**1.a Course Name: HTML5 - The Language**

**Module Name: Case-insensitivity, Platform-independency, DOCTYPE Declaration, Types of Elements, HTML Elements - Attributes, Metadata Element 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.**

**Program:**

<!DOCTYPE html>

<html >

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

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

<title>EXp1a</title>

</head>

<body>

<center>

<h1>WELCOME TO IKART'S ONLINE SERVICE'S</h1>

</center>

</body>

</html>

# Output:



**1.b Course Name: HTML5**

**The Language Module Name: Sectioning Elements Enhance the Homepage.html of IEKart's Shopping Application by adding appropriate sectioning elements.**

# Program:

<!DOCTYPE html>

<html >

<head>

<title>EXp1a</title>

<style>

#header{

border:1px solid black;width:100%;height:60px;

}

#nav{

text-align:center;border:1px solid black;width:100%;height:60px;

}

#aside{

display:inline;float:left;width:20%;border:1px solid lack;height:500px;

}

#section{

width:80%;height: 500px;border:1px solid black;margin-left:305px;text-align: center;

}

#footer{

border:1px solid black;width:100%;height:100px;text-align:center}

</style>

</head>

<body>

<div id="header">

<center><h1>

<header>

WELCOMETOIKART'SNLINESERVICE'S</header>

</h1></center>

</div>

<div id="nav"><nav><b>This is navigation section</b></nav></div>

< div id="aside"><b><aside>This is aside Section</aside></b></div>

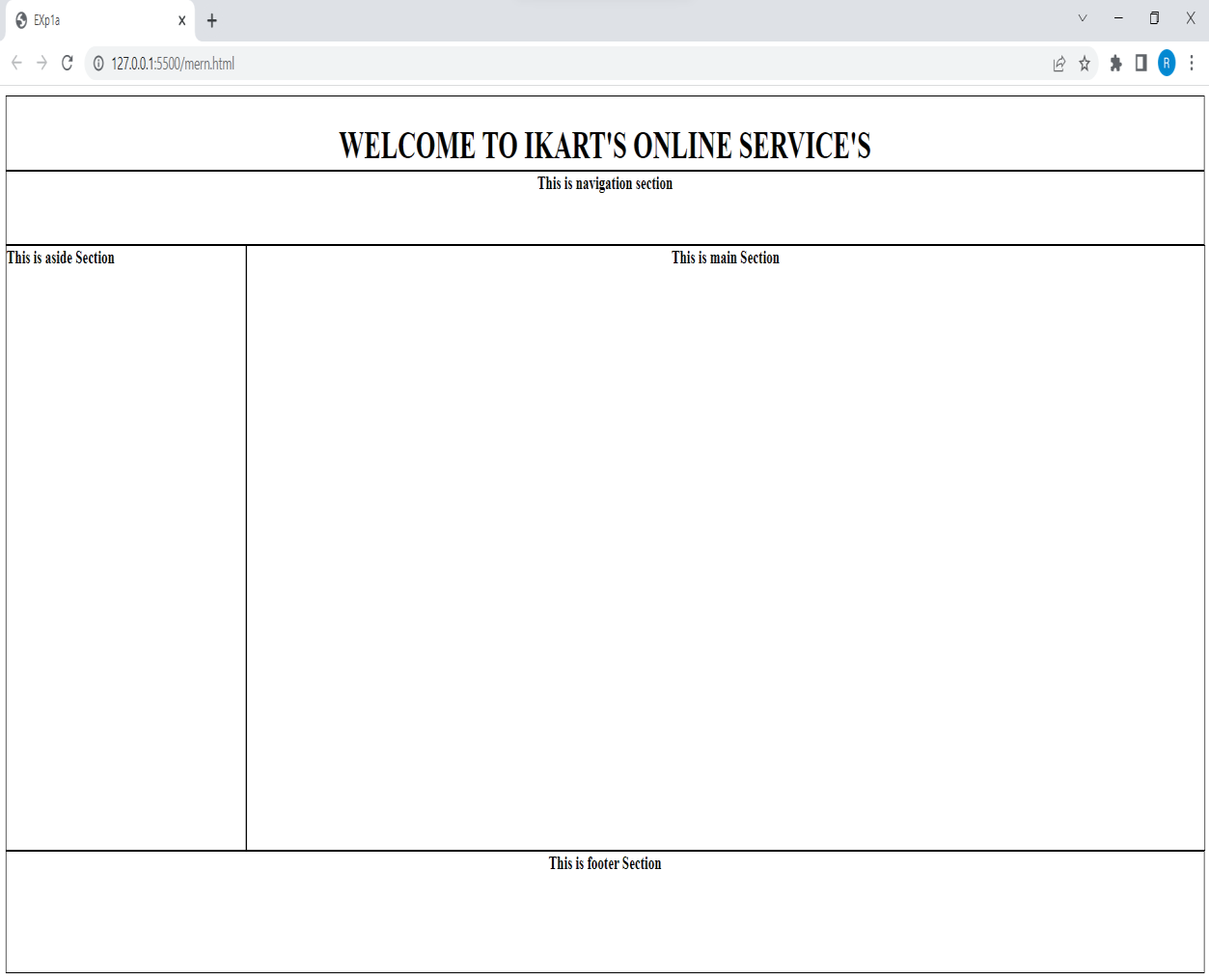
<div id="section"><b><section>This is main Section</section></b></div>

<div id="footer"><b><footer >This is footer Section</footer></b></div>

</body>

</html>

# Output:



**1.c Course Name: HTML5 - The Language Module Name: Paragraph Element, Division and Span Elements, List Element Make use of appropriate grouping elements such as list items to "About Us" page of IEKart's Shopping Application.**

**Program:**

<!DOCTYPE html>

<html >

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

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

<title>EXp1a</title>

<style>

#header

{

border:1px solid black;width:100%;height:60px;

}

#nav

{

text-align:center;border:1px solid black;width:100%;height:60px;

}

#aside

{

display:inline;float:left;width:20%;border:1pxsolidblack;height:500px;

}

#section

{

width:80%;height: 500px;border:1px solid black;margin-left:305px;text-align: center;

}

#footer

{border:1px solid black;width:100%;height:100px;text-align:center}

</style>

</head>

<body bgcolor="antiquewhite">

<div id="header">

<center>

<h1><header>WELCOME TO IKART'S ONLINESERVICE'S</header>

</h1>

</center></div>

<div id="nav"><nav><b>This is navigation section</b></nav></div>

<div id="aside"><aside> <details>

<summary>About Us</summary>

<ul>

<li>Limited License

<p>Limited Licence Refers to a licence to an intellectual property right, typically copyright, which does not grant all the rights which the ownerof the intellectual property possesses, but only some of those rights.</p>

</li>

<li>Copy Rights

<p>According to the US Copyright Office, copyright applies to original works of creative expression that are captured in a tangible form.

</p>

</li>

<li>Terms Policy

<ul> <li>Trademark or Copyright Infringement Cease and Desist Letter</li>

<li>Website Cookies Policy</li>

<li>Shipping Policy for the Website</li></ul>

</li>

</ul>

</details>

</aside></div>

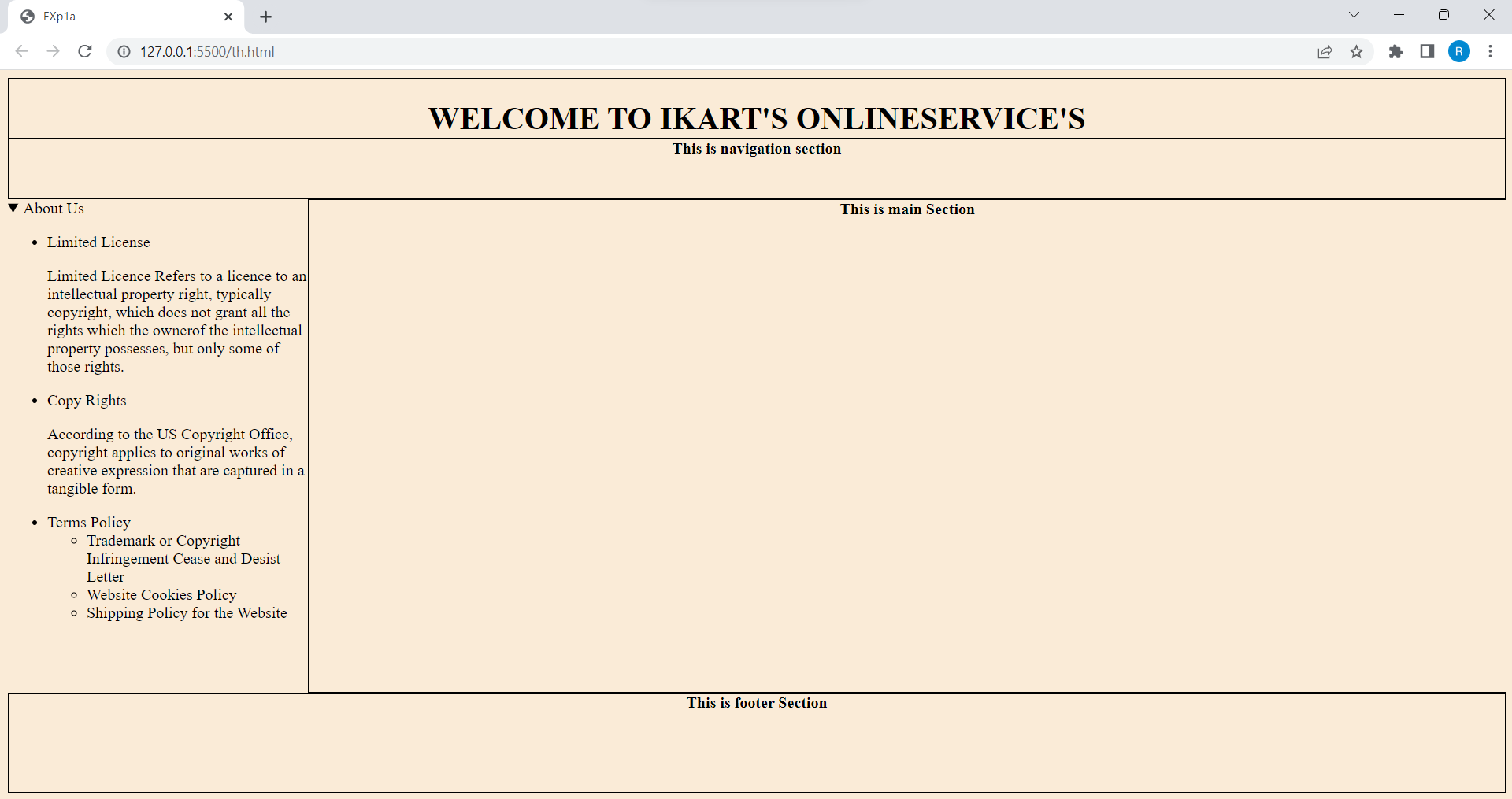
<div id="section"><b><section>This is main Section</section></b></div>

<div id="footer"><b><footer >This is footer Section</footer></b></div>

</body>

</html>

**Output:**



**1.d Course Name: HTML5 - The Language Module Name: Link Element 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.**

**Program:**

<!DOCTYPE html>

<html >

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

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

<title>EXp1a</title>

<style>

#header{

width:100%;height:120px;background:black;

}

#header h1{

color:white;

}

#nav{

text-align:right;width:100%;height:60px;

}

#nav span{

background-color:white; border:1px solid black; padding:5px;

border-radius: 4px;

}

#nav span a{

text-decoration: none;

}

#aside{

float:left;width:400px;height:700px;

}

#section{

width:80%;height: 600px;margin-left:305px;text-align: center;

}

#footer{

width:100%;height:100px;text-align:center

}

</style>

</head>

<body bgcolor="antiquewhite">

<div id="header">

<center><h1><header>WELCOME TO IKART'S ONLINE SERVICE'S</header></h1>

</center>

<div id="nav"><nav>

<span>Track </span><span><a href="sign.html">Sign UP</a></span>

<span><a href="login.html">Login</a></span></nav></div></div>

<div id="aside"><aside> <details>

<a href="details.html">Inventory</a>

<summary>About Us</summary>

<ul>

<li>Limited License

<p>Limited Licence Refers to a licence to an intellectual property right,

typically copyright, which does not grant all the rights which the ownerof the intellectual property possesses, but only some of those rights.</p>

</li>

<li>Copy Rights

<p>According to the US Copyright Office, copyright applies to original works of creative expression that are captured in a tangible form.

</p>

</li>

<li>Terms Policy

<ul> <li>Trademark or Copyright Infringement Cease and Desist Letter</li>

<li>Website Cookies Policy</li>

<li>Shipping Policy for the Website</li></ul>

</li>

</ul>

</details>

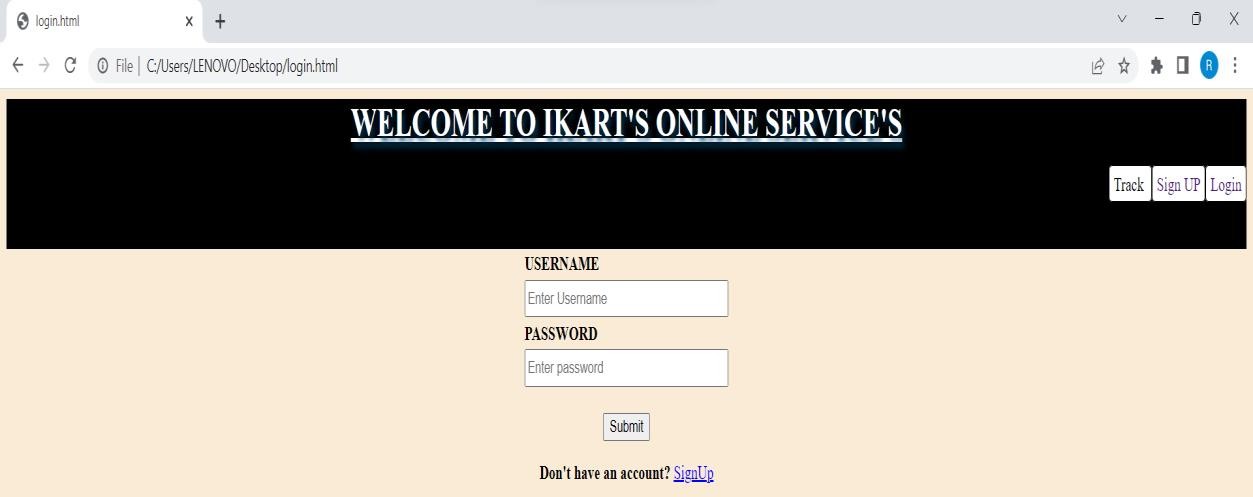
</div>

</section></b>

</div>

</body>

</html>

**Output:**

**1.e Course Name: HTML5 - The Language Module Name: Character Entities Add the © symbol in the Home page footer of IEKart's Shopping application.**

**PROGRAM:**

<!DOCTYPE html>

<html >

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

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

<title>EXp1a</title>

<style>

#header

{

border:1px solid black;width:100%;height:60px;

}

#nav

{

text-align:center;border:1px solid black;width:100%;height:60px;

}

#aside

{

display:inline;float:left;width:20%;border:1pxsolidblack;height:500px;

}

#section

{

width:80%;height: 500px;border:1px solid black;margin-left:305px;text-align: center;

}

#footer

{border:1px solid black;width:100%;height:100px;text-align:center}

</style>

</head>

<body bgcolor="antiquewhite">

<div id="header">

<center>

<h1><header>WELCOME TO IKART'S ONLINESERVICE'S</header>

</h1>

</center></div>

<div id="nav"><nav><b>This is navigation section</b></nav></div>

<div id="aside"><aside> <details>

<summary>About Us</summary>

<ul>

<li>Limited License

<p>Limited Licence Refers to a licence to an intellectual property right, typically copyright, which does not grant all the rights which the ownerof the intellectual property possesses, but only some of those rights.</p>

</li>

<li>Copy Rights

<p>According to the US Copyright Office, copyright applies to original works of creative expression that are captured in a tangible form.

</p>

</li>

<li>Terms Policy

<ul> <li>Trademark or Copyright Infringement Cease and Desist Letter</li>

<li>Website Cookies Policy</li>

<li>Shipping Policy for the Website</li></ul>

</li>

</ul>

</details>

</aside></div>

<div id="section"><b><section>This is main Section</section></b></div>

<div id="footer"><b><footer >

<a href="https://www.amazon.in/gp/help/customer/display.html

?nodeId=200545940&ref\_=footer\_cou"> Conditions of Use &SALES</a>

</span>

<p>

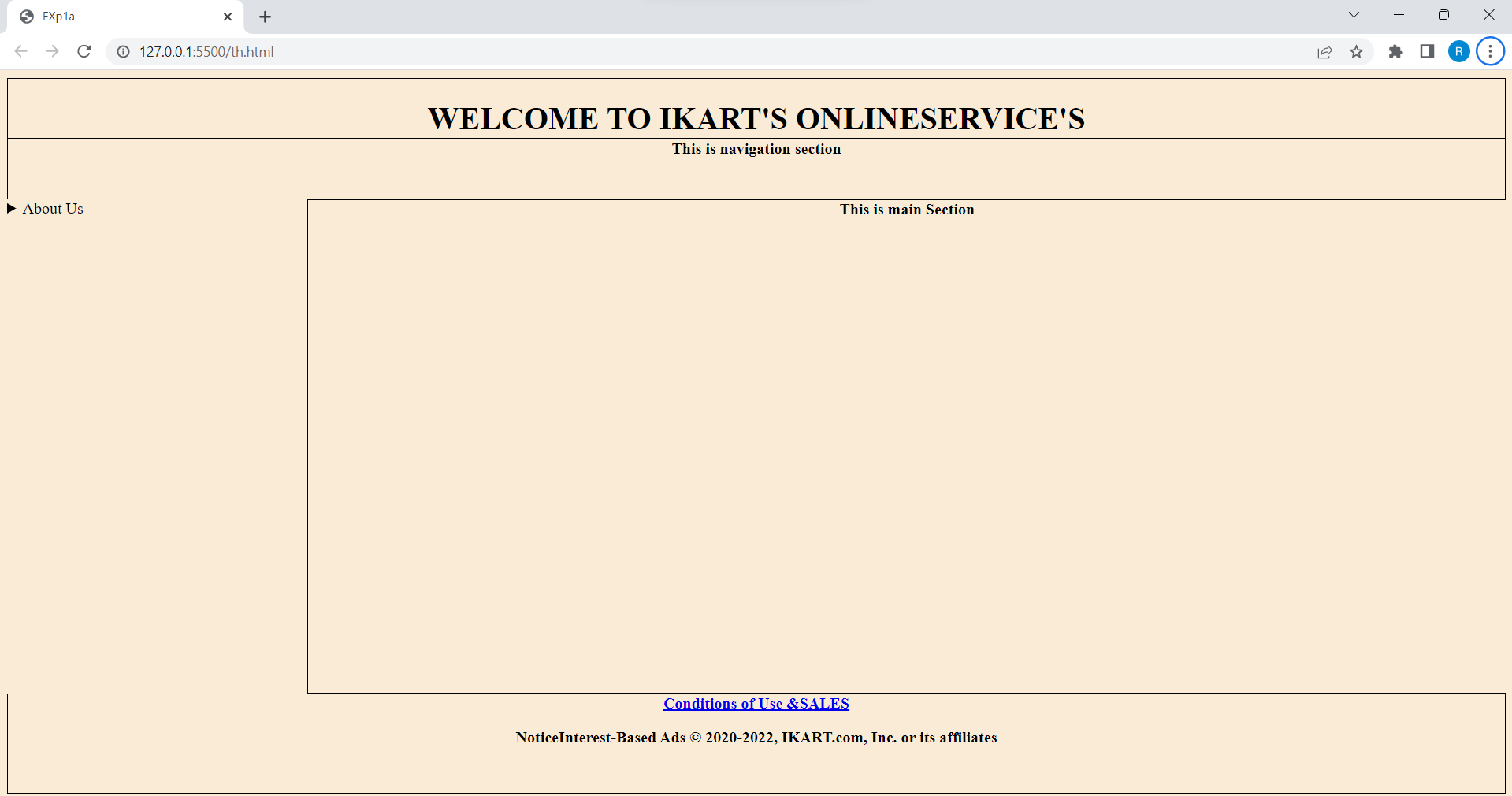
NoticeInterest-Based Ads &#169; 2020-2022, IKART.com, Inc. or its affiliates</p>

</footer></b></div>

</body>

</html>

**OUTPUT:**



**1.f Course Name: HTML5 - The Language Module Name: HTML5 Global Attributes Add the global attributes such as contenteditable, spellcheck, id etc. to enhance the Signup Page functionality of IEKart's Shopping application.**

**Program:**

<!DOCTYPE html>

<html >

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

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

<title>EXp1a</title>

<style>

#header{

width:100%;height:100px;background:black;

}

#header h1{

color:white;

}

#nav{

text-align:right;width:100%;height:60px;

}

#nav span{

background-color:white; border:1px solid black; padding:5px;

border-radius: 4px;

}

#nav span a{

text-decoration: none;

}

#aside{

float:left;width:400px;height:700px;

}

#section{

width:80%;height: 600px;margin-left:305px;text-align: center;

}

#footer{

width:100%;height:100px;text-align:center

}

</style>

</head>

<body bgcolor="antiquewhite">

<center>

<div id="header">

<center><h1><header>WELCOME TO IKART'S ONLINE SERVICE'S</header></h1></center>

<div id="nav">

<nav><span>Track </span><span><a href="sign.html">Sign UP</a></span>

<span><a href="login.html">Login</a></span></nav></div></div>

<div class="page">

<form align="center">

<br>

<label>First Name:</label>

<input type="text"placeholder="Enter Your First-Name" spellcheck="true">

<br><br>

<label>Last Name:</label>

<input type="text"placeholder="Enter Your Last-Name" spellcheck="true">

<br><br>

<label>Date of birth:</label><input type="date">

<br><br>

<label>Email:</label><input type="email" placeholder="Enter your mail Id" spellcheck="true">

<br><br>

<label>Username:</label>

<input type="text" pattern="[A-Za-z] UniqueUserName"spellcheck="true">

<br><br>

<label>Password:</label><input type="password" ><br><br>

<label>Confirm Password:</label><input type="password">

<br><br>

<tr rowspan="3">

<td>

<label>Address :<br></label><textarea rows="3" cols="30" spellcheck="true"></textarea>

<br><br>

<tr>

<td> <input type="checkbox" name="declare"></td>

<td><label contenteditable="true">I here by declare that the above facts are true in future if it is proved false I have no

<br> objection if my sanctioned

pension is cancelled more over I shall return all the pension amount <br> which I obtained from the

Government by my false representation.</label> <br></td></tr>

<br><button type="submit">Signup</button>

</form>

</div>

</center>

</body>

</html>

**OUTPUT:**

Graphical user interface

Description automatically generated

**2.a Course Name: HTML5 - The Language Module Name: Creating Table Elements, Table Elements : Colspan/Rowspan Attributes, border, cellspacing, cellpadding attributes Enhance the details page of IEKart's Shopping application by adding a table element to display the available mobile/any inventories.**

**Program:**

<html>

<head>

<style>

h1{

text-shadow:3px 4px 5px #003860; text-decoration: underline;

}

#header{

width:100%;height:120px;background:black;

}

#header h1{

color:white;

}

#nav{

text-align:right;width:100%;height:60px;

}

#nav span{

background-color:white; border:1px solid black; padding:5px;border-radius: 4px;

}

#nav span a{

text-decoration: none;

}

</style>

</head>

<body>

<div id="header">

<center>

<h1><header>WELCOME TO IKART'S ONLINE SERVICE'S</header></h1>

</center>

<div id="nav">

<nav>

<span>Track </span><span><a href="sign.html">Sign UP</a></span>

<span><a href="login.html">Login</a></span>

</nav>

</div>

</div>

<center><h1>Inventory Details</h1></center>

<table cellspacing="30px" cellpadding="30px">

<tr>

<td><a href="https://www.amazon.in/mobile- phones/b/?ie=UTF8&node=1389401031&ref\_=nav\_cs\_mobiles"><img src="mobiles.jpg" alt="Reload" width="280px" height="200px"></a></td>

<td><a href="https://www.amazon.in/b/ref=AF\_WIN\_bub\_w\_cml\_t\_4?pf\_rd\_r=S2H2VPJFXQE7EWRH8Q5G &pf\_rd\_p=719b0698-0dc6-48a3-aa6c- 219cc1c7a9ba&pf\_rd\_m=A1VBAL9TL5WCBF&pf\_rd\_s=merchandised-search-

2&pf\_rd\_t=&pf\_rd\_i=1389401031&node=21541572031"><img src="earphones.jpg" alt="Reload" width="280px" height="200px"></a>

</td>

<td>

<a href="https://www.amazon.in/s?k=headphones&rh=n%3A1389401031&ref=nb\_sb\_noss"><img src="headphones.jpg" alt="Reload" width="280px" height="200px"></a>

</td>

<td>

<a href="https://www.amazon.in/s?k=ear+buds&rh=n%3A1389401031&ref=nb\_sb\_noss"><img src="earbuds.jpg" alt="Reload" width="280px" height="200px"></a>

</td>

</tr>

<tr>

<td> <a href="https://www.amazon.in/b/?\_encoding=UTF8&node=1388977031&ref\_=sv\_top\_elec\_mega\_5 "><img src="camera.jpg" alt="Reload" width="280px" height="200px"></a>

</td>

<td>

<a href="https://www.amazon.in/s?k=grocery&ref=nb\_sb\_noss"><img src="groc.jpg" alt="Reload" width="280px" height="200px"></a>

</td>

<td><a href="https://www.amazon.in/Books/b/?ie=UTF8&node=976389031&ref\_=nav\_cs\_books"><img src="books.jpg" alt="Reload" width="280px" height="200px"></a>

</td>

<td>

<a

href="https://www.amazon.in/b?node=1983518031&pf\_rd\_r=XB9M6FNQD8XGHP0D56K1&pf\_rd\_p

=d66c2cf9-86d6-4691-8f1c-4e26ecf8278a&pd\_rd\_r=40360906-074c-4ad8-87ce- 425d3645e7ea&pd\_rd\_w=YhuEz&pd\_rd\_wg=yOBl7&ref\_=pd\_gw\_unk"><img src="shoes.jpg" alt="Reload" width="280px" height="200px"></a>

</td>

</tr>

<tr>

<td>

<a href="https://www.amazon.in/b/?\_encoding=UTF8&node=7459781031&ref\_=sv\_top\_ap\_mega\_2">

<img src="cloths.jpg" alt="Reload" width="280px" height="200px"></a>

</td>

<td> <a href="https://www.amazon.in/s?k=jewelry&rh=n%3A1983518031&ref=nb\_sb\_noss"><img src="jwel.jpg" alt="Reload" width="280px" height="200px"></a>

</td>

<td><a href="https://www.amazon.in/electronics/b/?ie=UTF8&node=976419031&ref\_=nav\_cs\_electronics"

><img src="elect.jpg" alt="Reload" width="280px" height="200px"></a>

</td>

<td>

<a href="https://www.amazon.in/s/ref=mega\_elec\_s23\_3\_4\_1\_1?rh=i%3Aelectronics

%2Cn%3A719857 0031&ie=UTF8&bbn=976419031"><img src="tv.webp" alt="Reload" width="280px" height="200px"></a>

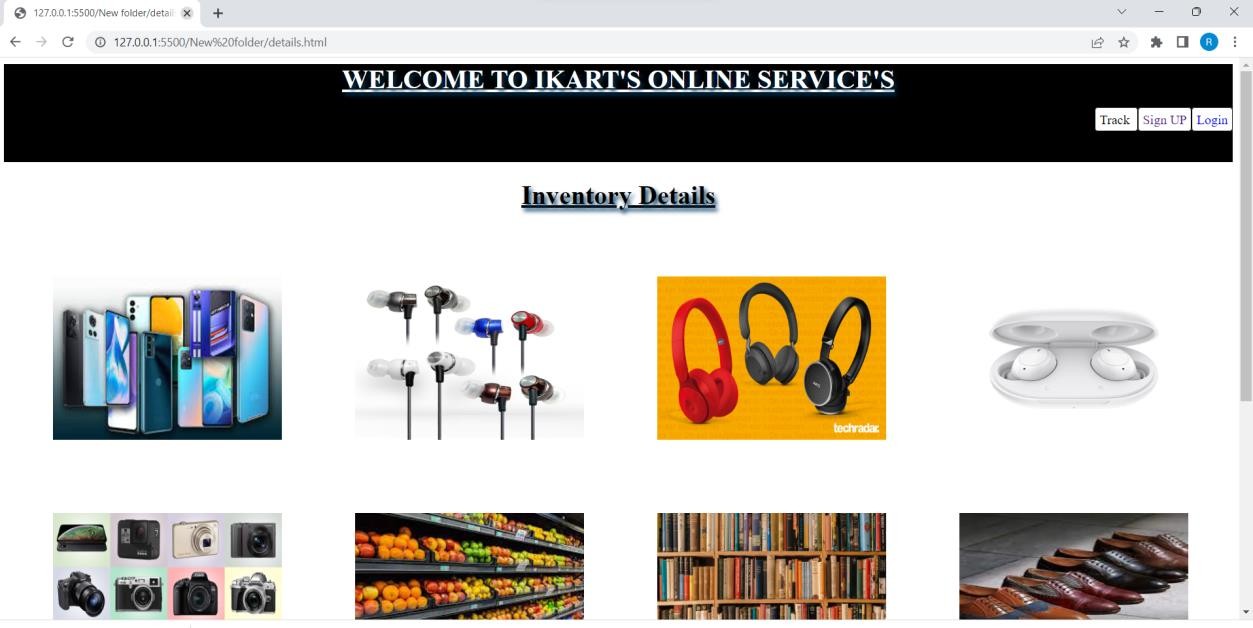
</td>

</tr>

</body>

</html>

**OUTPUT:**



**2.b Course Name: HTML5 - The Language Module Name: Creating Form Elements, Color and Date Pickers, Select and Datalist Elements Using the form elements create Signup page for IEKart's Shopping application.**

**Program:**

<!DOCTYPE html>

<html >

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

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

<title>EXp1a</title>

<style>

#header{

width:100%;height:100px;background:black;

}

#header h1{

color:white;

}

#nav{

text-align:right;width:100%;height:60px;

}

#nav span{

background-color:white; border:1px solid black; padding:5px;

border-radius: 4px;

}

#nav span a{

text-decoration: none;

}

#aside{

float:left;width:400px;height:700px;

}

#section{

width:80%;height: 600px;margin-left:305px;text-align: center;

}

#footer{

width:100%;height:100px;text-align:center

}

</style>

</head>

<body bgcolor="antiquewhite">

<center>

<div id="header">

<center><h1><header>WELCOME TO IKART'S ONLINE SERVICE'S</header></h1></center>

<div id="nav">

<nav><span>Track </span><span><a href="sign.html">Sign UP</a></span>

<span><a href="login.html">Login</a></span></nav></div></div>

<div class="page">

<form align="center">

<br>

<label>First Name:</label>

<input type="text"placeholder="Enter Your First-Name" spellcheck="true">

<br><br>

<label>Last Name:</label>

<input type="text"placeholder="Enter Your Last-Name" spellcheck="true">

<br><br>

<label>Date of birth:</label><input type="date">

<br><br>

<label>Email:</label><input type="email" placeholder="Enter your mail Id" spellcheck="true">

<br><br>

<label>Username:</label>

<input type="text" pattern="[A-Za-z] UniqueUserName"spellcheck="true">

<br><br>

<label>Password:</label><input type="password" ><br><br>

<label>Confirm Password:</label><input type="password">

<br><br>

<tr rowspan="3">

<td>

<label>Address :<br></label><textarea rows="3" cols="30" spellcheck="true"></textarea>

<br><br>

<tr>

<td> <input type="checkbox" name="declare"></td>

<td><label contenteditable="true">I here by declare that the above facts are true in future if it is proved false I have no

<br> objection if my sanctioned

pension is cancelled more over I shall return all the pension amount <br> which I obtained from the

Government by my false representation.</label> <br></td></tr>

<br><button type="submit">Signup</button>

</form>

</div>

</center>

</body>

</html>

**OUTPUT:**

Graphical user interface

Description automatically generated

**2.c Course Name: HTML5 - The Language Module Name: Input Elements - Attributes Enhance Signup page functionality of IEKart's Shopping application by adding attributes to input elements.**

**Program:**

<!DOCTYPE html>

<html >

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

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

<title>EXp1a</title>

<style>

#header{

width:100%;height:100px;background:black;

}

#header h1{

color:white;

}

#nav{

text-align:right;width:100%;height:60px;

}

#nav span{

background-color:white; border:1px solid black; padding:5px;

border-radius: 4px;

}

#nav span a{

text-decoration: none;

}

#aside{

float:left;width:400px;height:700px;

}

#section{

width:80%;height: 600px;margin-left:305px;text-align: center;

}

#footer{

width:100%;height:100px;text-align:center

}

</style>

</head>

<body bgcolor="antiquewhite">

<center>

<div id="header">

<center><h1><header>WELCOME TO IKART'S ONLINE SERVICE'S</header></h1></center>

<div id="nav">

<nav><span>Track </span><span><a href="sign.html">Sign UP</a></span>

<span><a href="login.html">Login</a></span></nav></div></div>

<div class="page">

<form align="center">

<br>

<label>First Name:</label>

<input type="text"placeholder="Enter Your First-Name" spellcheck="true">

<br><br>

<label>Last Name:</label>

<input type="text"placeholder="Enter Your Last-Name" spellcheck="true">

<br><br>

<label>Date of birth:</label><input type="date">

<br><br>

<label>Email:</label><input type="email" placeholder="Enter your mail Id" spellcheck="true">

<br><br>

<label>Username:</label>

<input type="text" pattern="[A-Za-z] UniqueUserName"spellcheck="true">

<br><br>

<label>Password:</label><input type="password" ><br><br>

<label>Confirm Password:</label><input type="password">

<br><br>

<tr rowspan="3">

<td>

<label>Address :<br></label><textarea rows="3" cols="30" spellcheck="true"></textarea>

<br><br>

<tr>

<td> <input type="checkbox" name="declare"></td>

<td><label contenteditable="true">I here by declare that the above facts are true in future if it is proved false I have no

<br> objection if my sanctioned

pension is cancelled more over I shall return all the pension amount <br> which I obtained from the

Government by my false representation.</label> <br></td></tr>

<br><button type="submit">Signup</button>

</form>

</div>

</center>

</body>

</html>

**OUTPUT:**

Graphical user interface

Description automatically generated

**2.d Course Name: HTML5 - The Language Module Name: Media, Iframe Add media content in a frame using audio, video, iframe elements to the Home page of IEKart's Shopping application.**

**PROGRAM:**

<!DOCTYPE html>

<html >

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

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

<title>EXp1a</title>

<style>

#header

{

border:1px solid black;width:100%;height:60px;

}

#nav

{

text-align:center;border:1px solid black;width:100%;height:60px;

}

#aside

{

display:inline;float:left;width:20%;border:1pxsolidblack;height:500px;

}

#section

{

width:80%;height: 500px;border:1px solid black;margin-left:305px;text-align: center;

}

#footer

{border:1px solid black;width:100%;height:100px;text-align:center}

</style>

</head>

<body bgcolor="antiquewhite">

<div id="header">

<center>

<h1><header>WELCOME TO IKART'S ONLINESERVICE'S</header>

</h1>

</center></div>

<div id="nav"><nav><b>This is navigation section</b></nav></div>

<div id="aside"><aside> <details>

<summary>About Us</summary>

<ul>

<li>Limited License

<p>Limited Licence Refers to a licence to an intellectual property right, typically copyright, which does not grant all the rights which the ownerof the intellectual property possesses, but only some of those rights.</p>

</li>

<li>Copy Rights

<p>According to the US Copyright Office, copyright applies to original works of creative expression that are captured in a tangible form.

</p>

</li>

<li>Terms Policy

<ul> <li>Trademark or Copyright Infringement Cease and Desist Letter</li>

<li>Website Cookies Policy</li>

<li>Shipping Policy for the Website</li></ul>

</li>

</ul>

</details>

</aside></div>

<div id="section"><b><section><iframe src="https://aec.edu.in" style="height:220px" title="W3Schools Free Online Web Tutorials">

</iframe>

<video src="movie.mp4" width="300px" controls></video>

</section></b></div>

<div id="footer"><b><footer >

<a href="https://www.amazon.in/gp/help/customer/display.html

?nodeId=200545940&ref\_=footer\_cou"> Conditions of Use &SALES</a>

</span>

<p>

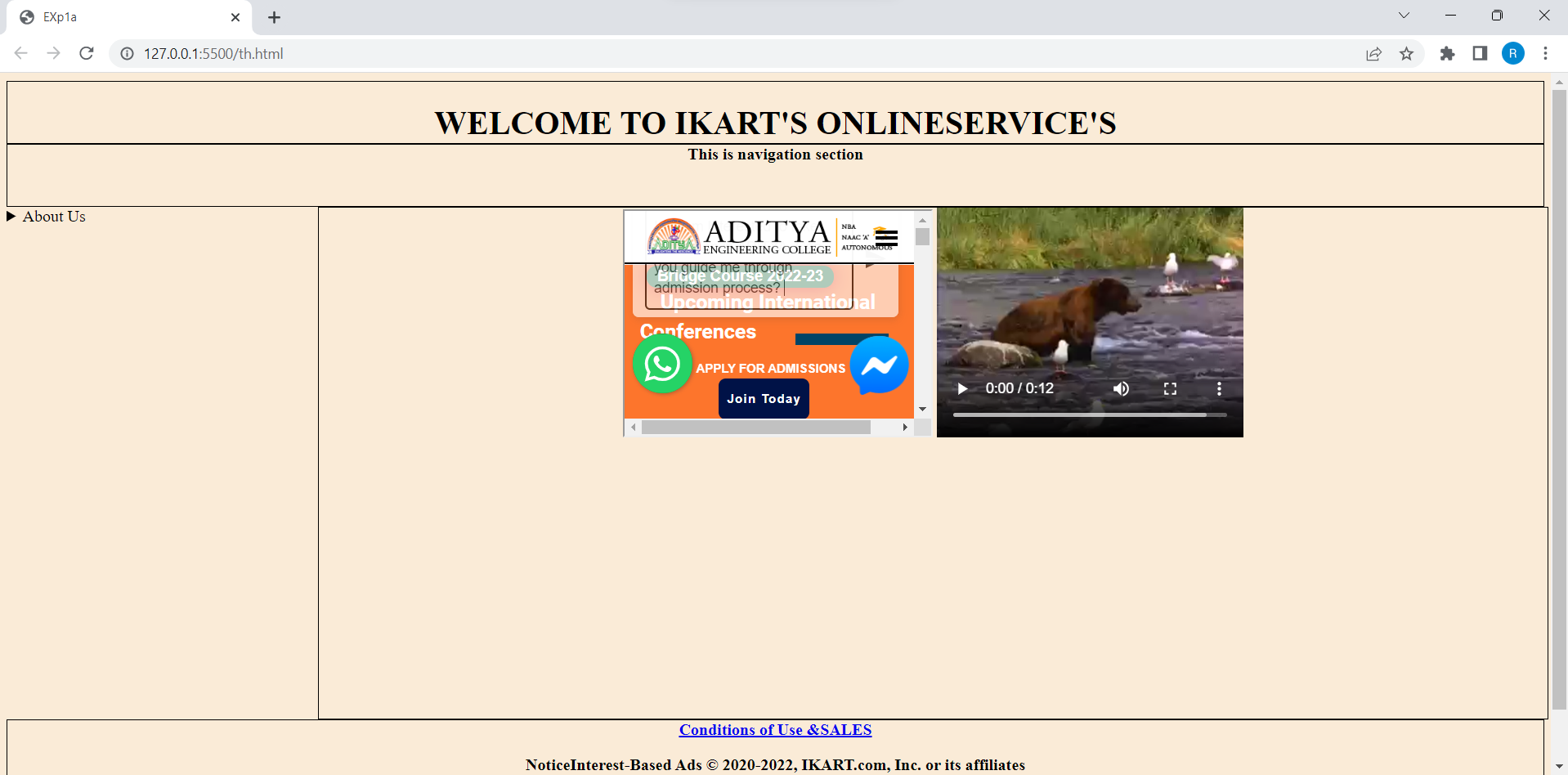
NoticeInterest-Based Ads &#169; 2020-2022, IKART.com, Inc. or its affiliates</p>

</footer></b></div>

</body>

</html>

**OUTPUT:**



**3.a Course Name: Javascript Module Name: Type of Identifiers 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).**

**Program:**

<!DOCTYPE html>

<html lang="en">

<head>

<title>Movie</title>

<style>

#maincontent { height: 200px; width: 800px;

border: 1px solid #CEE2FA; text-align: left;

color: #a84e81; font-family: calibri; font-size: 20; padding: 5px;

}

#heading {

text-decoration: bold; text-align: center; margin-top: 80px; width: 800px;

border: 1px solid #CEE2FA; text-align: center;

color: #08438E;

background-color: #CEE2FA; font-family: calibri;

font-size: 20; padding: 5px;

}

</style>

</head>

<body>

<center>

<div id="heading">To find the Area Of Circle</div>

<div id="maincontent">

<p>Click below to enter the rdius of Circle</p>

<button onclick="fun()">click</button>

<br>

<p id="one"><br> </p>

<p id="two"><br></p>

<script> function fun(){

var a=prompt("enter the radius"); let b=a;

const pi=3.14;

document.getElementById("one").innerHTML=" The area Of Circle with radius as var :::

="+(pi\*a\*a);

document.getElementById("two").innerHTML=" The area Of Circle with radius as let :::

="+(pi\*b\*b);

}

</script>

</div>

</body>

</html>

**Output:**

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

**3.b Course Name: Javascript Module Name: Primitive and Non Primitive Data Types 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.**

**Program:**

<html>

<head>

<title>review</title>

</head>

<body bgcolor="antiquewhite">

<div id="container"style="border:0px solid blue;width:80%;margin:auto">

<div id="header" style="text-align:center;color:orange"><h1>Bimbisara Movie Details</h1></div>

</div>

<br><br>

<div id="body"style="text-align:center;">

<script>

var a=`"Bimbisara"`; var s=`"`;

const b=["kalyan ram","Samyuktha","Catharine","Srinivas Reddy"]; const l=["Telugu","Hindi","Tamil","kannada"];

var j=`"`;

for (let i = 0; i < b.length; i++) { s += b[i] + ",";

j+=l[i]+",";

}

s+=`"`; j+=`"`;

document.write("<h2>"+"Movie Name :"+a+"</h2>");

document.write("<h2>"+"Movie Cast:"+s+"</h2>"); document.write("<h2>"+"Languages:"+j+"</h2>"); document.write("<h2>"+"Rating: 4.5"+"</h2>");

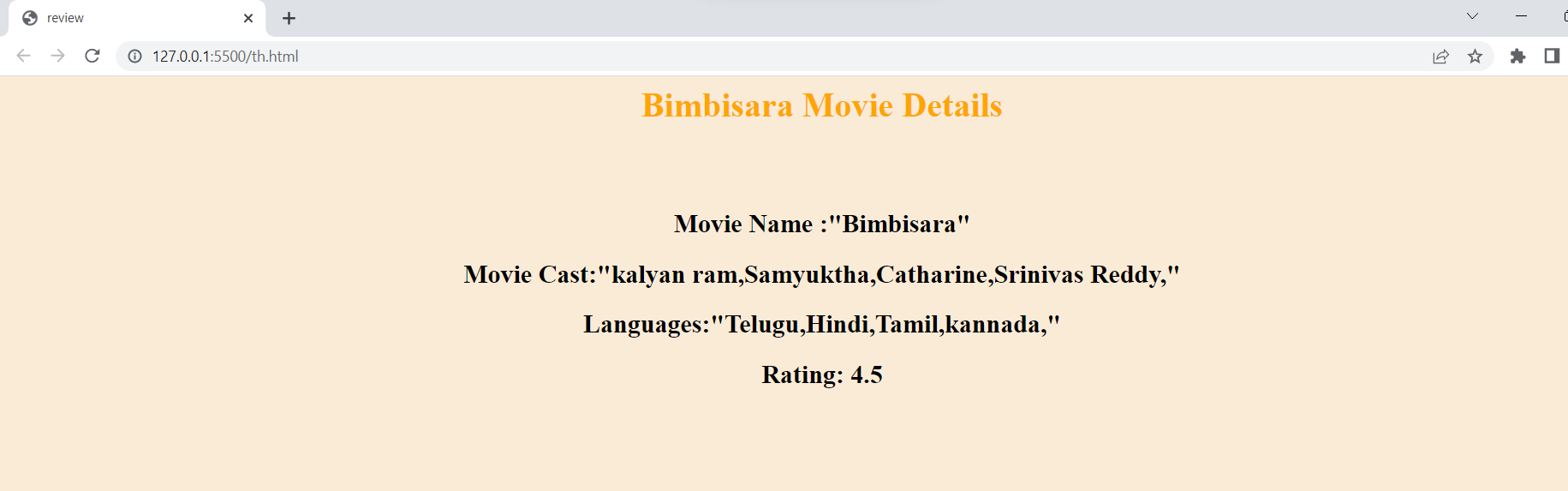
</script>

</div>

</body>

</html>

**Output:**



**3.c Course Name: Javascript Module Name: Operators and Types of Operators 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.**

**Program:**

<html>

<head>

<title>movie tickets</title>

</head>

<body>;

<div id="main" style="border:1px solid black;width:100%;height:auto">

<div id="header " style="margin-bottom:30px;margin-left:50px">

<center><h1>WELCOME TO IMAX THEATER</h1></center>

<H3>CLICK BELOW TO BOOK TICKETS</H3>

<button onclick="f1()">BOOK</button>

<h1 id="re"></h1>

<h1 id="id"></h1>

<button id="k" onclick="fun()">Pay Now</button>

<script>

var p=150; var d=0.1; function f1(){

var a=prompt("enter number of tickets"); var s=(p\*a);

s=s-(s\*d);

document.getElementById('re').innerHTML="THE AMOUNT YOU HAVE TO PAY:"; document.getElementById('id').innerHTML=s;

}

function fun()

{

alert("PAYMENT SUCCESSFULL"); alert("COLLECT TICKETS AT THEATRE");

}

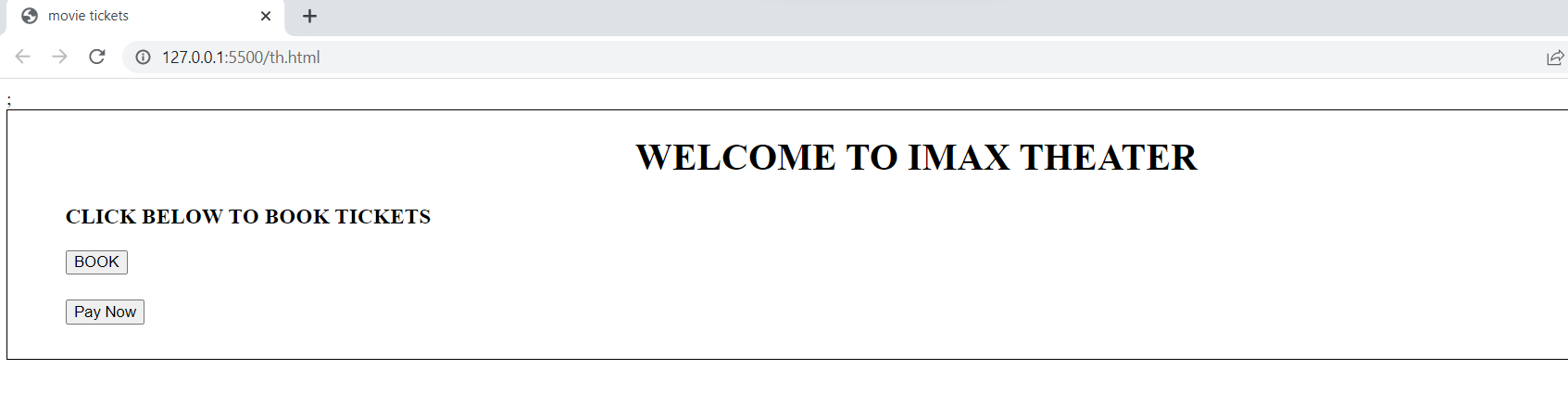
</script>

</div></div>

</body>

</html>

**Output:**



Graphical user interface, application, Word

Description automatically generated

**3.d Course Name: Javascript Module Name: Types of Statements, Non - Conditional Statements, Types of Conditional Statements, if Statements, switch Statements 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 seats to be booked are more than 2 but less than 6, based on the number of seats booked, do the following - Calculate total cost by applying discounts of 3, 5, 7, 9, 11 percent, and so on for customer 1,2,3,4 and 5. Try the code with different values for the number of seats.**

**PROGRAM:**

<html>

<head>

<title> exp3d </title>

<script>

function fun(){

var a=prompt("enter no of ticket's to book");

if(a<6)

{

document.getElementById("id").innerHTML="The Amount Needed To Book Tickets::";

document.getElementById("id1").innerHTML="Rs."+calculateCost(a);

document.getElementById("id2").innerHTML="Discount Offered Rs."+calculateDiscount(a);

}

else

{

document.getElementById("id").innerHTML="Upto 5 Tickets Only You Can Book in Online";

document.getElementById("id1").innerHTML="";

document.getElementById("id2").innerHTML="";

}

}

const p=150;

function calculateCost(a){

var i=1;

s=0;

j=0;

k=0.03

if(a>2 && a<6)

{

do

{

j=p-(p\*k);

s+=j;

j=0;

k+=0.02;

i+=1

}while(i<=a);

}

else if(a<=2)

{

s=p\*a;

}

else

s=0;

return s;

}

function calculateDiscount(a)

{

var g=calculateCost(a);

var z=a\*p;

return z-g;

}

</script>

</head>

<body>

<center><font color="white"><h1 style="background-color:orange">WELCOME TO TICKET BOOKING</h1></font></center>

<h2>Click below to Book Tickets</h2>

<br>

<input type="button" value="BOOK" onclick="fun()">

<center><h2 id="id"> </h2>

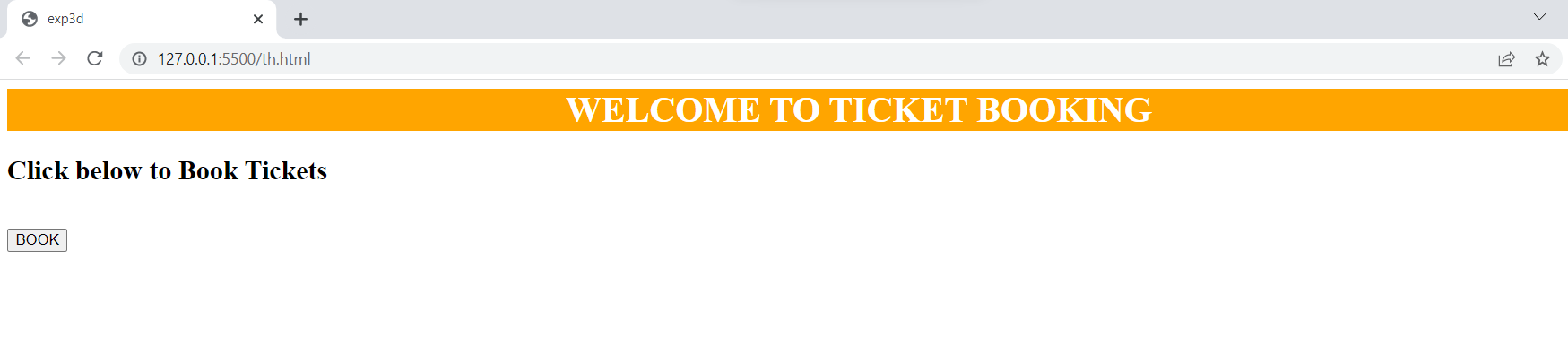
<h2 id="id1"> </h2>

<h2 id="id2"> </h2></center>

</body>

</html>

**OUTPUT:**



Graphical user interface, text, application

Description automatically generated

**3.e Course Name: Javascript Module Name: Types of Loops 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 seats to be booked are more than 2 but less than 6, based on the number of seats booked, do the following - Calculate total cost by applying a discount of 3, 5, 7, 9, 11 percent, and so on for customers till 5 respectively. Try the code with different values for the number of seats. Implement the problem statement using 'for' loop, 'while' loop and 'do-while' loop.**

**PROGRAM:**

<html>

<head>

<title> exp3d </title>

<script>

var a;

function fun2(){

a=prompt("enter no of ticket's to book");

if(a<6)

{

document.getElementById("id").innerHTML="The Amount Needed To Book Tickets::";

document.getElementById("id1").innerHTML="Rs."+DOCost(a);

document.getElementById("id2").innerHTML="Discount Offered Rs."+calculateDiscount(a);

}

else

{

document.getElementById("id").innerHTML="Upto 5 Tickets Only You Can Book in Online";

document.getElementById("id1").innerHTML="";

document.getElementById("id2").innerHTML="";

}

}

function fun1(){

a=prompt("enter no of ticket's to book");

if(a<6)

{

document.getElementById("id").innerHTML="The Amount Needed To Book Tickets::";

document.getElementById("id1").innerHTML="Rs."+whileCost(a);

document.getElementById("id2").innerHTML="Discount Offered Rs."+calculateDiscount(a);

}

else

{

document.getElementById("id").innerHTML="Upto 5 Tickets Only You Can Book in Online";

document.getElementById("id1").innerHTML="";

document.getElementById("id2").innerHTML="";

}

}

function fun(){

a=prompt("enter no of ticket's to book");

if(a<6)

{

document.getElementById("id").innerHTML="The Amount Needed To Book Tickets::";

document.getElementById("id1").innerHTML="Rs."+forCost(a);

document.getElementById("id2").innerHTML="Discount Offered Rs."+calculateDiscount(a);

}

else

{

document.getElementById("id").innerHTML="Upto 5 Tickets Only You Can Book in Online";

document.getElementById("id1").innerHTML="";

document.getElementById("id2").innerHTML="";

}

}

const p=150;

function DOCost(a){

var i=1;

s=0;

j=0;

k=0.03

if(a>2 && a<6)

{

do

{

j=p-(p\*k);

s+=j;

j=0;

k+=0.02;

i+=1

}while(i<=a);

}

else if(a<=2)

{

s=p\*a;

}

else

s=0;

return s;

}

function whileCost(a){

var i=1;

s=0;

j=0;

k=0.03

if(a>2 && a<6)

{

while(i<=a)

{

j=p-(p\*k);

s+=j;

j=0;

k+=0.02;

i+=1

}

}

else if(a<=2)

{

s=p\*a;

}

else

s=0;

return s;

}

function forCost(a){

s=0;

j=0;

k=0.03

if(a>2 && a<6)

{

for(let i=1;i<=a;i++)

{

j=p-(p\*k);

s+=j;

j=0;

k+=0.02;

}

}

else if(a<=2)

{

s=p\*a;

}

else

s=0;

return s;

}

function calculateDiscount(a)

{

var g=calculateCost(a);

var z=a\*p;

return z-g;

}

</script>

</head>

<body>

<center><font color="white"><h1 style="background-color:orange">WELCOME TO TICKET BOOKING</h1></font></center>

<h2>Click below to Book Tickets</h2>

<br>

<input type="button" value="FORLOOPBOOK" onclick="fun()">

<input type="button" value="WHILELOOPBOOK" onclick="fun1()">

<input type="button" value="DOWHILEBOOK" onclick="fun2()">

<center><h2 id="id"> </h2>

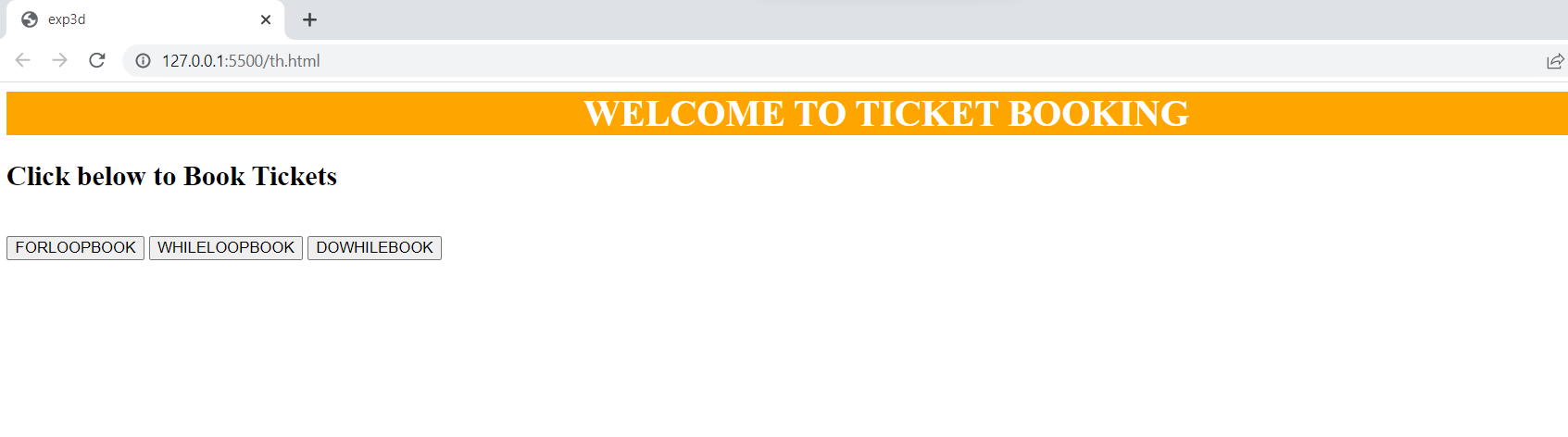
<h2 id="id1"> </h2>

<h2 id="id2"> </h2></center>

</body>

</html>

**OUTPUT:**



Graphical user interface, text, application, Word

Description automatically generated

**4.a Course Name: Javascript Module Name: Types of Functions, Declaring and Invoking Function, Arrow Function, Function Parameters, Nested Function, Built-in Functions, Variable Scope in Functions 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 seats to be booked are more than 2 but less than 6, based on the number of seats booked, do the following - Calculate total cost by applying a discount of 3, 5, 7, 9, 11 percent, and so on for customers till 5 respectively. Try the code with different values for the number of seats. Write the following custom functions to implement given requirements: i. calculateCost(seats): Calculate and display the total cost to be paid by the customer for the tickets they have bought. ii. calculateDiscount(seats): Calculate discount on the tickets bought by the customer. Implement using arrow functions.**

**PROGRAM:**

<html>

<head>

<title> exp4a </title>

<script>

function fun(){

var a=prompt("enter no of ticket's to book");

if(a<6)

{

document.getElementById("id").innerHTML="The Amount Needed To Book Tickets::";

document.getElementById("id1").innerHTML="Rs."+calculateCost(a);

document.getElementById("id2").innerHTML="Discount Offered Rs."+calculateDiscount(a);

}

else

{

document.getElementById("id").innerHTML="Upto 5 Tickets Only You Can Book in Online";

document.getElementById("id1").innerHTML="";

document.getElementById("id2").innerHTML="";

}

}

const p=150;

calculateCost=(a)=>{

var i=1;

s=0;

j=0;

k=0.03

if(a>2 && a<6)

{

do

{

j=p-(p\*k);

s+=j;

j=0;

k+=0.02;

i+=1

}while(i<=a);

}

else if(a<=2)

{

s=p\*a;

}

else

s=0;

return s;

}

CalculateDiscount=(a)=>

{

var g=calculateCost(a);

var z=a\*p;

return z-g;

}

</script>

</head>

<body>

<center><font color="white"><h1 style="background-color:orange">WELCOME TO TICKET BOOKING</h1></font></center>

<h2>Click below to Book Tickets</h2>

<br>

<input type="button" value="BOOK" onclick="fun()">

<center><h2 id="id"> </h2>

<h2 id="id1"> </h2>

<h2 id="id2"> </h2>

</center>

</body>

</html>

**OUTPUT:**

Graphical user interface, text, application

Description automatically generated

Graphical user interface, application, Word

Description automatically generated

**4.b Course Name: Javascript Module Name: Working With Classes, Creating and Inheriting Classes 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 and contact (iv) The constructor of the Employee to accept the name, age, role and contact where name and age are initialized through a call to super to invoke the base class constructor (v)Add a method getDetails() to display all the details of Employee.**

**PROGRAM:**

|  |  |
| --- | --- |
|  | <html> |
|  | <head> |
|  | <title>exp4b</title> |
|  | <script> |
|  | class Person |
|  | { |
|  |  |
|  | constructor(name,age) |
|  | { |
|  | this.name=name; |
|  | this.age=age; |
|  | } |
|  | det() |
|  | { |
|  | return "Name:"+this.name+"<br>"+"<br>"+"Age:"+this.age; |
|  | } |
|  |  |
|  | } |
|  | class Employee extends Person |
|  | { |
|  | constructor(name,age,role,contact) |
|  | { |
|  | super(name,age); |
|  | this.roll=role; |
|  | this.contact=contact; |
|  | } |
|  | getDetails() |
|  | { |
|  | return this.det()+"<br>"+"<br>"+"Role:"+this.roll+"<br>"+"<br>"+"Contact"+this.contact; |
|  | } |
|  |  |
|  | } |
|  | function fun() |
|  | { |
|  | let v=new Employee("abcds",20,"Software Developer","1015278878"); |
|  | document.getElementById("id1").innerHTML=v.getDetails(); |
|  | } |
|  | let personalDetails = { |
|  | name: "Stian Kirkeberg", |
|  | country: "Norway" |
|  | }; |
|  | let dynamicProperty = "age"; |
|  | personalDetails[dynamicProperty] = 45; |
|  |  |
|  | </script> |
|  | </head> |
|  | <body bgcolor="antiquewhite"> |
|  | <center><h1>We are Using Classes and Inheritance</h1></center> |
|  | <h3>Click below to get details </h3> |
|  | <center><h2 id="id1"> </h2></center> |
|  | <input type="button"value="Click" onclick="fun()"> |
|  |  |
|  |  |
|  | </body> |
|  | </html> |

**OUTPUT:**

**Graphical user interface, text, application

Description automatically generated**

**Graphical user interface, text

Description automatically generated**

**4.c Course Name: Javascript Module Name: In-built Events and Handlers 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 seats to be booked are more than 2 but less than 6, based on the number of seats booked, do the following - Calculate total cost by applying discounts of 3, 5, 7, 9, 11 percent, and so on for customer 1,2,3,4 and 5. Try the code with different values for the number of seats. Write the following custom functions to implement given requirements: (i) calculateCost(seats): Calculate and display the total cost to be paid by the customer for the tickets he has bought. (ii) calculateDiscount(seats): Calculate discount on the tickets bought by the customer. Invoke this function only when the user clicks on a link/button.**

**PROGRAM:**

<html>

<head>

<title> exp3d </title>

<script>

var a;

function fun(){

a=prompt("enter no of ticket's to book");

if(a<6)

{

document.getElementById("id").innerHTML="The Amount Needed To Book Tickets::";

document.getElementById("id1").innerHTML="Rs."+calculateCost(a);

document.getElementById("dis").style.display="inline";

}

else

{

document.getElementById("id").innerHTML="Upto 5 Tickets Only You Can Book in Online";

document.getElementById("id1").innerHTML="";

document.getElementById("id2").innerHTML="";

}

}

const p=150;

function calculateCost(a){

var i=1;

s=0;

j=0;

k=0.03

if(a>2 && a<6)

{

do

{

j=p-(p\*k);

s+=j;

j=0;

k+=0.02;

i+=1

}while(i<=a);

}

else if(a<=2)

{

s=p\*a;

}

else

s=0;

return s;

}

function fun2(){

document.getElementById("id2").innerHTML="Discount Offered Rs."+calculateDiscount(a);

}

function calculateDiscount(a)

{

var g=calculateCost(a);

var z=a\*p;

return z-g;

}

</script>

</head>

<body>

<center><font color="white"><h1 style="background-color:orange">WELCOME TO TICKET BOOKING</h1></font></center>

<h2>Click below to Book Tickets</h2>

<br>

<input type="button" value="BOOK" onclick="fun()">

<br> <br>

<input type="button" value="Discount" onclick="fun2()" style="display:none;" id="dis">

<center><h2 id="id"> </h2>

<h2 id="id1"> </h2>

<h2 id="id2"> </h2></center>

</body>

</html>

**OUTPUT:**

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application, Word, email

Description automatically generated

Graphical user interface, text

Description automatically generated

**4.d Course Name: Javascript Module Name: Working with Objects, Types of Objects, Creating Objects, Combining and cloning Objects using Spread operator, Destructuring Objects, Browser Object Model, Document Object Model 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 different color in the background.**

**PROGRAM:**

<html>

<head>

<title>exp5a</title>

</head>

<body bgcolor="yellow">

<style>a{

color:rgb(102, 7, 7);

font-size: 25px;}

</style>

<script>

var c=0;

function fun()

{

if(c==0)

{

document.body.style.backgroundColor = "blue";

document.getElementById("id1").innerHTML="FILL THE CONE";

document.getElementById("imag").src="coneemp.webp";

document.getElementById("link").innerHTML="FILL";

c=1;

}

else{

document.body.style.backgroundColor = "yellow";

document.getElementById("id1").innerHTML="EAT THE CONE";

document.getElementById("imag").src="conefii.webp";

document.getElementById("link").innerHTML="EAT";

c=0;

}

}

</script>

<center>

<h1 id="id1"> EAT THE CONE</h1>

<br><br>

<img src="conefii.webp" alt="Reload" height="300px" width="200px" id="imag"><br><br>

<a href="javascript:fun()" id="link">EAT</a>

</center>

</body>

</html>

**OUTPUT:**

**A picture containing diagram

Description automatically generated**

Graphical user interface, text

Description automatically generated

**5.a Course Name: Javascript Module Name: Creating Arrays, Destructuring Arrays, Accessing Arrays, Array Methods 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.**

**PROGRAM:**

<html>

<head>

<title>exp5a</title>

</head>

<body>

<script>

var a={"Movie":'Bimbisara','Starring':'KalyanRam',

"Language":'Telugu','Rating':'4.5'};

var b={ "Movie":"RRR","Starring":"RAMcharan&NTR",

"Language":"Telugu,Hindi",'Rating':"4.5"};

var c={ "Movie":"KGF","Starring":"Yash",

"Language":"English,Hindi,Knnada",'Rating':"5"};

var d={ "Movie":"ROBO","Starring":"RajiniKanth",

"Language":"Telugu,Hindi",'Rating':"5"};

const arr=[a,b,c,d]

document.write("<center>"+"<h1>"+"SOME HIT MOVIEDETAILS"+

"</h1>"+"</center>"+"<br>"+"<br>")

document.write("<center>")

for(let i=0;i<arr.length;i++)

{

document.write("<br>"+"Movie\_Name:"+arr[i].Movie+"<br>"

+"Starring:"+arr[i].Starring+"<br>");

document.write("Language:"+arr[i].Language+"<br>"

+"Rating:"+arr[i].Rating+"<br>");

document.write("<br>"+"<br")

}

document.write("</center>");

</script>

</body>

</html>

**Output:**

**A picture containing graphical user interface

Description automatically generated**

**5.b Course Name: Javascript Module Name: Introduction to Asynchronous Programming, Callbacks, Promises, Async and Await, Executing Network Requests using Fetch API 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 the count is 5 – use the setInterval method. (iii) Since setInterval is an async method, enclose the code in a Promise and handle the response generated in a success callback. (iv) The random value returned from the method every time can be used as a stock price and displayed on the console.**

**PROGRAM:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

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

<title>Exp\_\_5b</title>

</head>

<body>

<script>

let c=0;

const stock=setInterval(stokc,3000);

function stokc(){

var myPromise = new Promise(function (resolve, reject) {

setTimeout(function () {

var a=Math.floor(Math.random() \* 10);

resolve(a);

}, 3000);

});

myPromise.then(

function (data) {

console.log(data);

},

function (error) {

console.log(error);

}

);

c+=1;

if(c==5)

{

Stop();

}

}

function Stop() {

clearInterval(stock);

}

</script>

</body>

</html>

**OUTPUT:**

**Graphical user interface, text, application

Description automatically generated**

**5.c Course Name: Javascript Module Name: Creating Modules, Consuming Modules 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 will return "Unauthorized access". (iv) Create an validateUser.html file with textboxes username and password and a submit button. (v) Add a script tag in HTML to include validateUser.js file. (vi) Create an validateUser.js file which imports login module and invokes validate method of User class. (vii) On submit of the button in HTML the validate method of the User class should be invoked. (viii) Implement the validate method to send the username and password details entered by the user and capture the return value to display in the alert.**

**PROGRAM:**

<!DOCTYPE html>

<html>

<head>

<title>Exp\_5c</title>

<script type="module" src="index.js"></script>

</head>

<body>

<center><h1>WELCOME TO THE WWEB SERVICES</h1></center>

<center>

<h4>please fill the details carefullys</h4>

<form name="fn1" method="post">

<table>

<tr>

<td>Name:</td>

<td><input type="text" name="cname" required minlength="5"></td>

</tr>

<tr>

<td>Password:</td>

<td><input type="password" name="pass1" required minlength="5"></td>

</tr>

</table>

<br><br>

<center> <input type=button id="btn" value="Submit"></center>

</form>

</center>

</body>

</html>

**INDEX.JS**

import {User} from '/login.js';

var btn=document.getElementById("btn");

btn.addEventListener('click',val);

function val()

{

var x=document.forms["fn1"]["cname"].value;

var y=document.forms["fn1"]["pass1"].value;

const user=new User();

user.valid(x,y);

}

**LOGIN.JS**

export class User

{

name='ABCDE';

pass='98765';

valid(name,pass)

{

if(this.name==name && this.pass==pass)

alert("Login Successful");

else

alert("Unaurthorized Access");

}

}

**OUTPUT:**

**Graphical user interface, application

Description automatically generated**

Graphical user interface, text, application

Description automatically generated

**6.a Course Name: Node.js Module Name: How to use Node.js Verify how to execute different functions successfully in the Node.js platform.**

# Program:

var http = require('http'); http.createServer(function(req,res){

res.writeHead(200, {'Content-Type': 'text/html'});

res.write('THIS IS MY FIRST NODE.JS PROGRAM');

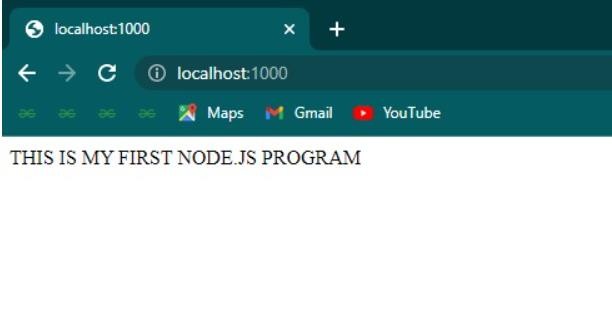
res.end();

}).listen(1000);

# Output:

Text

Description automatically generated



**6.b Course Name: Node.js Module Name: Create a web server in Node.js Write a program to show the workflow of JavaScript code executable by creating web server in Node.js.**

# Program:

var http = require('http'); http.createServer(function(req,res){

res.writeHead(200, {'Content-Type': 'text/html'});

res.write('THIS IS MY FIRST NODE.JS PROGRAM');

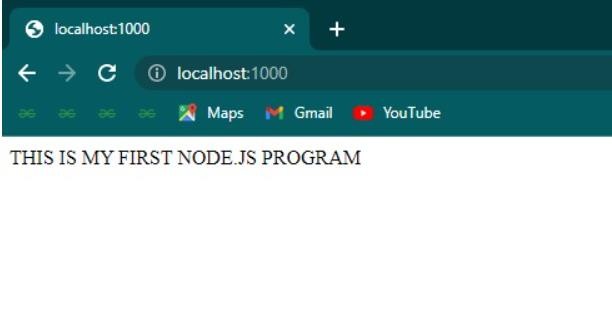
res.end();

}).listen(1000);

# Output:

Text

Description automatically generated



**6.c Course Name: Node.js Module Name: Modular programming in Node.js Write a Node.js module to show the workflow of Modularization of Node application.**

**PROGRAM:**

//myfile1.js

exports.data = function () { return "FRIENDS!!!!";};

**6C.JS**

var http = require('http'); var d = require('./myfile1');

http.createServer(function (req, res) { res.writeHead(200, {'Content-Type': 'text/html'}); res.write("WELCOME " + d.data());

res.end();

}).listen(8080);

**Output:**

# Text Description automatically generated

Graphical user interface, application

Description automatically generated

**6.d Course Name: Node.js Module Name: Restarting Node Application Write a program to show the workflow of restarting a Node application.**

# program:

const http = require("http");

http.createServer((req, res) => { res.write("I have created the server!"); res.end();

}).listen(8080);

console.log("Server started... Running on localhost:8080");

**output:**

Text

Description automatically generated

**MODIFIED PROGRAM:**

const http = require("http");

var server = http.createServer((req, res) => { res.write("I have modified the server!"); res.end();

});

server.listen(8080);

console.log("Server started... Running on localhost:8080");

# output:

Text

Description automatically generated

**6.e Course Name: Node.js Module Name: File Operations Create a text file src.txt and add the following data to it. Mongo, Express, Angular, Node.**

# Program:

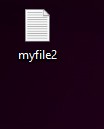
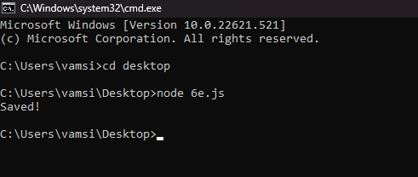
var fs = require('fs');

fs.appendFile('myfile2.txt', 'Mongo,Express,Angular,Node', function (err) { if (err) throw err;

console.log('Saved!');

});

# Graphical user interface, text, application Description automatically generatedOutput:



**7.a Course Name: Express.js Module Name: Defining a route, Handling Routes, Route Parameters, Query Parameters Implement routing for the AdventureTrails application by embedding the necessary code in the routes/route.js file.**

**App.js**

var express=require('express')

var route=require('./7aroute.js')

var app=express()

app.use('/',route)

app.listen(4000,function(){

console.log('server runnig....')

**Route.js**

var express=require('express')

var router=express.Router()

router.get('/hii',function(req,res){

res.status(200).send('HII everyone');

})

router.get('/bye',function(req,res){

res.status(200).json({message:"good to see you"});

})

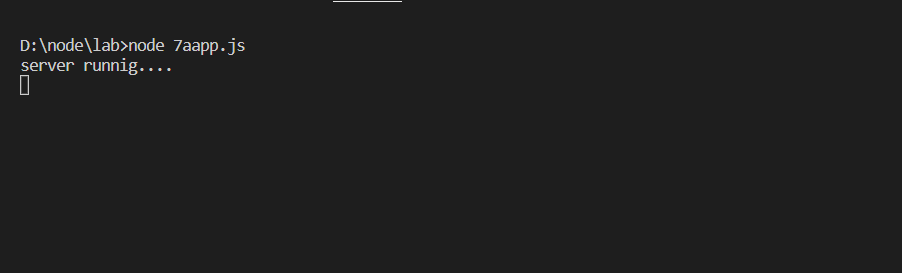
router.all('\*',function(req,res){

res.status(200).json({status:'fail',message:"Invalid"});

})

module.exports=router;

**OUTPUT:**



Graphical user interface, application

Description automatically generated with medium confidence

Graphical user interface, application, Word

Description automatically generated

**7.b Course Name: Express.js Module Name: How Middleware works, Chaining of Middlewares, Types of Middlewares In myNotes application: (i) we want to handle POST submissions. (ii) display customized error messages. (iii) perform logging.**

**App.js**

var express=require('express')

var route=require('./7aroute.js')

var app=express()

app.use('/',route)

app.listen(4000,function(){

console.log('server runnig....')

**Route.js**

var express=require('express')

var router=express.Router()

router.get('/hii',function(req,res){

res.status(200).send('HII everyone');

})

router.get('/bye',function(req,res){

res.status(200).json({message:"good to see you"});

})

router.all('\*',function(req,res){

res.status(200).json({status:'fail',message:"Invalid"});

})

module.exports=router;

**OUTPUT:**

A picture containing graphical user interface

Description automatically generated

Graphical user interface, application

Description automatically generated with medium confidence

Text

Description automatically generated

Graphical user interface, application, Word

Description automatically generated

**7.c Course Name: Express.js Module Name: Connecting to MongoDB with Mongoose, Validation Types and Defaults Write a Mongoose schema to connect with MongoDB.**

**PROGRAM:**  
var express=require('express')

var mongoose=require('mongoose');

const url='mongodb://0.0.0.0:27017/Hell';

mongoose.set('strictQuery', true);

mongoose.connect(url).then(function(){

console.log('connected....');

})

const myNotesSchema = new mongoose.Schema(

{

notesID: {

type: Number,

},

name: {

type: String,

},

data: {

type: String,

},

},

{

timestamps: {

createdAt: true,

updatedAt: true,

},

}

);

**OUTPUT:**

Shape

Description automatically generated with low confidence

**7.d Course Name: Express.js Module Name: Models Write a program to wrap the Schema into a Model object.**

**PROGRAM:**//to create model simply means creating collection in database

//syntax:: variable name= mongoose.model('coolectionname',schema);

//7D

var express=require('express')

var mongoose=require('mongoose');

const url='mongodb://0.0.0.0:27017/Hell';

mongoose.set('strictQuery', true);

mongoose.connect(url).then(function(){

console.log('connected....');

})

const myNotesSchema = new mongoose.Schema(

{

notesID: {

type: Number,

},

name: {

type: String,

},

data: {

type: String,

},

},

{

timestamps: {

createdAt: true,

updatedAt: true,

},

}

);

var bo=mongoose.model("12345",myNotesSchema)

console.log('model created......');

**OUTPUT**:



**8.a Course Name: Express.js Module Name: CRUD Operations Write a program to perform various CRUD (Create-Read-Update-Delete) operations using Mongoose library functions.**

**PROGRAM:**

//CRUD operations Create,Read,UPdate,Delete

var mongoose=require('mongoose');

const url='mongodb://0.0.0.0:27017/Hell';

mongoose.set('strictQuery', true);

mongoose.connect(url,{useNewUrlParser:true},{useUnifiedTopology:true}).then(function(){

console.log('connected....');})

var schema=mongoose.Schema({name:String,roll:Number});

var bo=mongoose.model("12345",schema);

//Create

var b1=new bo({name:"Rajesh",roll:897});

b1.save(function(err,res){

if(err)console.log(err)

console.log('Saved')

})

//update

bo.updateOne({roll:897},{$set:{name:"rambo"}},function(err,res){

console.log(res);

})

// //Delete

bo.deleteOne({name:"Rajesh"},function(er,res){

console.log(res)

console.log('deleted successfully');

})

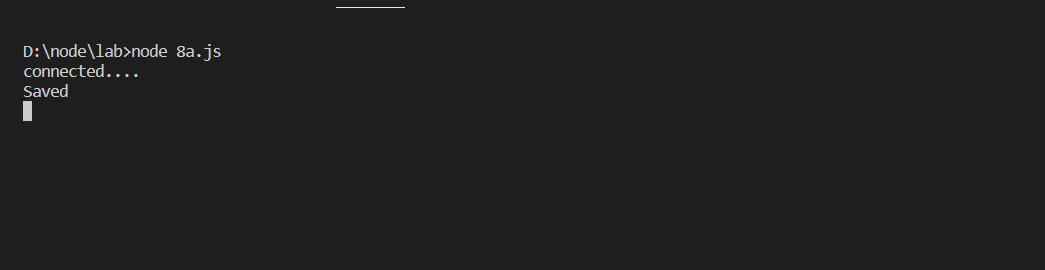
//Read

bo.find(function(err,res){

console.log(res);

})

OUTPUT:

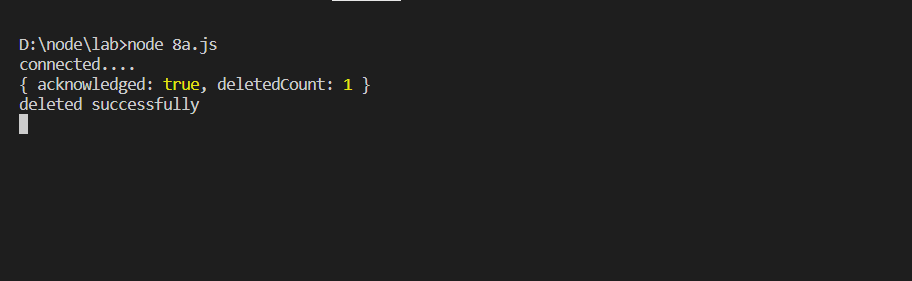


Text

Description automatically generated

Shape

Description automatically generated with medium confidence



**8.b Course Name: Express.js**

**Module Name: API Development**

**In the myNotes application, include APIs based on the requirements provided. (i) API should fetch the details of the notes based on a notesID which is provided in the URL. Test URL -http://localhost:3000/notes/7555 (ii) API should update the details based on the name which is provided in the URL and the data in the request body. Test URL http://localhost:3000/notes/Mathan Note: Only one document in the collection needs to be updated. (iii) API should delete the details based on the name which is provided in the URL. Test URL - http://localhost:3000/notes/Mathan Note: Only one document in the collection needs to be deleted**

**Program:**

**Routing.js**

const express = require('express');

const routing = express.Router();

const nodes = require('./myNotes');

routing.get('/notes', nodes.getNotes);

routing.get('/update/:id', nodes.updateNotes);

routing.get('/delete/:id', nodes.deleteNotes);

routing.get('/find/:id',nodes.findwithId);

routing.all('\*', nodes.invalid);

module.exports = routing;

**Mynotes.js**

const NotesModel = require('./myNotesSchema');

var a;

exports.getNotes = async (req, res) => {

try {

NotesModel.find( function(err, result){

if ( err )

console.log(err);

else

a=result;

});

res.status(200).json({

status: 'success',

results: a.length,

data: {

a,

},

});

}

catch (err) {

res.status(404).json({

status: 'fail',

message: 'loading fail',

});

}

};

exports.findwithId=async (req,res)=>{

try {

NotesModel.find({notesID:req.params.id}, function(err, result){

if ( err )

console.log(err);

else

a=result;

});

res.status(200).json({

status: 'success',

results: a.length,

data: {

a,

},

});

}

catch (err) {

res.status(404).json({

status: 'fail',

message: 'loading fail',

});

}

}

exports.updateNotes = async (req, res) => {

try {

NotesModel.updateOne({name:req.params.id},{$set:{data:"Data is just modified now successfully"}},function(er,result){

if(er)

console.log(er)

else a=result;

})

res.status(200).json({

status: 'success',

data: {

a,

},

});

} catch (err) {

res.status(404).json({

status: 'updatefailed',

message: err,

});

}

};

exports.deleteNotes = async (req, res) => {

const delDet = NotesModel.deleteOne({ name: req.params.id },function(err,result){if(err)console.log(err) ;else console.log(result)});

if (delDet.deletedCount === 0) {

res.status(404).json({

status: 'fail',

message: 'No notes available for this ID',

});

} else {

res.status(200).json({

status: 'success',

message: `Notes with ${req.params.id} ID deleted`,

});

}

};

exports.invalid = async (req, res) => {

console.log(params.req.id);

res.status(404).json({

status: 'fail',

message: 'Invalid path',

});

};

**Mynotesschema.js**

const mongoose = require('mongoose');

mongoose.connect('mongodb://0.0.0.0:27017/Hell', {

useNewUrlParser: true,

useUnifiedTopology: true,

}).then(() => console.log('DB connection successful!'));

//Schema

const myNotesSchema = new mongoose.Schema(

{

notesID: {

type: Number,

unique: true,

required: [true, 'Required field'],

},

name: {

type: String,

required: [true, 'Required field'],

},

data: {

type: String,

},

},{

timestamps: {

createdAt: true,

updatedAt: true,

},});

//Model

const NotesModel = mongoose.model('mynotes', myNotesSchema);

module.exports = NotesModel;

**App.js**

const express = require('express');

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

const route = require('./routing');

const app = express();

app.use(bodyparser.json());

app.use('/', route);

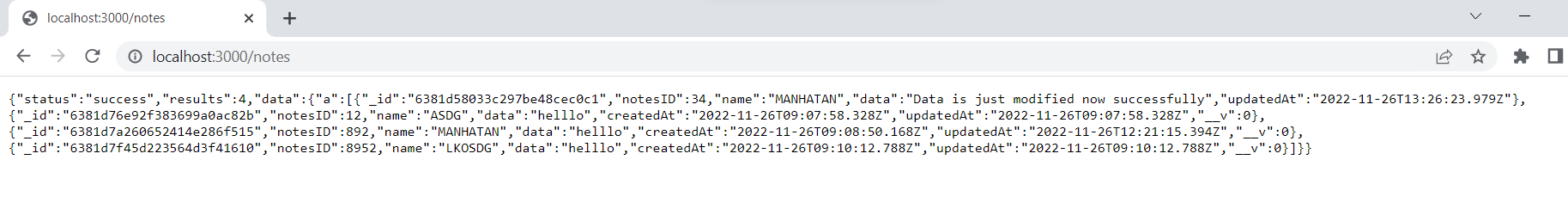
const port = process.env.PORT || 3000;

app.listen(port, () => {

console.log(`App running on port .....`);

});

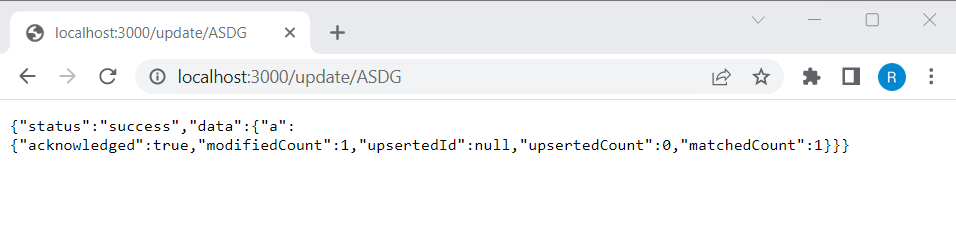
**Output:**

****

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

**8.c Course Name: Express.js Module Name: Why Session management, Cookies Write a program to explain session management using cookies.**

**PROGRAM:**  
var express=require('express')

var cookie=require('cookie-parser')

var app=express()

app.use(cookie())

app.get('/set',function(req,res){

res.cookie('name','age');

res.cookie('and','45');

res.status(200).send('cookie set');

})

app.get('/get',function(req,res){

res.status(200).send(req.cookies);

})

app.listen(4000,function(){

console.log('runingg......');

})

**OUTPUT:**

Table

Description automatically generated

Graphical user interface, application, Word

Description automatically generated

**8.d Course Name: Express.js Module Name: Sessions Write a program to explain session management using sessions.**

**PROGRAM:**

var express=require('express')

var session=require('express-session')

var app=express()

app.use(session({secret:'YOUR\_KEY',resave:true,saveUninitialized:true}))

app.get('/',function(req,res){

req.session.name="SEssion arl"

return res.send("Session set")

})

app.get('/session',function(req,res){

return res.send(req.session.name)

})

app.listen(3000,function(){

console.log("runnig....")

})

**OUTPUT:**

Application

Description automatically generated with low confidence

Table

Description automatically generated with low confidence

**8.e Course Name: Express.js Module Name: Why and What Security, Helmet Middleware Implement security features in myNotes application**

**PROGRAM:**

**.HTML**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

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

<title>Middle</title>

</head>

<body>

<iframe src="http://localhost:4000/ab" height="300" width="300"></iframe>

</body>

</html>

**.js**

var ex=require('express')

var hel=require('helmet')

var app=ex()

app.use(hel())

app.get('/ab',(req,res)=>{

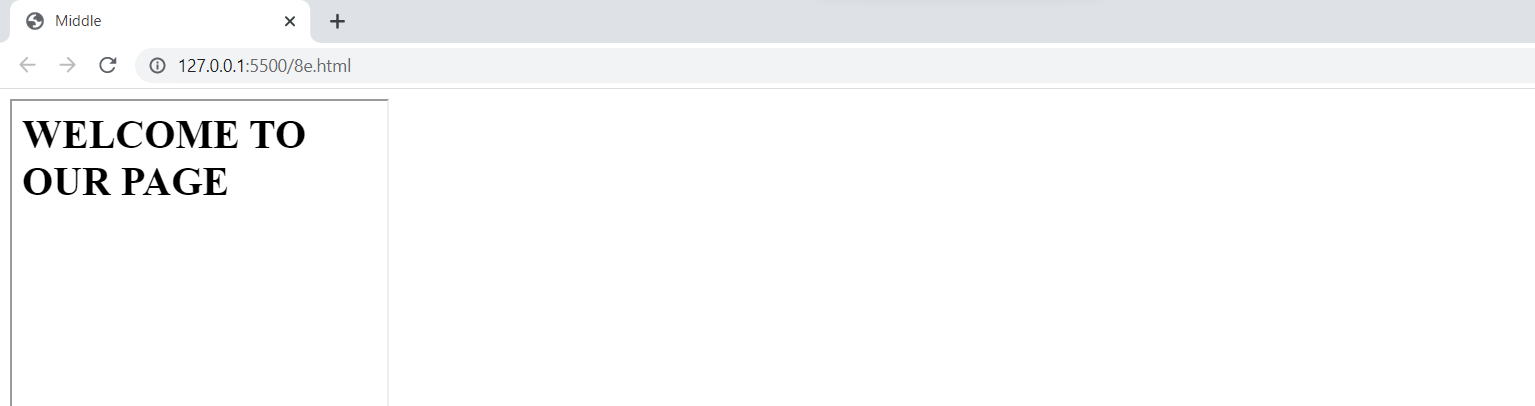
res.send('<h1>WELCOME TO OUR PAGE</h1>');

})

app.listen(4000,()=>{console.log('rer..')});

**OUTPUT:**

**WITHOUT USING HELMET:**



**WITH USING HELMET:**

Graphical user interface, application, Word

Description automatically generated

**9.a Course Name: Typescript Module Name: Basics of TypeScript 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.**

**Program:**

function price(a: string){ if(a=="GoldPlatinum ")

{

return 10000;

}

else if(a=="PinkGold")

{

return 12000;

}

else if(a=="SilverTitanium")

{

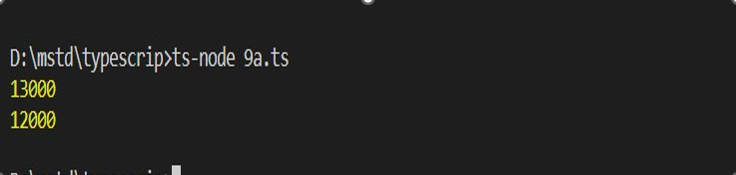
return 13000;

}

}

console.log(price("SilverTitanium")); console.log(price("PinkGold"));

**Output:**



**9.b Course Name: Typescript Module Name: Function 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.**

**Program:**

var manufacturers = [{ productId:121,id: 'Samsung', price: 150 },

{ productId:122,id: 'Microsoft', price: 200 },

{ productId:123,id: 'Apple', price: 400 },

{ productId:124,id: 'Micromax', price: 100 }

];

var i:number=0;

var getproductdetails=(product : number):string=>{

for(let i=0;i<manufacturers.length;i++)

{

if(manufacturers[i].productId==product)

{

break;

}

}

return "productID:"+manufacturers[i].productId+"\n ProductName:"+manufacturers[i].id+"\n Price: "+manufacturers[i].price;

};

console.log(getproductdetails(1234));

**Output:**



**9.c Course Name: Typescript Module Name: Parameter Types and Return Types Consider that developer needs to declare a function - getMobileByVendor which accepts string as input parameter and returns the list of mobiles.**

**Program:**

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;

}

else if (manufacturer === 'Apple') {

mobileList = ['Apple iPhone 5s', 'Apple iPhone 6s ', 'Apple iPhone 7']; return mobileList;

} else {

mobileList = ['Nokia 105', 'Nokia 230 Dual Sim']; return mobileList;

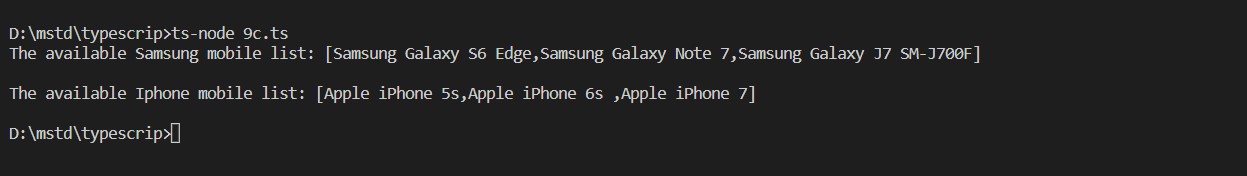
}

}

console.log('The available Samsung mobile list: [' + getMobileByManufacturer('Samsung')+']');

console.log('\nThe available Iphone mobile list: [' + getMobileByManufacturer('Apple')+"]");

**Output:**



**9.d Course Name: Typescript Module Name: Arrow Function 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 equal to 150 dollars then below mentioned code-snippet would fit into this requirement.**

**Program:**

var manufacturers = [{ id: 'Samsung', price: 150 },

{ id: 'Microsoft', price: 200 },

{ id: 'Apple', price: 400 },

{ id: 'Micromax', price: 100 }

];

console.log('Details of Manufacturer array are : ');

function myFunction() {

var test = manufacturers.filter((m) =>m.price >= 150);

for (var item of test) {

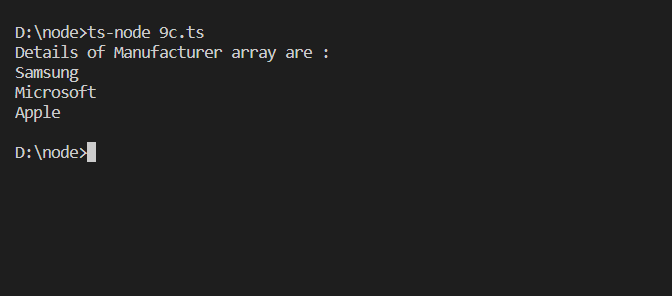
console.log(item.id);

}

}

myFunction();

**OUTPUT:**



**9.e Course Name: Typescript Module Name: Optional and Default Parameters Declare a function - getMobileByManufacturer with two parameters namely manufacturer and id, where manufacturer value should passed as Samsung and id parameter should be optional while invoking the function, if id is passed as 101 then this function should return Moto mobile list and if manufacturer parameter is either Samsung/Apple then this function should return respective mobile list and similar to make Samsung as default Manufacturer. Below mentioned code-snippet would fit into this requirement.**

**Program:**

function getMobileByManufacturer(manufacturer: string = 'Samsung', id?: number): string[]{

let mobileList: string[]; if (id) {

if (id === 101) {

mobileList = ['Moto G Play, 4th Gen', 'Moto Z Play with Style Mod']; return mobileList;

}}

category if (manufacturer === 'Samsung') {

mobileList = [' Samsung Galaxy S6 Edge', ' Samsung Galaxy Note 7', ' Samsung Galaxy J7 SM-J700F'];

return mobileList;

} else if (manufacturer === 'Apple') {

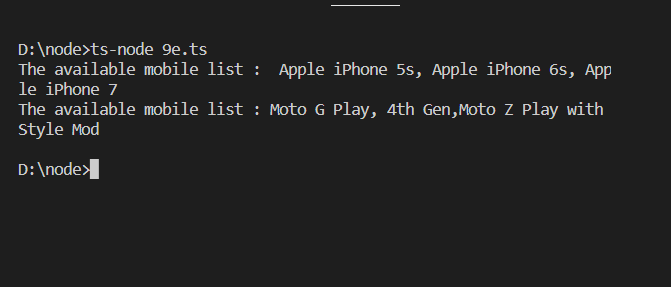
mobileList = [' Apple iPhone 5s', ' Apple iPhone 6s', ' Apple iPhone 7']; return mobileList;

} else {

mobileList = [' Nokia 105', ' Nokia 230 Dual Sim']; return mobileList;}}

console.log('The available mobile list : ' + getMobileByManufacturer('Apple')); console.log('The available mobile list : ' + getMobileByManufacturer(undefined, 101))

**Output:**



**10.a Course Name: Typescript Module Name: Rest Parameter Implement business logic for adding multiple Product values into a cart variable which is type of string array.**

**Program:**

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:

# 

**10.b Course Name: Typescript Module Name: Creating an Interface 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.**

# Program:

interface Product { productId: number ; productName: string ;

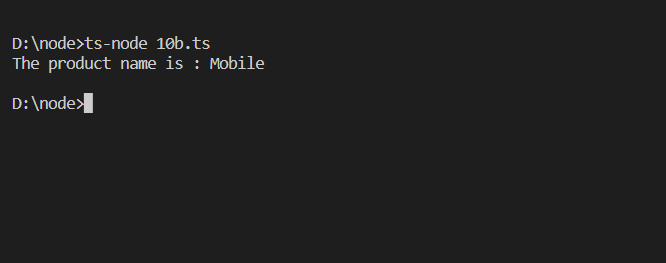
}

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:



**10.c Course Name: Typescript Module Name: Duck Typing 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.**

**Program:**

interface Product { productId: number; productName: string;

}

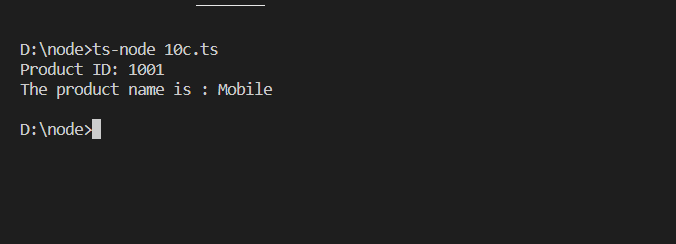
function getProductDetails(productobj: Product): string { return "Product ID: "+productobj.productId+'\nThe product name is : ' + productobj.productName;

}

const prodObject = {productId: 1001, productName: 'Mobile', productCategory: 'Gadget'};

const productDetails: string = getProductDetails(prodObject); console.log(productDetails);

# Output:



**10.d Course Name: Typescript Module Name: Function Types Declare an interface with function type and access its value.**

# Program:

function CreateCustomerID(name: string, id: number): string { return 'The customer id is ' + name + ' ' + id;

}

interface StringGenerator {

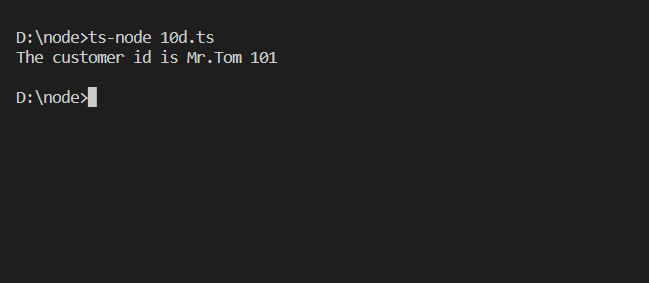
(chars: string, nums: number): string;

}

let idGenerator: StringGenerator; idGenerator = CreateCustomerID;

const customerId: string = idGenerator('Mr.Tom', 101); console.log(customerId);

# Output:



**11.a Course Name: Typescript Module Name: Extending Interfaces 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.**

**PROGRAM:**

interface Category {

categoryName: string;

}

interface Product {

productName: string;

productId: number;

}

interface ProductList extends Category, Product {

list: Array<string>;}

const productDetails: ProductList = {

categoryName: 'Gadget',

productName: 'Mobile',

productId: 1234,

list: ['Samsung', 'Motorola', 'LG']};

const listProduct = productDetails.list;

const pname: string = productDetails.productName;

console.log('Product Name is ' + pname);

console.log('Product List is ' + listProduct);

**Output:**



**11.b Course Name: Typescript Module Name: Classes Consider the Mobile Cart application, Create objects of the Product class and place them into the productlist array.**

**Program:**

class Product {

productId: number;

productName:string;

constructor(productId: number,productName:string) {

this.productId = productId;

this.productName=productName;

}

getDetails(): string {

return 'Product id is : ' + this.productId+'\n Product name is :'+this.productName;

}

}

const ar:Product[]=[];

let i:number=1;

for(i=1;i<=4;i++)

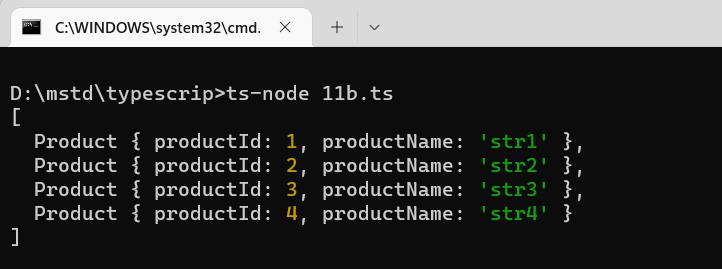
{

ar.push(new Product(i,"str"+i));

}

console.log(ar);

**Output:**

****

**11.c Course Name: Typescript Module Name: Constructor 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 <>".**

**Program:**

class Product

{

productId: number;

constructor(productId: number)

{

this.productId = productId;

}

getProductId(): string {

return 'Product id is : ' + this.productId;

}

}

const product: Product = new Product(1234);

console.log(product.getProductId());

**Output:**

**Text

Description automatically generated with medium confidence**

**11.d) Course Name: Typescript Module Name: Access Modifiers 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.**

**Program:**

class Product {

static productPrice = 150;

private productId: number;

public productName: string;

protected productCategory: string;

constructor(productId: number, productName:string , productCategory:string) {

this.productId = productId;

this.productName = productName;

this.productCategory = productCategory;

}

getProductId() {

console.log('The Product id is : ' + this.productId);

}

}

class Gadget extends Product {

getProduct(): void {

console.log('Product category is : ' + this.productCategory);

}

}

const g: Gadget = new Gadget(1234, 'Mobile', 'SmartPhone');

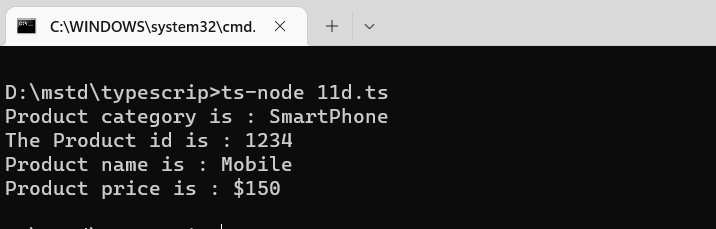
g.getProduct();

g.getProductId();

console.log('Product name is : ' + g.productName);

console.log('Product price is : $' + Product.productPrice);

**Output:**

****

**12.a Course Name: Typescript Module Name: Properties and Methods Create a Product class with 4 properties namely productId and methods to setProductId() and getProductId().**

**Program:**

class Product {

public productPrice = 150;

private productId: number=0;

public productName: string="Iphone";

public productCategory: string="Mobile";

set setProductId(productId:number){

this.productId=productId;

}

get getProductId():string{

return 'Product id :'+this.productId;

}

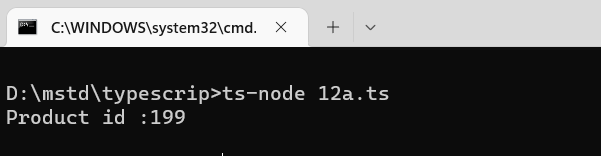
}

const ProductDetails:Product=new Product();

ProductDetails.setProductId=199;

console.log(ProductDetails.getProductId);

**Output:**

****

**12.b Course Name: Typescript Module Name: Creating and using Namespaces Create a namespace called ProductUtility and place the Product class definition in it. Import the Product class inside productlist file and use it.**

**Utility.ts**

namespace ProductUtility{

export class Product1 {

static productPrice: string;

productId: number;

constructor(productId: number) {

this.productId = productId;

}

getProductId(): string {

return 'Product id is : ' + this.productId;

}}}

**12b.ts**

///<reference path="./Utility.ts"/>

import prod=ProductUtility;

let pe=new prod.Product1(8);

interface ProductList {

list: Array<string>;

}

const productDetails: ProductList= {

list: ['Samsung', 'Motorola', 'LG']

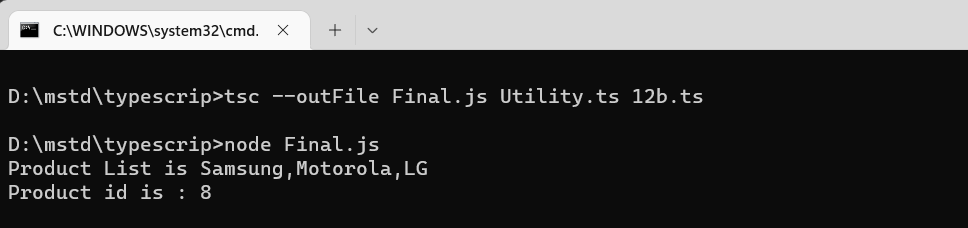
};

const listProduct = productDetails.list;

console.log('Product List is ' + listProduct);

console.log(pe.getProductId())

**Output:**

****

**12.c Course Name: Typescript Module Name: Creating and using Modules 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.**

**Module.ts**

class Utility {

CalculateAmount(price: number, quantity: number): number {

return price \* quantity;

}

}

export{Utility};

**12c.ts**

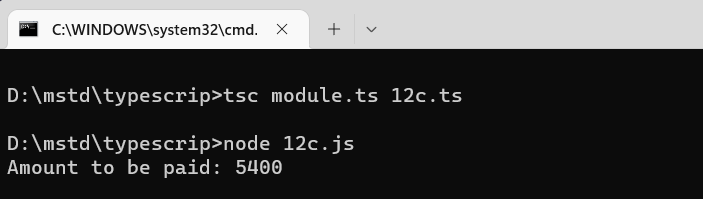
import { Utility as mainUtility} from "./module";

const util = new mainUtility();

const price = util.CalculateAmount(1350, 4);

console.log(`Amount to be paid: ${price}`);

**Output:**

****

**12.d Course Name: Typescript Module Name: What is Generics, What are Type Parameters, Generic Functions, Generic Constraints Create a generic array and function to sort numbers as well as string values.**

**Program:**

function sortArray<T>(arg: Array<T>): Array<T> {

return arg.sort();

}

const numbers: Array<number> = [1,5,2,8,4];

const strings: Array<string> = ['footwear', 'dress', 'cds', 'toys'];

console.log('\nBefore Sorting\n');

console.log(numbers);

const idList = sortArray(numbers);

console.log('\nAfter sorting\n');

console.log(idList);

console.log('\nBefore Sorting\n');

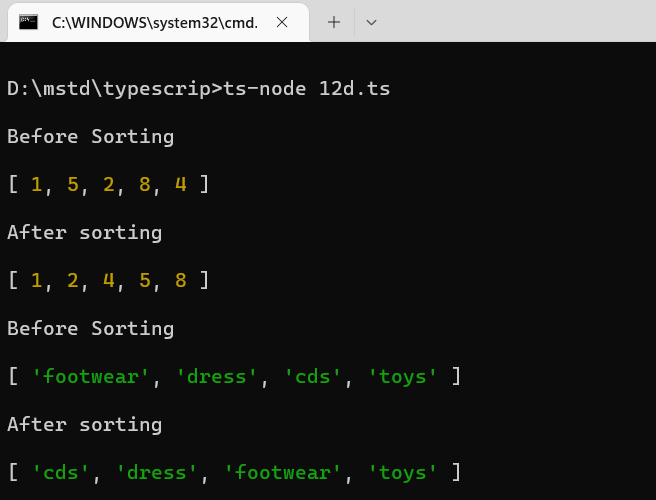
console.log(strings);

console.log('\nAfter sorting\n');

const nameList = sortArray(strings);

console.log(nameList);

**Output:**

****