Ex. No:1	STATIC WEB PAGE USING HTML 5.0

AIM

To design a static webpage to display bio data using html 5.0.

ALGORITHM

STEP 1: Open any html editor.

STEP 2: Insert html,head and title tags .Specify the title for the webpage inside the title tag for the webpage inside the title tag.

STEP 3: Open body tag and display all your details using header tag.

STEP 4: Open an article tag and enter the appropriately.

STEP 5: Enter close tags of all tags opened.

STEP 6: Save the file with html extension.

STEP 7: Open the file using any browser.

SOURCE CODE

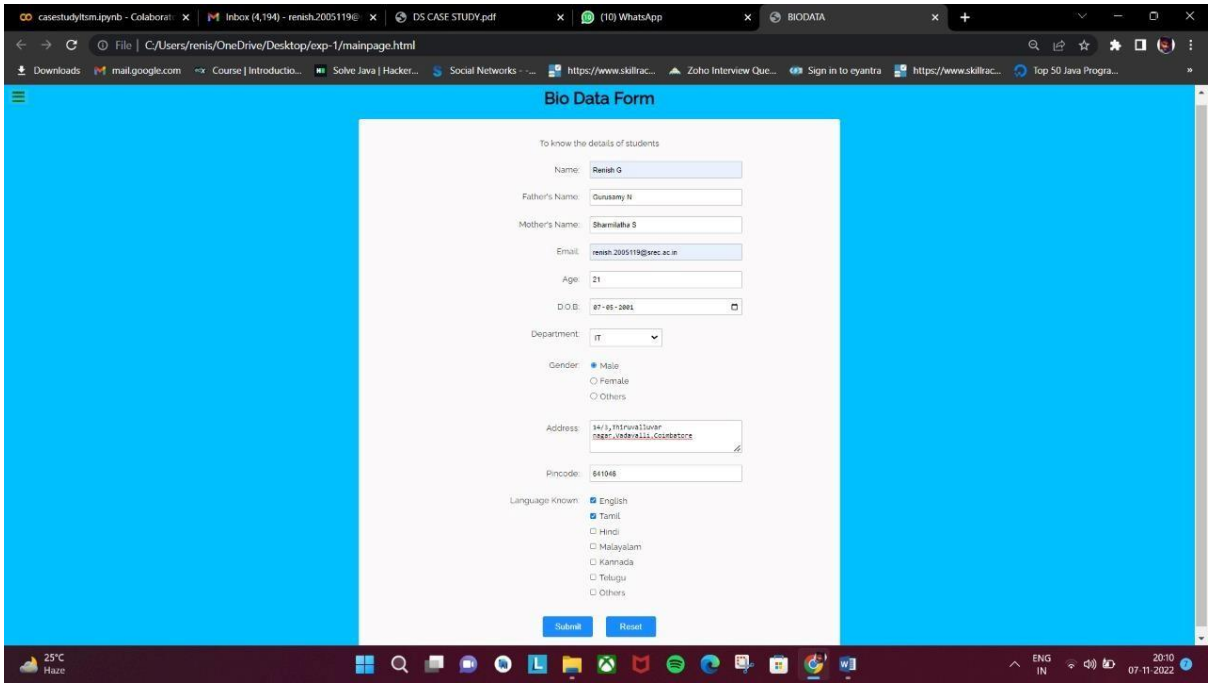
```
<!DOCTYPE html>
<html>
  <head>
<meta http-equiv="Content-Type" content="text/html; charset=windows-1252">
      <title>BIODATA</title>
   </head>
   <body>
      <header>
         <h1>MY BIODATA</h1>
         <h4>NIKITHA U P</h4>
         <address>
            75,
         krishnasamy street,
         municipal colony,
         Erode
         </address>
         <h4>nikitha.2005101@srec.ac.in</h4>
         <h4>6383450061</h4>
         <img src="E:\INTERNET PROGRAMMING\biodata in ip\flower.jpg" alt="profile
photo">
      </header>
      <article>
         <section>
            <h4>OBJECTIVE:</h4>
            <p>
```

To secure a challenging position in a reputed organization to expand my learnings, skills and knowledge.

```
</p>
<table border="2">
  <tbody><tr>
  <th>Standard</th>
  <th>Medium</th>
  <th>Board</th>
  <th>Institution</th>
  <th>Percentage/CGPA</th>
  </tr>
  <tr>
  <td>SSLC</td>
  <td>ENGLISH</td>
  <td>State Board</td>
  <td>Vivekananda Memorial Matriculation School ,Palada</td>
  <td>94.08%</td>
  </tr>
  <tr>
  <td>HSC</td>
  <td>ENGLISH</td>
  <td>State Board</td>
  <td>Stanes Anglo Indian Higher Secondary School, Coonoor</td>
  <td>96%</td>
  </tr>
  <tr>
  <td>B.Tech IT SEM-3</td>
  <td>ENGLISH</td>
  <td>AICTE</td>
  <td>Sri Ramakrishna Engineering College, Coimbatore</td>
  <td>8.2</td>
  </tr>
  </tbody></table>
  <ul style="list-style: square">
  <h2>Programming Skills</h2>
  <li>C Programming</li>
  <li>Python Programming</li>
  <li>C++</li>
  <li>Java</li>
  </ul>
  <ul>
    <h4>PROJECTS</h4>
    <li>Deploying application using Docker and Kubernetes</li>
    <li>Face Mask Alert System</li>
  </ul>
  <ul>
    <img> LinkedIn:<a href="#">nikitha-u-p</a><br><br>
    <img> Instagram: <a href="#">nikitha.26_</a><br><br>

  </ul>
  <ul>
    <h4>PARTICIPATION AND CERTIFICATIONS</h4>
```

```
                <li>Smart India Hackathon (Internals) </li>
                <li>SREC Hackfest</li>
        </ul>
</section>
      </article>

  </body>
</html>
```

OUTPUT



RESULT

Thus the static webpage to display Bio-data has been implemented and output wasverified successfully.

Ex No : 2	AUTO MECHANIC SHOP WEBSITE USING HTML 5.0

AIM:

To develop an Auto Mechanic Shop Website using HTML.

ALGORITHM:

STEP 1: Open any html editor.

STEP 2: Insert html, head and title tags. Specify the title for the webpage inside the title tag for the webpage inside the title tag.

STEP 3: Open the body tag to display all your details using the navigation tag.

STEP 4: Open a navigation tag and enter them appropriately**.**

STEP 5: Create a home page that renders information about the company.

STEP 6: Create a service page that displays the list of services provided by the company.

STEP 7: Create a booking page that displays a booking form that takes various inputs from the user.

STEP 8: Creates a link that opens in the user's email application.

STEP 9: Enter close tags of all tags opened.

STEP 10: Save the file with html extension.

STEP 11: Open file using any browser.

SOURCE CODE:

Home.html:

```
<!DOCTYPE html>
<html>
  <head>
    <title>HOME|AUTO MECHANIC SHOP</title>
  </head>
  <body>
    <h3>Automob-mechanic </h3>
    <a href="home.html">Home</a>|
    <a href="Services.html">Services</a>|
    <a href="Booking.html">Booking</a>|
    <a href="Contact.html">Contact</a><br>
    <br>
```

```
        <img src="E:\INTERNET PROGRAMMING\exp-2\home1.jpg" alt="picture not
    accessible">
        <article>
          <section>
            <p><b>"One stop solution to get your car repaired and serviced".</b><br>
            </p>
            <p>we aim to make the car repair and service experience</p>
          </section>
        </article>
        <br><br><br><br><br><br><br><br><br><br>
        <footer>
          Copyright @ 2020 Automob-mechanic.All Rights Reserved
        </footer>
      </body>
    </html>
```

Service.html:

```
        <!DOCTYPE html>
        <html>
          <head>
            <title>SERVICE|AUTO MECHANIC SHOP</title>
          </head>
          <body>
            <h3>Automob-mechanic </h3>
            <a href="home.html">home</a>|
            <a href="Services.html">Services</a>|
            <a href="Booking.html">Booking</a>|
            <a href="Contact.html">Contact</a><br>
            <br>
            <p>our services</p>
            <h1>preventive maintanence service</h1>
            <img src="service1.jpg" alt="picture not accessible">
            <p>periodically check and car running<br>
            offer:20% off</p>
            <br><br><br><br><br><br><br><br>
```

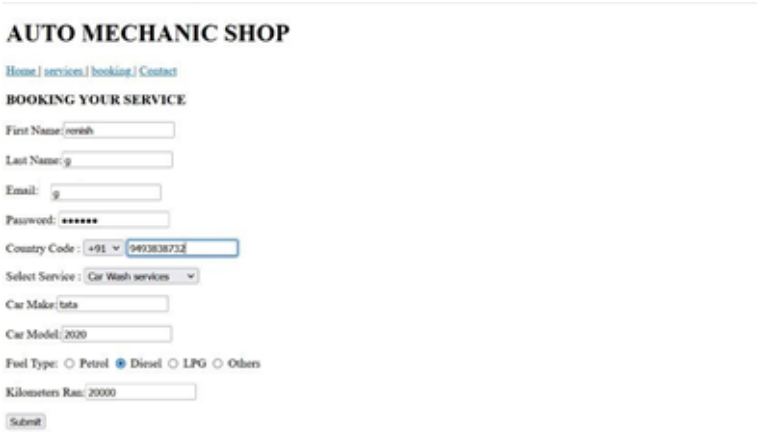
```
                    <footer>
                        Copyright @ 2020 Automob-mechanic.All Rights Reserved
                    </footer>
                </body>
            </html>
```

Booking.html:

```
<!DOCTYPE html>
<html>
    <head>
        <title>AUTO MECHANIC SHOP</title>
    </head>
    <body>
        <h1> Auto-BOB mechanic shop</h1>
        <a href ="home.html">home</a> |
        <a href ="Services.html">services </a>|
        <a href ="Booking.html">booking </a>|
        <a href ="contact">contact</a>|
        <p> BOOKING SERVICES</p><br>
        <form>
            First Name:<input type ="text" name ="firstname"/> <br/><br> Last
            Name:<input type ="text" name ="lastname"/> <br/><br> Car
            model:<input type ="text" name ="car model"/> <br/><br> Car
            Number:<input type ="text" name ="car number"/> <br/><br>
            Fuel type:<input type="radio"id="fuel type"name="fuel
type"value="petrol"/>petrol
            <input type="radio"id="fuel type"name="fuel
type"value="diesel"/>diesel
            <input type="radio"id="fuel type"name="fuel type"value="LPG"/>LPG
            <input type="radio"id="fuel type"name="fuel
type"value="others"/>others<br><br>
            <label for ="email">Email:</label>
            <input type="email" name="email"/> <br><br>
            <label for ="Contact num">Contact num:</label>
            <input type="tel"id="phone"name="phone"pattern="[0-9]{3}-[0-9]{2}-[0-
9]{4}" value="+91"> <br><br>
            <B>Select services:</B>
            <select>
                <option value="Oil/oil filter changed">oil/oil filter
```

```
changed</option>
                    <option value="Wipes blade replacement">wipes blade
replacement</option>filter</option><option value="Replace air filter">Replace air
<option value="New tires">New tires</option>
<option value="Battery replacement">Batteryreplacement</option>
                    <option value="Wheels aligned/balanced">Wheels
aligned/balanced</option>                    "
                </select><br><br><br>
                <button type ="button">Fix appointment!</button>

            </form>
        </body><br/><br><br/><br><br/><br><br/><br><br/><br><br/><br><br/>
        <footer>
            <p> Copyright @ 2022 AUTO MECHANIC SHOP. All rights reserved</p>
        </footer>
    </html>
```

OUTPUT:



RESULT:

Thus, the Auto Mechanic Shop Website using HTML is implemented and output was verified successfully.

Ex No : 3	WAYFAR TRAVEL WEBSITE USING HTML & CSS

AIM:

To develop a WayFar Travel website and verify the output using HTML & CSS.

ALGORITHM:

STEP 1: Create a new folder and create HTML files for Home page and Places page.

STEP 2: Create a CSS file for styling the document.

STEP 3: In the Home.html, create the link for the CSS file.

STEP 4: Add header, footer, navigation bar, image and description usingnecessary tags.

STEP 5: In the Places.html, create the link for the CSS file.

STEP 6: Add header, footer, navigation bar, and links for places descriptionusing necessary tags.

STEP 7: Go to the CSS style sheet and apply necessary styles for the html elements.

STEP 8: Close the tags appropriately in HTML files.

STEP 9: Save the files and open using any browser to display the output.

SOURCE CODE:

Home.html:

```
<html>
<head>
<link rel="stylesheet" href="a.css">
<title> WayFar Travel Website</title>
</head>
<body>
   <header>
      <h1>WAYFAR TRAVEL</h1>
   </header>
<nav>
   <ul>
      <li><a class="active" href="Home.html">Home</a></li>
      <li><a href="place.html">Famous places</a></li>
      <li><a href="Booking.html">Ticket Booking</a></li>
    </ul>
</nav>
   <div class="lineblock">
```

```
        <div class="i">
           <img src="travel.jpg" >
        </div>
        <div class="p">
           A comfort zone is a beautiful place… but nothing ever grows there.
        </div>
    </div>
</body>
<footer style="color: gray;">
    <h5>Copyrights@2022.All Rights Reserved.</h5>
</footer>
</html>
```

Places.html

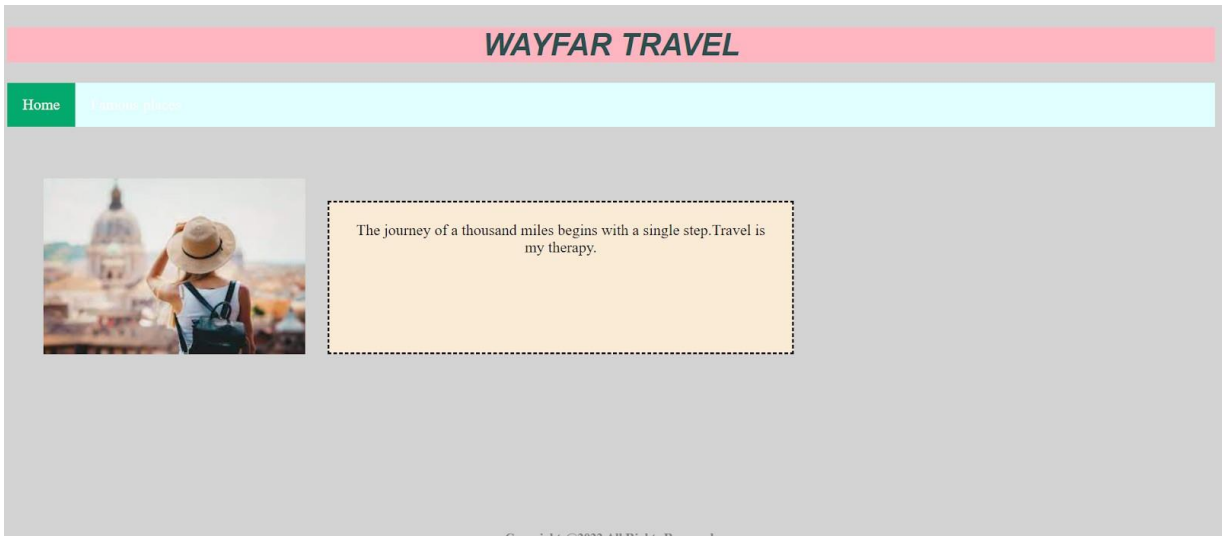
```
<html>
<head>
<link rel="stylesheet" href="a.css">
<title> WayFar Travel Website</title>
</head>
<body>
    <header>
        <h1>WAYFAR TRAVEL</h1>
    </header>
<nav>
        <ul>
            <li><a href="a.html">Home</a></li>
            <li><a class="active" href="place.html">Famous places</a></li>
            <li><a href="Booking.html">Ticket Booking</a></li>
          </ul>
        </nav>
    <div>
        <div class="design"><a
href="https://en.wikipedia.org/wiki/Coimbatore">COIMBATORE</a></div>
        <div class="design"><a
href="https://en.wikipedia.org/wiki/Chennai">CHENNAI</a></div>
        <div class="design"><a href="https://en.wikipedia.org/wiki/Erode">ERODE</a></div>
        <div class="design"><a
href="https://en.wikipedia.org/wiki/Salem,_Tamil_Nadu">SALEM</a></div>
        <div class="design"><a
href="https://en.wikipedia.org/wiki/Kumbakonam">KUMBAKONAM</a></div>
        <div class="design"><a
href="https://en.wikipedia.org/wiki/Madurai">MADURAI</a></div>
    </div>
</body>
<footer style="color: gray;" class="scroll">
    <h5>Copyrights@2022.All Rights Reserved.</h5>
</footer>
</html>
```

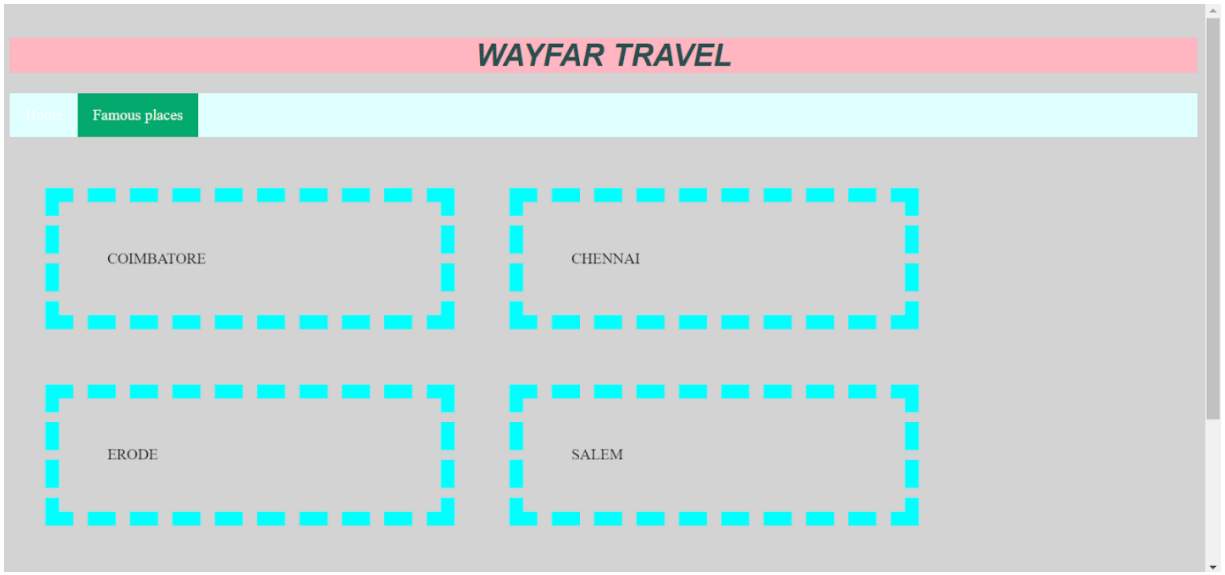
style.css

```
ul {
    list-style-type: none;
    padding: 0;
    overflow: hidden;
    top:0;
    background-color:pink;
     width: 100%;
  }
  body{
    background-color:lightskyblue;
  }
  header{
    font-style: italic;
    font-family: Arial, Helvetica, sans-serif;
    text-align: center;
    margin-top: 3%;
    width: 100%;
    background-color:aquamarine;
    color: darkslategray;
  }
  li {
    float: left;
  }
  li a {
    display: block;
    color: white;
    text-align: center;
    padding: 14px 16px;
    text-decoration: none;
  }
  li a.active {
    background-color: #04AA6D;
    color: white;
  }
  /* Change the link color to #111 (black) on hover */
  li a:hover:not(.active) {
    background-color: #555;
    color: white;
  }
  footer{
    clear:both;
    position:fixed;
    bottom:0;
    left:0;
    width:100%;
    text-align: center;
  }
  .scroll{
    clear:both;
    position:relative;
```

```
  bottom:0;
  left:0;
  width:100%;
  text-align: center;
 }
 .lineblock {
  display: flex;
  float:left;
  margin-top: 3%;
  margin-left: 3%;
 }
 .i{
 border: #333;
 }
 .p{
  margin-top: 3%;
  border-style: dashed;
  text-align: center;
  padding: 2.5%;
  background-color: antiquewhite;
  text-align: center;
  margin-left: 3%;
 }
 .design{
  background-color: lightgrey;
  width: 300px;
  border: 15px solid green;
  padding: 50px;
  display: flex;
  float:left;
  margin: 20px;
  margin-top: 3%;
  margin-left: 3%;
 }
 .designclear{
  background-color: lightgrey;
  width: 300px;
  border: 15px plum;
  padding: 50px;
  margin: 20px;
  clear: left;
  float:left;
  margin-top: 3%;
  margin-left: 3%;
  display: flex;
 }
 div a{
  text-align: center;
  color:#333;
  text-decoration: none;
 }
```

OUTPUT:





Result:

Thus the WayFar travel website has been developed using HTML and CSS and the output was verified successfully.

Ex No : 4	BOOTSTRAP WEB APPLICATION

AIM:

To develop a web page using Bootstrap basic concepts.

ALGORITHM:

Step 1: Create basic html tags to display the contents.

Step 2: Using .container class and .container-fluid create a class and also create a gridsystem.

Step 3: Use <nav> tag to create navigation tabs.

Step 4: Add contents to the webpage that has to be displayed.

Step 5: Add some images to make your webpage interactive.

Step 6: Use <button> tag for creating different kinds of buttons

Step 7: Save and run the program.

Step 8: Observe the output in any browser.

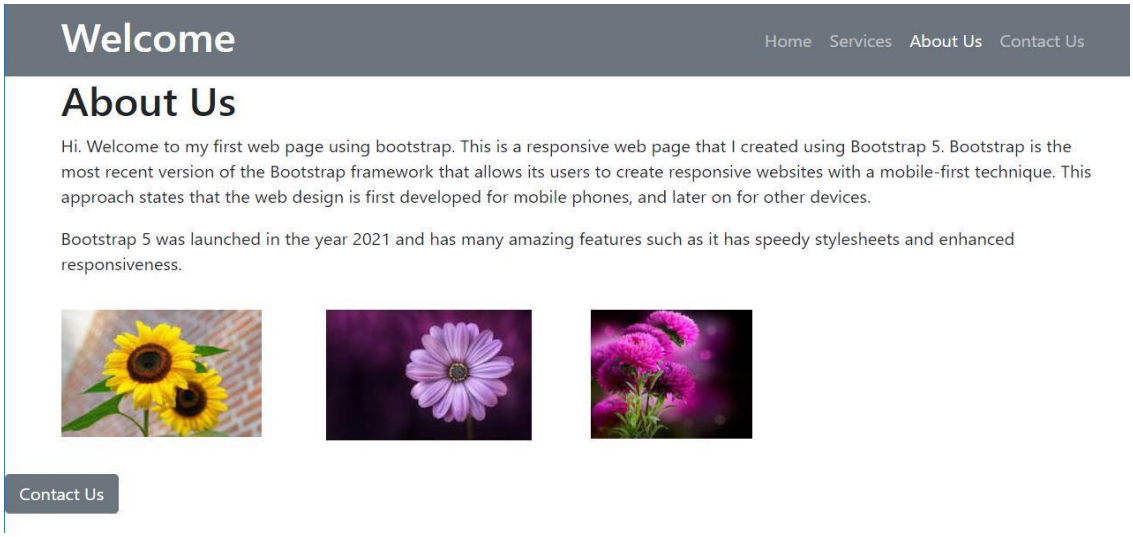
SOURCE CODE:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>My First Bootstrap 5 Website</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
rel="stylesheet">
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"></script>
</head>
<body>
  <nav class="navbar navbar-expand-sm bg-secondary navbar-dark">
    <div class="container">
      <div class="text-white text-center">
        <h1>Welcome</h1>
```

```
    </div>
    <ul class="navbar-nav ml-auto">
      <li class="nav-item">
        <a class="nav-link" href="#">Home</a>
      </li>
      <li class="nav-item">
        <a class="nav-link" href="#">Services</a>
      </li>
      <li class="nav-item">
        <a class="nav-link active" href="#">About Us</a>
      </li>
      <li class="nav-item">
        <a class="nav-link" href="#">Contact Us</a>
      </li>
    </ul>
  </div>
 </nav>
 <section id="aboutus">
  <div class="container">
    <div class="row align-items-left">
      <div class="col text-col">
        <h1>About Us</h1>
        <p>Hi. Welcome to my first web page. This is a responsive web page that I createdusing Bootstrap 5. Bootstrap is the most recent version of the Bootstrap framework that allows its users to create responsive websites with a mobile-first technique. This approachstates that the web design is first developed for mobile phones, and later on for other devices.</p>
        <p>Bootstrap 5 was launched in the year 2021 and has many amazing features such as it has speedy stylesheets and enhanced responsiveness.</p>
      </div>
    </div>
  </div>
 </section>
 <div class="container">
```

```
    <div class="row">
      <div class="col-md-4 mt-3 col-lg-3">
        <img src="img6.jpg" class="img-fluid" alt="image">
      </div>
      <div class="col-md-4 mt-3 col-lg-3">
        <img src="img1.jpg" class="img-fluid" alt="image">
      </div>
      <div class="col-md-4 mt-3 col-lg-3">
        <img src="img2.jpg" class="img-fluid" alt="image">
      </div>
    </div>
    </div>
    <iv class="container">
     <div class="mt-2">
      <button type="button" class="btn btn-secondary btn-large">Contact Us</button>
     </div>
    </div>
</body>
</html>
```

OUTPUT:



RESULT:

Thus the webpage using bootstrap basic concepts is implemented successfully.

Ex No : 5a	TELEVISION SHOP WEBPAGE WITH DATA VERIFICATION USING XML DTD

AIM:

To develop a webpage for television shops and verify data using XML DTD.

ALGORITHM:

STEP 1: Create the Root element.

STEP 2: Create child elements for the root element that is created.

STEP 3: Create the attributes that are required for every child element.

STEP 4: Give the values for every attribute of the child element.

STEP 5: Apply the internal DTD for the web page.

STEP 6: In order to display the XML file using CSS, link the XML file with CSS.

STEP 7: Define each element as a block by using the display property of CSS.

STEP 8: Identify the titles and bold them.

STEP 9: Save the file with the ".xml" extension and open the file using any browser.

SOURCE CODE:

television.html:

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/css" href="s.css"?>
<!DOCTYPE BEA [
<!ELEMENT BEA (BEA*)>
<!ELEMENT tv (name,type,screen_size,resolution,price,viewing_angle,os)>
<!ELEMENT type (#PCDATA)>
<!ELEMENT screen_size (#PCDATA)>
<!ELEMENT resolution (#PCDATA)>
<!ELEMENT price (#PCDATA)>
<!ELEMENT viewing_angle (#PCDATA)>
<!ELEMENT os (#PCDATA)>
<!ATTLIST tv brand (CDATA) #REQUIRED>
]>
<BEA>
   <tv brand="LG">
```

```
        <name>LG</name>
        <type>LED</type>
        <screen_size>32 inch</screen_size>
        <resolution>1080</resolution>
        <price>40,000</price>
        <viewing_angle>20</viewing_angle>
        <os>Android</os>
    </tv>
    <tv brand="MI">
        <name>MI</name>
        <type>OLED</type>
        <screen_size>40 inch</screen_size>
        <resolution>4K</resolution>
        <price>50,000</price>
        <viewing_angle>30</viewing_angle>
        <os>MIUI</os>
    </tv>
    <tv brand="Sony">
        <name>Sony</name>
        <type>LED</type>
        <screen_size>42 inch</screen_size>
        <resolution>4K</resolution>
        <price>70,000</price>
        <viewing_angle>50</viewing_angle>
        <os>Android</os>
    </tv>
</BEA>
```

s.css

```
BEA
{
margin:10px;
background-color:pink;
font-family:verdana;
}
```

```
name
{
display:block;
font-weight:bold;
text-align: center;
line-break: auto;
}
type,screen_size,resolution,price,viewing_angle,os
{
text-align: center;
display:block;
color:blue;
font-size:small;
font-style:italic;
}
```

OUTPUT:

RESULT:

Thus to develop a webpage for television shops and verify data using XML DTD is successfully studied and implemented.

Ex No : 5b	FOOD RECIPE WEB PAGE WITH DATA VERIFICATION USING XML SCHEMA

AIM:

To develop a webpage for food recipe and verify data using XML Schema.

ALGORITHM:

STEP 1: Create the Root element.

STEP 2: Create child elements for the Root element created.

STEP 3: Create the items that are required for every food recipe.

STEP 4: Create <xs:complex type> after the element name.

STEP 5: Create <xs:sequence> after complex type.

STEP 6: In order to display the XML file using CSS, link the XML file with CSS.

STEP 7: Define each element as a block by using the display property of CSS.

STEP 8: Identify the titles and bold them.

STEP 9: Save the file with .xml extension and open the file using any browser.

SOURCE CODE:

Food.html:

```
<?xml version="1.0" encoding="UTF-8"?>

<?xml-stylesheet type="text/css" href="style.css"?>
<food>
   <title>Food recipes</title>
   <section>
      <name>Noodles</name>
      <i1>Water</i1>
      <i2>Masala</i2>
      <i3>Noodles</i3>
      <i4>Salt</i4>
      <recipe>Recipe: Boil 6 cups water in a large pot or a deep pan and bring it to boil over medium flame. When it comes to rolling boil, add dried noodles,
                  1/2 tablespoon oil and 1/2 teaspoon salt. Boil until noodles are soft, it will take around 4-5 minutes. Stir occasionally in between.</recipe>
   </section>
   <section>
      <name>Coconut chutney</name>
      <i1>coconut</i1>
      <i2>green chillies</i2>
      <i3>ginger and chana dal</i3>
```

```
            <i4>Salt</i4>
      <i5>water</i5>
            <recipe>Recipe: Coconut chutney is made by grinding coconut, green chillies,
ginger and chana dal with water to a fine paste. Then a tempering
            of mustard seeds, red chilli and curry leaves is added to the fine paste to make
nariyal chutney.</recipe>
   </section>
   <section>
      <name>Kesari</name>
      <i1>rava </i1>
      <i2>sugar</i2>
      <i3>ghee</i3>
      <i4>saffron and dry fruits</i4>
            <recipe>Recipe: Kesari recipe is a popular and delicious South Indian sweet
made from rava (cream of wheat), sugar, ghee (clarified butter), saffron and dry fruits. This
melt in the mouth orange-colored sweet is alsoknown as Rava Kesari.</recipe>
   </section>
</food>
```

Style.css

```
food {
            font-size:60;
            margin:0.5em;
            font-family: ThreeDShadow;
            display:block;}
section {
            display:block;
            border: 1px yellow;
            margin:0.5em;
            padding:0.5em;
            background-color:olive;}
title {
     display:block;
     font-weight:bolder;
     text-align:center;
     font-size:30px;
     background-color:bl;
     color:brown;}
name, i1,i2,i3,i4,i5{
                        display:block;
                        text-align:cente }
name {
    color:cyan;

    text-decoration: underline ;
    font-weight:bolder;
    font-size:20px;
    }
```

OUTPUT:

Food recipies

Noodles
Water
Masala
Noodles
Salt
Recipe: Boil 6 cups water in a large pot or a deep pan and bring it to boil over medium flame. When it comes to rolling boil, add dried noodles, 1/2 tablespoon oil and 1/2 teaspoon salt. Boil until noodles are soft, it will take around 4-5 minutes. Stir occasionally in between.

Coconut chutney
coconut
green chillies
ginger and chana dal
Salt
water
Recipe: Coconut chutney is made by grinding coconut, green chillies, ginger and chana dal with water to a fine paste. Then a tempering of mustard seeds, red chilli and curry leaves is added to the fine paste to make nariyal chutney.

Kesari
rava
sugar
ghee
saffron and dry fruits
Recipe: Kesari recipe is a popular and delicious South Indian sweet made from rava (cream of wheat), sugar, ghee (clarified butter), saffron and dry fruits. This melt in the mouth orange-colored sweet is also known as Rava Kesari.

RESULT:

Thus to develop a webpage for food recipes and verify data using XML schema is successfully studied and implemented.

Ex No : 6a	MARRIAGE VALIDATOR USING JAVASCRIPT

AIM:

To write a JavaScript program to check whether the marriage is valid or not.

ALGORITHM:

STEP 1: Open the Visual Studio code.

STEP 2: Create the new folder.

STEP 3: Create a new file with .html extension.

STEP 4: Give the required header files or tags.

STEP 5: Give the required style for the web page like background, font color etc.

STEP 6: Use form tag for text box.

STEP 7: Create a function called validate.

STEP 8: If the bride's age is greater than 18 and groom's age is greater than 21, then it is valid marriage else not a valid marriage.

STEP 9: Close all the tags.

STEP 10: Save and Compile the program.

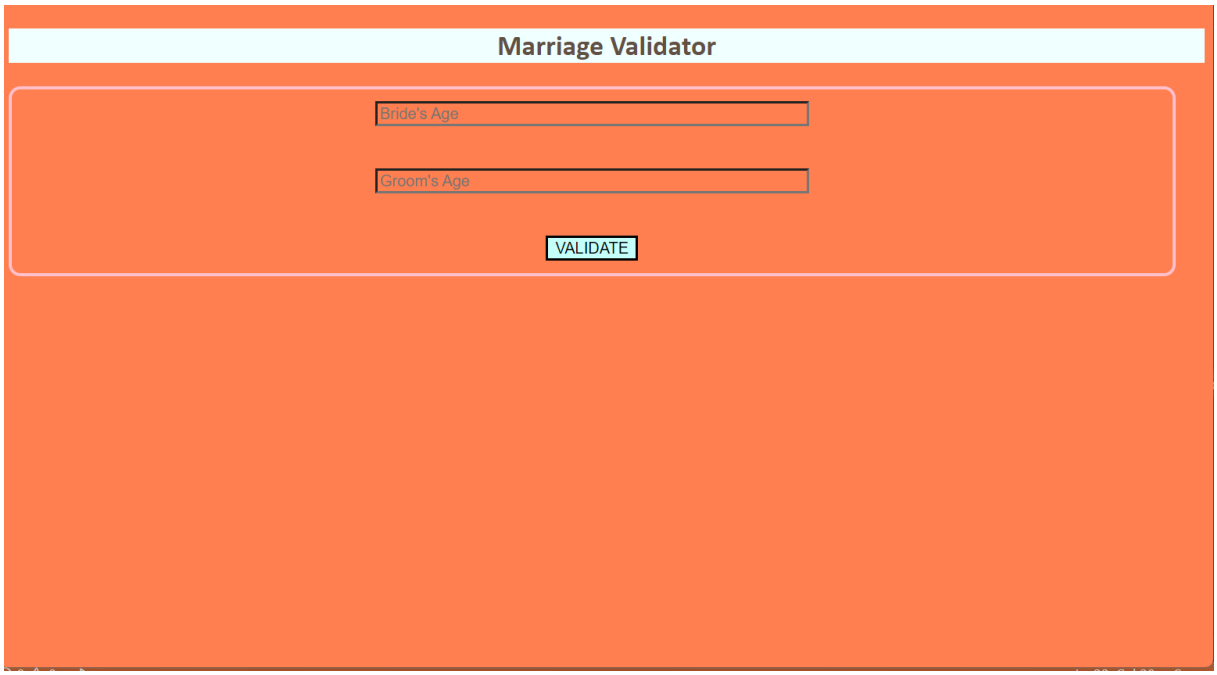
SOURCE CODE:

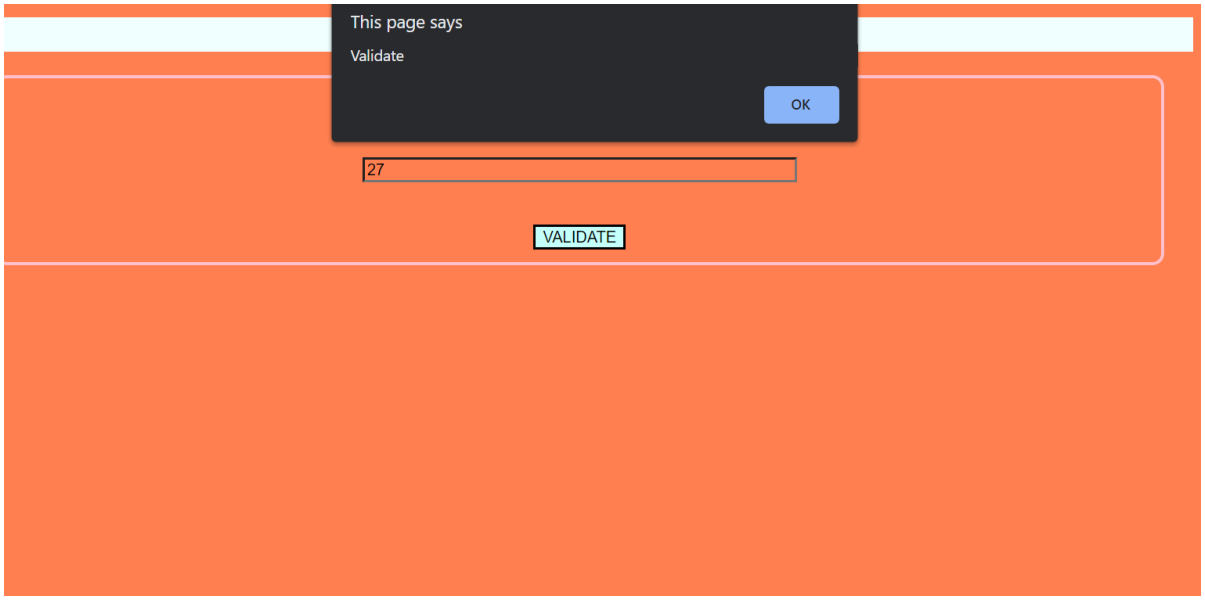
```
<!DOCTYPE html>
<html>
   <head>
      <title>Marriage Validator</title>
      <script language="Javascript" type="text/javascript"></script>
   </head>
   <style>
   *{
      background-color:lavender;
    }
    .new {
   width: 95%;
   border: 3px solid darkblue;
```

```
    border-radius: 10px;
    padding: 10px;
    align-items: center;
    }
    .btn {background-color:bisque;}
    </style>
<body>
    <form>
        <h2 align=center style="color:brown;font-family:calibri;background-
color:azure">Marriage Validator</h2>
        <div align=center class="new" style="color:rebeccapurple">
        <input type="text" placeholder="Bride's Age" id="bride" size="50"/></br></br></br>
        <input type="text" placeholder="Groom's Age" id="groom"
size="50"/></br></br></br>
        <div class="container">
            <input type="button" class="btn btn" onclick="check()" value="VALIDATE"/>
        </div>
    </div>
    </form>
    <script>
        function check()
        {
            var a=document.getElementById("bride").value;
            var b=document.getElementById("groom").value;
            if(a>=18 && b>=21)
            {
                alert("Validate");

            }
            else
            {
                 alert("Not validate");
            }
        }
```

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

OUTPUT:





RESULT:

Thus the JavaScript program to validate the age for marriage is verified and executed successfully.

Ex No : 6b	PALINDROME CHECKER USING JAVASCRIPT

AIM:

To write a JavaScript program to check whether the given string is palindrome or not.

ALGORITHM:

STEP 1: Open the Visual Studio code.

STEP 2: Give the title as Palindrome.

STEP 3: Get the strings or number from the user.

STEP 4: Take the temporary variable that holds a number.

STEP 5: Reverse the given string.

STEP 6: Compare the original numbers with the reversed number.

STEP 7: If the temporary and original number are the same, the number or string is apalindrome and else the given string is not a palindrome.

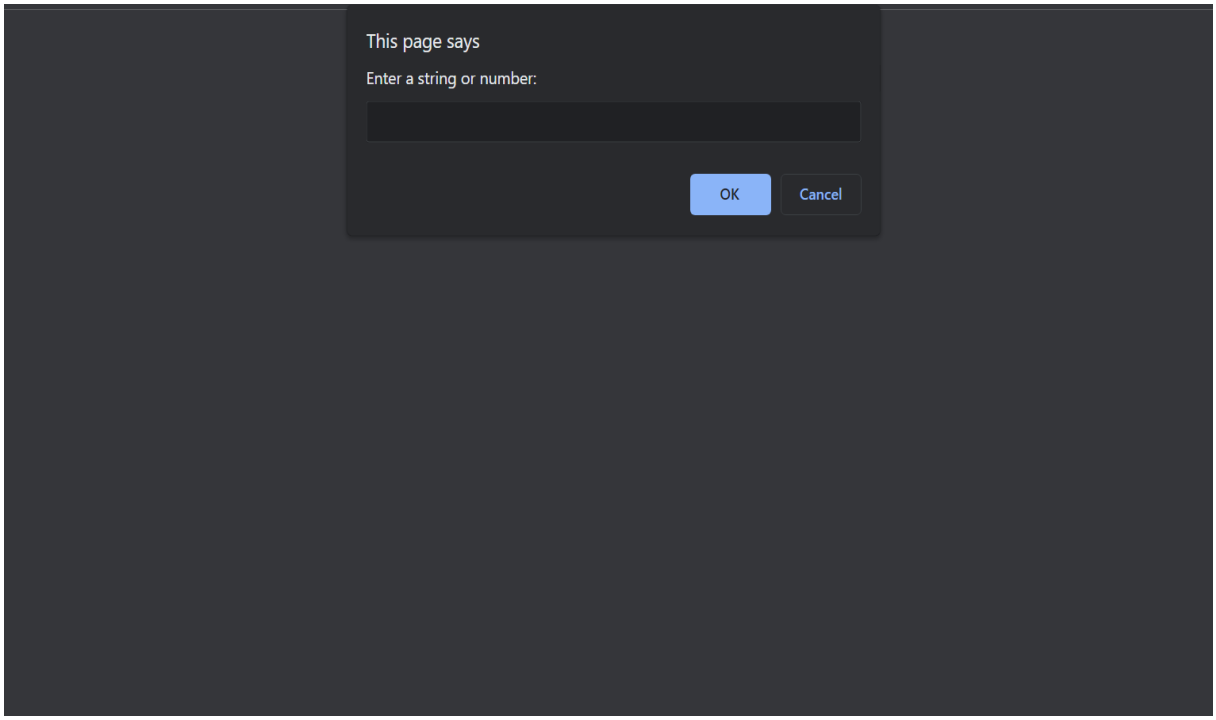
STEP 8: Save and Compile the program.

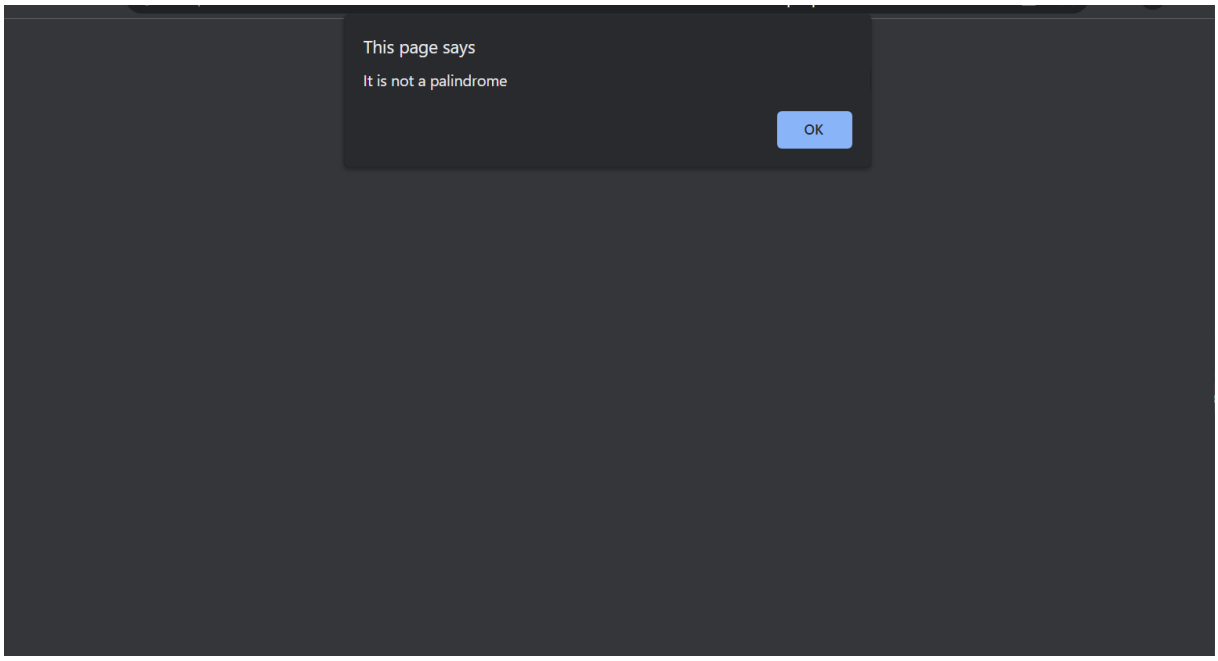
SOURCE CODE:

```
<html>
<head> <title> JavaScript Palindrome </title>
</head>
<body>
<script>
function validatePalin(str) {
   const len = string.length;
   for (let i = 0; i < len / 2; i++) {
      if (string[i] !== string[len - 1 - i]) {
         alert( 'It is not a palindrome');
      }
   }
   alert( 'It is a palindrome');
}
const string = prompt('Enter a string or number: ');
const value = validatePalin(string);
console.log(value);
```

</script></body> <html>

OUTPUT:





RESULT:

Thus the JavaScript program to check whether the given string is palindrome or not isverified and executed successfully.

Ex No : 6c	ELECTIVE COURSE REGISTRATION USING JAVASCRIPT

AIM:

To write a JavaScript program for open elective course registration.

ALGORITHM:

STEP 1: Write a JavaScript program inside the script tag

STEP 2: Use document.formname.name.value function to get the value of textbox.

STEP 3: If the username and password are empty, it alerts fill the text fields.

STEP 4: Else the username and password are correct it leads to the elective registration page.

STEP 5: If any of the text fields are empty, it alerts please fill the text boxes.

STEP 6: If the email syntax is wrong, it alerts please enter the valid email.

STEP 7: Else all details are entered, it alerts the user Login Successfully.

STEP 8: Apply required styles in style tag.

SOURCE CODE:

```
<!DOCTYPE html>
<html>
<head>
<title>Login Credentials</title>
<script language="Javascript" type="text/javascript">
function login()
{
var k=document.myform.user.value;
var m=document.myform.pass.value;
if(k=="" || m==""){
   alert("fill the username or password or both");
   return false;
}
if(k=="2005034" && m=="srec@123")
```

```
{
   alert("login sucessfully");
   window.location="course selection.html";
   return false;
}
else{
   alert("Enter valid password or username");
   return false;
}
}
</script>
<style type="text/css" media="all">

h1 {
   text-align: center;
   padding: 30px;
   background-color:khaki;
   color: white;
   }
   .mail {
   margin: auto;
   width: 50%;
   border: 3px solid gray;
   border-radius: 5px;
   padding: 10px;
   background-color: lightslategray;
   margin-top: 40px;
   }
   #mail {
   width: 100%;
   border: 3px solid gray;
   border-radius: 5px;
   padding: 5px;
   }
   body{background-image: url("8img.jpg");}
```

```
        </style>
        <link
    href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
    rel="stylesheet">
        <script
    src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"></
    script>
        </head>
        <body>
        <h1 align=center style="color:brown;font-family:calibri "> SREC COURSE
    REGISTRATION </h1>
        <form name="myform" onsubmit="login()">
        <div align=center class="mail" style="color:white;font-size:19px">
        UserName : <input type="text"name="user" placeholder="Enter UserName"
    id="mail" required><br><br>
        Password : <input type="password" name="pass" placeholder="Enter
    password" id="mail" required  title="dont leave the field empty"><br><br>
        <div class="container">
          <input type="button" class="btn btn-warning btn-sm" onclick="login()"
    value="submit"/>
        </div>
        <input type="reset"><br><br>

        </form>
        </body>
</html>
```

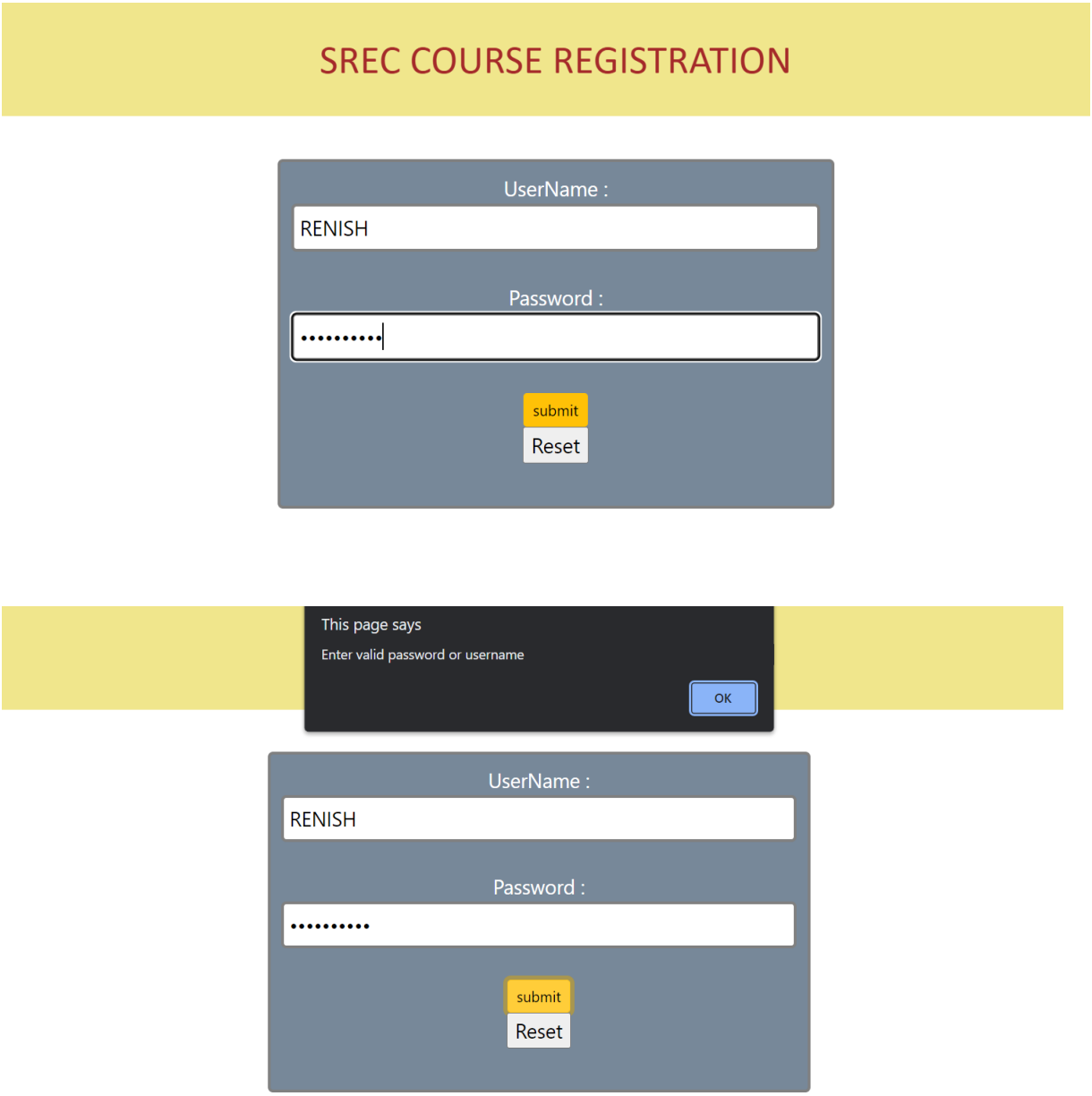
course selection.html

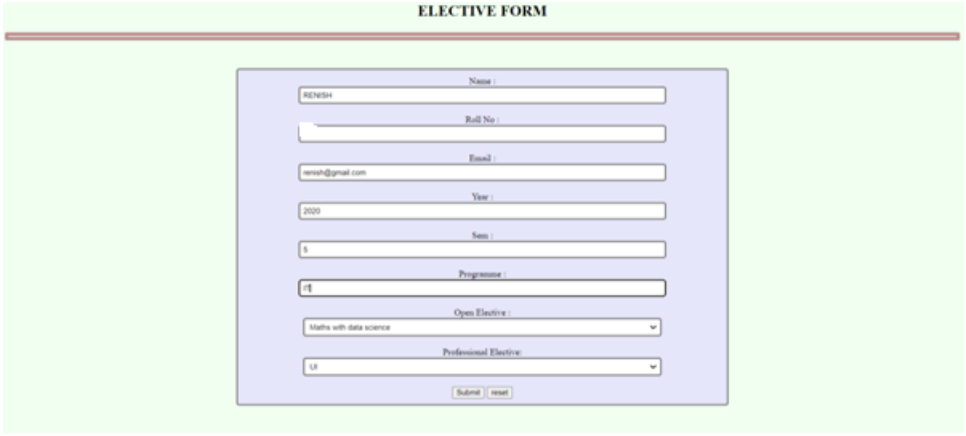
```
<!DOCTYPE html>
<html>
<head>
<title>Form</title>
<style>
   hr{
         border: 7px solid rosybrown;
         border-radius:3px;
       }
        .mail {
   margin: auto;
   width: 50%;
   border: 3px solid gray;
   border-radius: 5px;
   padding: 10px;
   background-color: lavender;
   margin-top: 50px;
   }
   #mail {
   width: 75%;
   border: 3px solid gray;
   border-radius: 5px;
   padding: 5px;
   }
</style>
<script language="Javascript" type="text/javascript">
   function elective()
   {
   var k=document.f.k.value;
   var m=document.f.m.value;
   var n=document.f.n.value;
   var o=document.f.o.value;
   var p=document.f.p.value;
   var q=document.f.q.value;
   if(k=="")
```

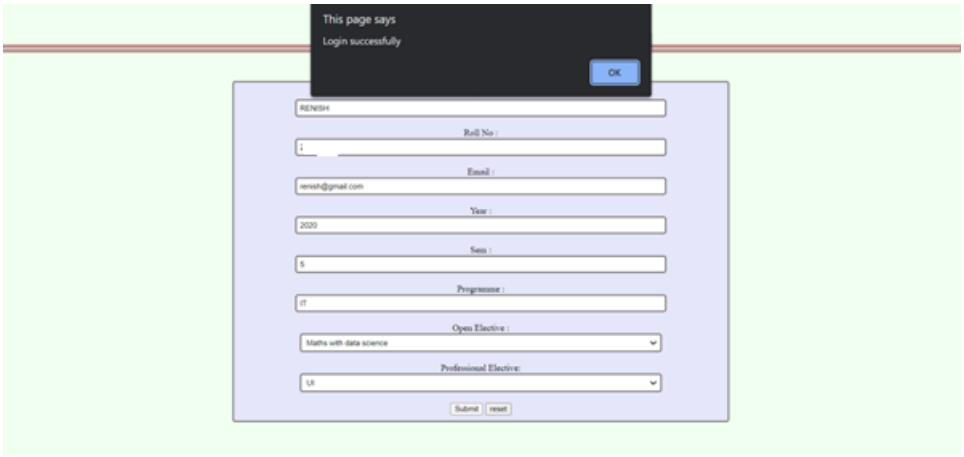
```
{
alert("please enter the name");
return false;
}
if(m==""){
    alert("please enter the Roll no");
    return false;
}
if(n=="" ){
    alert("please enter the Email");
    return false;
}
if(o==""){
    alert("please enter the Year");
    return false;
}
if(p==""){
    alert("please enter the sem");
    return false;
}
if(q==""){
    alert("please enter the programe");
    return false;
}
else{
    alert("Login successfully");
    return false;
}
}
</script>
<body style="background-color: honeydew">
<form  name="f" onsubmit="elective()">
<font face = "italic" size=4>
   <center >
<h2 align=center >  ELECTIVE FORM </h2></font>
<hr>
<div class="mail">
```

```
Name :<br><input type="text" placeholder="Enter Name" name="k"
id="mail" required>
<br><br>
Roll No :<br><input type="text" placeholder="Enter Roll no" name="m"
id="mail" required>
<br><br>
Email :<br><input type="email" id="mail" placeholder="Enter Email"
name="n" required title="Enter Valid Email">
<br><br>
Year :<br><input type="text" id="mail" placeholder="Enter Year" name="o"
required>
<br><br>
Sem :<br><input type="text" id="mail" placeholder="Enter semester"
name="p" required>
<br><br>
Programme :<br><input type="text" id="mail" placeholder="Enter
Programme" name="q" required>
<br><br>
Open Elective :<br>
<select id="mail" placeholder="Elective"  required>
<option value = "Maths with data science">Maths with data science
<option value = "Diaster Management">Diaster Management
<option value = "ESS">ESS
</select>
<br><br>
Professional Elective:<br>
<select id="mail" required>
<option value = "UI">UI
<option value = "AI">AI
<option value = "Digital Marketting">Digital Marketting
</select>
<br><br>
<input type = "submit" onClick="elective()" value="Submit">
<input type="reset" value="reset">
</div>
</center>
</font>
</form></body></html>
```

OUTPUT:


SREC COURSE REGISTRATION
UserName :
RENISH
Password :
submit
Reset
This page says
Enter valid password or username
OK
UserName :
RENISH
Password :
submit
Reset





RESULT:

Thus the open elective course registration webpage using JavaScript had beendesigned successfully and output has been verified.

Ex No : 7	ANGULAR WEB APPLICATION

AIM:

To create an Angular web application

ALGORITHM:

Step1: Install the Angular cli using the command npm install -g @angular/cli.

Step2: Create a new project by using the command ng new myNewApp.

Step3: Go to the project directory by using the command cd myNewApp.

Step4: In app.component.ts, add the necessary variables and methods for the website.

Step5: Write the HTML contents in the app.component.html.

Step6: For more styling write the contents in app.component.css.

Step6: Run the application using the command ng serve–o.

SOURCE CODE:

app.component.ts:

```
import { Component } from '@angular/core';


@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title= "Angular Kitchen";
  changeName() {
  this.title="Typescript";
  }
  run:boolean=false;
  user=[{name:'Gopi',age:26,gender:'m'},{name:'kannan',age:35,gender:'m'},
  {name:'john',age:23,gender:'m'},
  {name:'Hasini',age:34,gender:'f'}];
  colorName='yellow';
```

```
  color='red';
  borderStyle = '1px solid black';
  isBordered = true;
  name: string = "Angular";
  day=new Date();
  num=2345.57898766;
  n=20;
}
```

app.component.html:

```
<div class="container">
  <div>
    <h1>Templates</h1>
    <p>Hello world...</p>
    <p>1 + 2 = {{ 1+2 }}</p>
    <p>10 is {{ 10%2==0?'Even':'Odd' }}</p>
  </div>
<hr>
  <h1>Template Statement- events </h1>
      <p>Welcome {{title|uppercase}} </p>
      <p (click) = "changeName()">Click Me</p>
<hr>
<h1>Structural Directives</h1>
<div *ngIf="run">
  <p>My App is running</p>
</div>
<div *ngFor="let u of user">
  {{u.name}}
  {{u.age}}
</div>
<div *ngFor="let u of user">
```

```
  <div [ngSwitch]="u.gender">
    <div *ngSwitchCase="'m'" id="male">
      {{u.name}}
    </div>
    <div *ngSwitchCase="'f'" id="female">
      {{u.name}}
    </div>
  </div>
</div>
<hr>
<h1>Attribute Directives</h1>
<div [style.background-color]="colorName" [style.color]="color"
[style.border]="borderStyle">
  Uses fixed yellow background
</div>
<p [ngStyle]="{
  color:colorName,
  borderBottom: borderStyle
}">
Demo for attribute directive ngStyle
</p>
<div [class.bordered]="isBordered">
  Border {{ isBordered ? "ON" : "OFF" }}
</div>

<input type="text" [(ngModel)]="name">
<br/>
<div>Hello , {{ name }}</div>
<hr>
<h1>Pipes</h1>
<p>{{name|uppercase}}</p>
<p>{{name|lowercase}}</p>
<p>{{name|titlecase}}</p>
<p>{{day|date}}</p>
```

```
<p>{{day|date:'dd-MMM-yy'}}</p>
<p>{{num|number}}</p>
<p>{{num|number:'5.2-4'}}</p>
<p>{{n|percent}}</p>
<p>{{n|currency}}</p>
<hr>
<h1>Hello Component</h1>
  <app-hello></app-hello>
<h1>App Component</h1>
  <div class="p-2">
    <header class="border border-secondary">
      <h1 class="text-center">{{title}} </h1>
    </header>
  </div>
  <div class="p-2">
    <section class="p-1 border border-secondary">
      <nav class="nav nav-pills nav-justified">
        <p>Navigation links</p>
        <a class="nav-link" [routerLink]="['/hello']" routerLinkActive="active">Hello</a>
        <a class="nav-link" [routerLink]="['/header']" routerLinkActive="active">Header</a>
      </nav>
    </section>
  </div>
  <div class="p-2">
    <section class="p-1 border border-secondary">
      <router-outlet> </router-outlet>
    </section>
  </div>
  <div class="p-2">
    <section class="p-1 border border-secondary">
      <h2>demos will appear here</h2>
    </section>
  </div>
  <h1>Footer Component</h1>
```

```
    <app-footer></app-footer>
  </div>
```

app.component.css:

```
span{
    color:blue
}
#male
{
    background-color: blueviolet;
}
#female
{
    background-color: deeppink;
}
.bordered {
        border: 1px dashed black;
        background-color: #eee;
}
```

app.module.ts:

```
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
import { HelloComponent } from './hello/hello.component';
import { FooterComponent } from './footer/footer.component';
import { HeaderComponent } from './header/header.component';
import { FormsModule } from '@angular/forms';

@NgModule({
  declarations: [
  AppComponent,
  HelloComponent,
  FooterComponent,
```

```
    HeaderComponent
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    FormsModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

app-routing.module.ts:

```
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { HeaderComponent } from './header/header.component';
import { HelloComponent } from './hello/hello.component';
const routes: Routes = [{ path:'hello',component:HelloComponent},
{ path:'header',component:HeaderComponent},
{ path: '', redirectTo: '/hello', pathMatch: 'full' },];

@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})
export class AppRoutingModule { }
```

hello.component.html:

```
<p>hello works!</p>
<p>
    Hello {{ courseName }}
  </p>
```

hello.component.css:

```
p {
    color:chocolate;
```

```
    font-size:20px;
}
```

hello.component.spec.ts:

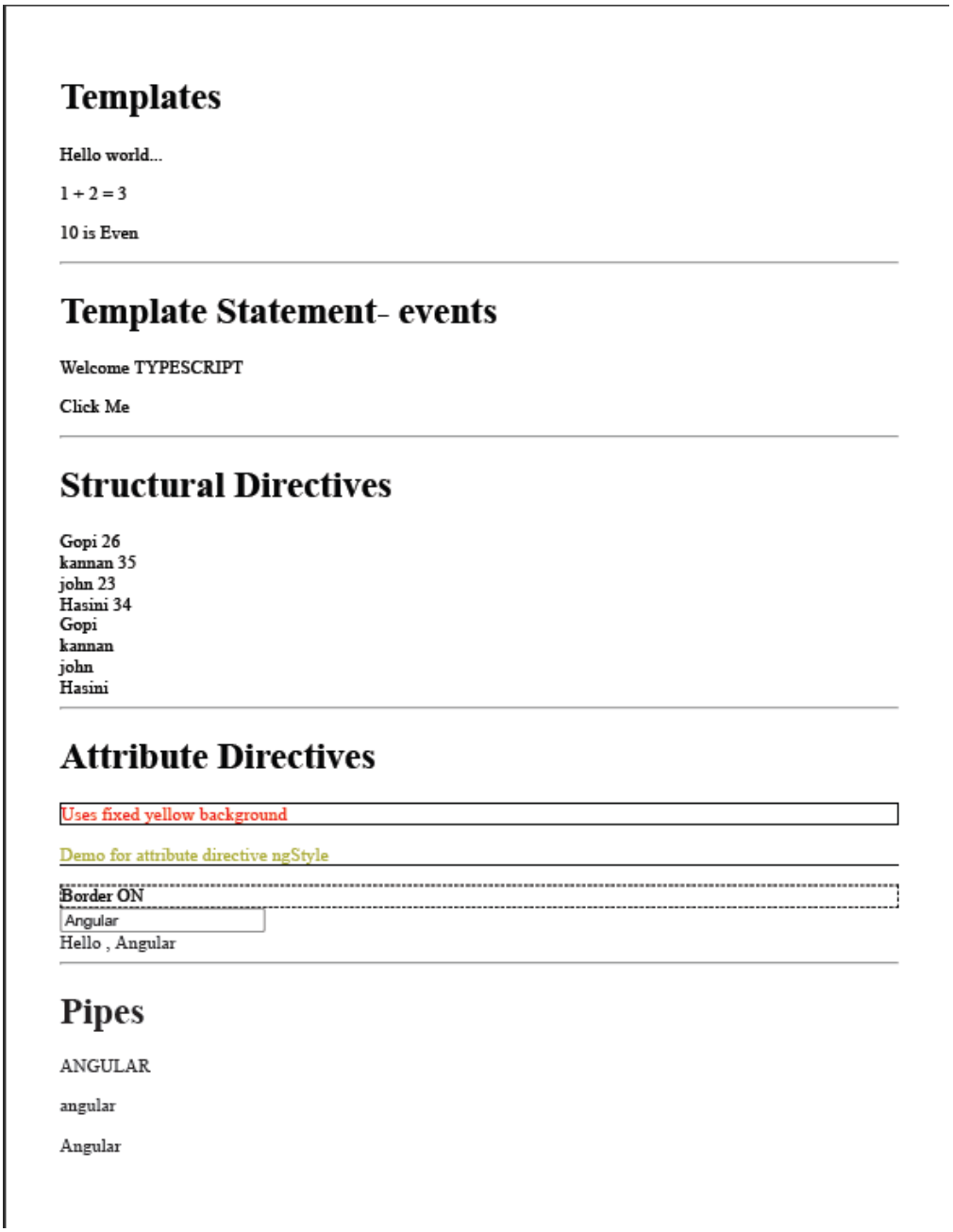
```
import { ComponentFixture, TestBed } from '@angular/core/testing';
import { HelloComponent } from './hello.component';
describe('HelloComponent', () => {
  let component: HelloComponent;
  let fixture: ComponentFixture<HelloComponent>;

  beforeEach(async () => {
    await TestBed.configureTestingModule({
      declarations: [ HelloComponent ]
    })
    .compileComponents();
  });


  beforeEach(() => {
    fixture = TestBed.createComponent(HelloComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

  it('should create', () => {
    expect(component).toBeTruthy();
  });
});
```

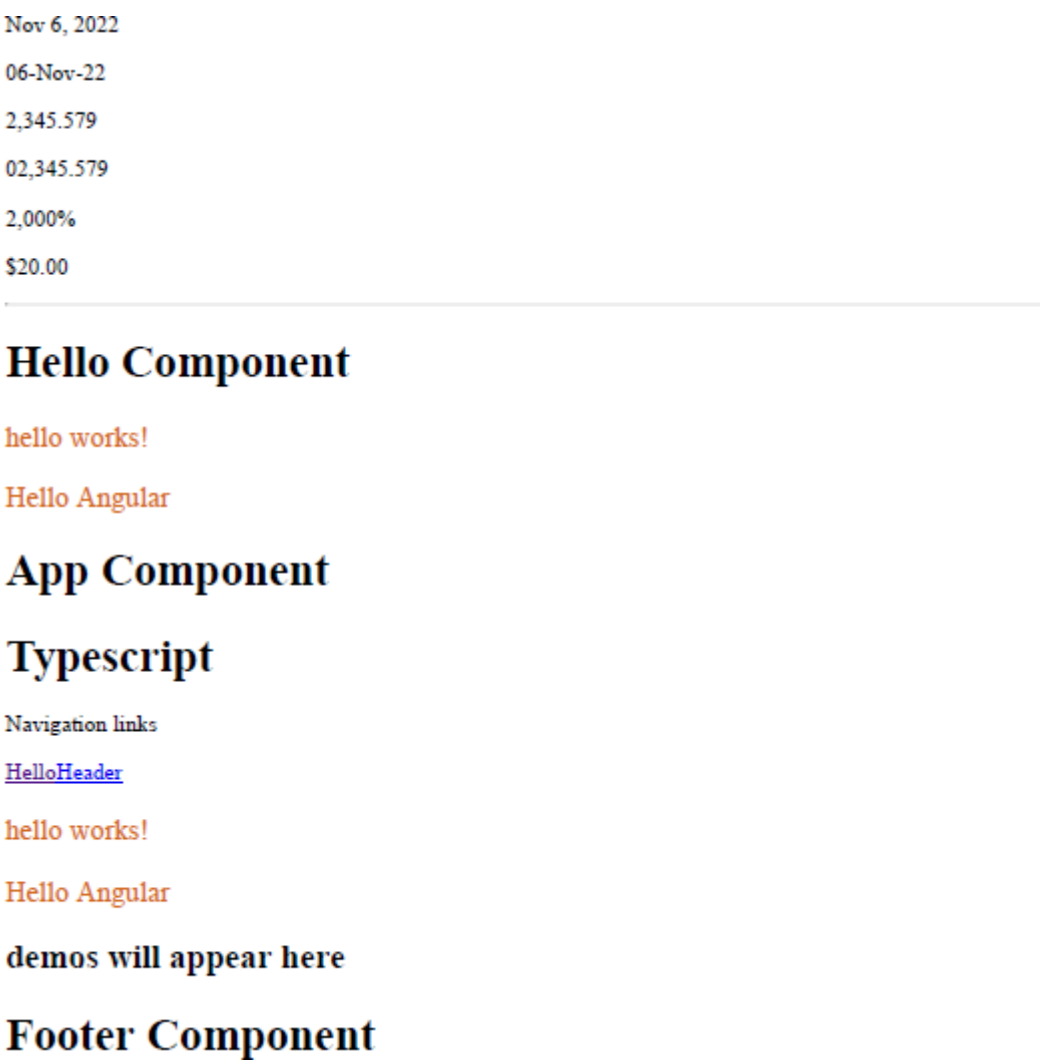
hello.component.ts:

```
import { Component, OnInit } from '@angular/core';

@Component({
  selector: 'app-hello',
  templateUrl: './hello.component.html',
```

```
  styleUrls: ['./hello.component.css']
})
export class HelloComponent implements OnInit {
  courseName: string = "Angular";
  constructor() { }

  ngOnInit(): void {
  }
}
```

OUTPUT:





RESULT:

Thus the program to create a web application using angular is verified and executed successfully.

Ex No : 8	BLOG ARTICLES APPLICATION USING REACTJS

AIM

To perform blog articles application using Reactjs.

ALGORITHM

STEP 1: Open the terminal and create a new application in react.
STEP 2: The new application is created by using the command: npxcreate-react-app (app name).
STEP 3: Open the project in VS code and open a new terminalwindow.
STEP 4: In the terminal window,execute the command :npm i -Dreact-router-dom.
STEP 5: In the src folder, create a new file directory named pages.
STEP 6: Include the Layout.js,Home.js,Blog.js,Contact.js files in thepage directory.
STEP 7: In the index.js file include the above pages and implementpage routing.
STEP 8: Save and run the command using npm start.
STEP 9: Observe the output.

SOURCE CODE

index.js:

```
import ReactDOM from "react-dom/client";
import { BrowserRouter, Routes, Route } from "react-router-dom";
import Layout from "./pages/Layout";
import Home from "./pages/Home";
import Blogs from "./pages/Blogs";
import Contact from "./pages/Contact";
import NoPage from "./pages/NoPage";

export default function App() {
  return (
    <BrowserRouter>
      <Routes>
        <Route path="/" element={<Layout />}>
          <Route index element={<Home />} />
          <Route path="blogs" element={<Blogs />} />
```

```
        <Route path="contact" element={<Contact />} />
        <Route path="*" element={<NoPage />} />
      </Route>
    </Routes>
  </BrowserRouter>
 );
}

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(<App />);
```

Layout.js:

```
import { Outlet, Link } from "react-router-dom";

const Layout = () => {

 return (
  <>
    <nav>

      <ul>

        <li>

          <Link to="/">Home</Link>

        </li>

        <li>

          <Link to="/blogs">Blogs</Link>

        </li>

        <li>

          <Link to="/contact">Contact</Link>

        </li>

      </ul>

    </nav>

    <Outlet />
```

```
    </>
  )
};
export default Layout;
```

Home.js:

```
const Home = () => {
  return <h1>Home</h1>;
};
export default Home;
```

Blogs.js:

```
const Blogs = () => {
  return <h1>Blog Articles</h1>;
};
export default Blogs;
```

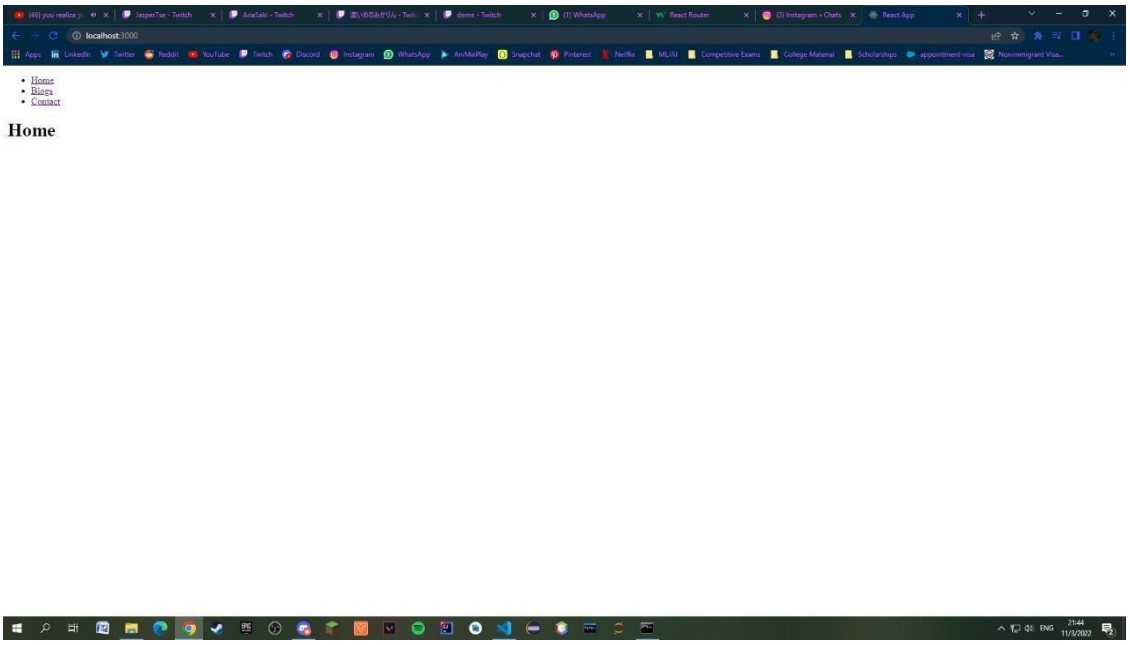
Contact.js:

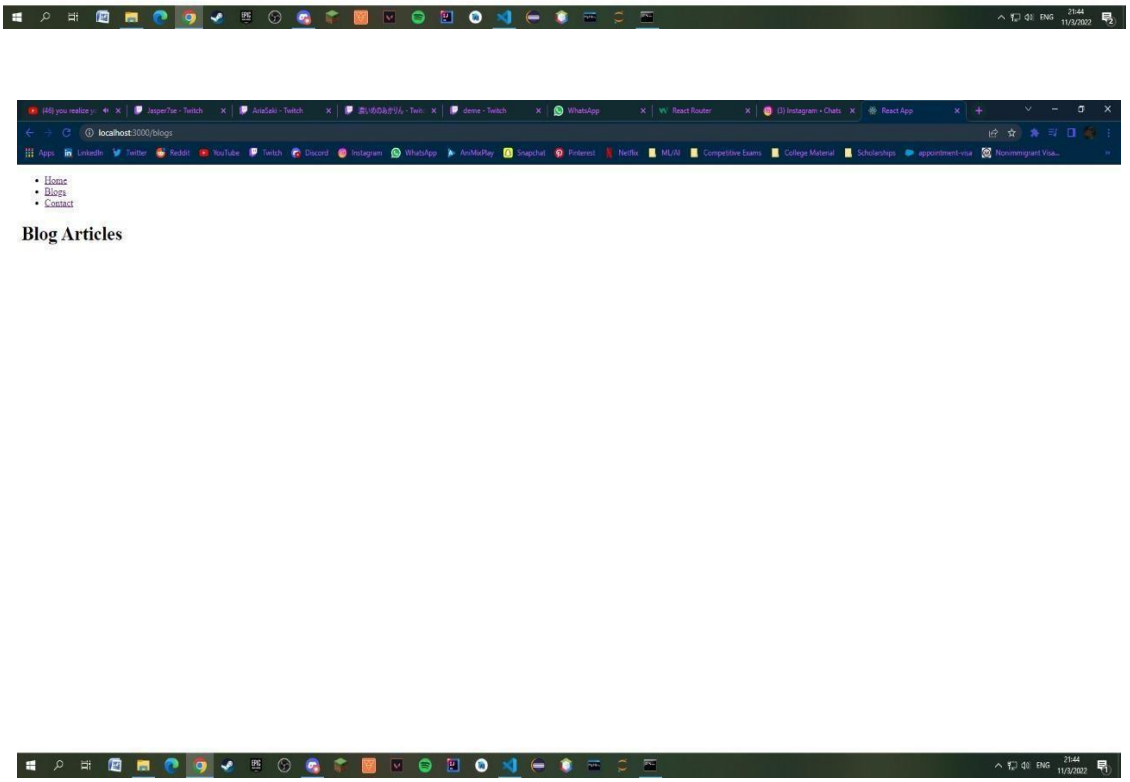
```
const Contact = () => {
  return <h1>Contact Me</h1>;
};
export default Contact;
```

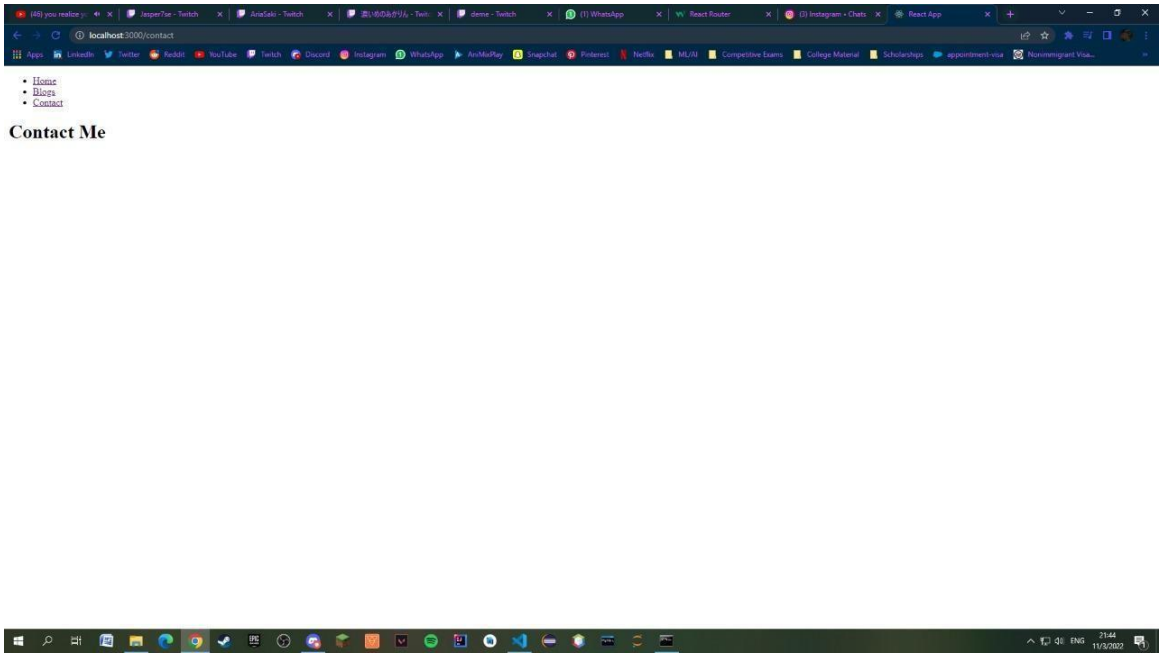
NoPage.js:

```
const NoPage = () => {
  return <h1>404</h1>;
};
export default NoPage;
```

OUTPUT:


localhost:3000
Home
Blogs
Contact
Home


localhost:3000/blogs
Home
Blogs
Contact
Blog Articles



RESULT:

Thus, the blog articles applications using React.js are verified and executedsuccessfully.

Ex No : 9	EVENT REGISTRATION USING NODE.JS

AIM

To design a node js application to make event registration by the user.

ALGORITHM

STEP 1: Install Node.js.

STEP 2: Create a Folder named Event and open it in visual studio code

STEP 3: Open terminal and type “npm init” to install package.json file

STEP 4: Install express by the command “npm install express” in the terminal

STEP 5: Create a HTML file named index.html and create a form forregistration of user by entering his/her details

STEP 6: Create a javascript file named example.js

STEP 7: In example.js, write an express code to connect to server and to get thedetails of form submitted by user by listening to the port 8000.

STEP 8: Execute the js file by the command “node example.js”

STEP 9: In the browser, paste the html file path and fill the form.

STEP 10: Browser will be redirected to server address and details submitted byuser will be displayed.

SOURCE CODE

index.html

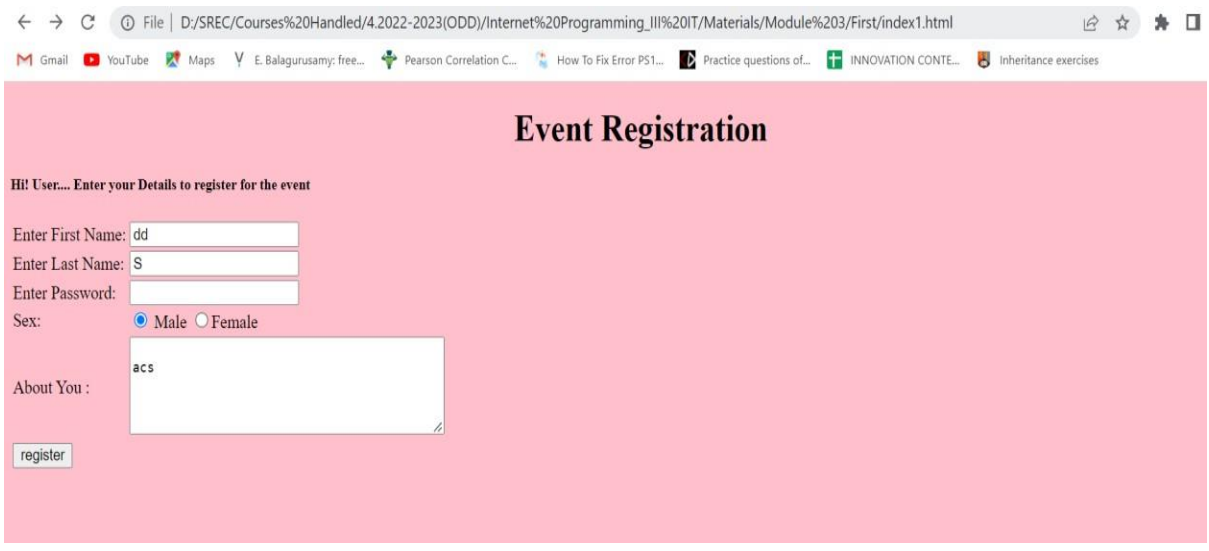
```
<!DOCTYPE html>
<html>
<body bgcolor="pink">
   <h1><center> Event Registration </center></h1>
   <h5>Hi! User. ...Enter your Details to register for the event</h5>
<form action="http://127.0.0.1:8000/example">
<table>
<tr><td>Enter First Name:</td><td><input type="text" name="firstname"/><td></tr>
<tr><td>Enter Last Name:</td><td><input type="text" name="lastname"/><td></tr>
```

```
<tr><td>Enter Password:</td><td><input type="password"
name="password"/></td></tr>
<tr><td>Sex:</td><td>
<input type="radio" name="sex" value="male"> Male
<input type="radio" name="sex" value="female">Female
</td></tr>
<tr><td>About You :</td><td>
<textarea rows="5" cols="40" name="aboutyou" placeholder="Write about yourself">
</textarea>
</td></tr>
<tr><td colspan="2"><input type="submit" value="register"/></td></tr>
</table>
</form>
</body>
</html>
```

example.js

```
var express = require('express');
var app=express();

app.get('/example', function (req, res) {
res.send('<p>Firstname: ' + req.query['firstname']+'</p><p>Lastname:
'+req.query['lastname']+'</p><p>Password: '+req.query['password']+'</p><p>AboutYou:
'+req.query['aboutyou']+'</p>');
})

var server = app.listen(8000, function () {
  var host = server.address().address
  var port = server.address().port
  console.log("Example app listening at http://%s:%s", host, port)
})
```

OUTPUT



Firstname: dd

Lastname: S

Password:

AboutYou: acs

RESULT

Thus, the node js application for event registration by the user has been implemented successfully.

Ex No : 10	WEB APPLICATION USING NODE.JS AND MYSQL WITH CRUD OPERATIONS

AIM

To design a node.js application to make CRUD operations in MySQL database for management of products.

ALGORITHM

STEP 1: Install node.js, mysql server

STEP 2: check node is properly installed using node -v command

STEP 3: Create a folder "nodemysql" and open in visual studio code

STEP 4: Connect to local host database using MySql Workbench

STEP 5: In vscode terminal, type command

npm install mysql – to install mysql module npm install express – to install express module

STEP 6: Create conn.js file

STEP 7: In the conn.js file, required modules are included

STEP 8: Create connection to server with the port number 3000. Once successful, it prints "connection successful" or else error will be thrown

STEP 9: Using the app of express, Connection to the database "product" is created.

STEP 10: Create two tables, "chocolates" and "stationary" and verify the successful creation in MySql Workbench

STEP 11: Insert the product details into the table using INSERT query

STEP 12: Display all the values inserted using the SELECT query

STEP 13: Update the name of the chocolate for a specific productid (pid) and verify thechange in database

STEP 14: Delete the product with specific pid from the table

SOURCE CODE

conn.js

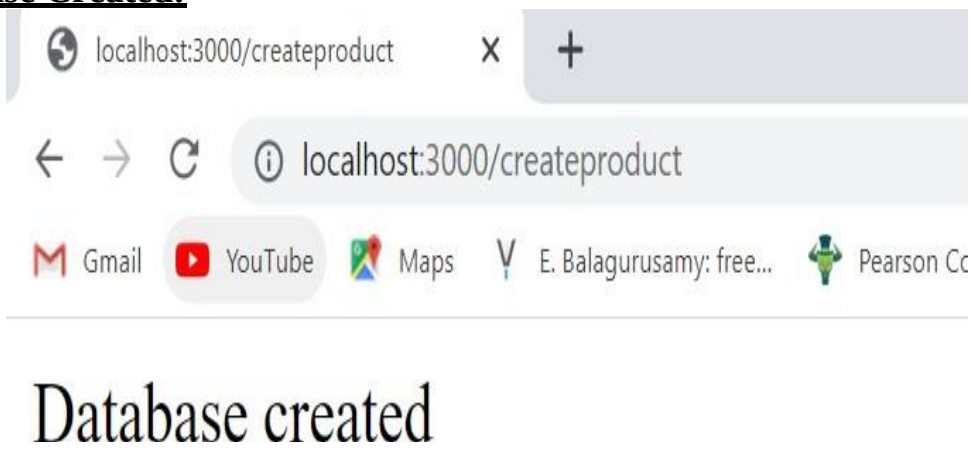
```
var mysql = require('mysql');
const express=require('express');
var connection=mysql.createConnection({
   host:"localhost",
   user:"root",
   password:"vaishu",
   database:"product" //this line entered after database creation
});
connection.connect(function(err)
{
   if(err)
      throw err;
   console.log("connection successful");

});
const app=express()
//create database product
app.get('/createproduct',(req,res)=>{
   let sql='CREATE DATABASE product'
   connection.query(sql,(err)=>{
      if(err)
      {
         throw err;
      }
      res.send("Database created");
   });
});
//create table chocolates
app.get('/createchocolates',(res,req)=>{
   let sql='CREATE TABLE chocolates(pid int AUTO_INCREMENT,nameVARCHAR(30),price
integer,PRIMARY KEY(pid))'
   connection.query(sql,(err)=>{
      if(err)
      {
         throw err;
      }
      res.send("stationary table created");
   });
});
app.get('/createstat',(res,req)=>{
   let sql='CREATE TABLE stationary(pid int AUTO_INCREMENT,name
VARCHAR(30),price integer,PRIMARY KEY(pid))'
   connection.query(sql,(err)=>{
      if(err)
      {
         throw err;
      }
      res.send("stationary table created");
   });
```

```
});

//insert data into tables
app.get('/chocolate1',(req,res)=>
{
   let post={name:'Nutties',price:100}
   let sql='INSERT INTO chocolates SET ?' let query=connection.query(sql,post,err=>{
      if(err)
      {
         throw err;
      }
      res.send("Chocolate added");
   });
});
app.get('/stationary1',(req,res)=>
{
   let post={name:'Poster Colours',price:500}let sql='INSERT INTO stationary SET ?' let
   query=connection.query(sql,post,err=>{
      if(err)
      {
         throw err;
      }
      res.send("Stationary added");
   });
});

//select chocolates & stationary
app.get('/getchocolates',(req,res)=>{
   let sql='SELECT * FROM chocolates'
   let query = connection.query(sql,(err,results)=>{if(err)
      {
         throw err;
      }
      console.log(results);
      res.send("Chocolates fetched");
   });
});

app.get('/getstat',(req,res)=>{
   let sql='SELECT * FROM stationary'
   let query = connection.query(sql,(err,results)=>{
      if(err)
      {
         throw err;
      }
      console.log(results);
      res.send(" Stationary fetched");
   });
});

// update chocolates
```

```
app.get('/updatechoco/:pid',(req,res)=>{
   let newName='Diary Milk'
   let sql=`UPDATE chocolates SET name='${newName}' WHERE pid= ${req.params.pid}
`;
   let query = connection.query(sql,(err,results)=>{
      if(err)
      {
         throw err;
      }

      res.send(" Chocolate updated");
   });
});
//deletion of chocolate app.get('/deletechoco/:pid',(req,res)=>{
   let sql=`DELETE FROM chocolates WHERE pid= ${req.params.pid}`;let query =
   connection.query(sql,(err,results)=>{
      if(err)
      {
         throw err;
      }

      res.send(" Chocolate deleted");
   });
});


app.listen('3000',()=>{
   console.log("server started at port 3000");
})
```

OUTPUT

Database Created:



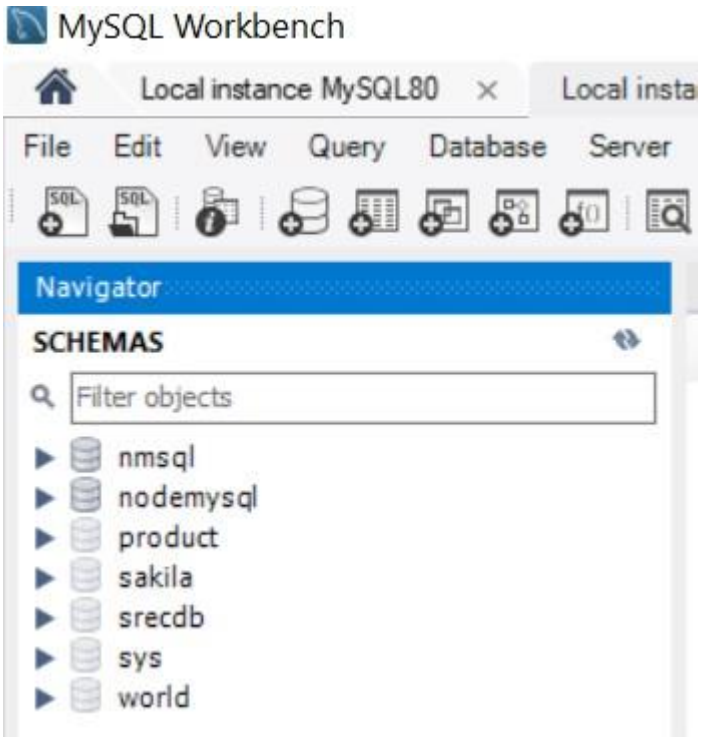
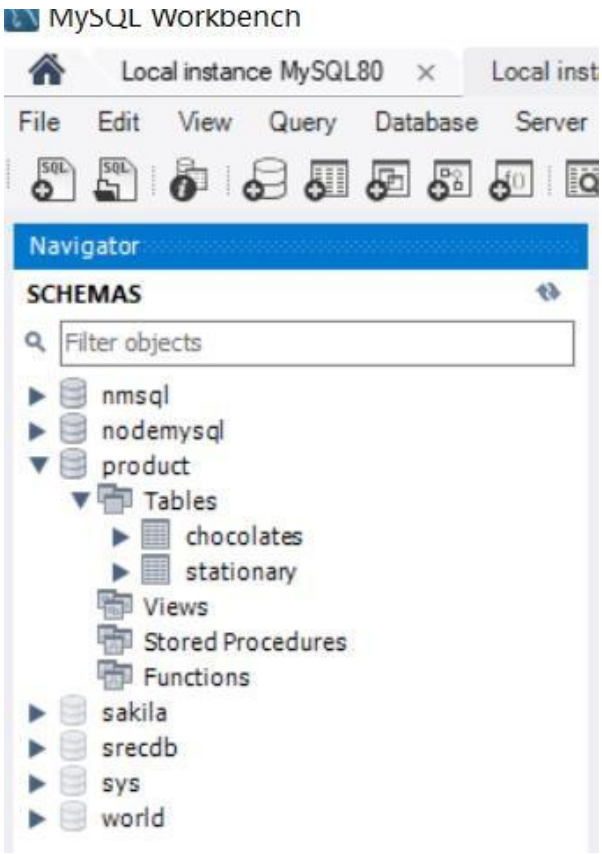
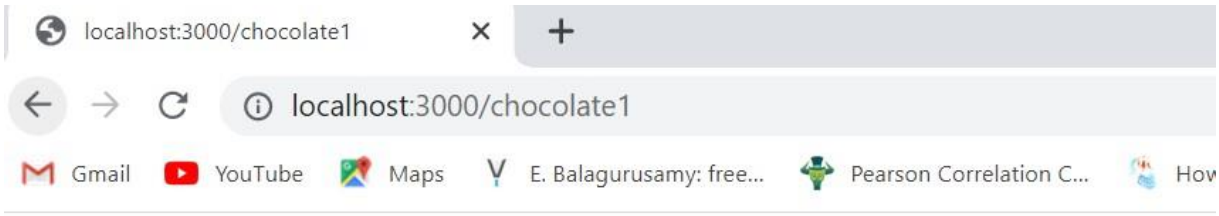


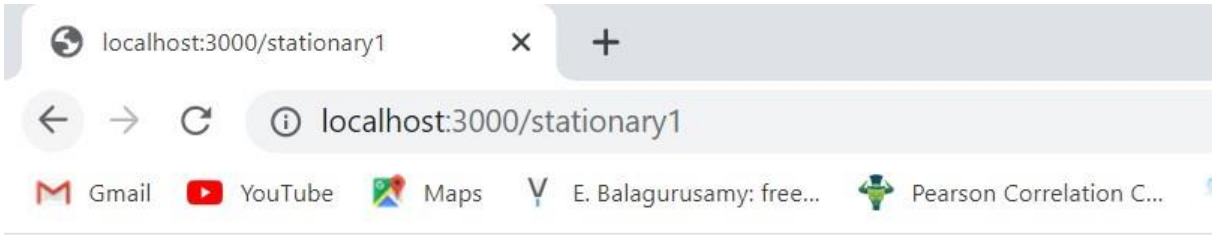
Table Created:



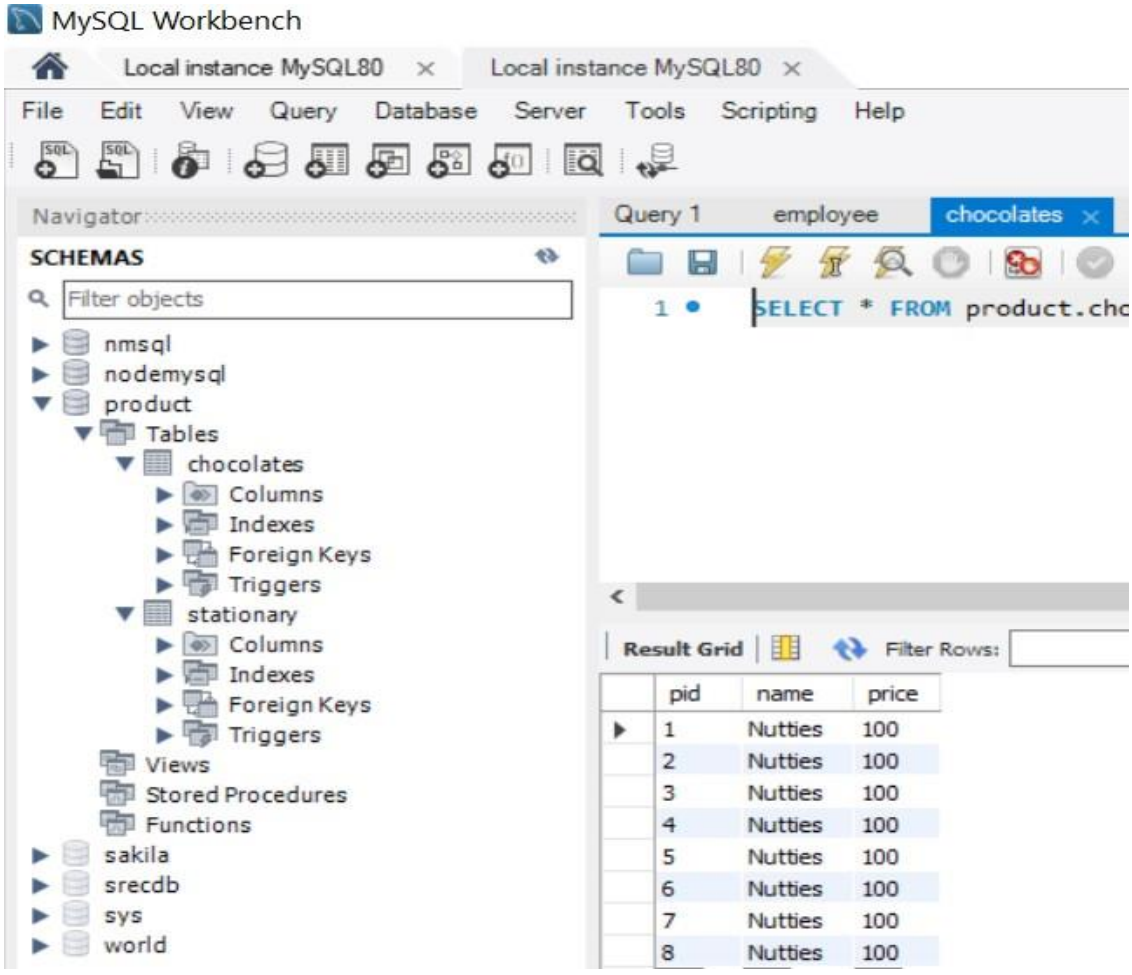
Insert Data into table chocolate & stationary:

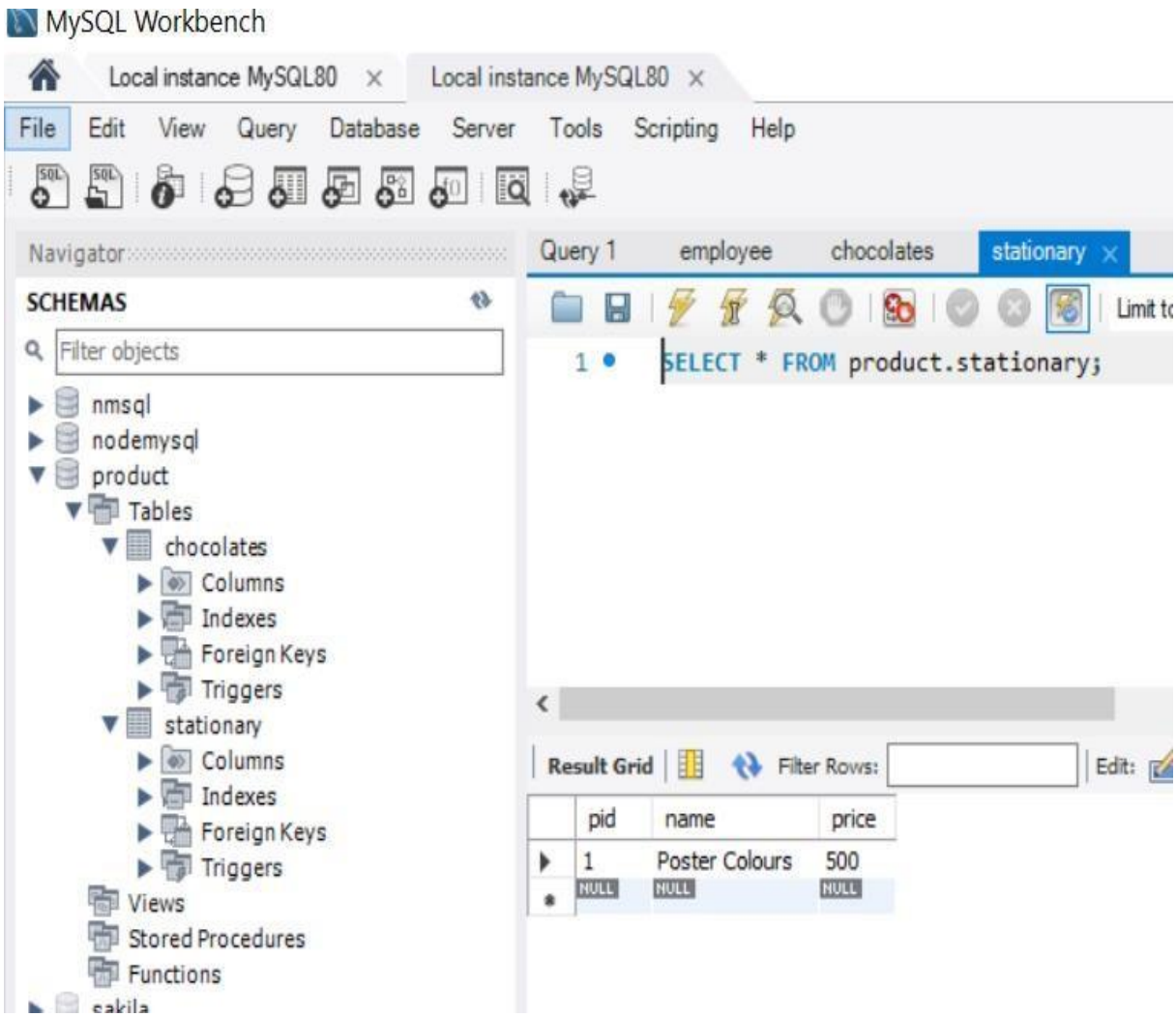


Chocolate added

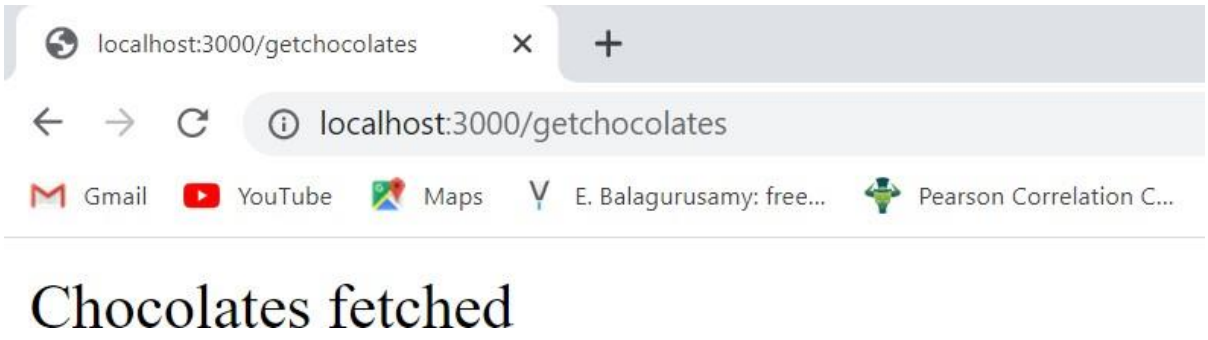


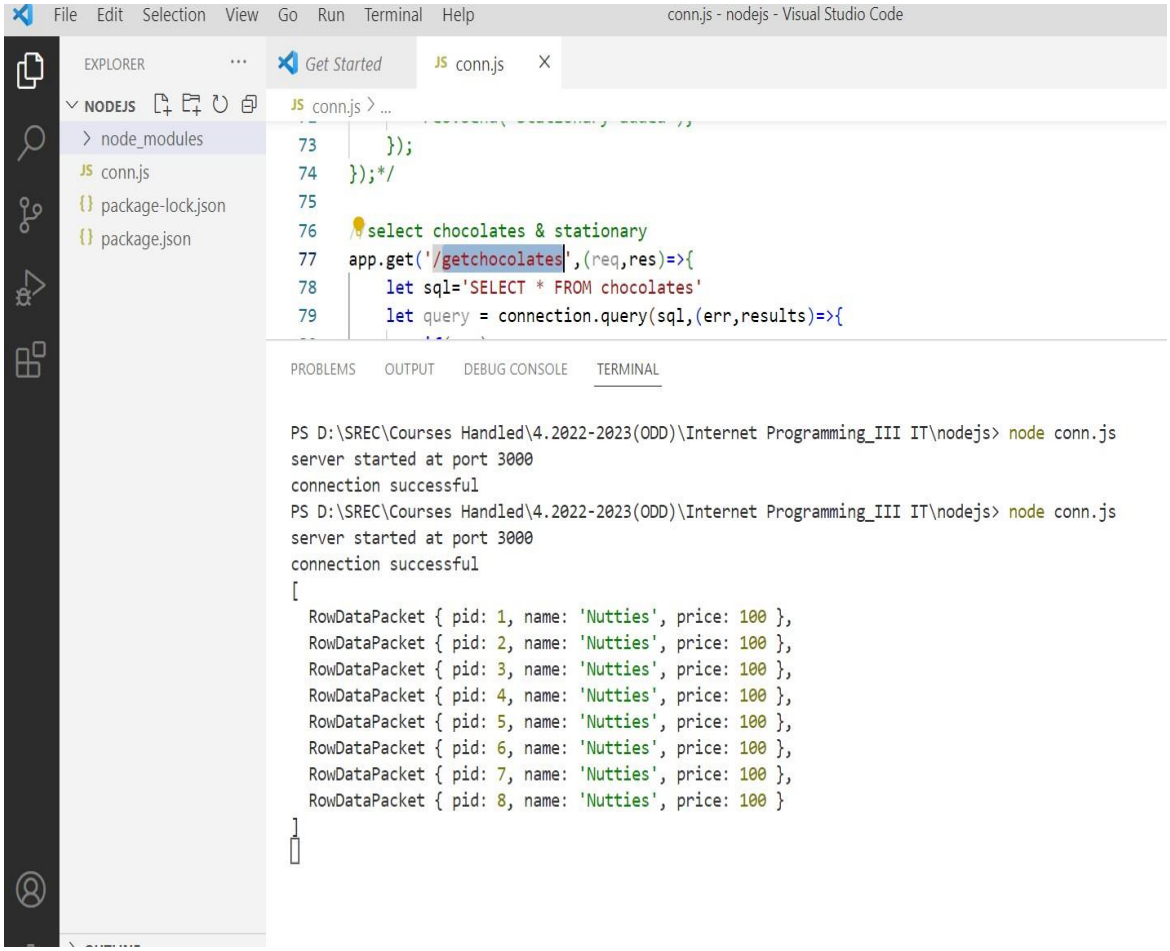
Stationary added

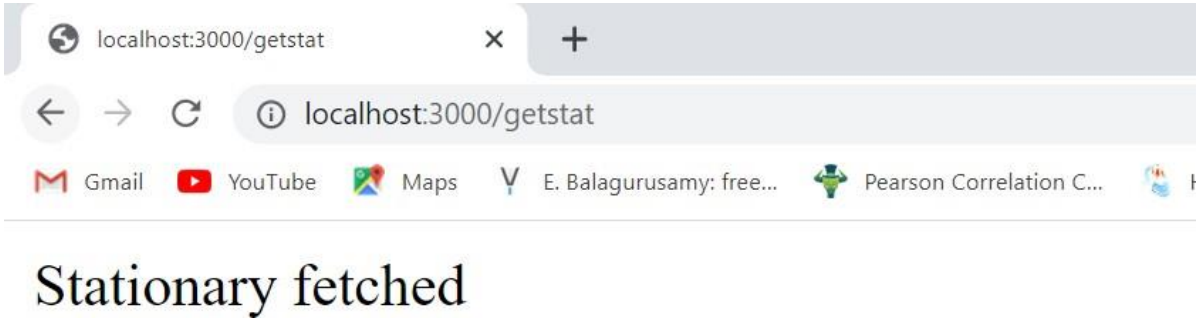


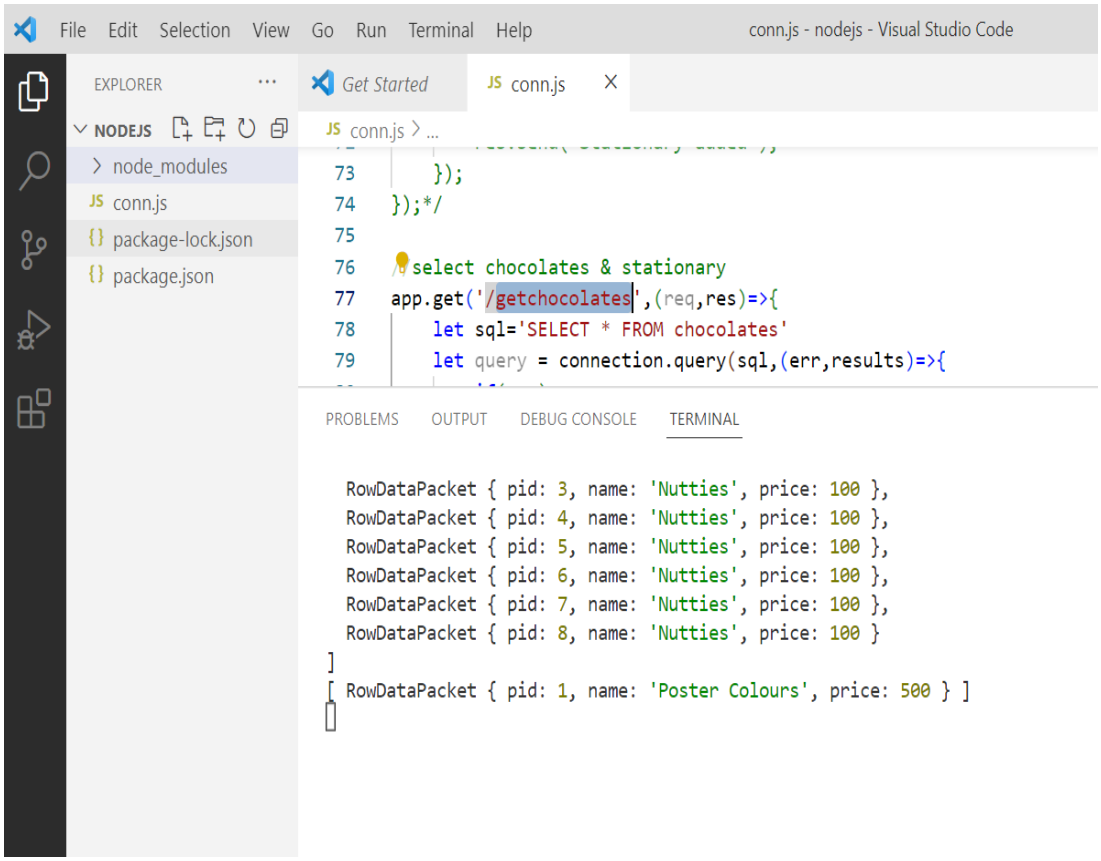


Selecting data from table

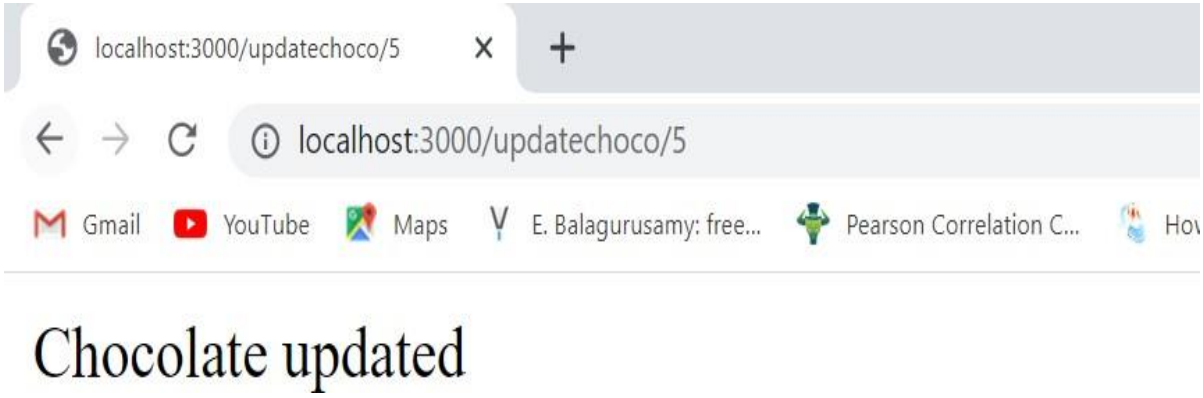


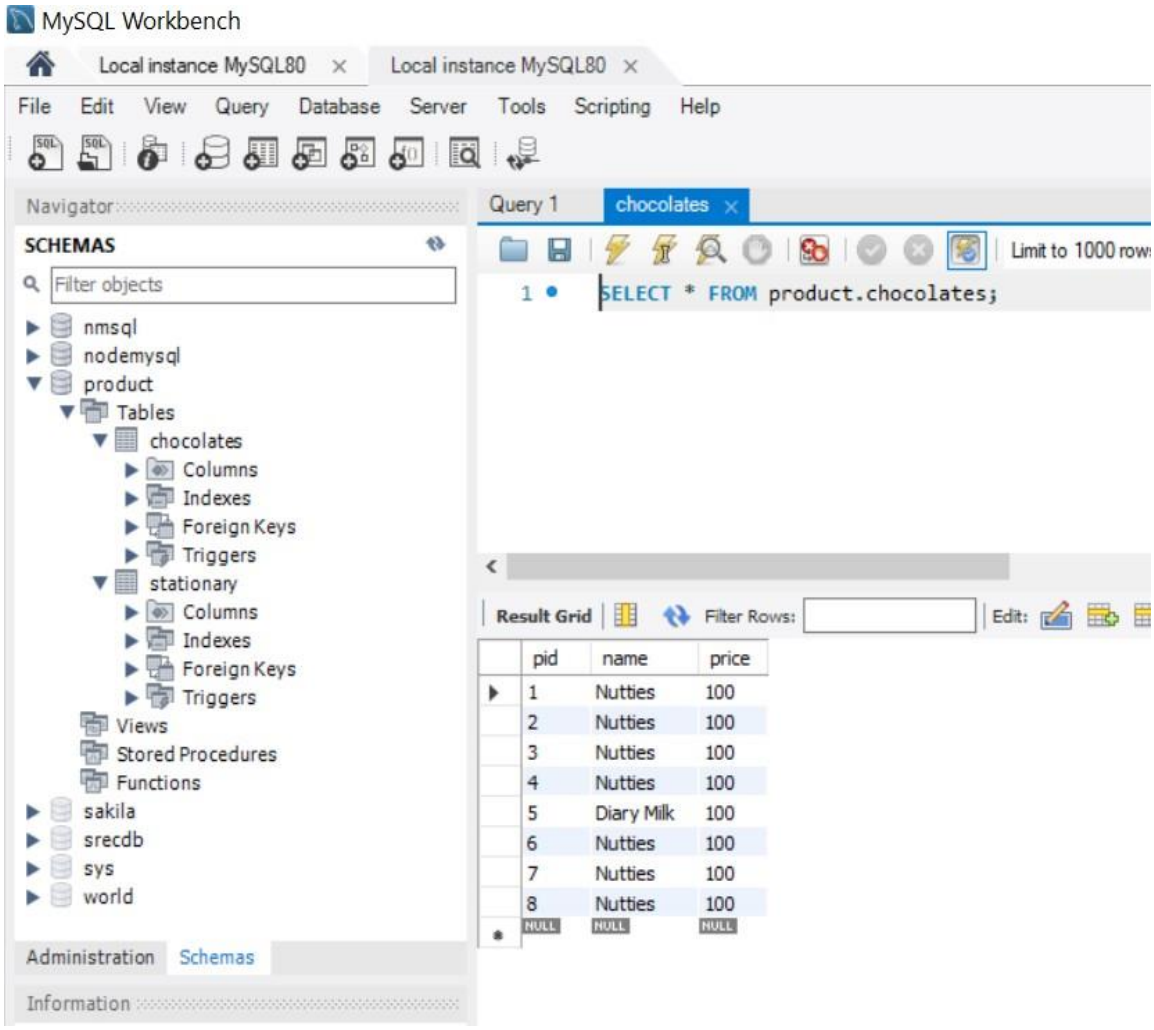

File Edit Selection View Go Run Terminal Help
conn.js - nodejs - Visual Studio Code
EXPLORER
NODEJS
node_modules
conn.js
package-lock.json
package.json
Get Started
conn.js
conn.js > ...
73 });
74 });*/
75
76 //select chocolates & stationary
77 app.get('/getchocolates',(req,res)=>{
78 let sql='SELECT * FROM chocolates'
79 let query = connection.query(sql,(err,results)=>{
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS D:\SREC\Courses Handled\4.2022-2023(ODD)\Internet Programming_III IT\nodejs> node conn.js
server started at port 3000
connection successful
PS D:\SREC\Courses Handled\4.2022-2023(ODD)\Internet Programming_III IT\nodejs> node conn.js
server started at port 3000
connection successful
[
RowDataPacket { pid: 1, name: 'Nutties', price: 100 },
RowDataPacket { pid: 2, name: 'Nutties', price: 100 },
RowDataPacket { pid: 3, name: 'Nutties', price: 100 },
RowDataPacket { pid: 4, name: 'Nutties', price: 100 },
RowDataPacket { pid: 5, name: 'Nutties', price: 100 },
RowDataPacket { pid: 6, name: 'Nutties', price: 100 },
RowDataPacket { pid: 7, name: 'Nutties', price: 100 },
RowDataPacket { pid: 8, name: 'Nutties', price: 100 }
]


localhost:3000/getstat
localhost:3000/getstat
Gmail
YouTube
Maps
E. Balagurusamy: free...
Pearson Correlation C...
Stationary fetched

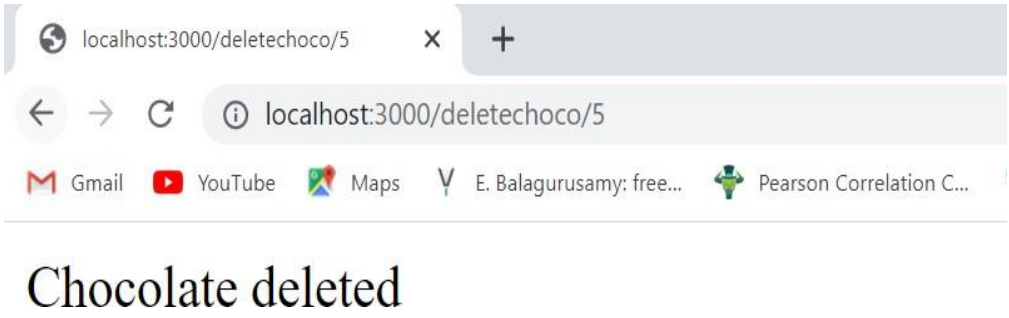


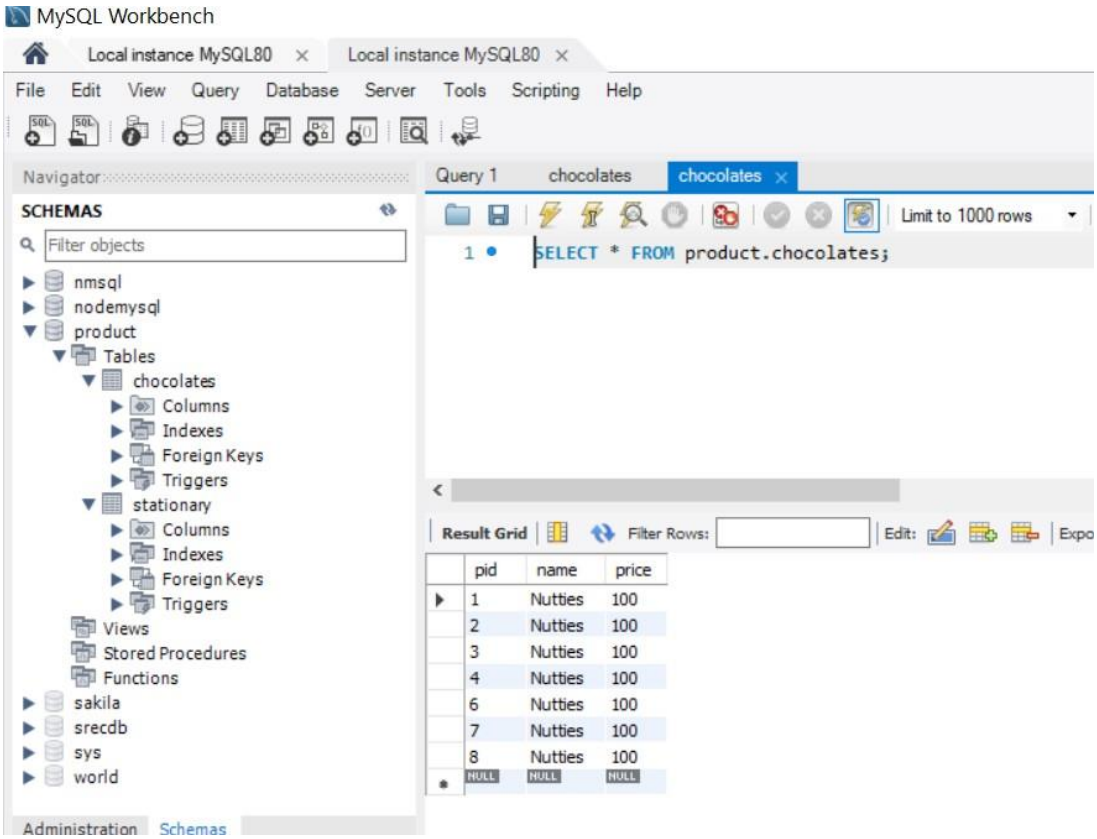
Updating Data in the tables:





Deletion of data:





RESULT

Thus, the node js application for product management has been implemented with database connectivity and CRUD operations performed successfully.

CONTENT BEYOND SYLLABUS

Ex No : 11	USER PERSONAL DATA VALIDATION IN A FORM USING PHP

AIM:

To write PHP programs to Validate the personal information form.

ALGORITHM:

STEP 1: use the PHP tags to denote start and end of the programs.

STEP 2: Design a form using HTML form controls like label, text, button and textarea

STEP 3: pass all variables through PHP's htmlspecialchars() function.

STEP 4: The user has to submits the form by strip unnecessary characters and remove backslashes

STEP 5: Call the test input function

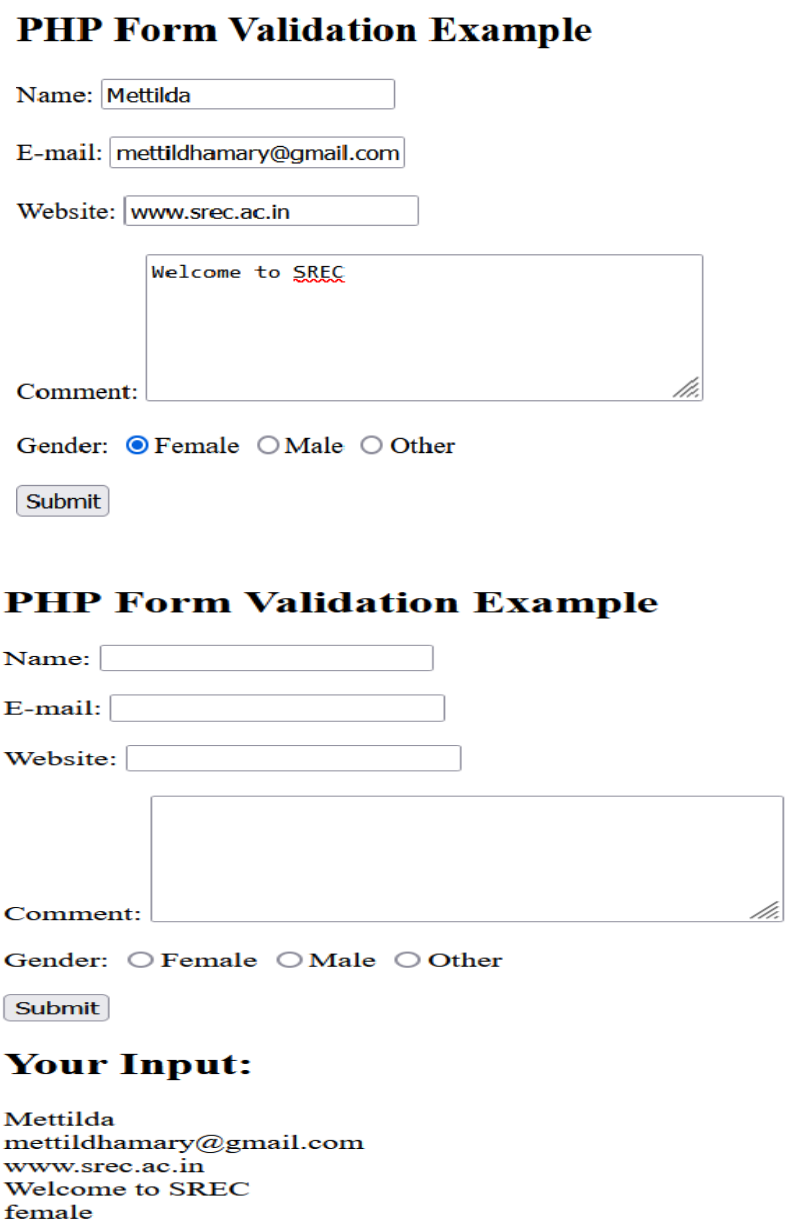
STEP 6: Print the user entered details while press the submit button.

SOURCE CODE:

```
<!DOCTYPE HTML>
<html>
<head>
</head>
<body>

<?php
// define variables and set to empty values
$name = $email = $gender = $comment = $website = "";

if ($_SERVER["REQUEST_METHOD"] == "POST") {
  $name = test_input($_POST["name"]);
  $email = test_input($_POST["email"]);
  $website = test_input($_POST["website"]);
  $comment = test_input($_POST["comment"]);
  $gender = test_input($_POST["gender"]);
}

function test_input($data) {
  $data = trim($data);
  $data = stripslashes($data);
  $data = htmlspecialchars($data);
  return $data;
```

```
}
?>

<h2>PHP Form Validation Example</h2>
<form method="post" action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]);?>">
  Name: <input type="text" name="name">
  <br><br>
  E-mail: <input type="text" name="email">
  <br><br>
  Website: <input type="text" name="website">
  <br><br>
  Comment: <textarea name="comment" rows="5" cols="40"></textarea>
  <br><br>
  Gender:
  <input type="radio" name="gender" value="female">Female
  <input type="radio" name="gender" value="male">Male
  <input type="radio" name="gender" value="other">Other
  <br><br>
  <input type="submit" name="submit" value="Submit">
</form>

<?php
echo "<h2>Your Input:</h2>";
echo $name;
echo "<br>";
echo $email;
echo "<br>";
echo $website;
echo "<br>";
echo $comment;
echo "<br>";
echo $gender;
?>

</body>
</html>
```

OUTPUT:



RESULT:

Thus, the simple PHP programs to sort an array have been executed and output is verified successfully.