DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

FULL STACK DEVELOPMENT

[R22A0589]

LABORATORY MANUAL

B. TECH CSE

(III YEAR–I SEM)

R22 REGULATION

(2024-25)

Name :	
Roll no:	
Section:	
Year :	

MALLAREDDY COLLEGE OF ENGINEERING & TECHNOLOGY

(Autonomous Institution– UGC, Govt. of India)

Recognized under2(f)and12(B) of UGC ACT 1956

(Affiliated to JNTUH, Hyderabad, Approved by AICTE - Accredited by NBA &NAAC–'A' Grade-ISO9001:2015 Certified)

Maisammaguda, Dhulapally (Post Via. Hakimpet), Secunderabad –500100, Telangana, India

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Vision

To acknowledge quality education and instill high patterns of discipline making the students technologically superior and ethically strong which involves the improvement in the quality of life in human race.

Mission

- ❖ To achieve and impart holistic technical education using the best of infrastructure, outstanding technical and teaching expertise to establish the students in to competent and confident engineers.
- ❖ Evolving the center of excellence through creative and innovative teaching learning practices for promoting academic achievement to produce internationally accepted competitive and world class professionals.

PROGRAMME EDUCATIONAL OBJECTIVES (PEOs)

PEO1–ANALYTICALSKILLS

- ❖ To facilitate the graduates with the ability to visualize, gather information, articulate, analyze, solve complex problems, and make decisions. These are essential to address the challenges of complex and computation intensive problems increasing their productivity.

PEO2–TECHNICALSKILLS

- ❖ Tofacilitatethegraduateswiththetechnicalskillsthatpreparethemforimmediateemployment and pursue certification providing a deeper understanding of the technology in advanced areas of computer science and related fields, thus encouraging to pursue higher education and research based on their interest.

PEO3– SOFTSKILLS

- ❖ To facilitate the graduates with the soft skills that include fulfilling the mission, setting goals, showing self confidence by communicating effectively, having a positive attitude, get involved in team-work, being a leader, managing their career and their life.

PEO4–PROFESSIONALETHICS

- ❖ To facilitate the graduates with the knowledge of professional and ethical responsibilities by paying attention to grooming, being conservative with style, following dress codes, safety codes, and adapting them to technological advancements.

PROGRAM SPECIFIC OUTCOMES (PSOs)

After the completion of the course, B.Tech Computer Science and Engineering, the graduates will have the following Program Specific Outcomes:

1. Fundamentals and critical knowledge of the Computer System: - Able to Understand the working principles of the computer System and its components, Apply the knowledge to build, asses, and analyze the software and hardware aspects of it.

2. The comprehensive and Applicative knowledge of Software Development: Comprehensive skills of Programming Languages, Software process models, methodologies, and able to plan, develop, test, analyze, and manage the software and hardware intensive systems in heterogeneous platforms individually or working in teams.

3. Applications of Computing Domain & Research: Able to use the professional, managerial, interdisciplinary skill set, and domain specific tools in development processes, identify their search gaps, and provide innovative solutions to them.

PROGRAM OUTCOMES (POs)

Engineering Graduates should possess the following:

1. Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2. Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

3. Design / development of solutions: Design solutions for complex engineering problems anddesignsystemcomponentsorprocessesthatmeetthespecifiedneedswithappropriateconsiderati on for the public health and safety, and the cultural, societal, and environmental considerations.

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.

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.

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.

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.

8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10. 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.

11. 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.

12. Life- long learning: Recognize the need for, and have the preparation and ability to engagein independent and life-long learning in the broadest context of technological change.

FULL STACK DEVELOPMENT LABORATORY

1. Course Objectives:

1. This course will enable the students:
2. Usage of various front and backend Tools
3. They can understand and create applications on their own
4. Demonstrate and Designing of Websites can be carried out.
5. Develop web based application using suitable client side and server side code.
6. Implement web based application using effective database access.

2. Course outcomes:

Students will be able to understand

1. Usage of various front and back end Tools
2. They can understand and create applications on their own
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.

3. Introduction about lab

Minimum System requirements:

Processors: Intel Atom® processor or Intel® Core™ i3 processor.
Disk space: 1 GB.
Operating systems: Windows* 7 or later, mac OS, and Linux.
Python* versions: 2.7.X, 3.6.X., 3.8.X

4. Guidelines to students

A. Standard operating procedure

a) Explanation on today's experiment by the concerned faculty using PPT covering the following aspects:

1) Name of the experiment
2) Aim

b) Writing the python programs by the students

c) Commands for executing programs

Writing of the experiment in the Observation Book

The students will write the today's experiment in the Observation book as per the following format:

a) Name of the experiment
b) Aim
c) Writing the program
d) Viva-Voce Questions and Answers
e) Errors observed (if any) during compilation/execution
f) Signature of the Faculty

B. Guide Lines to Students in Lab

Disciplinary to be maintained by the students in the Lab

- Students are required to carry their lab observation book and record book with completed experiments while entering the lab:
- Students must use the equipment with care. Any damage is caused student is punishable·
- Students are not allowed to use their cell phones/pen drives/ CDs in labs. ·
- Students need to be maintain proper dress code along with ID Card·
- Students are supposed to occupy the computers allotted to them and are not supposed to talk or make noise in the lab. Students, after completion of each experiment they need to be updated in observation notes and same to be updated in the record. ·
- Lab records need to be submitted after completion of experiment and get it corrected with the concerned lab faculty:
- If a student is absent for any lab, they need to be completed the same experiment in the free time before attending next lab.

Instructions to maintain the record

· Before start of the first lab they have to buy the record and bring the record to the lab.·

· Regularly (Weekly) update the record after completion of the experiment and get it corrected with concerned lab in-charge for continuous evaluation.

· In case the record is lost inform the same day to the faculty in charge and get the new record within 2 days the record has to be submitted and get it corrected by the faculty.·

· If record is not submitted in time or record is not written properly, the evaluation marks (5M) will be deducted.

C. General laboratory instructions

1. Students are advised to come to the laboratory at least 5 minutes before (to the starting time), those who come after 5 minutes will not be allowed into the lab.
2. Plan your task properly much before to the commencement, come prepared to the lab with the synopsis / program / experiment details.
3. Student should enter into the laboratory with:
a. Laboratory observation notes with all the details (Problem statement, Aim, Algorithm, Procedure, Program, Expected Output, etc.,) filled in for the lab session.
b. Laboratory Record updated up to the last session experiments and other utensils (if any) needed in the lab. c. Proper Dress code and Identity card.
4. Sign in the laboratory login register, write the TIME-IN, and occupy the computer system allotted to you by the faculty.
5. Execute your task in the laboratory, and record the results / output in the lab observation note book, and get certified by the concerned faculty.
6. All the students should be polite and cooperative with the laboratory staff, must maintain the discipline and decency in the laboratory.
7. Computer labs are established with sophisticated and high end branded systems, which should be utilized properly.
8. Students / Faculty must keep their mobile phones in SWITCHED OFF mode during the lab sessions. Misuse of the equipment, misbehaviors with the staff and systems etc., will attract severe punishment.
9. Students must take the permission of the faculty in case of any urgency to go out; if anybody found loitering outside the lab / class without permission during working hours will be treated seriously and punished appropriately.
10. Students should LOG OFF/ SHUT DOWN the computer system before he/she leaves the lab after completing the task (experiment) in all aspects. He/she must ensure the system / seat is kept properly.

INDEX

S.No	Date	Name of the Activity/Experiment	Grade/ Marks	Faculty Signature

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:

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}
```

OUTPUT:

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>
```

OUTPUT:

C) 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>
```

OUTPUT:

Mouseout()

```
<html>
<head>
<title>mouse events</title>
<script language="javascript">
function fun2()
{
alert("bye");
}
</script>
</head>
<body>
<button  onmouseout="fun2()" >onClickme</button>
</body>
</html>
```

OUTPUT:

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>
```

OUTPUT:

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>
```

OUTPUT:

Keydown()

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

OUTPUT:

Keyup()

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

OUTPUT:

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>
```

OUTPUT:

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>
```

OUTPUT:

Load() and unload()

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

OUTPUT:

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 JQuery.

```
<html>
<body>
    <head>
        <style>
            #frmCheckPassword {
                border-top: #F0F0F0 2px solid;
                background: #808080;
                padding: 10px;
            }

            .demoInputBox {
                padding: 7px;
                border: #F0F0F0 1px solid;
                border-radius: 4px;
            }

            #password-strength-status {
                padding: 5px 10px;
                color: #FFFFFF;
                border-radius: 4px;
```

```
                    margin-top: 5px;
                }

                .medium-password {
                    background-color: #b7d60a;
                    border: #BBB418 1px solid;
                }

                .weak-password {
                    background-color: #ce1d14;
                    border: #AA4502 1px solid;
                }

                .strong-password {
                    background-color: #12CC1A;
                    border: #0FA015 1px solid;
                }
            </style>
        </head>
        <div name="frmCheckPassword" id="frmCheckPassword">
            <label>Password:</label>
            <input type="password" name="password" id="password"
class="demoInputBox" onKeyUp="checkPasswordStrength();" />
```

```
        <div id="password-strength-status"></div>
      </div>
      <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
      <script>
        function checkPasswordStrength() {
          var number = /([0-9])/;
          var alphabets = /([a-zA-Z])/;
          var special_characters = /([~,!,@,#,$,%,^,&,*,-,_,+,=,?,>,<])/;
          if ($('#password').val().length < 6) {
            $('#password-strength-status').removeClass();
            $('#password-strength-status').addClass('weak-password');
            $('#password-strength-status').html("Weak (should be atleast 6
characters.)");
          } else {
            if ($('#password').val().match(number) &&
$('#password').val().match(alphabets) &&
$('#password').val().match(special_characters)) {
              $('#password-strength-status').removeClass();
              $('#password-strength-status').addClass('strong-password');
              $('#password-strength-status').html("Strong");
            } else {
              $('#password-strength-status').removeClass();
              $('#password-strength-status').addClass('medium-password');
```

```
                    $('#password-strength-status').html("Medium (should include
alphabets, numbers and special characters.)");
                  }
                }
              }
          </script>
      </body>
      </html>
```

OUTPUT:

Week-5. Write a program to create and Build a star rating system using JQuery.

```
<!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 = "https://cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/4.4.1/css/bootstrap.min.css">
    <script src = "https://cdnjs.cloudflare.com/ajax/libs/jquery/3.4.1/jquery.js"> </script>
    <script src = "https://cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/4.4.1/js/bootstrap.min.js"> </script>
    <link rel = "stylesheet" href = "https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
    <script src = "https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
    <title> jQuery simple star rating example </title>
    <style>
    body {
        background-color: aquamarine;
        margin : 0px;
    }
    .fa-star {
        font-size : 50px;
        align-content: center;
    }
    .container {
        height: 100px;
        width: 600px;
        margin: auto;
    }
    </style>
</head>
<body>
    <div class = "container">
        <h2 style="margin-top: 50px;">jQuery simple star rating example</h2>
       <div class = "con">
        <h3 style = "margin-top : 80px; color: green;">Rate our product :-</h3>
        <i class = "fa fa-star" aria-hidden = "true" id = "st1"></i>
       <i class = "fa fa-star" aria-hidden = "true" id = "st2"></i>
       <i class = "fa fa-star" aria-hidden = "true" id = "st3"></i>
       <i class = "fa fa-star" aria-hidden = "true" id = "st4"></i>
       <i class = "fa fa-star" aria-hidden = "true" id = "st5"></i>
       </div>
    </div>
    <script>
        $(document).ready(function() {
```

```
$("#st1").click(function() {
    $(".fa-star").css("color", "black");
    $("#st1").css("color", "yellow");

});
$("#st2").click(function() {
    $(".fa-star").css("color", "black");
    $("#st1, #st2").css("color", "yellow");

});
$("#st3").click(function() {
    $(".fa-star").css("color", "black")
    $("#st1, #st2, #st3").css("color", "yellow");

});
$("#st4").click(function() {
    $(".fa-star").css("color", "black");
    $("#st1, #st2, #st3, #st4").css("color", "yellow");

});
$("#st5").click(function() {
    $(".fa-star").css("color", "black");
    $("#st1, #st2, #st3, #st4, #st5").css("color", "yellow");
```

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

OUTPUT:

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

index.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>AJAX Example</title>
    <script>
        // Function to send AJAX request
        function sendRequest() {
            // Create a new XMLHttpRequest object
            var xhr = new XMLHttpRequest();

            // Configure it: GET-request for the URL /server-response
            xhr.open('GET', '/server-response', true);

            // What to do when the request is completed
            xhr.onload = function() {
                if (xhr.status == 200) {
                    // Successfully received response
                    document.getElementById('response').innerText =
xhr.responseText;
                  } else {
                    // Error occurred
                    document.getElementById('response').innerText = 'Error: ' +
xhr.status;
                  }
                };

                // Send the request
                xhr.send();
              }
          </script>
        </head>
        <body>
          <h1>AJAX Request Example</h1>
          <button onclick="sendRequest()">Send Request</button>
          <p id="response"></p>
        </body>
        </html>
```

Server.js

```
const express = require('express');
const app = express();
const port = 3000;


// Serve static files (HTML) from the current directory
app.use(express.static(__dirname));


// Endpoint to handle AJAX request
app.get('/server-response', (req, res) => {
    res.send('Hello from the server!');
});


app.listen(port, () => {
    console.log(`Server is running at http://localhost:${port}`);
});
```

- Save index.html and server.js in the same directory.
- Run the following commands to start the server:

```
npm install express
node server.js
```

- Open your browser and navigate to http://localhost:3000/index.html.

OUTPUT

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>
```

OUTPUT:

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>
```

OUTPUT:

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="language">
                                                    <div className="voteCount">
                                                         {lang.votes}
                                                    </div>
    <div className="languageName">
         {lang.name}
         </div>
    <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.

OUTPUT:

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

OUTPUT:

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
```

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.

OUTPUT:

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

OUTPUT:

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

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();
  });
});
```

OUTPUT:

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: