "demo_mongodb_insert_multiple.js" and run the file

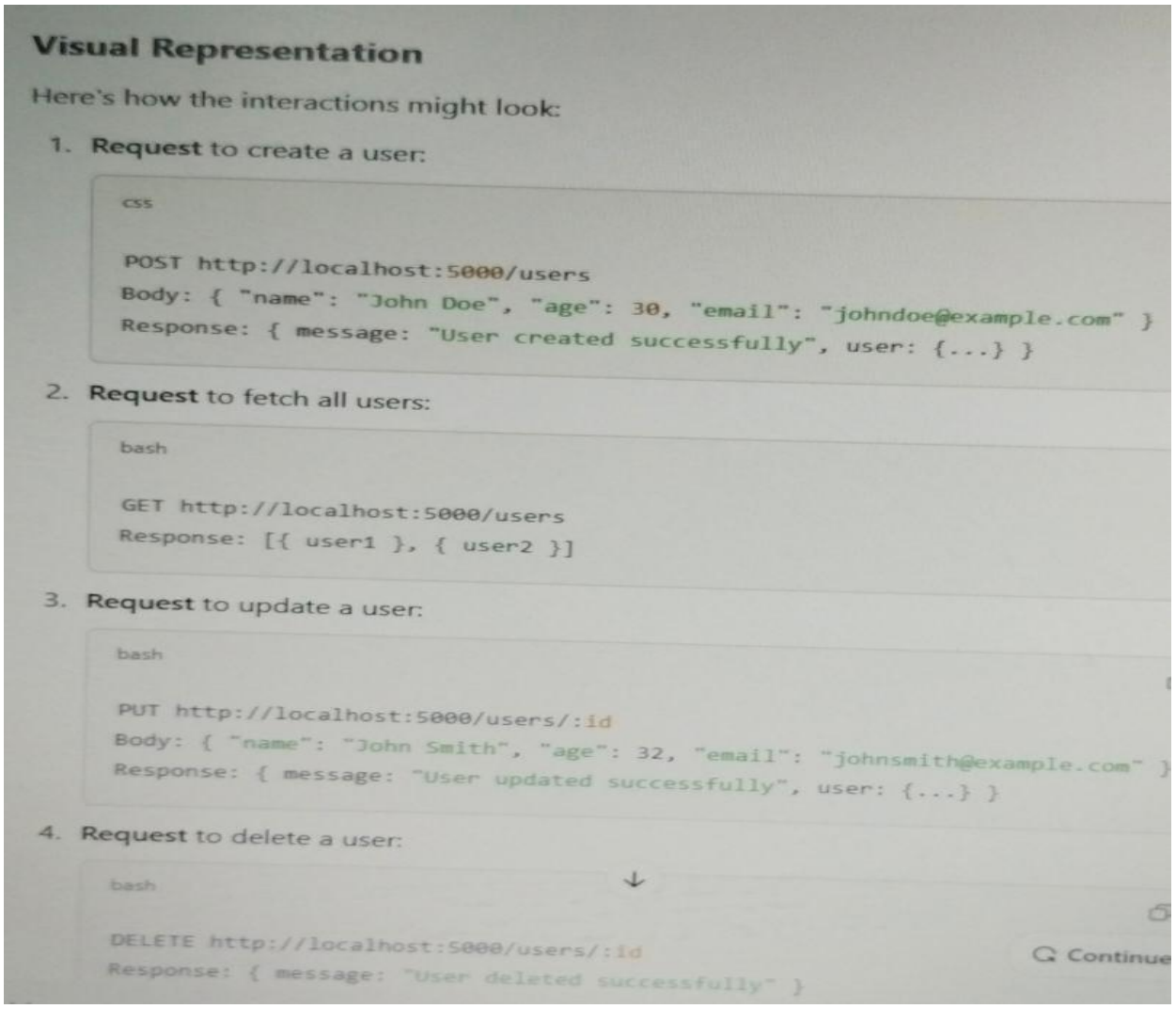
Step 4: Accessing elements

```
var MongoClient = require('mongodb').MongoClient;
var url = "mongodb://localhost:27017/";

MongoClient.connect(url, function(err, db) {
  if (err) throw err;
  var dbo = db.db("mydb");
  dbo.collection("customers").find({}).toArray(function(err, result) {
```

```
  if (err) throw err;
    console.log(result);
    db.close();
    });
```

Note: Save the code above in a file called "demo_mongodb_find.js" and run the file

OUTPUT



Week-11. Write a server side program for Manipulating MongoDB from Node.js

Step 1: Create a database called "mydb":

```
var MongoClient = require('mongodb').MongoClient;
var url = "mongodb://localhost:27017/mydb";

MongoClient.connect(url, function(err, db) {
  if (err) throw err;
  console.log("Database created!");
```

```
  db.close();
});
```

Save the code above in a file called "demo_create_mongo_db.js" and run the file:

Run "demo_create_mongo_db.js"

C:\Users*Your Name*>node demo_create_mongo_db.js

Step 2:Creating a Collection

Create a collection called "customers":

To create a collection in MongoDB, use the createCollection() method:

```
var MongoClient = require('mongodb').MongoClient;
var url = "mongodb://localhost:27017/";
MongoClient.connect(url, function(err, db) {
if (err) throw err;
  var dbo = db.db("mydb");
  dbo.createCollection("customers", function(err, res) {
  if (err) throw err;
    console.log("Collection created!");
    db.close();
  });
});
```

Note: Save the code above in a file called "demo_mongodb_createcollection.js" and run the file.

Step 3: Insert into Collection

```
var MongoClient = require('mongodb').MongoClient;
var url = "mongodb://localhost:27017/";

MongoClient.connect(url, function(err, db) {
  if (err) throw err;
  var dbo = db.db("mydb");
  var myobj = [
    { name: 'John', address: 'Highway 71'},
    { name: 'Peter', address: 'Lowstreet 4'},
    { name: 'Amy', address: 'Apple st 652'},
    { name: 'Hannah', address: 'Mountain 21'},
    { name: 'Michael', address: 'Valley 345'},
    { name: 'Sandy', address: 'Ocean blvd 2'},
```

```
    { name: 'Betty', address: 'Green Grass 1'},
    { name: 'Richard', address: 'Sky st 331'},
    { name: 'Susan', address: 'One way 98'},
    { name: 'Vicky', address: 'Yellow Garden 2'},
    { name: 'Ben', address: 'Park Lane 38'},
    { name: 'William', address: 'Central st 954'},
    { name: 'Chuck', address: 'Main Road 989'},
    { name: 'Viola', address: 'Sideway 1633'}
  ];
  dbo.collection("customers").insertMany(myobj, function(err, res) {

    if (err) throw err;
    console.log("Number of documents inserted: " + res.insertedCount);
    db.close();
  });
});
```

Note: Save the code above in a file called "demo_mongodb_insert_multiple.js" and run the file

Step 4: Accessing elements

```
var MongoClient = require('mongodb').MongoClient;
var url = "mongodb://localhost:27017/";

MongoClient.connect(url, function(err, db) {
  if (err) throw err;
  var dbo = db.db("mydb");
  dbo.collection("customers").find({}).toArray(function(err, result) {
  if (err) throw err;
    console.log(result);
    db.close();
     });
});
```

Note: Save the code above in a file called "demo_mongodb_find.js" and run the file

Step 5: Delete the document with the address "Mountain 21": var MongoClient = require('mongodb').MongoClient;

```
var url = "mongodb://localhost:27017/";
MongoClient.connect(url, function(err, db) {
if (err) throw err;
```

```
  var dbo = db.db("mydb");
  var myquery = { address: 'Mountain 21' };
  dbo.collection("customers").deleteOne(myquery, function(err, obj) {
  if (err) throw err;
    console.log("1 document deleted");
    db.close();
  });
});
```

Step 6: Update the document with the address "Valley 345" to name="Mickey" and address="Canyon 123":

```
var MongoClient = require('mongodb').MongoClient;
var url = "mongodb://127.0.0.1:27017/";

  MongoClient.connect(url, function(err, db) {
    if (err) throw err;
    var dbo = db.db("mydb");
    var myquery = { address: "Valley 345" };
    var newvalues = { $set: {name: "Mickey", address: "Canyon 123" } };
    dbo.collection("customers").updateOne(myquery, newvalues, function(err, res) {
    if (err) throw err;
      console.log("1 document updated");
      db.close();
    });
  });
```

OUTPUT: