

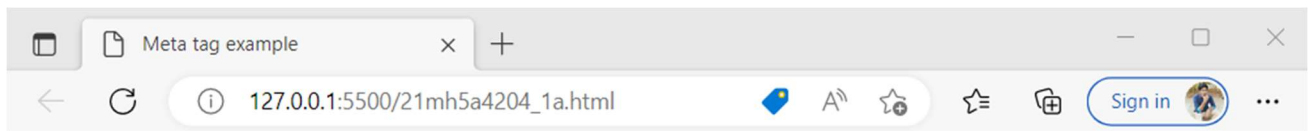
Experiment-1a:**Aim:**

Include the Metadata element in Homepage.html for providing description as "IEKart's is an online shopping website that sells goods in retail. This company deals with various categories like Electronics, Clothing, Accessories etc.

Description:**Source Code:**

```
<!DOCTYPE html>
<html lang="en-US">
  <head>
    <title>Meta tag example</title>
    <!-- Specifies the character encoding for the document. -->
    <!-- ISO-8859-1 -->
    <meta charset="utf-8">
    <meta name="author" content="Susan">
    <meta name="description" content="This is a sample demo to understand HTML.">
    <!-- A web app wants to allow content from a current domain and trust domain xyz
    and all its subdomains -->
    <meta http-equiv="content-security-policy" content="default-src 'self' http://xyz.com" >
    <!-- Specify the preferred style sheet to use.-->
    <meta http-equiv="default-style" content="/style.css">
    <!-- page content display to be adjusted to the device width being used to view the content with initial
    zoom level as 100% -->
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
  </head>
  <body>
    <p> Sample demo to explain meta tag with attributes </p>
  </body>
</html>
```

Output:



Sample demo to explain meta tag with attributes

Experiment-1b:

Aim:

Enhance the Homepage.html of IEKart's Shopping Application by adding appropriate sectioning elements.

Description:

Source Code:

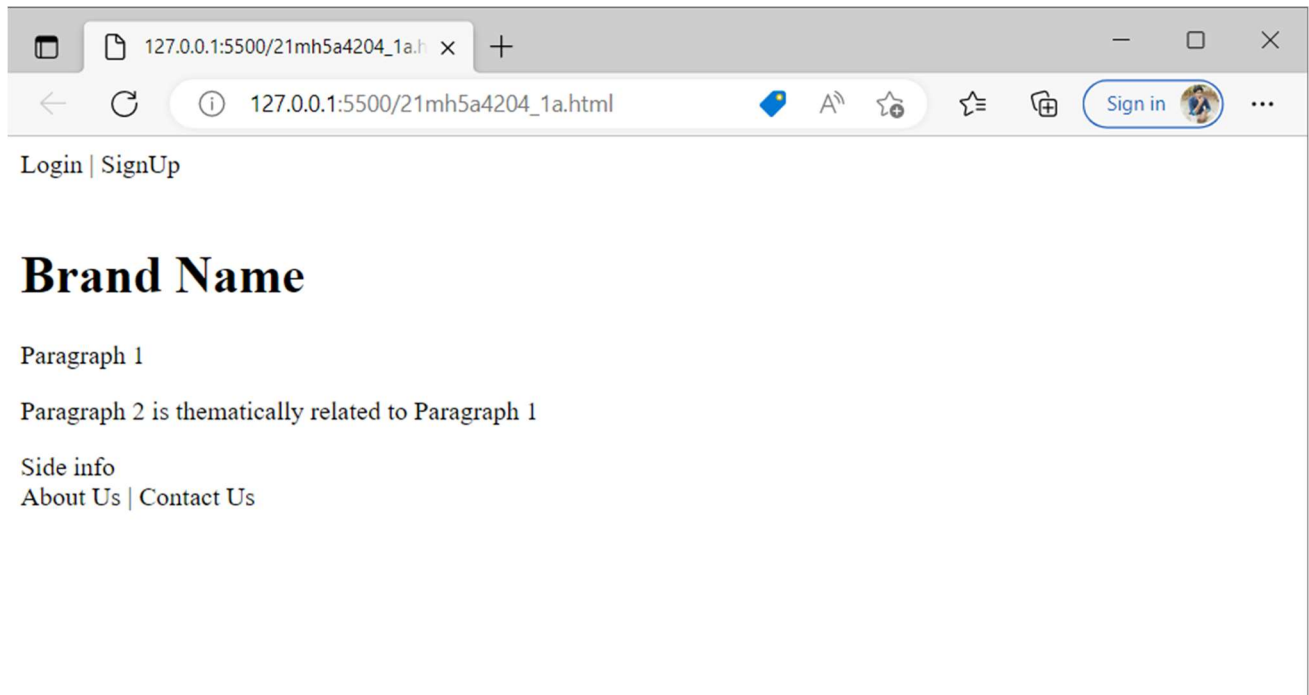
```
<!DOCTYPE html>
<html>
  <body>
    <header>
      <nav> Login | SignUp</nav>
      <h1> Brand Name </h1>
    </header>
    <article>
      <section>
        <p>Paragraph 1</p>
        <p>Paragraph 2 is thematically related to Paragraph 1</p>
      </section>
    </article>
    <aside>
```

```

    Side info
  </aside>
  <footer>
    <nav> About Us | Contact Us </nav>
  </footer>
</body>
</html>

```

Output:



Experinment-1c:

Aim:

Make use of appropriate grouping elements such as list items to "About Us" page of IEKart's Shopping Application

Description:

Source Code:

```

<!DOCTYPE html>
<html>

  <head>

```

```

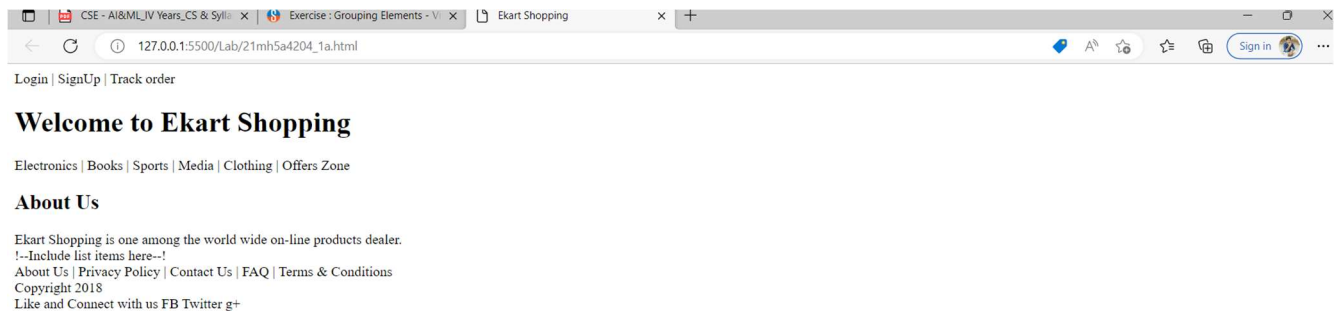
<title>Ekart Shopping</title>
<meta name="keywords" content="Online, Shopping" />
</head>

<body>
  <header>
    <nav> Login | SignUp | Track order </nav>
    <h1> Welcome to Ekart Shopping </h1>
    <nav> Electronics | Books | Sports | Media | Clothing | Offers Zone </nav>
  </header>
  <article>
    <h2>About Us</h2>Ekart Shopping is one among the world wide on-line products dealer.
    <section>!--Include list items here--!</section>
  </article>
  <footer>
    <nav> About Us | Privacy Policy | Contact Us | FAQ | Terms & Conditions </nav> Copyright 2018
  </footer>
  <aside> Like and Connect with us FB Twitter g+ </aside>
</body>

</html>

```

Output:



Experiment-1d:

Aim:

Link "Login", "SignUp" and "Track order" to "Login.html", "SignUp.html" and "Track.html" page respectively. Bookmark each category to its details of IEKart's Shopping application.

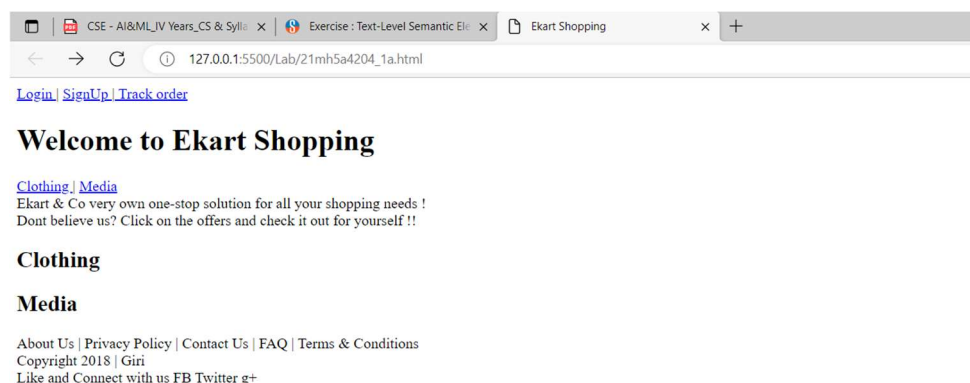
Description:

Source Code:

```

<!DOCTYPE html>
<html>
  <head>
    <meta name="keywords" content="Online, Shopping" />
    <title>Ekart Shopping</title>
  </head>
  <body>
    <header>
      <nav>
        <a href="Login.html"> Login </a> | <a href="Signup.html">SignUp</a> |<a
href="TrackOrder.html"> Track order </a>
      </nav>
      <h1>Welcome to Ekart Shopping</h1>
      <nav>
        <a href="#Clothing"> Clothing </a> | <a href="#Media"> Media </a>
      </nav>
    </header>
    <article>Ekart& Co very own one-stop solution for all your shopping needs!
    <br/>
    Dont believe us? Click on the offers
    and check it out for yourself !!
    <h2 id="Clothing"> Clothing </h2>
    <h2 id="Media"> Media </h2>
  </article>
  <footer>
    <nav> About Us | Privacy Policy | Contact Us | FAQ | Terms & Conditions </nav> Copyright
    <!-- Add character entity for Copyright symbol -->2018 | Giri
  </footer>
  <aside> Like and Connect with us FB Twitter g+ </aside>
</body>
</html>

```

Output:**Experinment-1e:****Aim:**

Add the © symbol in the Home page footer of IEKart's Shopping application.

Description:

Source Code:

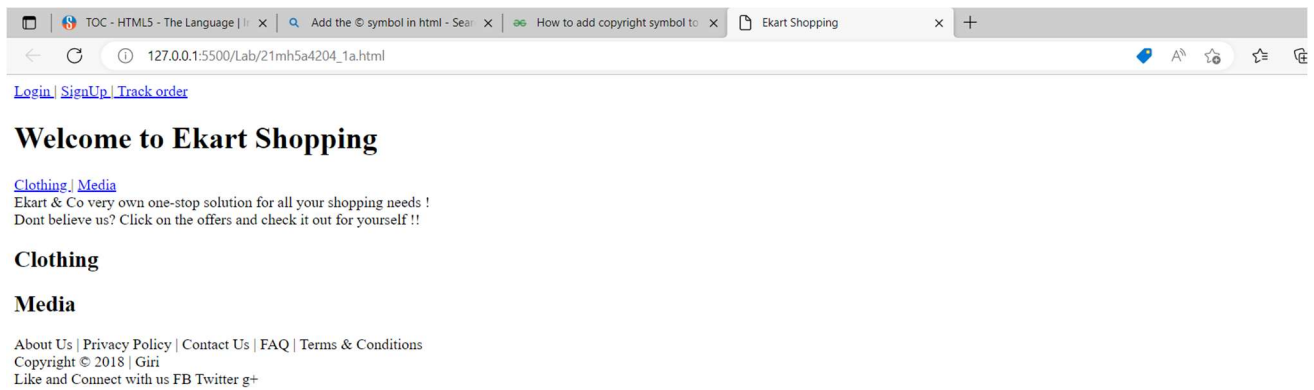
```
<!DOCTYPE html>
<html>

  <head>
    <meta name="keywords" content="Online, Shopping" />
    <title>Ekart Shopping</title>
  </head>

  <body>
    <header>
      <nav>
        <a href="Login.html"> Login </a> | <a href="Signup.html">SignUp</a> |<a
href="TrackOrder.html"> Track order </a>
      </nav>
      <h1>Welcome to Ekart Shopping</h1>
      <nav>
        <a href="#Clothing"> Clothing </a> | <a href="#Media"> Media </a>
      </nav>
    </header>
    <article>Ekart& Co very own one-stop solution for all your shopping needs !
      <br />
      Dont believe us? Click on the offers
      and check it out for yourself !!
      <h2 id="Clothing"> Clothing </h2>
      <h2 id="Media"> Media </h2>
    </article>
    <footer>
      <nav> About Us | Privacy Policy | Contact Us | FAQ | Terms & Conditions </nav> Copyright ©
      <!-- Add character entity for Copyright symbol -->2018 | Giri
    </footer>
    <aside> Like and Connect with us FB Twitter g+ </aside>
  </body>

</html>
```

Output:



Experiment-1f:

Aim:

Add the global attributes such as contenteditable, spellcheck, id etc. to enhance the Signup Page functionality of IEKart's Shopping application.

Description:

Source Code:

```
<!DOCTYPE html>
<html>

  <head>
    <meta name="keywords" content="Online, Shopping" />
    <title>Ekart Shopping</title>
  </head>

  <body>
    <header>
      <h1>Sign Up!</h1>
    </header>
    <article>
      <form action="success.html" method="get">
        <table>
          <tr>
            <td><br><label for="username">Username:</label></td>
```

```

        <td><br><input type="text" id="username" placeholder="Enter your username" required
/></td>
    </tr>
    <tr>
        <td><label for="email_id">Email ID:</label></td>
        <td><input type="email" id="email_id" placeholder="Enter your email ID" required /></td>
    </tr>
    <tr>
        <td><label for="password">Password:</label></td>
        <td><input type="password" id="password" placeholder="Enter your password" required
/></td>
    </tr>
    <tr>
        <td><label for="gender">Gender:</label></td>
        <td><label for="male">Male<input type="radio" id="male" name="gender" value="M"
/>&nbsp; <label for="female">Female<input
        type="radio" id="female" name="gender" value="F" /></td>
    </tr>
    <tr>
        <td><label for="dob">DOB:</label></td>
        <td><input type="date" id="dob" required /></td>
    </tr>
    <tr>
        <td><label for="phone_no">Phone number:</label></td>
        <td><input type="text" id="phone_no" placeholder="Enter your contact number" pattern="+
[0-9] {12}" required /></td>
    </tr>
    <tr>
        <td><label for="country">Country:</label></td>
        <td><select id="country" placeholder="Select your country">
            <option value="India" />India
            <option value="India" />USA
            <option value="India" />UK
            <option value="India" />Canada
            <option value="India" />Belgium
            <option value="India" />France </select></td>
    </tr>
    <tr>
        <td><label id="language">Languages Known:</label></td>
        <td><input type="checkbox" name="language" id="english" value="English"
checked="checked" /><label for="english">
            English </label><input type="checkbox" name="language" id="hindi" value="Hindi"
/><label for="hindi"> Hindi
            </label><input type="checkbox" name="language" id="french" value="French" /><label
for="french"> French </label></td>
    </tr>
    <tr>
        <td><label for="pic">Profile pic:</label></td>
        <td><input type="file" id="pic" required /></td>
    </tr>
    <tr>
        <td><label for="yourself" dir="ltr">About yourself:</label></td>

```



```

        <td><textarea></textarea></td><!-- Add contenteditable and spellcheck attribute -->
    </tr>
    <tr>
        <td><input type="submit" value="Register" />
        <td><input type="reset" value="Reset" />
    </tr>
</table>
</form><br /><br /><br /><br />
</article>
<footer>
    <nav> About Us | Privacy Policy | Contact Us | FAQ | Terms & Conditions </nav> Copyright &copy;
2018 | Giri
</footer>
<aside> Like and Connect with us FB Twitter g+ </aside>
</body>

</html>

```

Output:

Sign Up!

Username:

Email ID:

Password:

Gender: ☒ Male ☐ Female

DOB:

Phone number:

Country:

Languages Known: ☒ English ☐ Hindi ☐ French

Profile pic: No file chosen

About yourself:

About Us | Privacy Policy | Contact Us | FAQ | Terms & Conditions
 Copyright © 2018 | Giri
 Like and Connect with us FB Twitter g+

Experiment-2a:**Aim:**

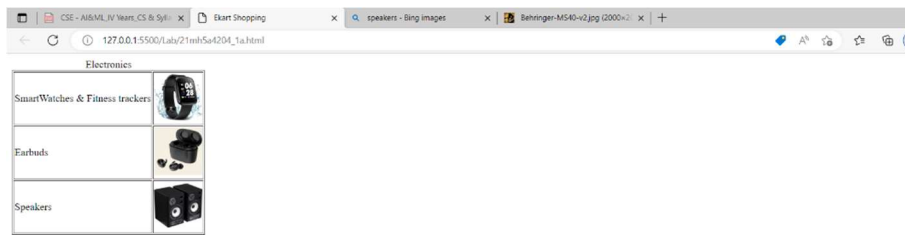
Enhance the details page of IEKart's Shopping application by adding a table element to display the available mobile/any inventories.

Description:**Source Code:**

```
<!DOCTYPE html>
<html>
  <head>
    <meta name="keywords" content="Online, Shopping" />
    <title>Ekart Shopping</title>
  </head>

  <body>
    <div id="tab">
      <table border="1" id="tables">
        <caption>Electronics</caption>
        <tr>
          <td>SmartWatches& Fitness trackers</td>
          <td></td>
        </tr>
        <tr>
          <td>Earbuds</td>
          <td></td>
        </tr>
        <tr>
          <td>Speakers</td>
          <td></td>
        </tr>
      </table>
    </div>
  </body>
</html>
```

Output:



Experiment-2b:

Aim:

Using the form elements create Signup page for IEKart's Shopping application.

Description:

Source Code:

```
<!DOCTYPE html>
<html>
  <head>
    <meta name="keywords" content="Online, Shopping" />
    <title>Ekart Shopping</title>
  </head>

  <body>
    <header>
      <h1>Sign Up!</h1>
    </header>
    <article>
      <form action="success.html" method="get">
        <table>
          <tr>
            <td><br><label for="username">Username:</label></td>
            <td><br><input type="text" id="username" placeholder="Enter your username" required
          </td>
        </tr>
        <tr>
            <td><label for="email_id">Email ID:</label></td>
            <td><input type="email" id="email_id" placeholder="Enter your email ID" required /></td>
        </tr>
        <tr>
            <td><label for="password">Password:</label></td>
```

```

        <td><input type="password" id="password" placeholder="Enter your password" required
/></td>
    </tr>
    <tr>
        <td><label for="gender">Gender:</label></td>
        <td><label for="male">Male<input type="radio" id="male" name="gender" value="M"
/>&nbsp;<label for="female">Female<input
            type="radio" id="female" name="gender" value="F" /></td>
    </tr>
    <tr>
        <td><label for="dob">DOB:</label></td>
        <td><input type="date" id="dob" required /></td>
    </tr>
    <tr>
        <td><label for="phone_no">Phone number:</label></td>
        <td><input type="text" id="phone_no" placeholder="Enter your contact number" pattern="+
[0-9] {12}" required /></td>
    </tr>
    <tr>
        <td><label for="country">Country:</label></td>
        <td><select id="country" placeholder="Select your country">
            <option value="India" />India
            <option value="India" />USA
            <option value="India" />UK
            <option value="India" />Canada
            <option value="India" />Belgium
            <option value="India" />France </select></td>
    </tr>
    <tr>
        <td><label id="language">Languages Known:</label></td>
        <td><input type="checkbox" name="language" id="english" value="English"
checked="checked" /><label for="english">
            English </label><input type="checkbox" name="language" id="hindi" value="Hindi"
/><label for="hindi"> Hindi
            </label><input type="checkbox" name="language" id="french" value="French" /><label
for="french"> French </label></td>
    </tr>
    <tr>
        <td><label for="pic">Profile pic:</label></td>
        <td><input type="file" id="pic" required /></td>
    </tr>
    <tr>
        <td><input type="submit" value="Register" />
        <td><input type="reset" value="Reset" />
    </tr>
</table>
</form><br /><br /><br /><br />
</article>
<footer>
    <nav> About Us | Privacy Policy | Contact Us | FAQ | Terms & Conditions </nav> Copyright &copy;
2018 | Giri
</footer>

```

```
<aside> Like and Connect with us FB Twitter g+ </aside>
</body>
```

```
</html>
```

Output:

Sign Up!

Username:

Email ID:

Password:

Gender: ☐ Male ☐ Female

DOB:

Phone number:

Country:

Languages Known: ☒ English ☐ Hindi ☐ French

Profile pic: No file chosen

About Us | Privacy Policy | Contact Us | FAQ | Terms & Conditions
 Copyright © 2018 | Giri
 Like and Connect with us FB Twitter g+

Experiment-2c:

Aim:

Enhance Signup page functionality of IEKart's Shopping application by adding attributes to input elements.

Description:

Source Code:

```
<!DOCTYPE html>
<html>
  <head>
    <meta name="keywords" content="Online, Shopping" />
    <title>Ekart Shopping</title>
  </head>
  <body>
    <header>
      <h1>Sign Up!</h1>
    </header>
    <article>
      <form>
        <table>
          <tr>
```

```

        <td><br><label>Username:</label></td>
        <td><br><input type="text" id="username" placeholder="Enter your username" required
/></td>
    </tr>
    <tr>
        <td><label>Email ID:</label></td>
        <td><input type="email" id="email_id" placeholder="Enter your email ID" required /></td>
    </tr>
    <tr>
        <td><label>Password:</label></td>
        <td><input type="password" id="password" placeholder="Enter your password" required
/></td>
    </tr>
    <tr>
        <td><label>Gender:</label></td>
        <td><label for="male">Male<input type="radio" id="male" name="gender" value="M"
/>&nbsp; <label for="female">Female<input
            type="radio" id="female" name="gender" value="F" /></td>
    </tr>
    <tr>
        <td><label>DOB:</label></td>
        <td><input type="date" id="dob" required /></td>
    </tr>
    <tr>
        <td><label>Phone number:</label></td>
        <td><input type="text" id="phone_no" placeholder="Enter your contact number" pattern="[0-
9]+"maxlength="10 required /></td>
    </tr>
    <tr>
        <td><label>Country:</label></td>
        <td><select id="country" placeholder="Select your country">
            <option value="India" />India
            <option value="India" />USA
            <option value="India" />UK
            <option value="India" />Canada
            <option value="India" />Belgium
            <option value="India" />France</select></td>
    </tr>
    <tr>
        <td><label id="language">Languages Known:</label></td>
        <td><input type="checkbox" name="language" id="english" value="English"
checked="checked" /><label for="english">
            English </label><input type="checkbox" name="language" id="hindi" value="Hindi"
/><label for="hindi"> Hindi
            </label><input type="checkbox" name="language" id="french" value="French" /><label
for="french"> French </label></td>
    </tr>
    <tr>
        <td><label for="pic">Profile pic:</label></td>
        <td><input type="file" id="pic" required /></td>
    </tr>

```

```

        <tr>
            <td><input type="submit" value="Register" />
            <td><input type="reset" value="Reset" />
        </tr>
    </table>
</form><br /><br /><br /><br />
</article>
<footer>
    <nav> About Us | Privacy Policy | Contact Us | FAQ | Terms & Conditions </nav> Copyright &copy;
2018 | Giri
</footer>
<aside> Like and Connect with us FB Twitter g+ </aside>
</body>

</html>

```

Output:

Sign Up!

Username:

Email ID:

Password:

Gender: ☒ Male ☐ Female

DOB:

Phone number:

Country:

Languages Known: ☒ English ☐ Hindi ☐ French

Profile pic: calc.js

About Us | Privacy Policy | Contact Us | FAQ | Terms & Conditions
 Copyright © 2018 | Giri
 Like and Connect with us FB Twitter g+

Experiment-2d:

Aim:

Add media content in a frame using audio, video, iframe elements to the Home page of IEKart's Shopping application.

Description:

Source Code:

```

<!DOCTYPE html>
<html>
    <head>
        <meta name="keywords" content="Online, Shopping" />
    </head>

```

```

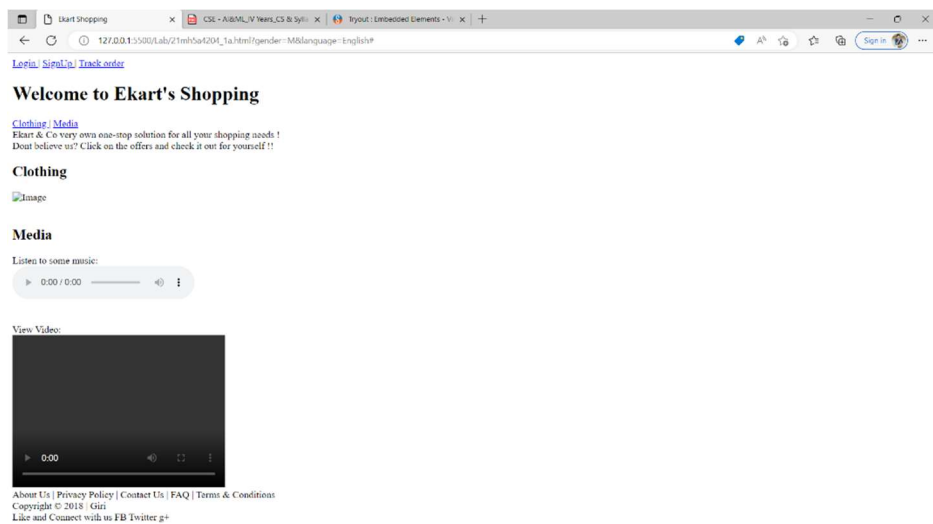
<title>Ekart Shopping</title>
</head>
<body>
  <header>
    <nav>
      <a href="Login.html"> Login </a> |
      <a href="Signup.html">SignUp</a> |
      <a href="TrackOrder.html"> Track order </a>
    </nav>
    <h1>Welcome to Ekart's Shopping</h1>
    <nav>
      <a href="#Clothing"> Clothing </a> |
      <a href="#Media"> Media </a>
    </nav>
  </header>
  <article>
    Ekart& Co very own one-stop solution for all your shopping needs !<br/>
    Dont believe us? Click on the offers and check it out for yourself !!

    <h2 id="Clothing"> Clothing </h2>
    <imgsrc="#.jpg" alt="Image" width="350" height="250"><br><br>

    <h2 id="Media"> Media </h2>
    Listen to some music: <br/>
    <audio src="music.mp3" controls="controls"></audio><br><br>
    <br/>
    View Video: <br/>
    <video src="seaView.mp4" controls="controls" width="350" height="250"></video>
  </article>
  <footer>
    <nav>
      About Us | Privacy Policy | Contact Us | FAQ | Terms & Conditions
    </nav>
    Copyright &copy; 2018 | Giri
  </footer>
  <aside>
    Like and Connect with us FB Twitter g+
  </aside>
</body>
</html>

```

Output:



Experiment-3a:**Aim:**

Write a JavaScript program to find the area of a circle using radius (var and let - reassign and observe the difference with var and let) and PI (const)

Description:**Source Code:**

```
<!DOCTYPE html>
<html>
  <head>
    <meta name="keywords" content="Online, Shopping" />
    <title>JavaScript</title>
  </head>
  <body>
    <script>
      let radius=10;
      var rad=10;
      const pi =3.14;
      document.write(pi*(rad**2));
      document.write("<br>");
      var rad=20;
      document.write(pi*(rad**2));
      document.write("<br>");

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

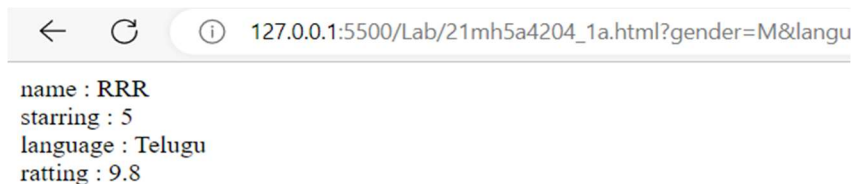
Output:

Experiment-3b:**Aim:**

Write JavaScript code to display the movie details such as movie name, starring, language, and ratings. Initialize the variables with values of appropriate types. Use template literals wherever necessary.

Description:**Source Code:**

```
<!DOCTYPE html>
<html>
  <head>
    <title>JavaScript</title>
  </head>
  <body>
    <script>
      let movie={
        name:"RRR",
        starring:5,
        language:"Telugu",
        rating: 9.8,
      };
      for(key in movie){
        document.write(key + " : ");
        document.write(movie[key]);
        document.write("<br>");
      };
    </script>
  </body>
</html>
```

Output:


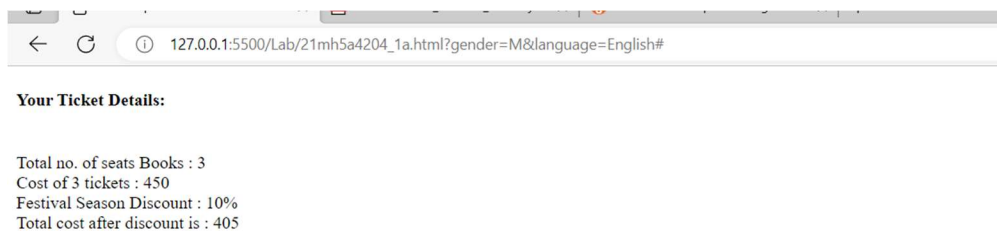
```
name : RRR
starring : 5
language : Telugu
rating : 9.8
```

Experiment-3c:**Aim:**

Write JavaScript code to book movie tickets online and calculate the total price, considering the number of tickets and price per ticket as Rs. 150. Also, apply a festive season discount of 10% and calculate the discounted amount.

Description:**Source Code:**

```
<!DOCTYPE html>
<html>
  <head>
    <title>JavaScript</title>
  </head>
  <body>
    <h4>Your Ticket Details:</h4>
    <br>
    <script>
      let x= Number(prompt("Enter No. of tickets : "));
      document.write("Total no. of seats Books : "+x);
      document.write("<br>");
      document.write("Cost of "+x+" tickets : " +(150*x));
      document.write("<br>");
      document.write("Festival Season Discount : 10%");
      document.write("<br>");
      document.write("Total cost after discount is : "+((x*150)-(x*(10/100)*150)));
    </script>
  </body>
</html>
```

Output:

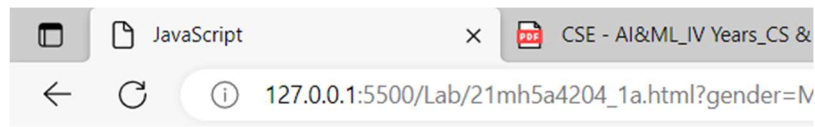
Experiment-3d:**Aim:**

Write a JavaScript code to book movie tickets online and calculate the total price based on the 3 conditions: (a) If seats to be booked are not more than 2, the cost per ticket remains Rs. 150. (b) If seats are 6 or more, booking is not allowed. (c) If se.

Description:**Source Code:**

```
<!DOCTYPE html>
<html>
  <head>
    <title>JavaScript</title>
  </head>
  <body>
    <h4>Your Ticket Details:</h4>
    <br>
    <script>
      let i=0;
      let y=[3,5,7,9,11];
      let x=prompt("No of Tickets?");
      x=Number(x);
      if(x<=2){
        document.getElementById("11").innerHTML="Tickets booked <br>"+ "Cost for "+x+" Tickets is
" + ((x*150));
      }
      else if(x>=6){
        document.getElementById("11").innerHTML="Booking is not allowed";
      }
      else{
        document.write("Tickets booked <br>"+ "Cost for "+x+" Tickets is " + ((x*150)-(x*(y[x-
1]/100)*150)));
      }
    </script>
  </body>
</html>
```

Output:



Your Ticket Details:

Tickets booked
Cost for 3 Tickets is 418.5

Experiment-3e:

Aim:

Write a JavaScript code to do online booking of theatre tickets and calculate the total price based on the below conditions:

1. If seats to be booked are not more than 2, the cost per ticket remains \$9.
2. If seats are 6 or more, booking is not allowed.
3. If seats to be booked are more than 2 but less than 5, based on the number of seats booked, do the following:
 - Calculate total cost by applying a discount of 5, 7, 9, 11 percent, and so on for customer 1,2,3 till 5.
 - Try the code with different values for the number of seats.

Implement the problem statement using 'for' loop, 'while' loop and 'do-while' loop.

Description:

Source Code:

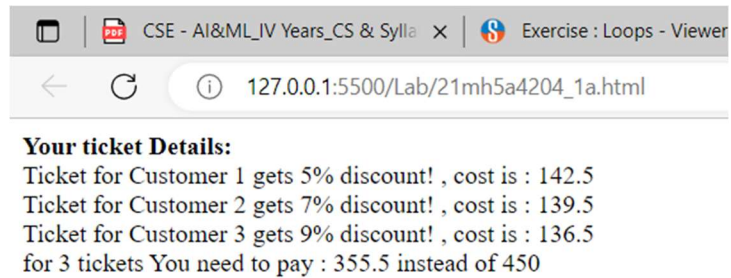
```
<!DOCTYPE html>
<html>
<head>
  <title>Booking Details</title>
</head>
<body>
  <script>
    let x=prompt("No of Tickets?");
    x=Number(x);
    let i=0;
    let y=5;
    let total=0;
    document.write("<b>Your ticket Details:</b><br>")
    for(i=0;i<x;i++){
```

```

        document.write("Ticket for Customer "+(i+1)+" gets "+y+"% discount! , cost is : " +((150)-
((y/100)*150))+"<br>");
        total+=((150)-(x*(y/100)*150))
        y+=2
    }
    document.write("for "+x+" tickets You need to pay : "+total+" instead of "+(x*150)+" <br>")

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

```

Output:

Experiment-4a:**Aim:**

Write a JavaScript code to do online booking of theatre tickets and calculate the total price based on the below conditions:

1. If seats to be booked are not more than 2, the cost per ticket remains 150rs.
2. If seats are 6 or more, booking is not allowed.
3. If seats to be booked are more than 2 but less than 5, based on the number of seats booked, do the following:
 - Calculate total cost by applying a discount of 5, 7, 9, 11 percent, and so on for customer 1,2,3 till 5.
 - Try the code with different values for the number of seats.

Write the following custom functions to implement given requirements:

1. calculateCost(seats): Calculate and display the total cost to be paid by the customer for the tickets they have bought.
2. calculateDiscount(seats): Calculate discount on the tickets bought by the customer. Implement using arrow functions.

Description:**Source Code:**

```
<!DOCTYPE html>
<html>
<head>
  <title>Booking Details</title>
</head>
<body>
  <script>
    let x=prompt("No of Tickets?");
    let total=0;
    x=Number(x);
    document.write("<b>Your ticket Details:</b><br>");
    document.write("Actual cost per ticker is : 150 <br>")
    if(x>2 && x<6){
      document.write("Seats are available <br>");
      let i=0;
      let y=5;
      for(i=0;i<x;i++){
        document.write("Ticket for Customer "+(i+1)+" gets "+y+"% discount! , cost is : " +((150)-
((y/100)*150))+ "<br>");
        total+=((150)-(x*(y/100)*150));
        y+=2;
      };
    }
```

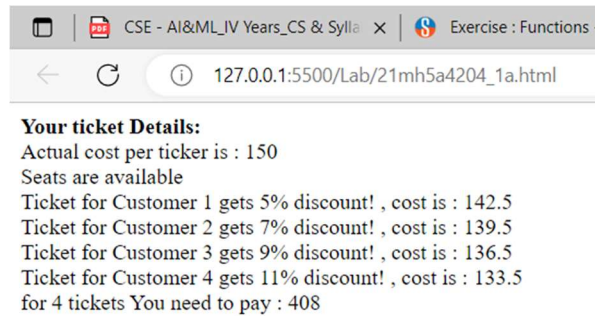


```

    }
    else{
        document.write("Seats are not available <br>");
    };
    document.write("for "+x+" tickets You need to pay : "+total+" <br>");

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

```

Output:**Experiment-4b:****Aim:**

Create an Employee class extending from a base class Person. Hints: (i) Create a class Person with name and age as attributes. (ii) Add a constructor to initialize the values (iii) Create a class Employee extending Person with additional attributes role

Description:**Source Code:**

```

<!DOCTYPE html>
<html>
<head>
    <title>Booking Details</title>
</head>
<body>
    <script>
        class Person{
            constructor(name,age){
                this.name=name;
                this.age=age;
            }
        }
        class Employee extends Person{
            constructor(name,age,role,salary){

```

```

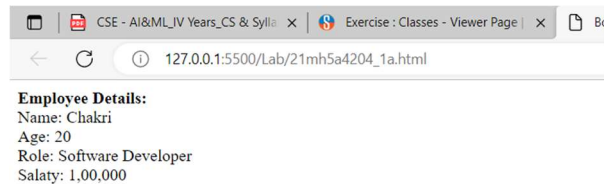
        super(name,age);
        this.role=role;
        this.salary=salary;
    }
    Details(){
        document.write("<b>Employee Details:</b><br>");
        document.write("Name: "+this.name+"<br>");
        document.write("Age: "+this.age+"<br>");
        document.write("Role: "+this.role+"<br>");
        document.write("Salaty: "+this.salary+"<br>");
    }
}
let c=new Employee("Chakri","20","Software Developer","1,00,000");
c.Details();

```

```

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

```

Output:**Experiment-4c:****Aim:**

Write a JavaScript code to make online booking of theatre tickets and calculate the total price based on the below conditions:

1. If seats to be booked are not more than 2, the cost per ticket remains 150.
2. If seats are 6 or more, booking is not allowed.
3. If seats to be booked are more than 2 but less than 5, based on the number of seats booked, do the following:
 - Calculate total cost by applying a discount of 5, 7, 9, 11 percent, and so on for customer 1,2,3 till 5.
 - Try the code with different values for the number of seats.

Description:**Source Code:**

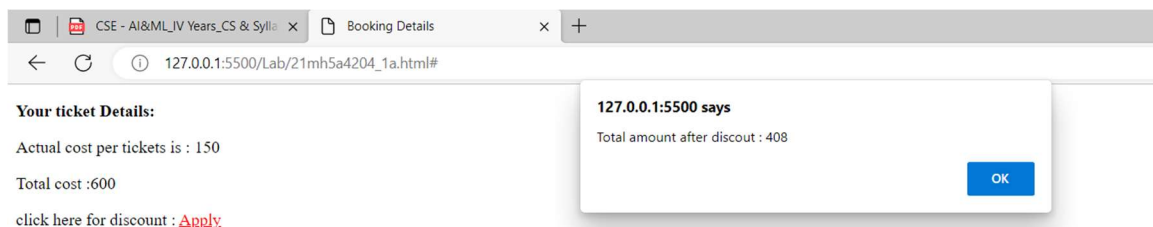
```

<!DOCTYPE html>
<html>
<head>
  <title>Booking Details</title>
</head>
<body>
  <p><b>Your ticket Details:</b></p>
  <p>Actual cost per tickets is : 150 </p>
  <p id="1"></p>
  <div><p id="2">click here for discount : <a onclick="toggle()" href="#">Apply</a></p></div>
  <script>
    let x=prompt("No of Tickets?");
    let total=0;
    x=Number(x);
    let c=x*150;
    c=String(c)
    document.getElementById("1").innerHTML="Total cost :"+c;
    function toggle(){
      if(x>2 && x<6){
        let i=0;
        let y=5;
        for(i=0;i<x;i++){
          total+=((150)-(x*(y/100)*150));
          y+=2;
        };
        alert("Total amount after discout : "+total);
      }
      else{
        alert("Seats are not available");
      };
    }
  </script>

</body>
</html>

```

Output:



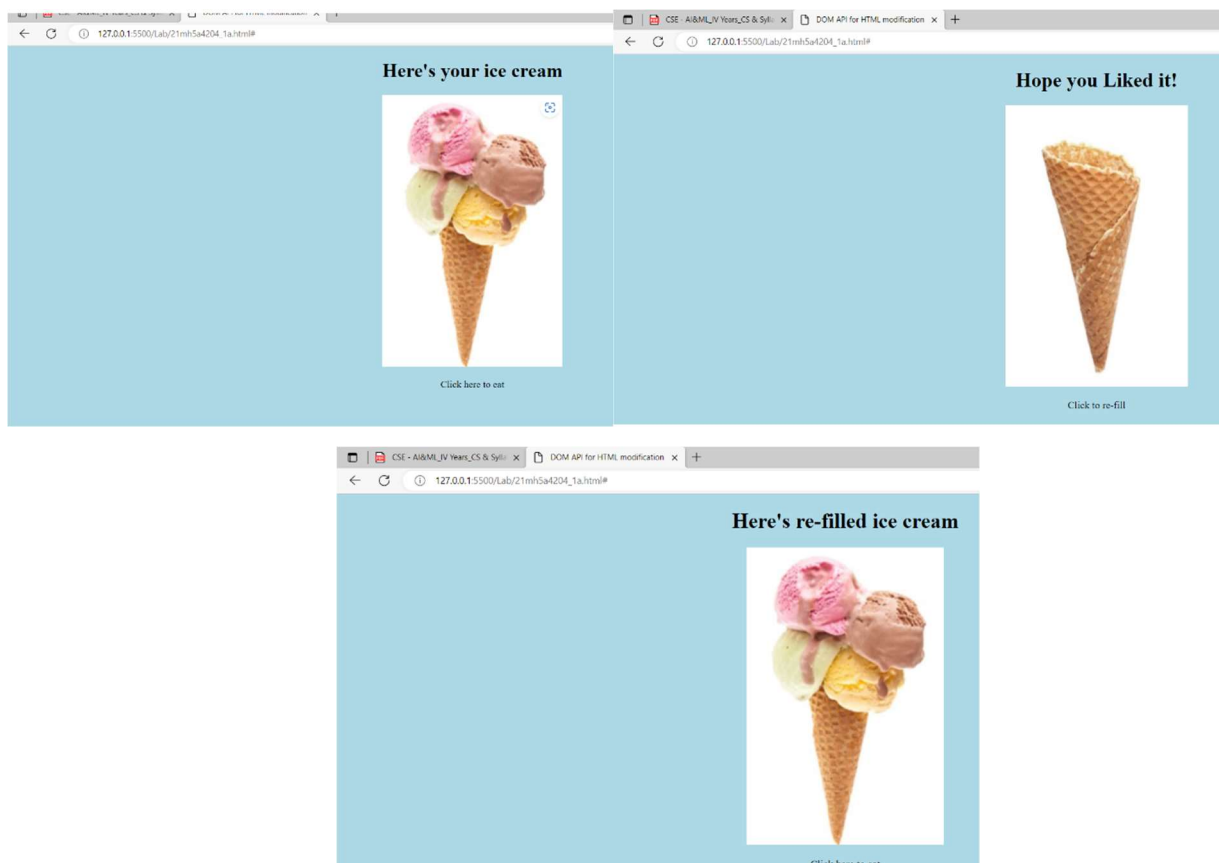
Experiment-4d:**Aim:**

If a user clicks on the given link, they should see an empty cone, a different heading, and a different message and a different background color. If user clicks again, they should see a re-filled cone, a different heading, a different message, and a differ

Description:**Source Code:**

```
<!DOCTYPE html>
<html>
<head>
  <title>DOM API for HTML modification</title>
</head>
<body style="background-color:lightblue;text-align:center">
  <h1 id="hdr1">Here's your ice cream</h1>
  
  <p id="p1" onclick="eat()">Click here to eat</p>
  <script>
    function eat(){
      let x=document.getElementById("hdr1").innerHTML;
      if(x=="Here's your ice cream"){
        x=document.getElementById("hdr1").innerHTML="Hope you Liked it!";
        document.getElementById("img1").src="ice1.jpg";
        x=document.getElementById("p1").innerHTML="Click to re-fill";
      }
      else if(x=="Hope you Liked it!"){
        x=document.getElementById("hdr1").innerHTML="Here's re-filled ice cream";
        document.getElementById("img1").src="ice.jpg";
        x=document.getElementById("p1").innerHTML="Click here to eat";
      }
      else{
        x=document.getElementById("hdr1").innerHTML="Hope you Liked it!";
        document.getElementById("img1").src="ice1.jpg";
        x=document.getElementById("p1").innerHTML="Click to re-fill";
      }
    }
  </script>
</body>
</html>
```

Output:

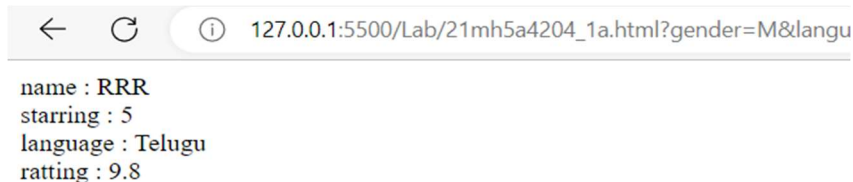


Experiment-5a:**Aim:**

Create an array of objects having movie details. The object should include the movie name, starring, language, and ratings. Render the details of movies on the page using the array.

Description:**Source Code:**

```
<!DOCTYPE html>
<html>
  <head>
    <title>JavaScript</title>
  </head>
  <body>
    <script>
      let movie={
        name:"RRR",
        starring:5,
        language:"Telugu",
        rating: 9.8,
      };
      for(key in movie){
        document.write(key + " : ");
        document.write(movie[key]);
        document.write("<br>");
      };
    </script>
  </body>
</html>
```

Output:


```
name : RRR
starring : 5
language : Telugu
rating : 9.8
```

Experiment-5b:**Aim:**

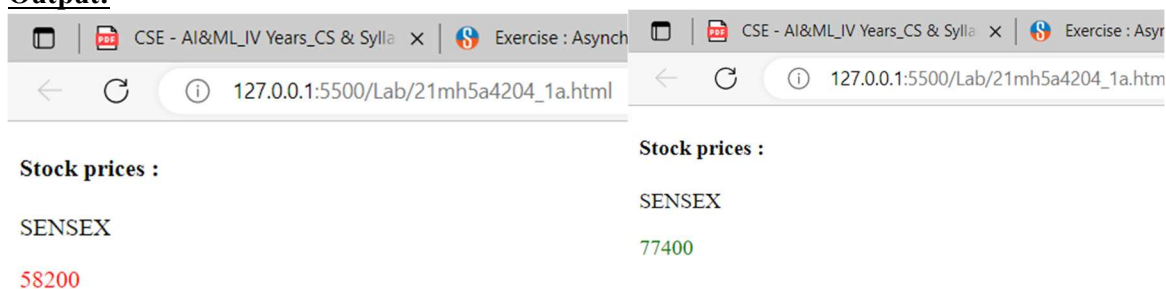
Simulate a periodic stock price change and display on the console. Hints: (i) Create a method which returns a random number - use Math.random, floor and other methods to return a rounded value. (ii) Invoke the method for every three seconds and stop when

Description:

Source Code:

```
<!DOCTYPE html>
<html>
<head>
  <title>DOM API for HTML modification</title>
</head>
<body>
  <h4>Stock prices :</h4>
  <p>SENSEX <p id="dem"></p></p>
  <script>
    let max=-1000,min=1000;
    function stock(){
      let marke=Math.floor((Math.random()*199)-99);
      if(60000+((marke/100)*60000)>60000){
        max=marke;
        document.getElementById("dem").style.color="green";
      }
      else if(60000+((marke/100)*60000)<60000){
        min=marke;
        document.getElementById("dem").style.color="red";
      }
      document.getElementById("dem").innerHTML=60000+((marke/100)*60000);
    }
    setInterval(stock,3000);
  </script>
</body>
</html>
```

Output:



Experiment-5c:**Aim:**

Validate the user by creating a login module. Hints: (i) Create a file login.js with a User class. (ii) Create a validate method with username and password as arguments. (iii) If the username and password are equal it will return "Login Successful" else w

Description:**Source Code:**

```
<!DOCTYPE html>
<html>
<head>
  <title>Login Validation</title>
  <style>
    body{
      min-height: 100vh;
      display: flex;
      align-items: center;
      justify-content: center;
    }
  </style>
</head>
<body>
<div id="logform">
  <div class="di">
    <div class="lab">Username</div>
    <div><input type="text" class="txt" placeholder="Username" maxlength="30" id="i"></div>
  </div>
  <div class="di">
    <div class="lab">Password</div>
    <div><input type="password" class="txt" placeholder="Password" maxlength="16" id="2"></div>
  </div>
  <div>
    <a href="#">Forget Password?</a>
  </div>
  <button onclick="vali()">Login</button>
  <div class="di">Don't have an account? <a onclick="register()" href="#">Signup Now</a></div>
</div>
<script>
  function vali(){
    let user=document.getElementById("i").value;
    let pass=document.getElementById("2").value;
    let c=new User(user,pass);
  }
}
```

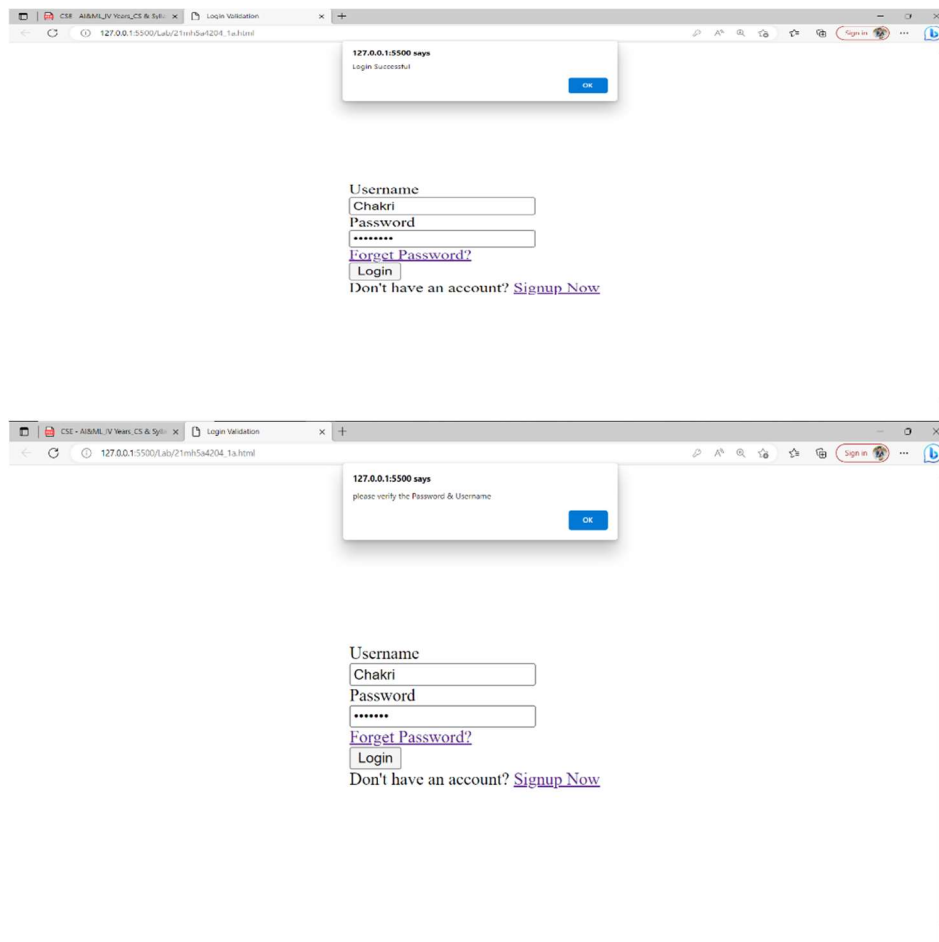


```

class User{
    constructor(user,pass){
        this.user=user;
        this.pass=pass;
        if(this.user.length<6 || this.pass.length<8){
            alert("please verify the Password & Username");
        }
        else{
            if(this.user=="Chakri" &&this.pass=="12345678"){
                alert("Login Successful");
            };
        };
    }
}
</script>
</body>
</html>

```

Output:



Experiment-6a:**Aim:**

Verify how to execute different functions successfully in the Node.js platform.

Description:

Step 1: Create a file **demo.js** in the folder created earlier.

```
function tester() {
  alert("Hello! I am an alert box!");
}
```

Step 3: Run the code and observe the output.

Step 4: Modify the code as shown below:

```
function tester() {
  var message;
  if (confirm("Press a button!")) {
    message = "You pressed OK!";
  } else {
    message = "You pressed Cancel!";
  }
  console.log(message);
}
tester();
```

Step 5: Run the code and observe the output.

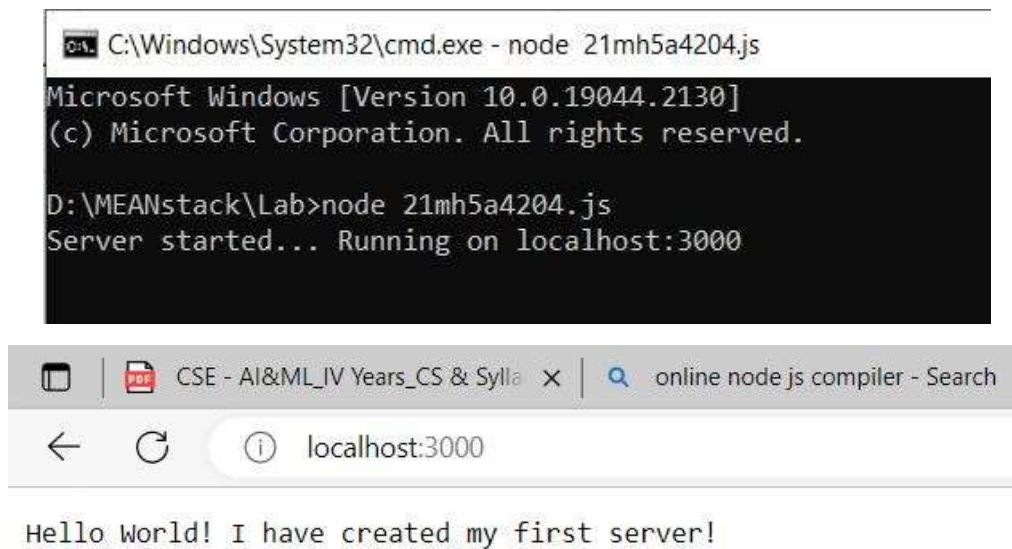
Experiment-6b:**Aim:**

Write a program to show the workflow of JavaScript code executable by creating web server in Node.js.

Description:**Source Code:**

```
const http = require("http");
var server=http.createServer((req,res)=>{
  res.write("Hello World! I have created my first server!");
  res.end();
});
server.listen(3000);
console.log("Server started... Running on localhost:3000");
```

Output:



The screenshot shows a Windows command prompt window with the title 'C:\Windows\System32\cmd.exe - node 21mh5a4204.js'. The text inside the prompt reads: 'Microsoft Windows [Version 10.0.19044.2130] (c) Microsoft Corporation. All rights reserved. D:\MEANstack\Lab>node 21mh5a4204.js Server started... Running on localhost:3000'. Below the command prompt is a web browser window with the address bar showing 'localhost:3000'. The browser page displays the text 'Hello World! I have created my first server!'.

Experiment-6c:**Aim:**

Write a Node.js module to show the workflow of Modularization of Node application.

Description:**Source Code:**

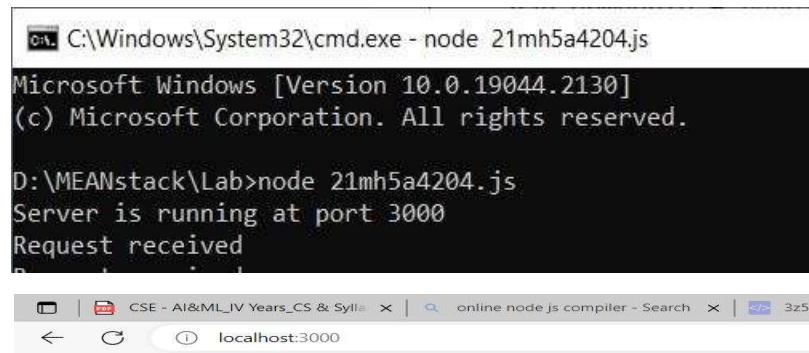
File: DBModule.js

```
exports.authenticateUser = (username, password) => {
  if (username === "admin" && password === "admin") {
    return "Valid User";
  }
  else return "Invalid User";
};
```

File: 21mh5a4204.js

```
const http = require("http");
var dbmodule = require("./DBModule");
var server = http.createServer((request, response) => {
  result = dbmodule.authenticateUser("admin", "admin");
  response.writeHead(200, { "Content-Type": "text/html" });
  response.end("<html><body><h1>" + result + "</h1></body></html>");
  console.log("Request received");
});
server.listen(3000);
```

```
console.log("Server is running at port 3000");
```

Output:

The screenshot shows a Windows command prompt window titled "C:\Windows\System32\cmd.exe - node 21mh5a4204.js". The prompt displays the following text: "Microsoft Windows [Version 10.0.19044.2130] (c) Microsoft Corporation. All rights reserved. D:\MEANstack\Lab>node 21mh5a4204.js Server is running at port 3000 Request received". Below the command prompt, a web browser window is visible with the address bar showing "localhost:3000". The browser tabs include "CSE - AI&ML_IV Years_CS & Syllabus" and "online node js compiler - Search".

Valid User

Experiment-7a:**Aim:**

On the page, display the price of the mobile-based in three different colors. Instead of using the number in our code, represent them by string values like GoldPlatinum, PinkGold, SilverTitanium.

Description:**Source Code:**

File:product.html

```
<!doctype html>
<html>
  <head>
    <title>Mobile Cart</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">

    <link href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css" rel="stylesheet">
    <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>

    <style>
      .navbar-inverse {
        background-color:#005580;
        background-image: none;
        background-repeat: no-repeat;
        color:#ffffff;
      }
      .navbar-inverse .navbar-nav > .active > a {
        color: #ffffff;
        background-color:transparent;
      }
      .navbar-inverse .navbar-nav > li > a:hover{
        text-decoration: none;
        color: #ffffff;
      }
    </style>
  </head>
  <body>
    <nav class="navbar navbar-inverse navbar-fixed-top">
      <div class="navbar-header">
        <button type="button" class="navbar-toggle collapsed" data-toggle="collapse" data-target="#navbar"
        aria-expanded="false" aria-controls="navbar">
```

```

    <span class="sr-only">Toggle navigation</span>
    <span class="icon-bar"></span>
    <span class="icon-bar"></span>
    <span class="icon-bar"></span>
  </button>
  <a class="navbar-brand" href="#">Mobile Cart</a>
</div>
<div id="navbar" class="collapse navbar-collapse">
  <ul class="nav navbar-nav">
    <li><a href="#">Home</a></li>

    </ul>
    <!--/.nav-collapse -->
    <ul class="nav navbar-nav navbar-middle" style="color:white; margin-right:30px;">
    <li><a href="Cart.html"><span class="glyphicon glyphicon-shopping-cart"
style="color:white"></span></a></li>
    </ul>
  </div>
</nav>
<div style="margin-top:7%">
  <center><h2>Your Favorite Online Mobile Shop!</h2></center>
</div>
<div class="container" style="padding-top:5%">
  <div class="row">
    <div class="col-md-4">
      <div style="text-align: center;">
        
      </div>
      <div style="padding-top:10px;">
        <div style="cursor:pointer;color:Steelblue;text-align: center;"><b>
          <span id="pName"></span></b></div>
        <div style="padding-top:10px;padding-left: 101px;"><b>Price:</b>&nbsp;&dollar;<span
id="pPrice"></span></div>
        <div style="padding-left: 100px;"><b>Status:</b><span id="pAvailable"></span></div>

      </div>
    </div>

  </div>
</div>
</div>
</body>
<!--Adding the converted js file -->
<script src="productlist.js"></script>
</html>

```

File:productlist.ts

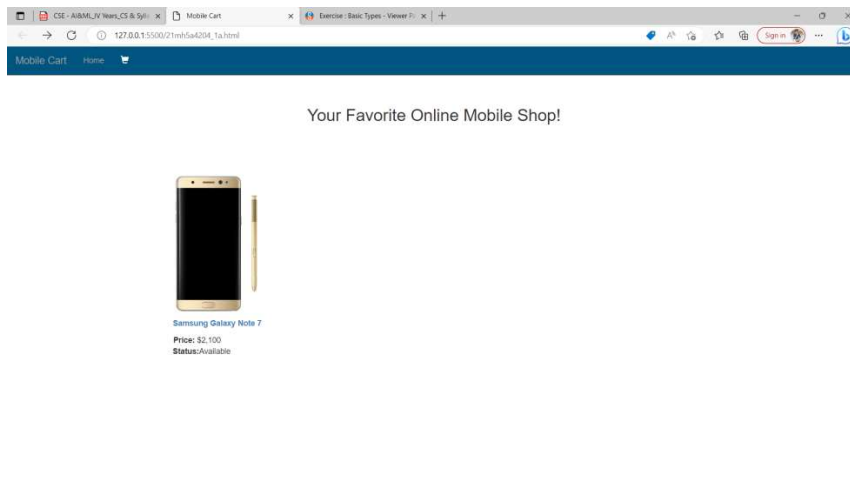
```

let mobileName:string="Samsung Galaxy Note 7";
let pPrice:string="2,100";
let pAvailable:string="Available";
document.getElementById("pName").innerHTML=mobileName;
document.getElementById("pPrice").innerHTML=pPrice;

```

```
document.getElementById("pAvailable").innerHTML=pAvailable;
```

Output:



Experiment-7b:

Aim:

Define an arrow function inside the event handler to filter the product array with the selected product object using the productId received by the function. Pass the selected product object to the next screen.

Description:

Source Code:

File: product.html

```
<!doctypehtml>
<html>
  <head>
    <title>Mobile Cart</title>
    <metacharset="UTF-8">
    <metaname="viewport"content="width=device-width, initial-scale=1">
    <linkhref="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css"rel="stylesheet">
    <scriptsrc="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
    <scriptsrc="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>
  </style>
  .navbar-inverse {
    background-color:#005580;
    background-image: none;
    background-repeat: no-repeat;
    color:#ffffff;
  }
  .navbar-inverse.navbar-nav>.active>a {
```

```

color: #ffffff;
background-color: transparent;

}
.navbar-inverse.navbar-nav>li>a: hover{
    text-decoration: none;
    color: #ffffff;
}
</style>
</head>
<body>
    <nav class="navbar navbar-inverse navbar-fixed-top">
        <div class="navbar-header">
            <button type="button" class="navbar-toggle collapsed" data-toggle="collapse" data-target="#navbar" aria-
expanded="false" aria-controls="navbar">
                <span class="sr-only">Toggle navigation</span>
                <span class="icon-bar"></span>
                <span class="icon-bar"></span>
                <span class="icon-bar"></span>
            </button>
            <a class="navbar-brand" href="#">Mobile Cart</a>
        </div>
        <div id="navbar" class="collapse navbar-collapse">
            <ul class="nav navbar-nav">
                <li><a href="#">Home</a></li>
            </ul>
            <!--/.nav-collapse -->
            <ul class="nav navbar-nav navbar-middle" style="color: white; margin-right: 30px;">
                <li><a href="Cart.html"><span class="glyphicon glyphicon-shopping-
cart" style="color: white"></span></a></li>

            </ul>
        </div>
    </nav>
    <div style="margin-top: 7%">
        <center><h2>Your Favorite Online Mobile Shop!</h2></center>
    </div>
    <div class="container" style="padding-top: 5%">
        <div class="row">
            <div class="col-md-4">
                <div style="text-align: center;">
                    
                </div>
                <div style="padding-top: 10px;">
                    <div onclick="getMobileDetails('Samsung',432);" style="cursor: pointer; color: Steelblue; text-align:
center;"><b><span id="pName0"></span></b></div>
                        <div style="padding-top: 10px; padding-left:
101px;"><b>Price:</b>&nbsp;&dollar;<span id="pPrice0"></span></div>
                        <div style="padding-left: 100px;"><b>Status:</b><span id="pAvailable0"></span></div>

                </div>

```



```

</div>
<div class="col-md-4">
  <div style="text-align: center;">
    
  </div>
  <div style="padding-top: 10px;">

<div onclick="getMobileDetails('samsungedge',231);" style="cursor:pointer;color:Steelblue;text-align:
center;"><b><span id="pName1"></span></b></div>
    <div style="padding-top: 10px;padding-left:
95px;"><b>Price:</b>&nbsp;&dollar;<span id="pPrice1"></span></div>
    <div style="padding-left: 94px;"><b>Status:</b><span id="pAvailable1"></span></div>

  </div>
</div>
<div class="col-md-4">
  <div style="text-align: center;">
    
  </div>
  <div style="padding-top: 10px;">
    <div onclick="getMobileDetails('lumia',875);" style="cursor:pointer;color:Steelblue;text-
align: center; "><b><span id="pName2"></span></b></div>
    <div style="padding-top: 10px;padding-left:
118px;"><b>Price:</b>&nbsp;&dollar;<span id="pPrice2"></span></div>
    <div style="padding-left: 117px;"><b>Status:</b><span id="pAvailable2"></span></div>
  </div>
</div>

</div>

</div>

</body>
<!--Adding the converted js file -->
<script src="arrays.js"></script>
</html>

```

File: arrays.ts

```

const productList : any[] = [
  {pId:432,productName:"Samsung Galaxy Note
7",productPrice:699,productAvailable:true,imageUrl:"SamsungGalaxy_Gold.jpg",productDescription:"Samsu
ng Galaxy Note 7 is a stylish mobile you can ever have. It has 64GB memory."},
  {pId:231,productName:"Samsung Galaxy s6
Edge",productPrice:630,productAvailable:true,imageUrl:"samsung_edge_silver.jpg",productDescription:"Sa
msung Galaxy Note 7 is a stylish mobile you can ever have. It has 64GB memory."},
  {pId:875,productName:"Nokia Lumina
640XL",productPrice:244,productAvailable:false,imageUrl:"lumia_640xl.jpg",productDescription:"Samsung
Galaxy Note 7 is a stylish mobile you can ever have. It has 64GB memory."},
]
for(let i=0;i<productList.length;i++){
  document.getElementById('pName${i}').innerHTML = productList[i].productName;
  document.getElementById('pPrice${i}').innerHTML = productList[i].productPrice;
}

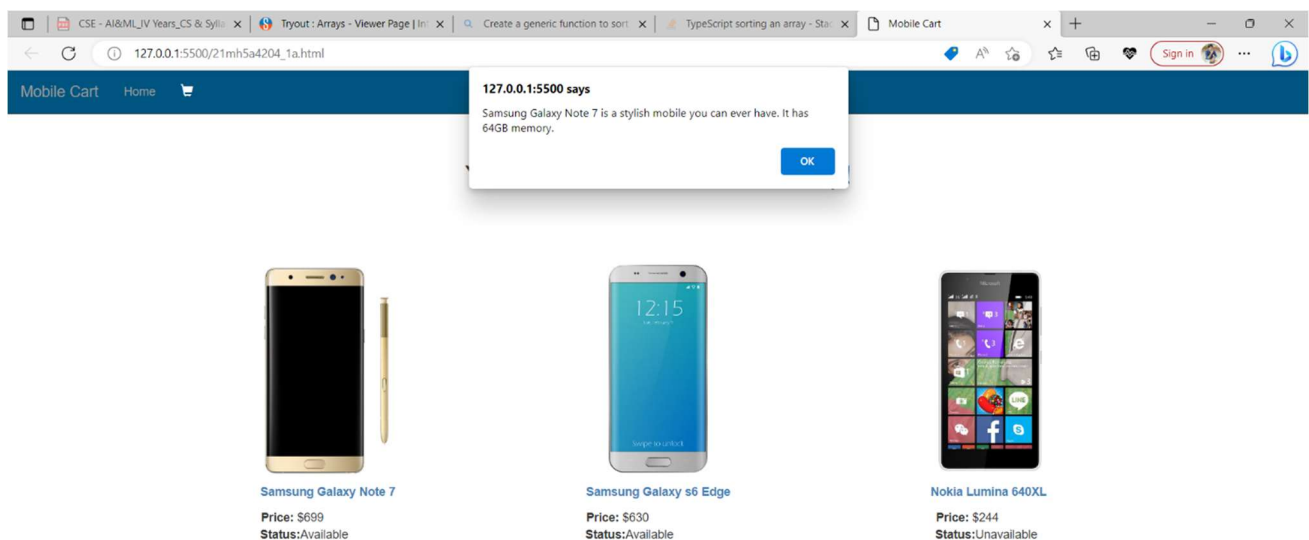
```

```

if (productlist[i].productAvailable){
    document.getElementById(`pAvailable${i}`).innerHTML = "Available";
}
else{
    document.getElementById(`pAvailable${i}`).innerHTML = "Unavailable"
}
}
function getMobileDetails(produc: string, po: Number): void {
    if(po==423){
        alert(productlist[0].productDescription);
    }
    elseif(po==231){
        alert(productlist[1].productDescription);
    }
    elseif(po==875){
        alert(productlist[2].productDescription);
    }
}
}

```

Output:



Experiment-7c:

Aim:

Consider that developer needs to declare a function - getMobileByVendor which accepts string as input parameter and returns the list of mobiles.

Description:

Source Code:

```
// declaring a function which accepts string datatype as parameter and returns string array
function getMobileByManufacturer(manufacturer: string): string[] {
  let mobileList: string[];
  if (manufacturer === 'Samsung') {
    mobileList = ['Samsung Galaxy S6 Edge', 'Samsung Galaxy Note 7',
'Samsung Galaxy J7 SM-J700F'];
    return mobileList;
  } elseif (manufacturer === 'Apple') {
    mobileList = ['Apple iPhone 5s', 'Apple iPhone 6s ', 'Apple iPhone 7'];
    return mobileList;
  } else {
    mobileList = ['Nokia 105', 'Nokia 230 Dual Sim'];
    return mobileList;
  }
}
// logic to populate the Samsung manufacturer details on console
console.log('The available mobile list: ' + getMobileByManufacturer('Samsung'));
```

Output:

```
D:\MEANstack\Lab>tsc g.ts
D:\MEANstack\Lab>node g.js
The available mobile list: Samsung Galaxy S6 Edge,Samsung Galaxy Note 7,Samsung Galaxy J7 SM-J700F
```

Experiment-7d:**Aim:**

Consider that developer needs to declare a manufacturer's array holding 4 objects with id and price as a parameter and needs to implement an arrow function - myfunction to populate the id parameter of manufacturers array whose price is greater than or equ.

Description:**Source Code:**

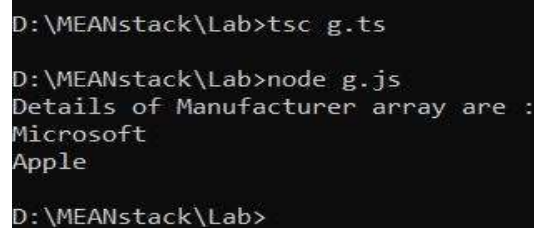
```
// declaring an Array with 3 objects
const manufacturers = [{ id:'Samsung', price:150 },
  { id:'Microsoft', price:200 },
  { id:'Apple', price:400 },
  { id:'Micromax', price:100 }
];
```

```

lettest;
// Arrow function to populate the details of Array whose price is greater than 200
functionmyFunction() {
  test = manufacturers.filter((manufacturer) =>manufacturer.price>= 200);
}
// self-invoking an arrow function
myFunction();
console.log('Details of Manufacturer array are : ');
// logic to populate the manufacturer array details based on id value
for (constitemoftest) {
  console.log(item.id);
}

```

Output:



```

D:\MEANstack\Lab>tsc g.ts
D:\MEANstack\Lab>node g.js
Details of Manufacturer array are :
Microsoft
Apple
D:\MEANstack\Lab>

```

Experiment-7e:

Aim:

Declare a function - getMobileByManufacturer with two parameters namely manufacturer and id, where manufacturer value should be passed as Samsung and id parameter should be optional while invoking the function, if id is passed as 101 then this function should

Description:

Source Code:

```

// declaring a function with optional parameter
functiongetMobileByManufacturer(manufacturer: string = 'Samsung', id?: number): string[] {
  letmobileList: string[];
  // logic to be evaluated if id parameter while invoking above declared function
  if (id) {
    if (id === 101) {
      mobileList = ['Moto G Play, 4th Gen', 'Moto Z Play with Style Mod'];
      returnmobileList;
    }
  }
  // logic to return mobileList based on manufacturer category
  if (manufacturer === 'Samsung') {
    mobileList = [' Samsung Galaxy S6 Edge', ' Samsung Galaxy Note 7',

```

```

    'Samsung Galaxy J7 SM-J700F'];
    return mobileList;
  } elseif (manufacturer === 'Apple') {
    mobileList = ['Apple iPhone 5s', 'Apple iPhone 6s', 'Apple iPhone 7'];
    return mobileList;
  } else {
    mobileList = ['Nokia 105', 'Nokia 230 Dual Sim'];
    return mobileList;
  }
}

// statement to invoke function with optional parameter
console.log('The available mobile list : ' + getMobileByManufacturer('Apple'));
// statement to invoke function with default parameter
console.log('The available mobile list : ' + getMobileByManufacturer(undefined, 101));

```

Output:

```

D:\MEANstack\Lab>tsc g.ts
D:\MEANstack\Lab>node g.js
The available mobile list : Apple iPhone 5s, Apple iPhone 6s, Apple iPhone 7
The available mobile list : Moto G Play, 4th Gen,Moto Z Play with Style Mod

```

Experiment-8a:**Aim:**

Implement business logic for adding multiple Product values into a cart variable which is type of string array.

Description:**Source Code:**

```
const cart: string[] = [];
const pushToCart = (item: string) => { cart.push(item); };
function addToCart(...productName: string[]): string[] {
    for (const item of productName) {
        pushToCart(item);
    }
    return cart;
}
console.log('Cart Items are:' + addToCart(' Moto G Play, 4th Gen', ' Apple iPhone 5s'));
```

Output:

```
PS D:\MEANstack\Lab> node productlist.js
Cart Items are: Moto G Play, 4th Gen, Apple iPhone 5s
PS D:\MEANstack\Lab> []
```

Experiment-8b:**Aim:**

Declare an interface named - Product with two properties like productId and productName with a number and string datatype and need to implement logic to populate the Product details.

Description:**Source Code:**

```
interface Product {
    productId: number ;
    productName: string ;
}
function getProductDetails(productObj: Product): string {
```

```

    return 'The product name is : ' + productobj.productName;
  }
  const prodObject = {productId: 1001, productName: 'Mobile'};
  const productDetails: string = getProductDetails(prodObject);
  console.log(productDetails);

```

Output:

```

PS D:\MEANstack\Lab> tsc.cmd productlist.ts
PS D:\MEANstack\Lab> node productlist.js
The product name is : _Mobile

```

Experiment-8c:**Aim:**

Declare an interface named - Product with two properties like productId and productName with the number and string datatype and need to implement logic to populate the Product details.

Description:**Source Code:**

```

interface Product {
  productId: number;
  productName: string;
}
function getProductDetails(productobj: Product): string {
  return 'The product name is : ' + productobj.productName;
}
const prodObject = {productId: 1001, productName: 'Mobile', productCategory: 'Gadget'};
const productDetails: string = getProductDetails(prodObject);
console.log(productDetails);

```

Output:

```

PS D:\MEANstack\Lab> tsc.cmd productlist.ts
PS D:\MEANstack\Lab> node productlist.js
The product name is : Mobile
PS D:\MEANstack\Lab> 

```

Experiment-8d:**Aim:**

Declare an interface with function type and access its value.

Description:

Source Code:

```
function CreateCustomerID(name: string, id: number): string {  
    return 'The customer id is ' + name + ' ' + id;  
}  
interface StringGenerator {  
    (chars: string, nums: number): string;  
}  
let idGenerator: StringGenerator;  
idGenerator = CreateCustomerID;  
const customerId: string = idGenerator('Mr.Tom', 101);  
console.log(customerId);
```

Output:

```
PS D:\MEANstack\Lab> tsc.cmd productlist.ts  
PS D:\MEANstack\Lab> node productlist.js  
The customer id is Mr.Tom 101
```


Experiment-9a:**Aim:**

Declare a productList interface which extends properties from two other declared interfaces like Category, Product as well as implementation to create a variable of this interface type.

Description:**Source Code:**

```
// declaring a Category interface
interface Category {
    categoryName: string;
}

// declaring a Product interface
interface Product {
    productName: string;
    productId: number;
}

// declaring a ProductList interface which extends from Category and Product interfaces
interface ProductList extends Category, Product {
    list: Array<string>;
}

// declaring a variable which is type of ProductList interface
const productDetails: ProductList = {
    categoryName: 'Gadget',
    productName: 'Mobile',
    productId: 1234,
    list: ['Samsung', 'Motorola', 'LG']
};

// assigning list value of productDetails variable into another variable
const listProduct = productDetails.list;
// assigning productName value of productDetails variable into another variable
const pname: string = productDetails.productName;
// line to populate Product name
console.log('Product Name is ' + pname);
// line to populate Product list
console.log('Product List is ' + listProduct);
```

Output:

```
PS D:\MEANstack\Lab> tsc.cmd productlist.ts
PS D:\MEANstack\Lab> node productlist.js
Product Name is Mobile
Product List is Samsung,Motorola,LG
```

Experiment-9b:

Aim:

Consider the Mobile Cart application, Create objects of the Product class and place them into the productlist array.

Description:

Source Code:

```
// declaring a Product interface
interface Product {
  displayProductName: (productId: number) => string;
  getProductDetails(): string[];
}
// declaring Gadget class which implements Product interface
class Gadget implements Product {
  getProductDetails(): string[] {
    return ['Samsung', 'LG', 'Moto'];
  }
  displayProductName(productId: number): string {
    if (productId === 101) {
      return 'Product Name is Mobile';
    } else if (productId === 201) {
      return 'Product Name is Tablet';
    }
  }
  getGadget(): string[] {
    return ['Mobile', 'Tablet', 'iPad', 'iPod'];
  }
}
// creating instance of class of interface type
const gadget: Product = new Gadget();
// creating variable to hold return value of displayProductName function
const productName: string = gadget.displayProductName(101);
// line to populate Product name on console
console.log(productName);
```

Output:

```
PS D:\MEANstack\Lab> tsc.cmd productlist.ts
PS D:\MEANstack\Lab> node productlist.js
Product Name is Mobile
```

Experiment-9c:**Aim:**

Declare a class named - Product with the below-mentioned declarations: (i) productId as number property (ii) Constructor to initialize this value (iii) getProductId method to return the message "Product id is <id va>".

Description:**Source Code:**

```
// declaring a Product class
class Product {
  static productPrice: string;
  productId: number;
// constructor declaration
  constructor(productId: number) {
    this.productId = productId;
  }
  getProductId(): string {
    return 'Product id is : ' + this.productId;
  }
}
// creation of Product class object
const product: Product = new Product(1234);
// line to populate the product id details
console.log(product.getProductId());
```

Output:

```
PS D:\MEANstack\Lab> tsc.cmd productlist.ts
PS D:\MEANstack\Lab> node productlist.js
Product id is : 1234
```

Experiment-9d:**Aim:**

Create a Product class with 4 properties namely productId, productName, productPrice, productCategory with private, public, static, and protected access modifiers and accessing them through Gadget class and its methods.

Description:**Source Code:**

```
// declaring a Product class with access modifiers
classProduct {
    staticproductPrice = 150;
    privateproductId: number;
    publicproductName: string;
    protectedproductCategory: string;
    // declaration of constructor with 3 parameters
    constructor(productId: number, productName , productCategory) {
        this.productId = productId;
        this.productName = productName;
        this.productCategory = productCategory;
    }
    // method ot display product id details
    getProductId() {
        console.log('The Product id is : ' + this.productId);
    }
}
// declaring a Gadget class extending the properties from Product class
classGadgetextendsProduct {
    // method to display productCategory property
    getProduct(): void {
        console.log('Product category is : ' + this.productCategory);
    }
}
// Gadget Class object creation
constg: Gadget = newGadget(1234, 'Mobile', 'SmartPhone');
// invoking getProduct method with the help of Gadget object
g.getProduct();
// invoking getProductId method with the help of Gadget object
g.getProductId();
// line to populate product name property with Gadget object
console.log('Product name is : ' + g.productName);
// line to populate static property product price directly with Product class name
console.log('Product price is : $' + Product.productPrice);
```

Output:

```
PS D:\MEANstack\Lab> tsc.cmd productlist.ts
PS D:\MEANstack\Lab> node productlist.js
Product category is : SmartPhone
The Product id is : 1234
Product name is : Mobile
Product price is : $150
PS D:\MEANstack\Lab> █
```

Experiment-10a:**Aim:**

Create a Product class with 4 properties namely productId and methods to setProductId() and getProductId().

Description:**Source Code:**

```
// declaring Product class
classProduct {
    protectedproductId: number;
    constructor(productId: number) {
        this.productId = productId;
    }
    getProduct(): void {
        console.log('Product id is : ' + this.productId);
    }
}
// declaring Gadget class which extends properties from Product class
classGadgetextendsProduct {
    constructor(publicproductName: string , productId: number ) {
        super(productId);
    }
    getProduct(): void {
        super.getProduct();
        console.log('Product id is : ' + this.productId + ' Product name is : ' + this.productName);
    }
}
// Gadget class object creation
constg = newGadget('Tablet', 1234);
// line to invoke getProduct method with the help of Gadget object
g.getProduct();
```

Output:

```
PS D:\MEANstack\Lab> tsc.cmd productlist.ts
PS D:\MEANstack\Lab> node productlist.js
Product id is : 1234
Product id is : 1234 Product name is : Tablet
PS D:\MEANstack\Lab> █
```

Experiment-10b:**Aim:**

Create a namespace called ProductUtility and place the Product class definition in it. Import the Product class inside productlist file and use it.

Description:

Step 1: Create a file namespace_demo.ts

```
namespaceUtility {
  exportnamespacePayment {
    exportfunctionCalculateAmount(price: number, quantity: number): number {
      returnprice * quantity;
    }
  }
  exportfunctionMaxDiscountAllowed(noOfProduct: number): number {
    if (noOfProduct>5) {
      return40;
    } else {
      return10;
    }
  }
  functionprivateFunc(): void {
    console.log('This is private...');
  }
}
```

Step 2: Create another file Namespace_import.ts

```
/// <referencepath="./namespace_demo.ts"/>
importutil = Utility.Payment;
letpaymentAmount = util.CalculateAmount(1255, 6);
console.log(`Amount to be paid: ${paymentAmount}`);
letdiscount = Utility.MaxDiscountAllowed(6);
console.log(`Maximum discount allowed is: ${discount}`);
```

Step 3: Open Node.js command prompt, change directory to the folder in which Namespace file resides and run the below command from the command line:

```
D:\MEANstack\Lab>tsc --outFile Final.js namespace_demo.ts Namespace_import.ts
```

Step 4: Run **node Final.js** from the command line

```
D:\MEANstack\Lab>node Final.js
Amount to be paid: 7530
Maximum discount allowed is: 40
```

Experinment-10c:

Aim:

Consider the Mobile Cart application which is designed as part of the functions in a module to calculate the total price of the product using the quantity and price values and assign it to a totalPrice variable.

Description:

Step 1: Create a file `module_demo.ts`

```
export function MaxDiscountAllowed(noOfProduct: number): number {
  if (noOfProduct > 5) {
    return 30;
  } else {
    return 10;
  }
}

class Utility {
  CalculateAmount(price: number, quantity: number): number {
    return price * quantity;
  }
}

interface Category {
  getCategory(productId: number): string;
}

export const productName = 'Mobile';
export { Utility, Category };
```

Step 2: Create another file `2.module_import.ts`

```
import { Utility as mainUtility, Category, productName, MaxDiscountAllowed } from './module_demo';

const util = new mainUtility();
const price = util.CalculateAmount(1350, 4);
const discount = MaxDiscountAllowed(2);
console.log(`Maximum discount allowed is: ${discount}`);
console.log(`Amount to be paid: ${price}`);
console.log(`ProductName is: ${productName}`);
```

Step 3: Open a command prompt and navigate to the folder in which module files resides and run the below command from the command line

```
D:\MEANstack\Lab>tsc module_demo.ts 2.module_import.ts
```

Step 4: Run `node 2.module_import.js` from the command line

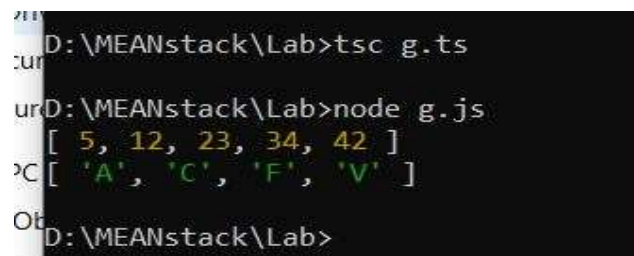
```
D:\MEANstack\Lab>node Final.js
Amount to be paid: 7530
Maximum discount allowed is: 40
```


Experiment-10d:**Aim:**

Create a generic array and function to sort numbers as well as string values.

Description:**Source Code:**

```
function sortedArr<T>(arg: Array<T>): Array<T> {
  var sortedArray: Array<T> = arg.sort((n1, n2) => {
    if (n1 > n2) {
      return 1;
    }
    if (n1 < n2) {
      return -1;
    }
    return 0;
  });
  return sortedArray;
}
const num: Array<number> = [5, 12, 23, 42, 34];
const str: Array<string> = ['C', 'F', 'V', 'A'];
const sortedNum = sortedArr(num);
console.log(sortedNum);
const sortedStr = sortedArr(str);
console.log(sortedStr);
var stringArray: string[] = ['G', 'Z', 'A', 'D'];
```

Output:


```
D:\MEANstack\Lab>tsc g.ts
D:\MEANstack\Lab>node g.js
[ 5, 12, 23, 34, 42 ]
[ 'A', 'C', 'F', 'V' ]
D:\MEANstack\Lab>
```