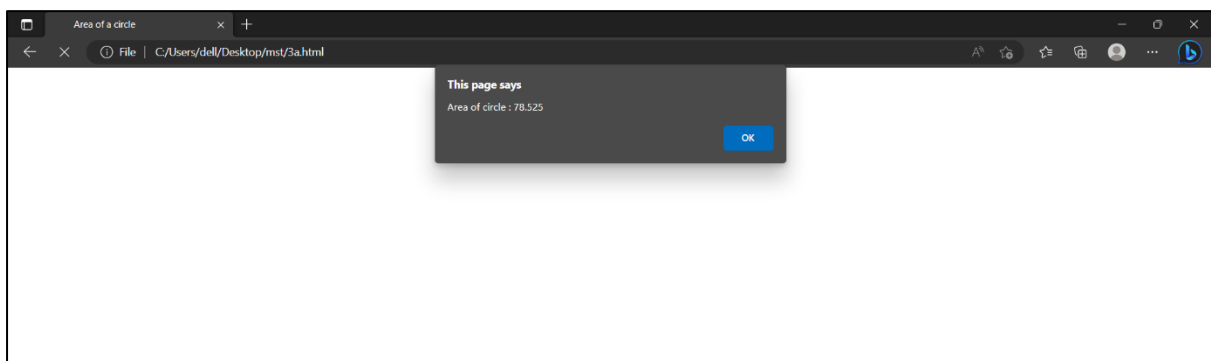
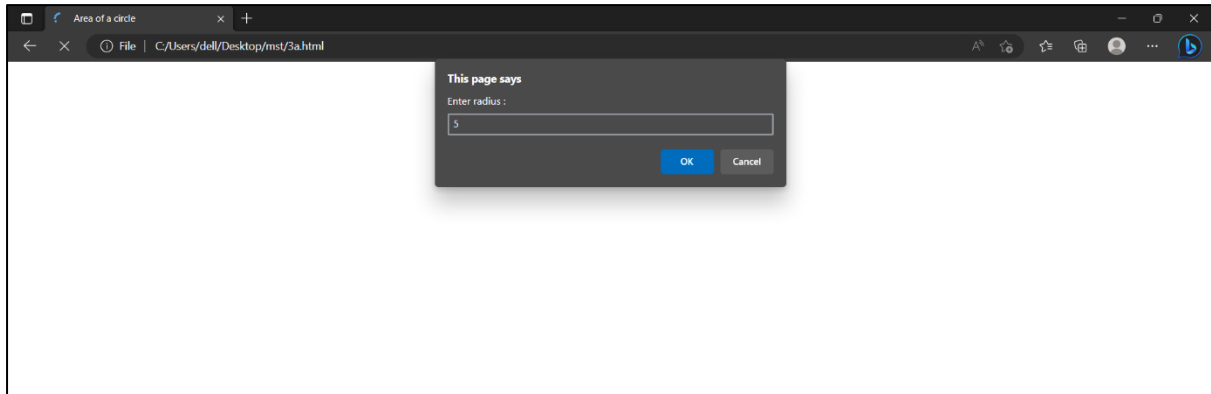


3. (a) Aim: Write a JavaScript program to find the area of a circle using radius (var and let - reassign and observe the difference with var and let) and PI (const)

Program:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Area of a circle</title>
  </head>
  <body>
    <script>
      var r=prompt("Enter radius : ");
      const pi=3.141
      let area = pi*r*r
      alert("Area of circle :"+area);
    </script>
  </body>
</html>
```

Output:



3. (b) 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.

Program:

```
<html>

<head>

<script>

var mv,st,rt,l,i;

    mv=window.prompt("Enter movie name : ");

    st=window.prompt("Enter hero, heroine Names : ");

    rt=window.prompt("Enter rating : ");

    l=window.prompt("Enter language : ");

    document.write("Movie Name : " + mv + "<br>");

    document.write("Hero, Heroine Names : " + st + "<br>");

    document.write("Rating : " + rt + "<br>");

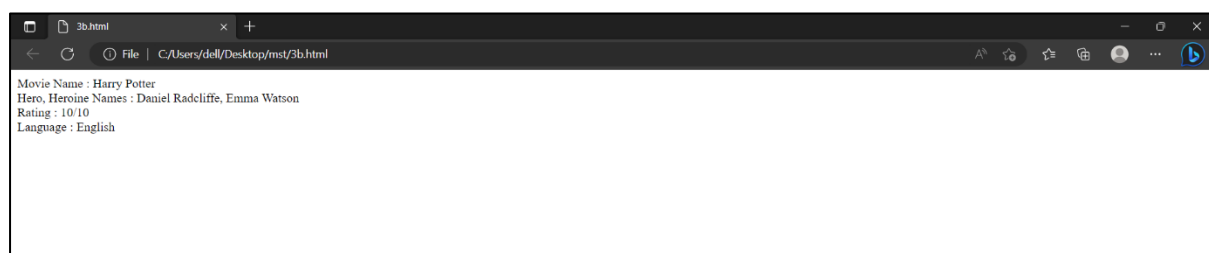
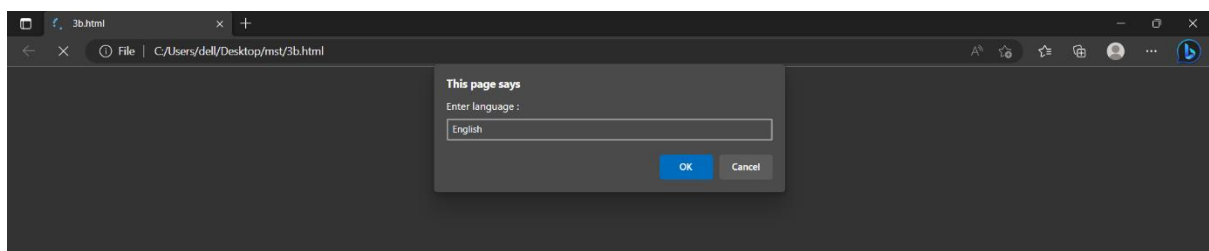
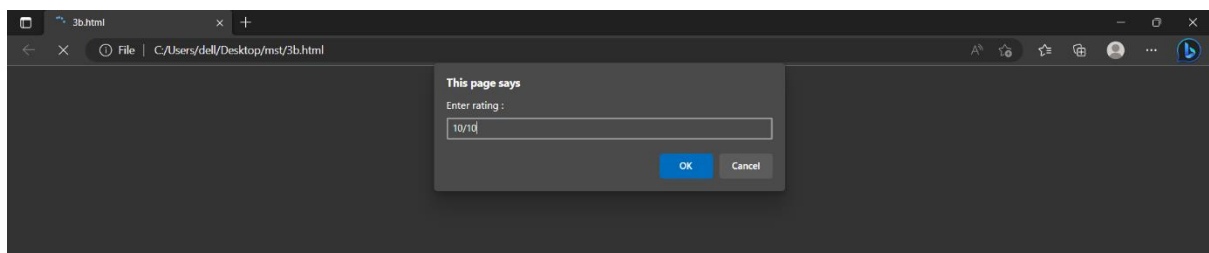
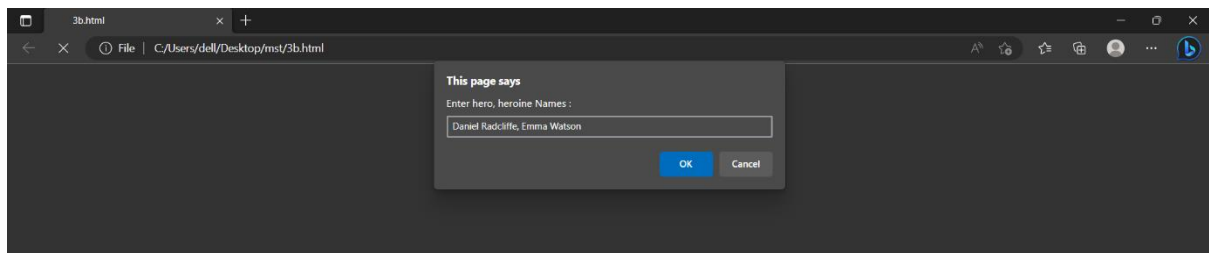
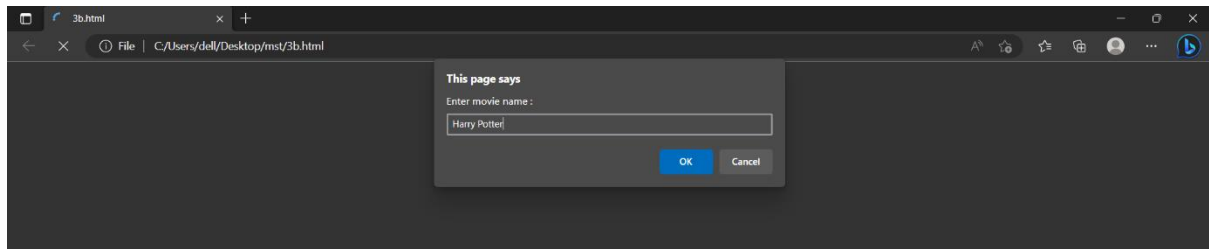
    document.write("Language : " + l + "<br>");

</script>

</head>

</html>
```

Output:



3. (c) 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.

Program:

```
<!DOCTYPE html>

<html>

  <head>

    <meta charset="UTF-8">

    <meta http-equiv="A-ML-compatiable">

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

  </head>

  <head>

    <title>Book a movie Ticket</title>

  </head>

  <body>

    <label for="tick">Enter number of tickets : </label>

    <input type="text" name="tick" id="tick"><br><br>

    <button onclick="getprice()">Calculate Price</button>

    <p id="price"></p>

    <script>

      function getprice(){

        var ticket;

        const price=150;

        const discount=10;

        var Total;

        var netPrice;

        ticket=document.getElementById("tick").value;

        total=ticket*price;
```

Output:

```
discountAmount=(total*discount)/100;

netPrice=total-discountAmount;

output=`<p>Number of tickets:${ticket}</p>

<p>Ticket price:${price}</p>

<p>${discount}%discountAmount:${discountAmount}</p>

<p>Total price:${netPrice}</p>`

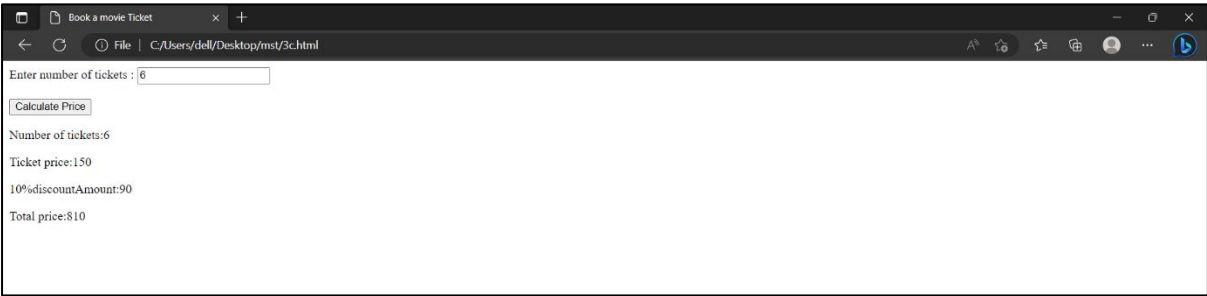
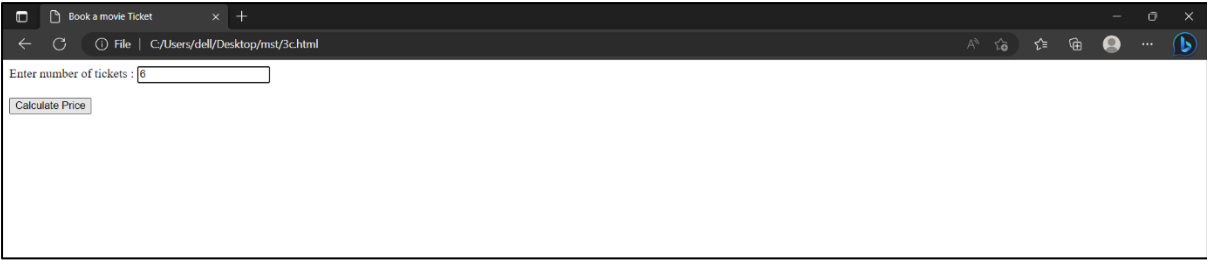
document.getElementById("price").innerHTML=output;

}

</script>

</body>

</html>
```



3. (d) 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 seats are more than 2 and less than 6, add discount of 10% to ticket.

Program:

```
<!DOCTYPE html>

<html>

  <head>

    <meta charset="UTF-8">

    <meta http-equiv="A-ML-compatiable">

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

    <title>Book Ticket</title>

  </head>

  <body>

    <label for="tick">Enter number of tickets : </label>

    <input type="text" name="tick" id="tick"><br><br>

    <button onclick="getprice()">Calculate Price</button>

    <p id="price"></p>

    <script>

      function getprice(){

        var ticket;

        var netPrice;

        ticket=document.getElementById("tick").value;

        if (ticket<= 2) {

          pcal(150,0,ticket)

        }

        else if (ticket >= 6) {

          document.getElementById("price").innerHTML="Booking Is Not Allowed. Not more
than 5 tickets can be booked at a time.";

        }

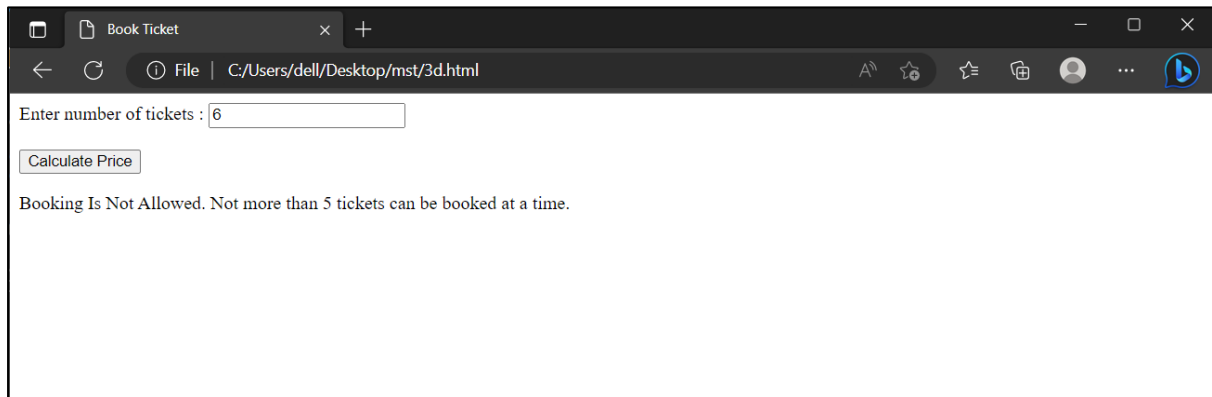
        else {
```



```
        pcal(150,10,ticket)
    }
}

function pcal(price,discount,ticket){
    total=ticket*price;
    discountAmount=(total*discount)/100;
    netPrice=total-discountAmount;
    output=`<p>Number of tickets:${ticket}</p>
    <p>Ticket price:${price}</p>
    <p>${discount}%discountAmount:${discountAmount}</p>
    <p>Total price:${netPrice}</p>`
    document.getElementById("price").innerHTML=output;
}
</script>
</body>
</html>
```

Output:

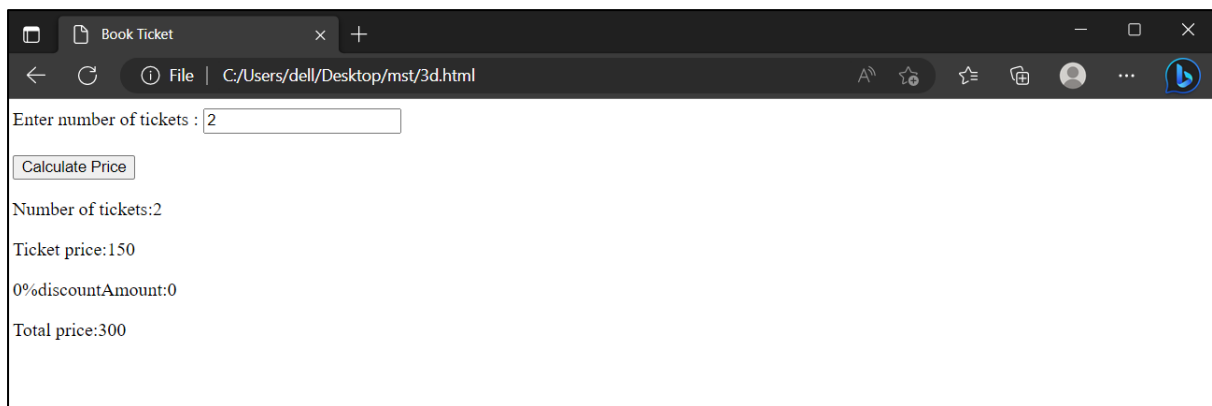


Book Ticket

File | C:/Users/dell/Desktop/mst/3d.html

Enter number of tickets :

Booking Is Not Allowed. Not more than 5 tickets can be booked at a time.

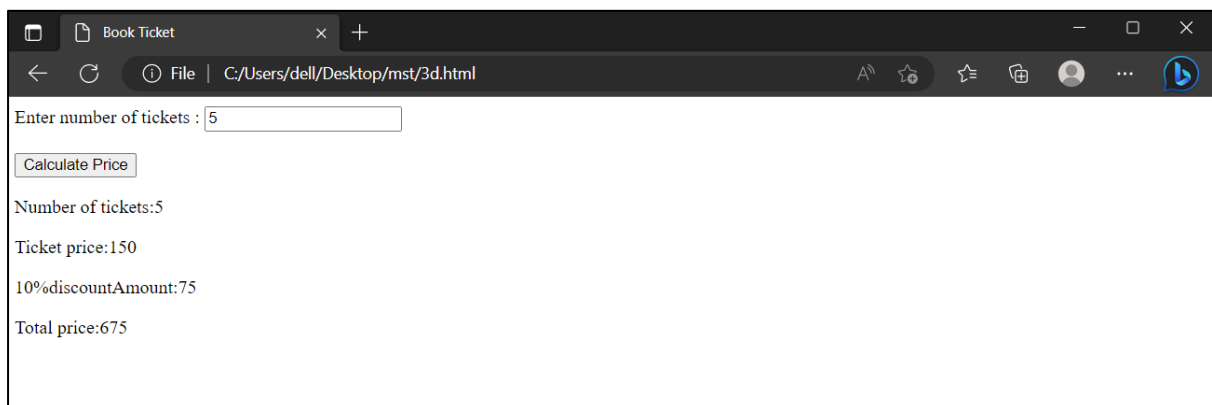


Book Ticket

File | C:/Users/dell/Desktop/mst/3d.html

Enter number of tickets :

Number of tickets:2
Ticket price:150
0%discountAmount:0
Total price:300



Book Ticket

File | C:/Users/dell/Desktop/mst/3d.html

Enter number of tickets :

Number of tickets:5
Ticket price:150
10%discountAmount:75
Total price:675

3. (e) 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 seats are more than 2 and less than 6, add discount of 10% to ticket.

Program:

```
<!DOCTYPE html>

<html>

  <head>

    <meta charset="UTF-8">

    <meta http-equiv="A-ML-compatiable">

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

    <title>Book a movie Ticket</title>

  </head>

  <body>

    <label for="tick">Enter number of tickets : </label>

    <input type="text" name="tick" id="tick"><br><br>

    <button onclick="getprice()">Calculate Price</button>

    <p id="price"></p>

    <script>

      function getprice(){

        var ticket;

        var netPrice;

        ticket=document.getElementById("tick").value;

        switch(ticket)

        {

          case '1':

            pcal(150,0,ticket)

            break;

          case '2':

            pcal(150,10,ticket)
```

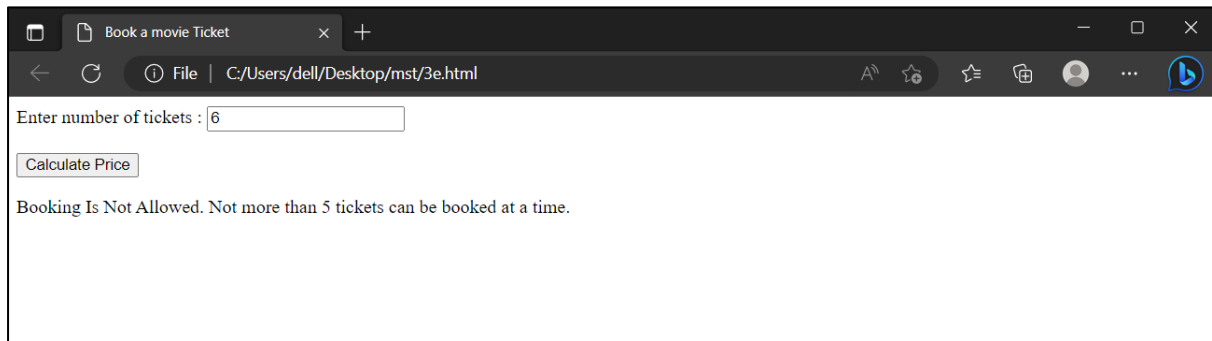
```

        break;
    case '3':
        pcal(150,10,ticket)
        break;
    case '4':
        pcal(150,10,ticket)
        break;
    case '5':
        pcal(150,10,ticket)
        break;
    default:
        document.getElementById("price").innerHTML="Booking Is Not Allowed. Not
more than 5 tickets can be booked at a time.";
    }
}

function pcal(price,discount,ticket){
    total=ticket*price;
    discountAmount=(total*discount)/100;
    netPrice=total-discountAmount;
    output=`<p>Number of tickets:${ticket}</p>
<p>Ticket price:${price}</p>
<p>${discount}%discountAmount:${discountAmount}</p>
<p>Total price:${netPrice}</p>`
    document.getElementById("price").innerHTML=output;
}
</script>
</body>
</html>

```

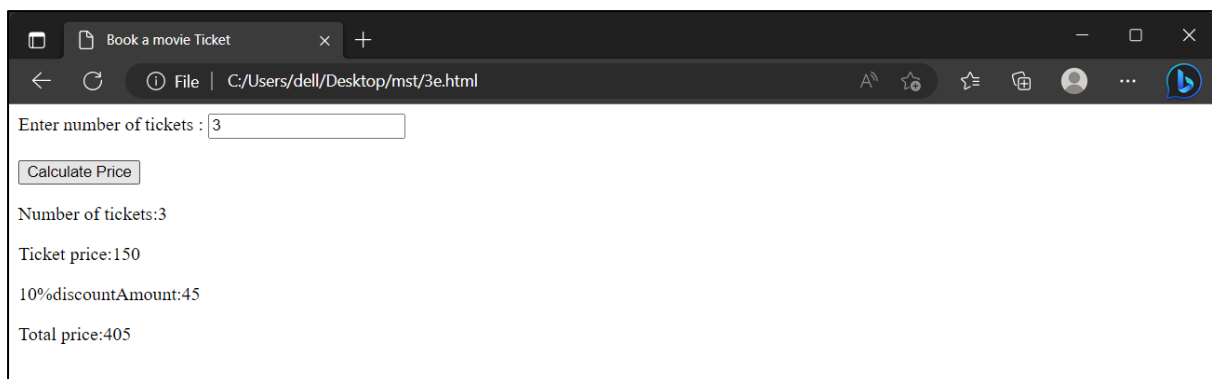
Output:



A screenshot of a web browser window titled "Book a movie Ticket". The address bar shows the file path "C:/Users/dell/Desktop/mst/3e.html". The page contains a text input field with the value "6" and a "Calculate Price" button. Below the button, a message states: "Booking Is Not Allowed. Not more than 5 tickets can be booked at a time."

Enter number of tickets :

Booking Is Not Allowed. Not more than 5 tickets can be booked at a time.



A screenshot of the same web browser window. The text input field now contains the value "3". After clicking the "Calculate Price" button, the page displays the following information: "Number of tickets:3", "Ticket price:150", "10%discountAmount:45", and "Total price:405".

Enter number of tickets :

Number of tickets:3
Ticket price:150
10%discountAmount:45
Total price:405

4. (a) 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 seats are more than 2 and less than 6, add discount of 10% to ticket.

Program:

```
<!DOCTYPE html>

<html>

  <head>

    <meta charset="UTF-8">

    <meta http-equiv="A-ML-compatiable">

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

    <title>Book a movie Ticket</title>

  </head>

  <body>

    <label for="tick">Enter number of tickets : </label>

    <input type="text" name="tick" id="tick"><br><br>

    <button onclick="getprice()">Calculate Price</button>

    <p id="price"></p>

    <script>

      getprice = () => {

        var ticket;

        var netPrice;

        ticket=document.getElementById("tick").value;

        if (ticket<= 2) {

          pcal(150,0,ticket)

        }

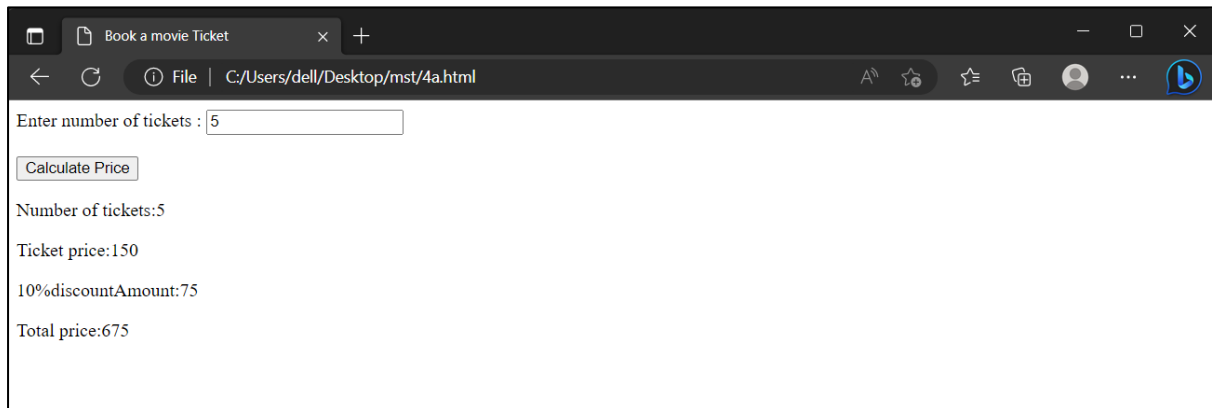
        else if (ticket >= 6) {

          document.getElementById("price").innerHTML="Booking Is Not Allowed. Not more
than 5 tickets can be booked at a time.";

        }

        else {
```

Output:



A screenshot of a web browser window titled "Book a movie Ticket". The address bar shows the file path "C:/Users/dell/Desktop/mst/4a.html". The page content includes a text input field with the label "Enter number of tickets :" and the value "5". Below the input field is a button labeled "Calculate Price". The output of the calculation is displayed below the button: "Number of tickets:5", "Ticket price:150", "10%discountAmount:75", and "Total price:675".

Enter number of tickets :

Calculate Price

Number of tickets:5
Ticket price:150
10%discountAmount:75
Total price:675

```
        pcal(150,10,ticket)
    }
}

pcal = (price,discount,ticket) => {
    total=ticket*price;
    discountAmount=(total*discount)/100;
    netPrice=total-discountAmount;
    output=`<p>Number of tickets:${ticket}</p>
    <p>Ticket price:${price}</p>
    <p>${discount}%discountAmount:${discountAmount}</p>
    <p>Total price:${netPrice}</p>`
    document.getElementById("price").innerHTML=output;
}
</script>
</body>
</html>
```


4. (b) 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.

Program:

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

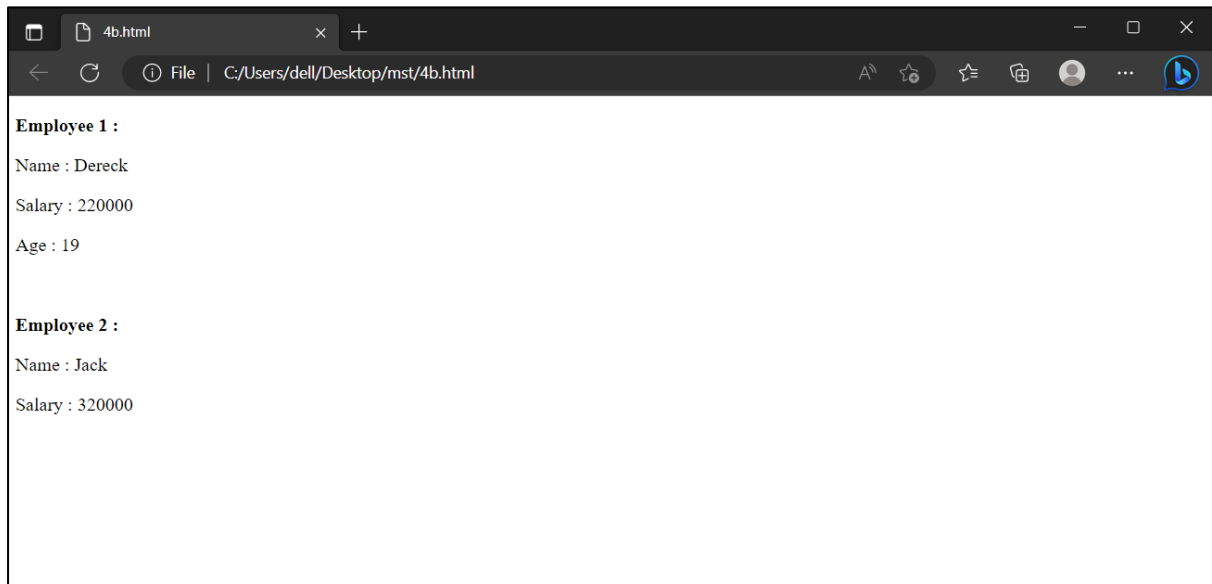
```
<script>
```

```
class person {
    constructor(name, s, age) {
        this.name = name;
        this.s = s;
        this.age = age;
    }
    fun() {
        this.s = this.s + 20000;
    }
}

class employee extends person {
    constructor(salary, name) {
        super(name);
        this.salary = salary;
    }
    add()
    {
        this.salary=this.salary+p.s;
    }
}

var p = new person("Dereck", 200000, 19)
var e = new employee(100000, "Jack")
```

Output:



```
p.fun();
e.add();
output = `

<b>Employee 1 : </b></p>
    <p>Name : ${p.name}</p>
    <p>Salary : ${p.s}</p>
    <p>Age : ${p.age}</p><br>
    <p><b>Employee 2 : </b></p>
    <p>Name : ${e.name}</p>
    <p>Salary : ${e.salary}</p>`

document.write(output);

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


```

4. (c) 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 seats are more than 2 and less than 6, add discount of 10% to ticket.

Program:

```
<!DOCTYPE html>

<html>

<body>

    <label for="movie_ticket">Enter number of tickets : </label>

    <input id="tickets" type="number" onfocus="of()" onblur="ob()"><br><br>

    <button type="button" onclick="tickets()">Calculate Price</button>

    <script >

function tickets()

{

    const price=150;

    const disc=10;

    var t=document.getElementById('tickets').value;

    total=t*price;

    if(t<=2 && t>0)

    {

        discount=0;

        netprice=total;

    }

    else if(t>=6 || t<=0)

    {

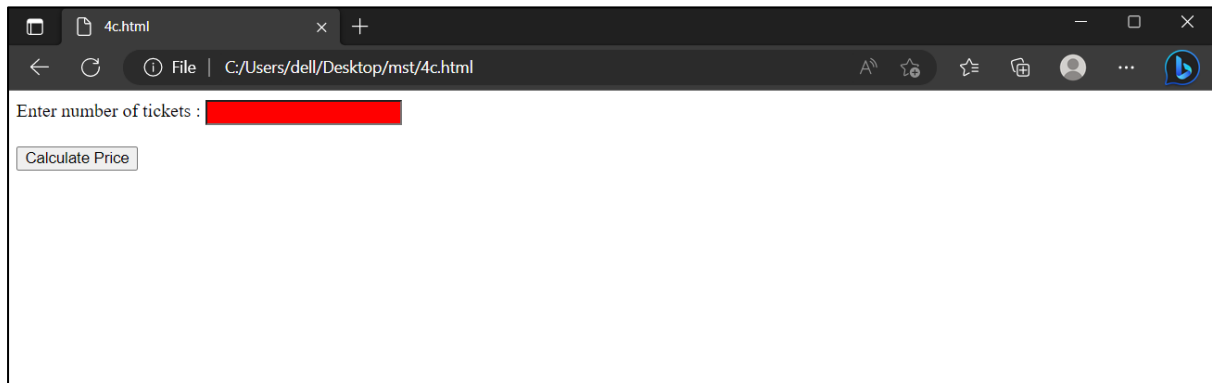
        document.write("Booking Is Not Allowed. Not more than 5 tickets can be booked at a
time.");

        return;

    }

    else
```

Output:

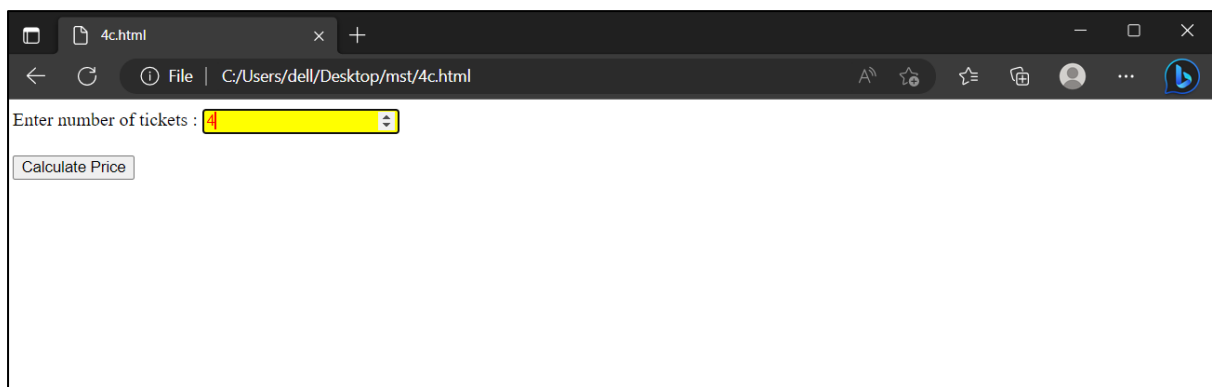


4c.html

File | C:/Users/dell/Desktop/mst/4c.html

Enter number of tickets :

Calculate Price

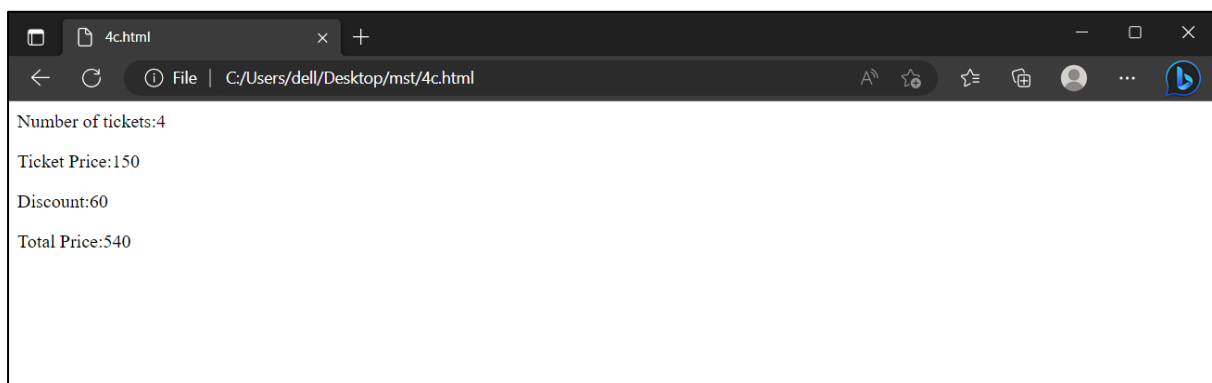


4c.html

File | C:/Users/dell/Desktop/mst/4c.html

Enter number of tickets :

Calculate Price



4c.html

File | C:/Users/dell/Desktop/mst/4c.html

Number of tickets:4
Ticket Price:150
Discount:60
Total Price:540

```

    {
        var total=t*price;
        var discount=(total*disc)/100;
        var netprice=total-discount;
    }
    output=`<p>Number of tickets:${t}</p>
    <p>Ticket Price:${price}</p>
    <p>Discount:${discount}</p>
    <p>Total Price:${netprice}</p>`
    document.write(output);
}
function of()
{
    document.getElementById('tickets').style='background-color:yellow;color:red';
}
function ob()
{
    document.getElementById('tickets').style='background-color:red;color:yellow';
}
</script>
</body>
</html>

```

4. (d) 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 and a different message.

Program:

```
<!DOCTYPE html>

<html>

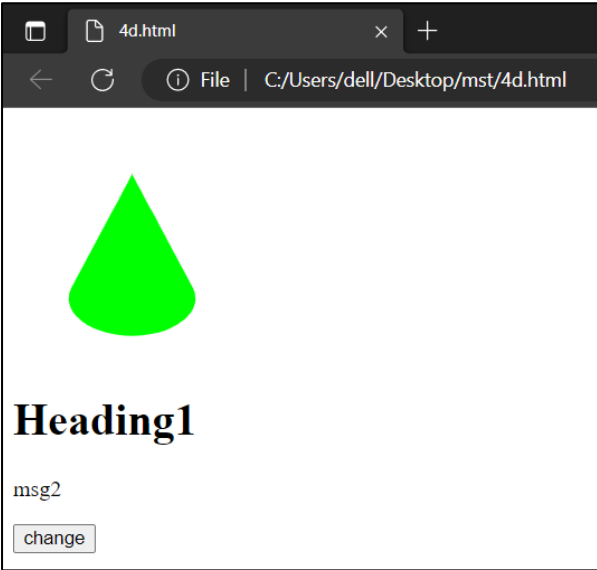
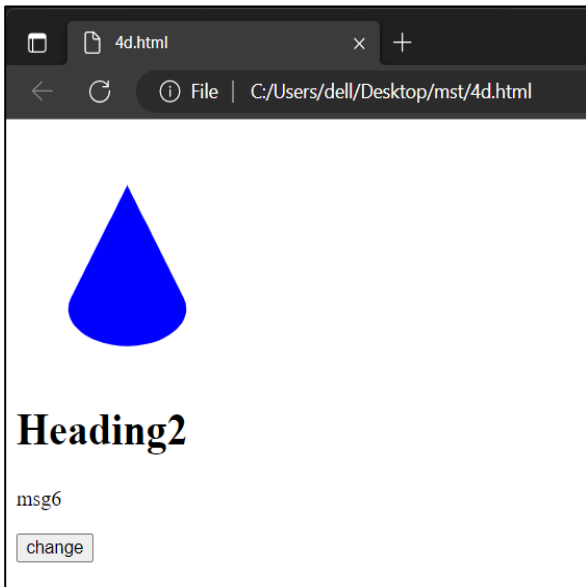
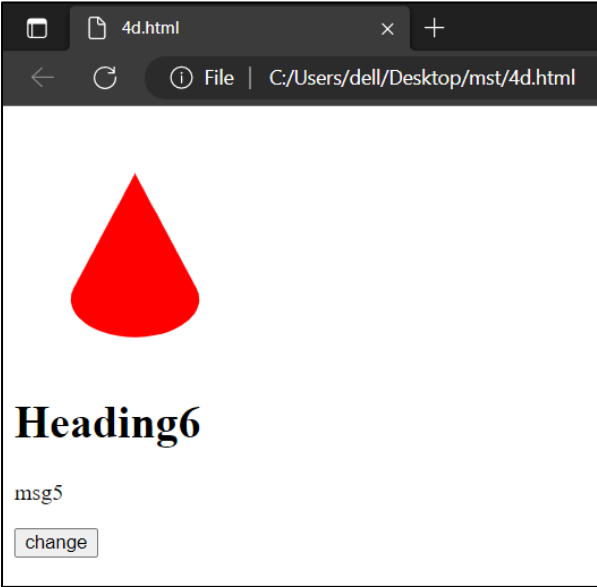
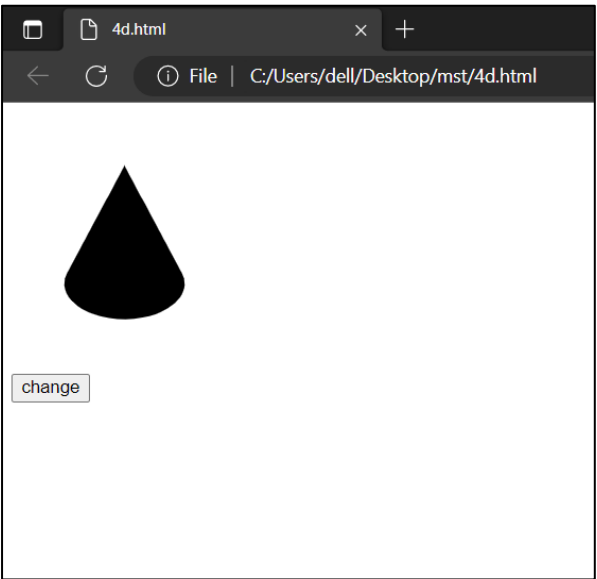
<body>

  <svg id="cone" version="1.0" xmlns="http://www.w3.org/2000/svg"
  width="128.000000pt" height="128.000000pt" viewBox="0 0 1280.000000 1280.000000"
  preserveAspectRatio="xMidYMid meet" fill="#ff0000">

    <g transform="translate(0.000000,1280.000000) scale(0.100000,-0.100000)"
    fill="#000000" stroke="none">

      <path d="M6380 9807 c-9 -18 -58 -111 -109 -207 -52 -96 -109 -202 -126 -235
      -18 -33 -65 -121 -105 -195 -40 -74 -107 -200 -150 -280 -43 -80 -107 -199
      -142 -265 -91 -168 -217 -405 -271 -505 -24 -47 -74 -139 -109 -205 -131 -242
      -309 -575 -408 -760 -99 -185 -277 -518 -408 -760 -35 -66 -85 -158 -109 -205
      -25 -47 -62 -116 -83 -155 -21 -38 -55 -101 -75 -140 -21 -38 -71 -133 -113
      -210 -41 -77 -105 -196 -142 -265 -37 -69 -102 -190 -145 -270 -43 -80 -94
      -176 -115 -215 -99 -185 -277 -518 -408 -760 -35 -66 -85 -158 -109 -205 -25
      -47 -70 -131 -99 -187 -54 -100 -115 -255 -128 -323 -3 -19 -9 -46 -12 -60
      -12 -50 -23 -237 -18 -315 16 -293 154 -586 412 -874 191 -214 493 -429 830
      -592 78 -38 145 -69 148 -69 3 0 24 -8 47 -18 39 -17 59 -25 165 -67 217 -85
      613 -186 927 -235 639 -100 1241 -92 1900 25 88 16 222 46 380 86 80 20 330
      98 395 123 75 29 127 50 167 68 23 10 45 18 48 18 19 0 308 149 420 217 129
      78 307 207 417 302 70 61 227 236 288 321 156 220 252 473 265 700 4 81 -7
      256 -19 310 -3 14 -9 41 -12 60 -13 68 -74 223 -128 323 -29 56 -74 140 -99
      187 -24 47 -74 139 -109 205 -131 242 -309 575 -408 760 -21 39 -72 135 -115
      215 -43 80 -108 201 -145 270 -37 69 -101 188 -142 265 -42 77 -92 172 -113
      210 -20 39 -54 102 -75 140 -21 39 -58 108 -83 155 -24 47 -74 139 -109 205
      -129 239 -304 565 -408 760 -21 39 -70 131 -110 205 -40 74 -107 200 -150 280
```

Output:




```

-43 80 -108 201 -145 270 -37 69 -88 163 -112 210 -54 100 -180 337 -271 505
-35 66 -99 185 -142 265 -43 80 -110 206 -150 280 -40 74 -87 162 -105 195
-17 33 -74 139 -126 235 -51 96 -100 189 -109 207 -8 18 -17 32 -20 32 -3 0
-12 -14 -20 -32z"/>

</g>

</svg>

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

<p id="msg"></p>

<button onclick="randomChange();">change</button>

<script>

    function randomChange(){

        let colors = ["#3cb371", "#FF0000", "#00FF00", "#0000FF", "#FFFFFF", "#808080"];

        let headings =
["Heading1", "Heading2", "Heading3", "Heading4", "Heading5", "Heading6"]

        let message = ["msg1", "msg2", "msg3", "msg4", "msg5", "msg6"];

        let rcolor = colors[Math.floor(Math.random() * 6)];

        let rheading = headings[Math.floor(Math.random() * 6)];

        let rmessage = message[Math.floor(Math.random() * 6)];

        document.getElementById("cone").childNodes[1].style.fill = rcolor;

        document.getElementById("head").innerHTML = rheading;

        document.getElementById("msg").innerHTML = rmessage;

    }

</script>

</body>

</html>

```

5. (a) 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.

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>ARRAYS</title>

</head>

<body>

  <div id="movietemplate"></div>

  <script>

    let html_code="";

    let movies = [{

      movieName : "movie1",

      starring : "starring1",

      language : "Telugu",

      rating : 5

    },

    {

      movieName : "movie2",

      starring : "starring2",

      language : "English",

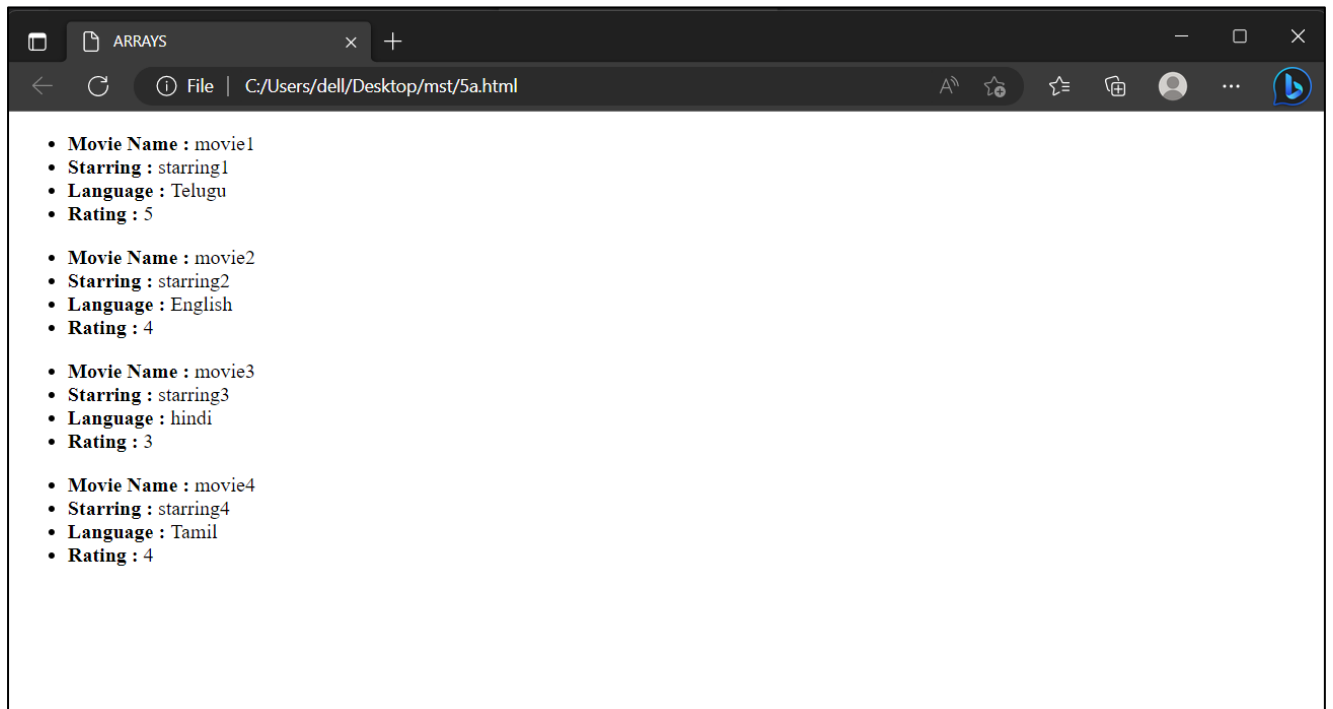
      rating : 4

    },

    {

      movieName : "movie3",
```

Output:



```

    starring : "starring3",
    language : "hindi",
    rating : 3
  },
  {
    movieName : "movie4",
    starring : "starring4",
    language : "Tamil",
    rating : 4
  }
];

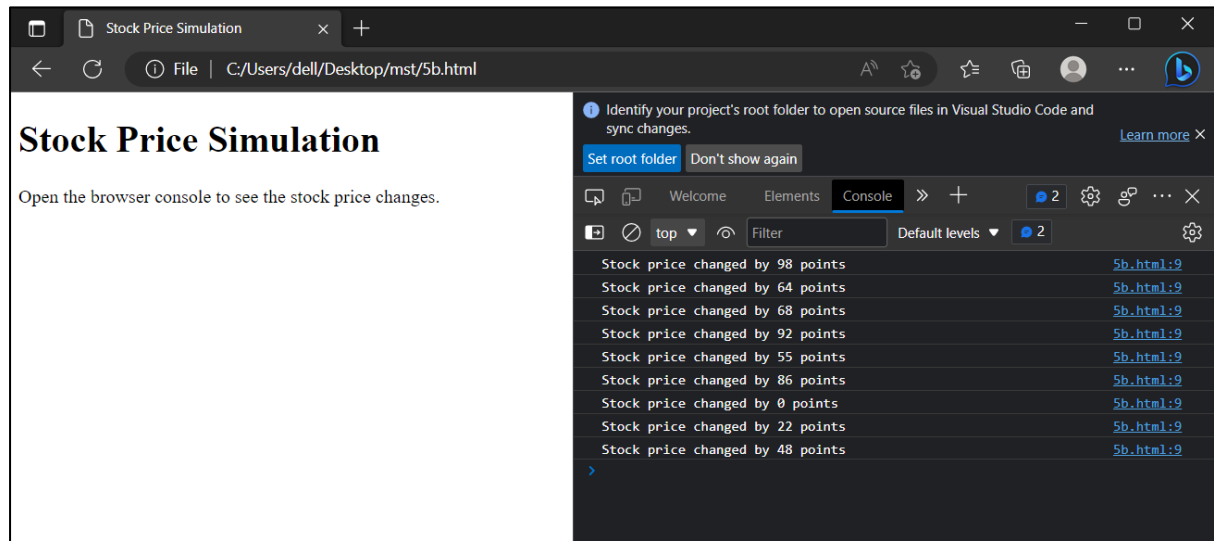
const movieDetails = (movie) => {
  let html;
  html = `<div><ul>
    <li><b>Movie Name : </b>${movie.movieName}</li>
    <li><b>Starring : </b>${movie.starring}</li>
    <li><b>Language : </b>${movie.language}</li>
    <li><b>Rating : </b>${movie.rating}</li>
  </ul></div>`;
  return html;
};

for (let x of movies){
  html_code += movieDetails(x);
}

document.getElementById('movietemplate').innerHTML = html_code;
</script>
</body>
</html>

```

Output:



5. (b) 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 stops when exited.

Program:

```
<!DOCTYPE html>

<html>

  <head>

    <title>Stock Price Simulation</title>

    <script>

      function simulateStockPriceChange() {

        setInterval(() => {

          const stockPriceChange = Math.floor(Math.random() * 100);

          console.log(`Stock price changed by ${stockPriceChange} points`);

        }, 3000);

      }

    </script>

  </head>

  <body onload="simulateStockPriceChange()">

    <h1>Stock Price Simulation</h1>

    <p>Open the browser console to see the stock price changes.</p>

  </body>

</html>
```

5. (c) 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 will return "Invalid User".

Program:

5c.html

```
<html>

<body>

  <form>

    <label for="user">USER ID : </label>

    <input type="text" name="user" id="user"><br><br>

    <label for="password">PASSWORD : </label>

    <input type="password" name="password" id="password"><br><br>

    <input type="button" id="clog" value="LOGIN" onclick="">&nbsp;

    <input type="reset" value="RESET">

  </form>

  <script type="module">

    import{database,validate as validate} from "./login.mjs";

    console.log(database);

    console.log(validate);

    function checkLogin(){

      let userid = document.getElementById("user").value;

      let pass = document.getElementById("password").value;

      validate(userid, pass);

    }

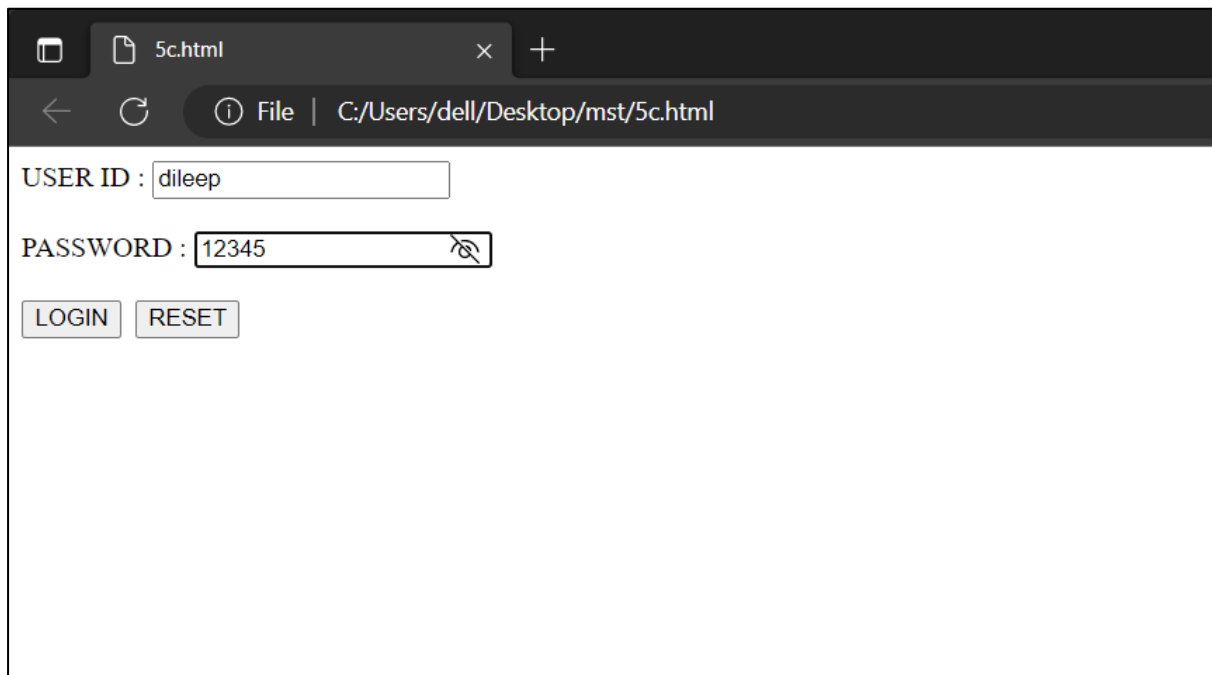
    document.getElementById("clog").addEventListener('click', checkLogin));

  </script>

</body>


</html>
```

Output:



A screenshot of a web browser window showing a login form. The browser's address bar displays the file path `C:/Users/dell/Desktop/mst/5c.html`. The form contains two input fields: "USER ID :" with the value "dileep" and "PASSWORD :" with the value "12345". The password field has a toggle icon (an eye with a slash) to its right. Below the input fields are two buttons: "LOGIN" and "RESET".

USER ID :

PASSWORD : 

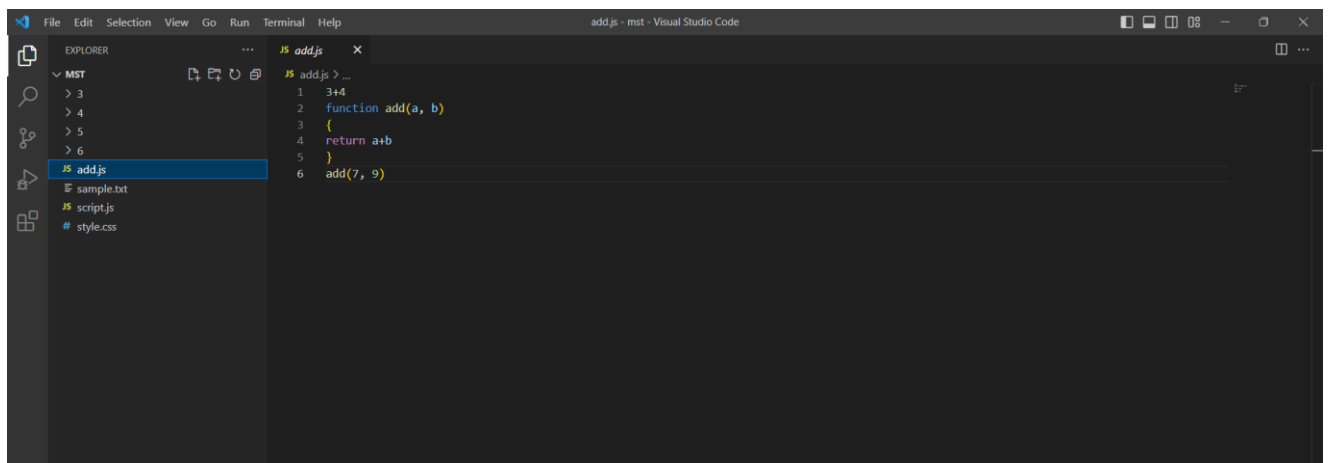
login.mjs

```
let database = {dileep : "12345",
               deepthi : "ap456",
               vinessa : "56lj*45"};

function validate(userid, pass){
  if(database[userid]==pass){
    alert("Login successful");
  }
  else{
    alert("Invalid userid/password");
  }
}

export {database,validate}
```

Output:

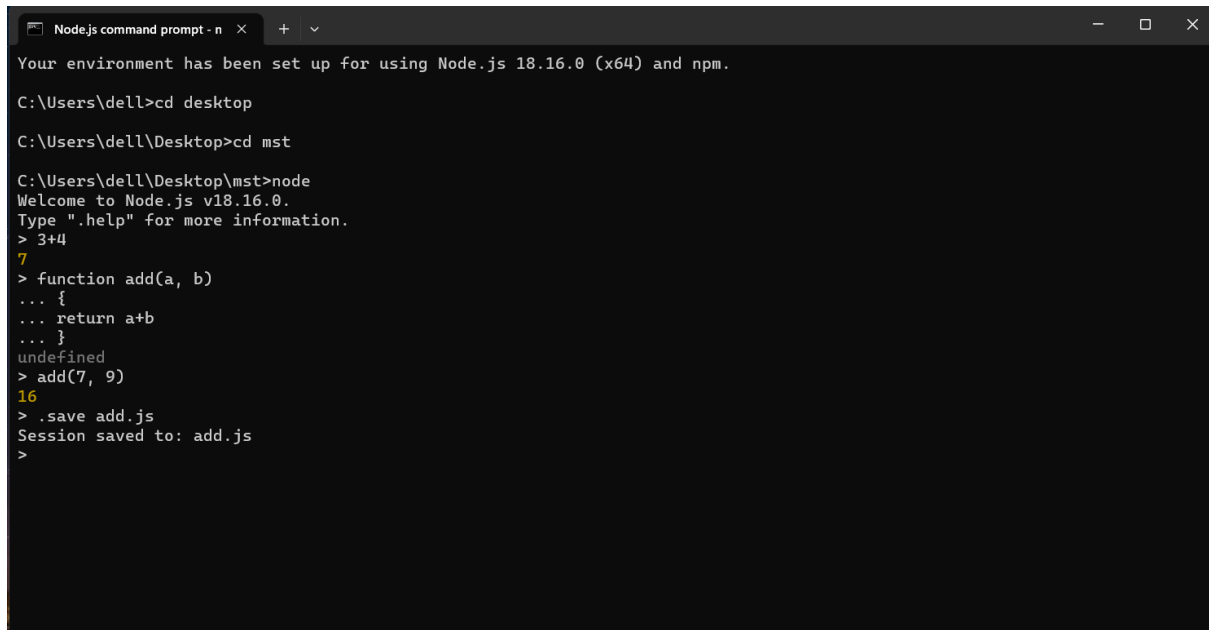


The image shows a screenshot of the Visual Studio Code editor interface. The title bar at the top reads "add.js - mst - Visual Studio Code". The Explorer sidebar on the left shows a project structure with a folder named "MST" containing files "sample.txt", "script.js", and "style.css". The "add.js" file is selected and open in the main editor. The code in the editor is as follows:

```
1 3+4
2 function add(a, b)
3 {
4   return a+b
5 }
6 add(7, 9)
```

6. (a) **Aim:** Verify how to execute different functions successfully in the Node.js platform.

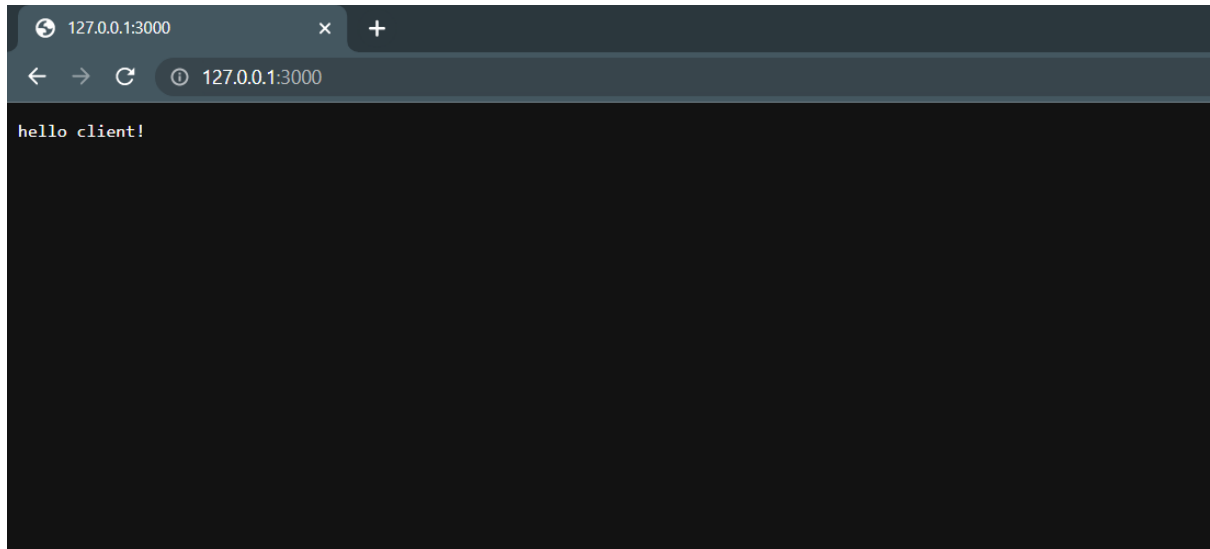
Program:



```
Node.js command prompt - n  X + v
Your environment has been set up for using Node.js 18.16.0 (x64) and npm.

C:\Users\dell>cd desktop
C:\Users\dell\Desktop>cd mst
C:\Users\dell\Desktop\mst>node
Welcome to Node.js v18.16.0.
Type ".help" for more information.
> 3+4
7
> function add(a, b)
... {
...   return a+b
... }
undefined
> add(7, 9)
16
> .save add.js
Session saved to: add.js
>
```

Output:



6. (b) Aim: Write a program to show the workflow of JavaScript code executable by creating web server in Node.js.

Program:

```
const hostname = '127.0.0.1';  
  
const port = 3000;  
  
var http = require('http');  
var server = http.createServer(function (request, response) {  
    console.log('request starting...');  
    response.write('hello client!');  
    response.end();  
});  
server.listen(port, hostname, () => {  
    console.log(`Server running at http://${hostname}:${port}`);  
});
```

Output:

6. (c) Aim: Write a Node.js module to show the workflow of Modularization of Node application.

Program:

```
import * as readline from 'node:readline/promises';
import {stdin as input, stdout as output} from 'node:process'
import {database, validate} from './login.mjs';
const rl=readline.createInterface((input, output));
async function main(){
    var username, password;
    username = await rl.question("Enter username : ");
    password = await rl.question("Enter password : ");
    validate(username, password);
    rl.close()
}
main();
```

login.mjs

```
let database = {dileep : "12345",
                deepthi : "ap456",
                vinessa : "56lj*45"};
function validate(userid, pass){
    if(database[userid]==pass){
        console.log("Login successful");
    }
    else{
        console.log("Invalid userid/password");
    }
}
export {database,validate};
```

Output:

6. (d) Aim: Write a program to show the workflow of restarting a Node application

Program:

```
const hostname = '127.0.0.1';
const port = 3000;
var http = require('http')
var server = http.createServer(function(request, response)
{
  console.log('request starting...');
  response.write('hello client!');
  response.end();
})
```

Output:

```
Node.js command prompt
Your environment has been set up for using Node.js 18.16.0 (x64) and npm.

C:\Users\dell>cd desktop
C:\Users\dell\Desktop>cd mst
C:\Users\dell\Desktop\mst>node 6e.js
file created
C:\Users\dell\Desktop\mst>
```

```
Visual Studio Code
sample.txt - mst - Visual Studio Code

EXPLORER
MST
  > 3
  > 4
  > 5
  < 6
    JS 6b.js
    JS 6c.js
    JS 6d.js
    JS 6e.js
    JS login.mjs
    JS sample.txt
    JS add.js
    JS script.js
    # style.css

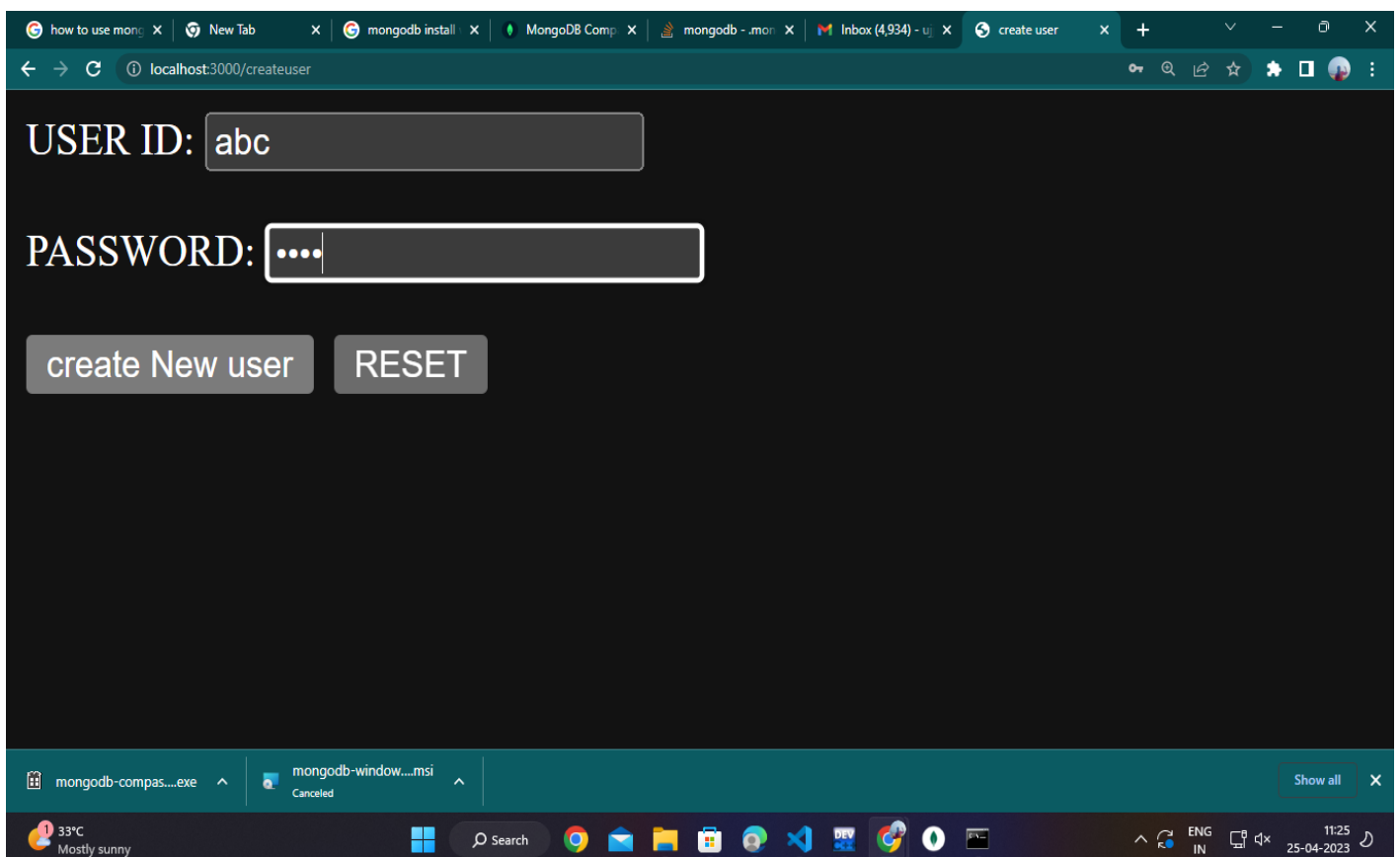
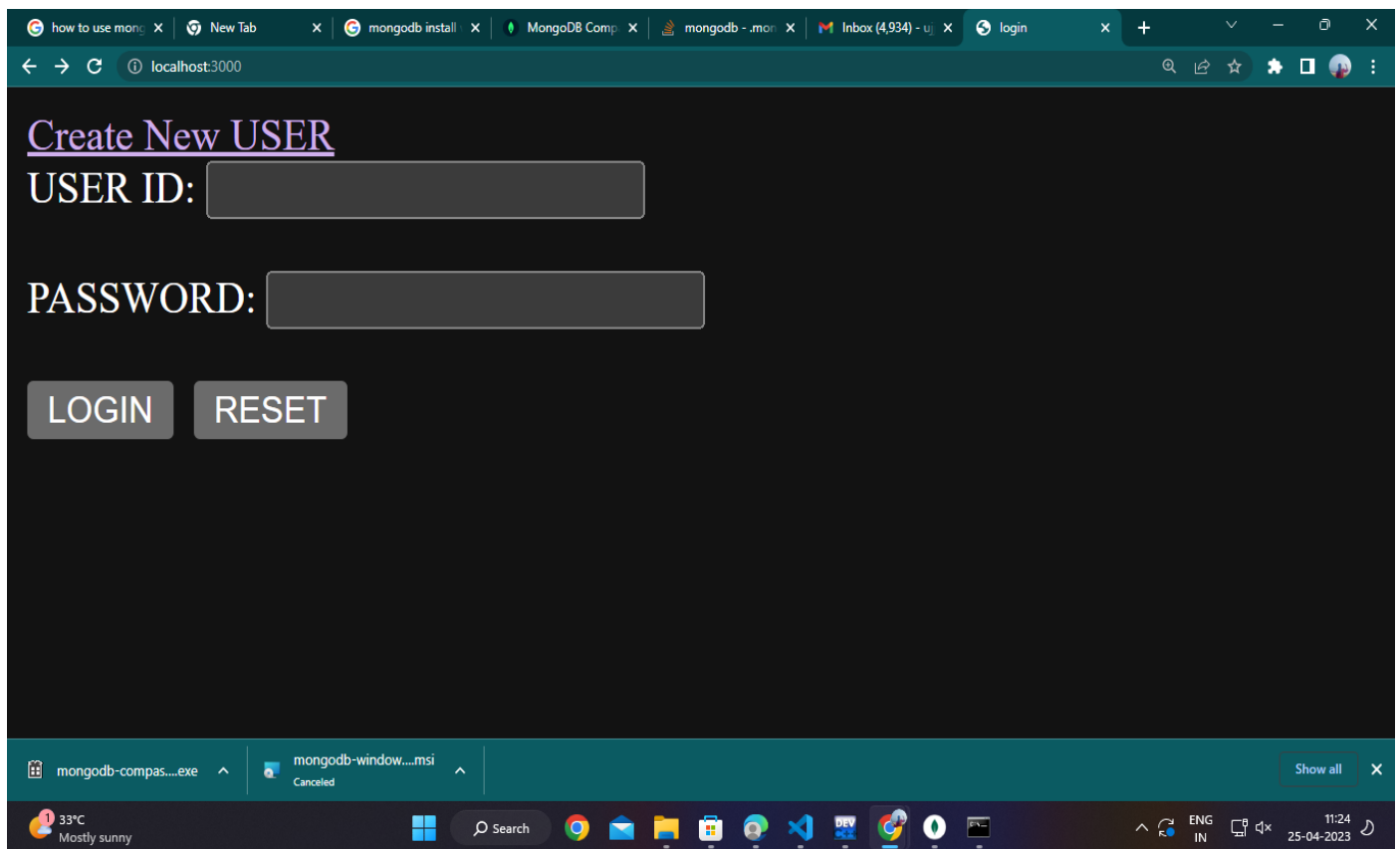
sample.txt
6 > sample.txt
1  Mongo Express Angular Node
```

6. (e) Aim: Create a text file src.txt and add the following data to it. Mongo, Express, Angular, Node.

Program:

```
const fs = require('fs');  
fs.writeFile("sample.txt","Mongo Express Angular Node",(err)=>{  
  if(err){  
    throw err;  
  }  
  console.log("file created");  
})
```

Output:



7. Aim:

(a) Implement routing for the AdventureTrails application by embedding the necessary code in the routes/route.js file.

(b) In myNotes application: (i) we want to handle POST submissions. (ii) display customized error messages. (iii) perform logging.

(c) Write a Mongoose schema to connect with MongoDB.

(d) Write a program to wrap the Schema into a Model object.

Program:**login.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>login</title>

</head>

<body>

<a href="/createuser">Create New USER</a>

<form action="/" method="post">

<label for="user">USER ID:</label>

<input type="text" name="user" id="user"><br /><br />

<label for="password">PASSWORD:</label>

<input type="password" name="password" id="password"><br /><br />

<input type="submit" id="clog" value="LOGIN" onclick="">&nbsp;

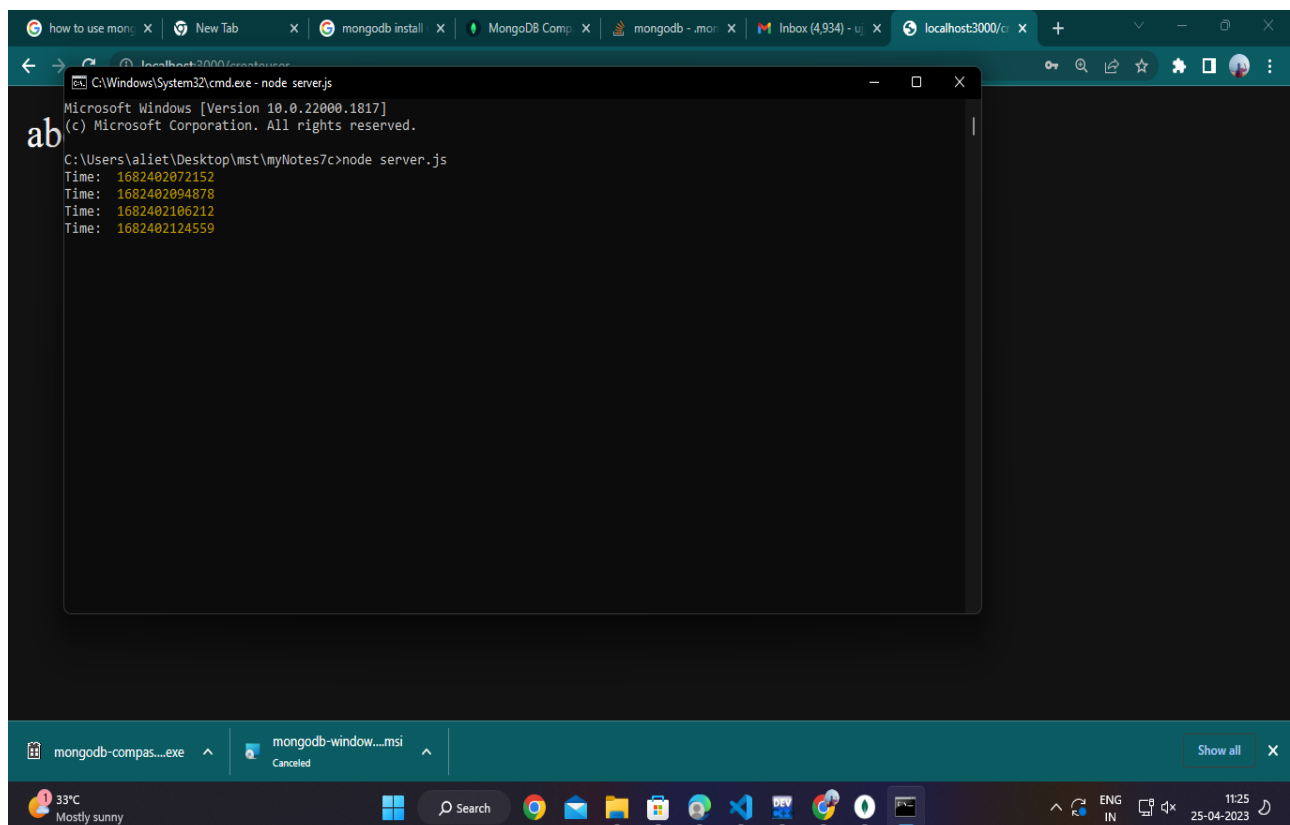
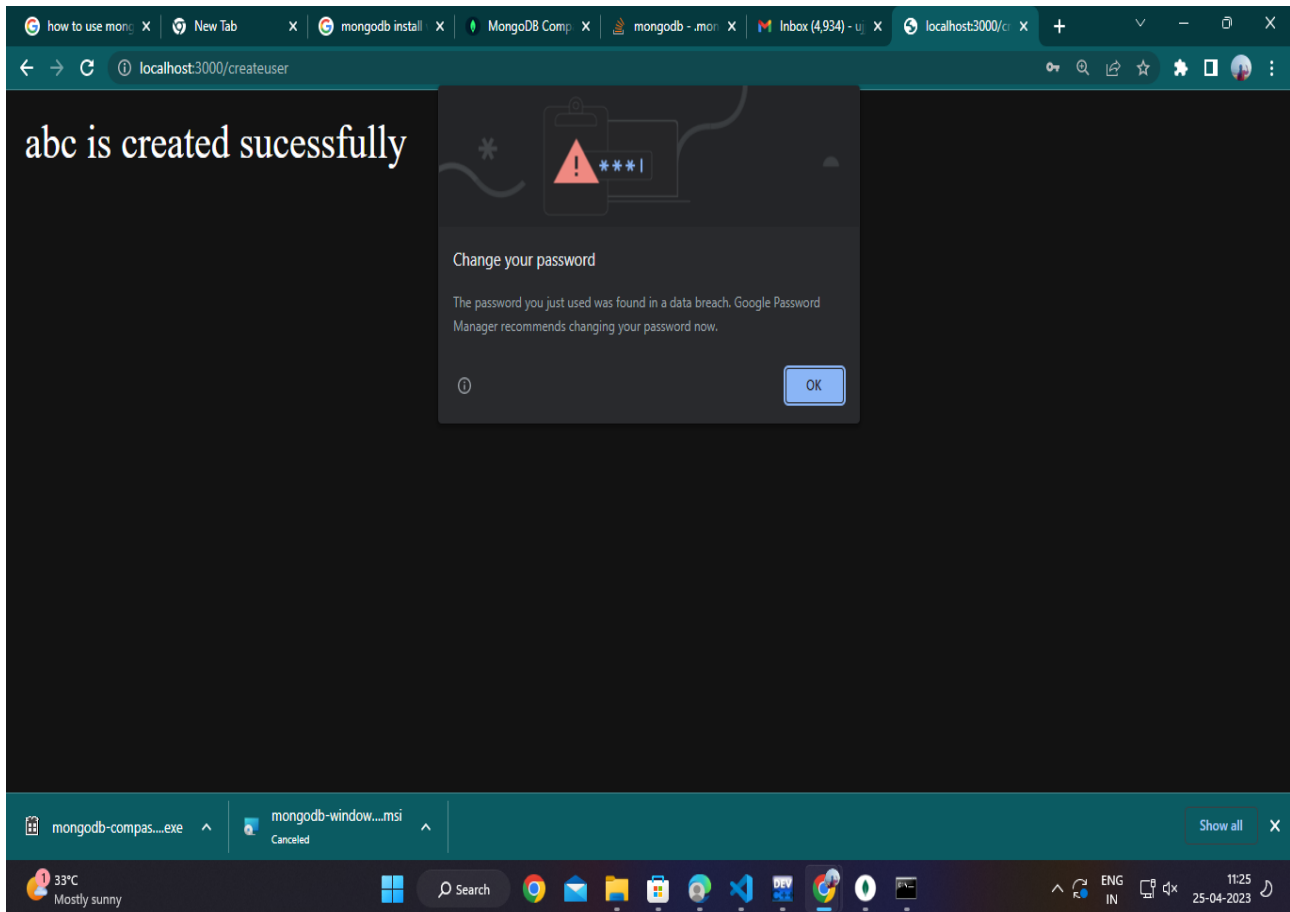
<input type="reset" value="RESET">

</form>

</body>

</html>
```

Output:



createuser.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>create user</title>

</head>

<body>

<form action="/createuser" method="post">

<label for="user">USER ID:</label>

<input type="text" name="user" id="user"><br /><br />

<label for="password">PASSWORD:</label>

<input type="password" name="password" id="password"><br /><br />

<input type="submit" id="clog" value="create New user" >&nbsp;

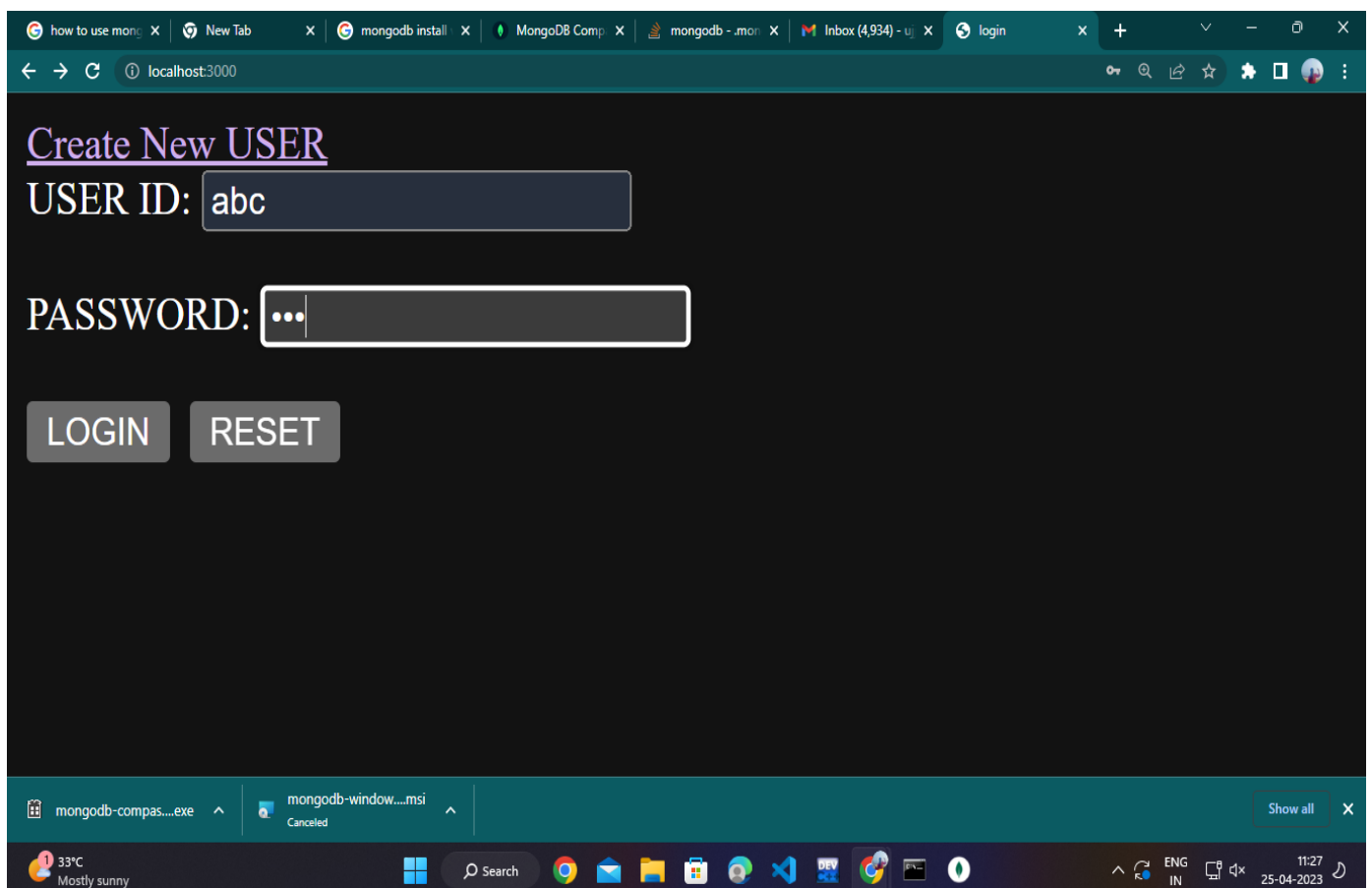
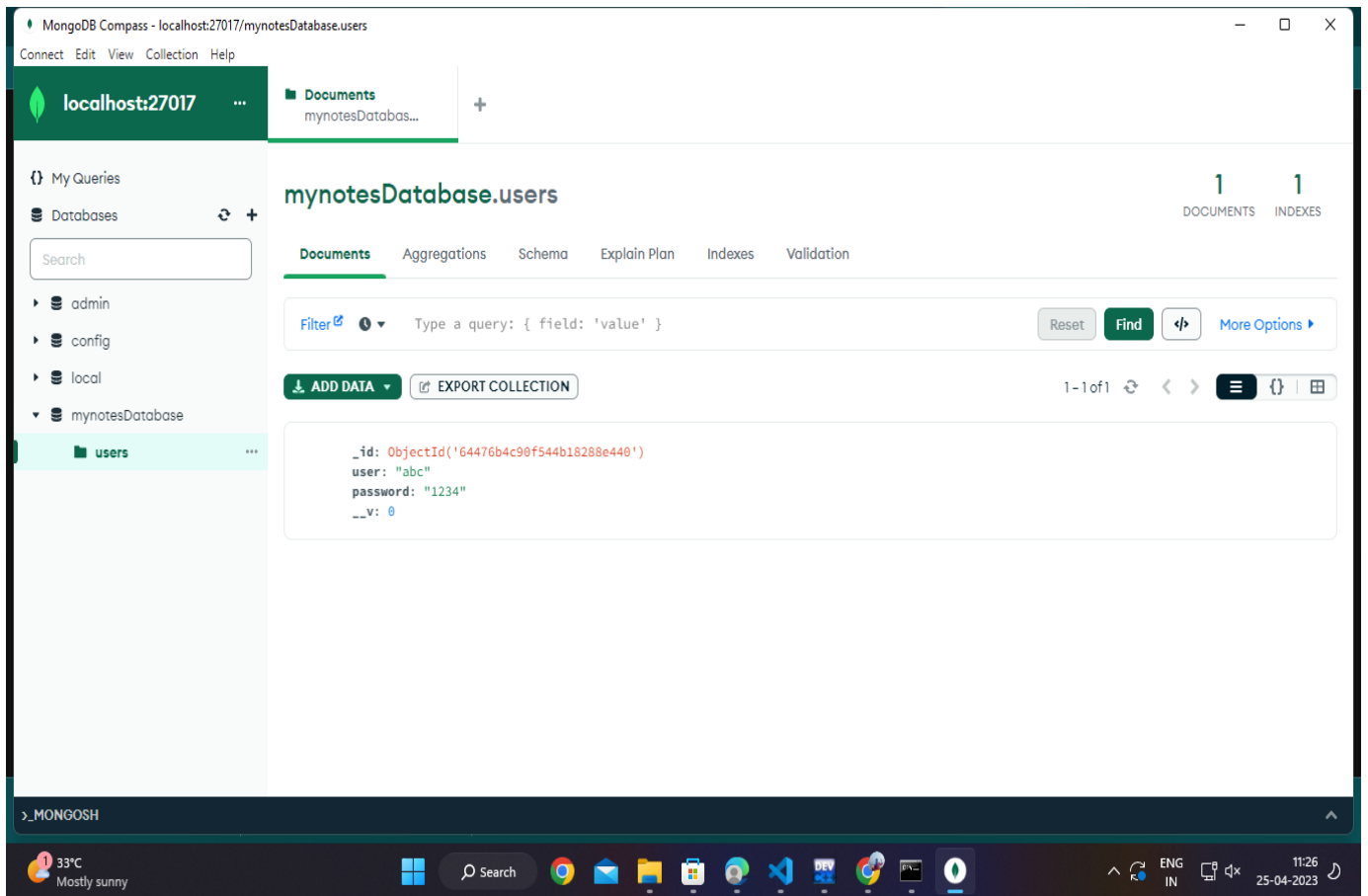
<input type="reset" value="RESET">

</form>

</body>

</html>
```

Output:



routes.js

```

const express = require('express')

const router = express.Router()

const path = require('path')

const mongoose = require('mongoose')

const Usermodel = require('../models/user')

// respond with "hello world" when a GET request is made to the homepage

// middleware that is specific to this router

router.use((req, res, next) => {

  console.log('Time: ', Date.now())

  next()

})

// define the home page route

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

  res.sendFile(path.join(__dirname, '../public/login.html'))

})

//create user

router.get('/createuser',(req,res) => {

  res.sendFile(path.join(__dirname, '../public/createuser.html'))

})

router.post('/createuser',(req,res) => {

  let userid = req.body.user;

  let pass = req.body.password;

  let newuser = new Usermodel({user : userid,password : pass})

  newuser.save().then(() => {

    res.send(` ${userid} is created sucessfully`)

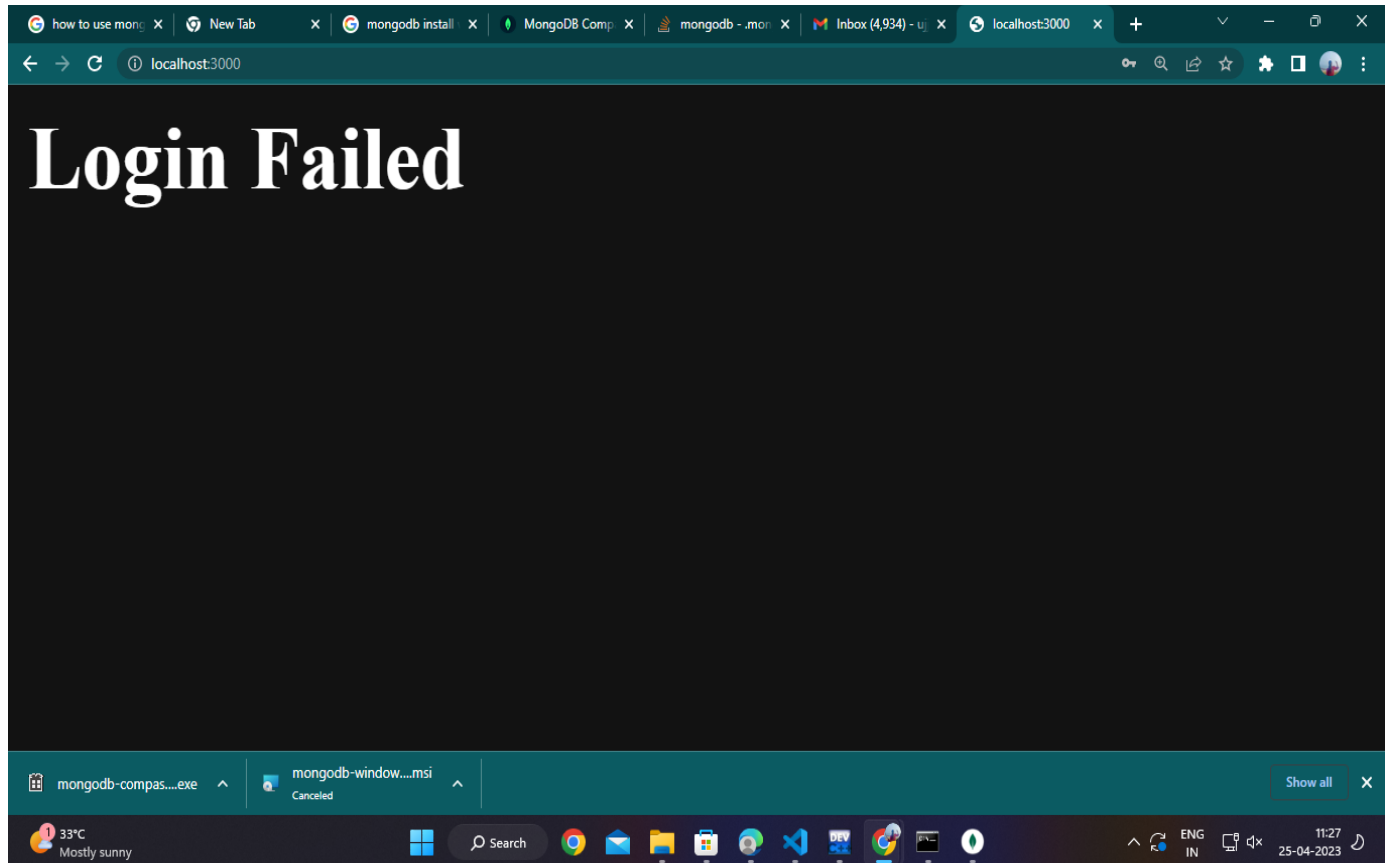
  })

})

router.post('/', (req, res) => {

```

Output:



```
//process form data
let userid = req.body.user;
let pass = req.body.password;
Usermodel.findOne({user : userid}).then((usr_rec) => {
  if ( usr_rec.password == pass ){
    res.send('<h1>login Successful</h1>');
  } else{
    res.send('<h1>Login Failed</h1>')
  }
})
})
module.exports = router
```

server.js

```
const express = require('express')
const app = express()
const bodyParser = require('body-parser')
const mongoose = require('mongoose')
const conn = mongoose.connect('mongodb://127.0.0.1:27017/mynotesDatabase')
const my_router = require("./routes/routes.js");
app.use(bodyParser.urlencoded({ extended: false }));
app.use(express.static('public'));
app.use("/", my_router);
app.listen(3000);
```

8. Aim:

(a) Write a program to perform various CRUD (Create-Read-Update-Delete) operations using Mongoose library functions.

(b) 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 base

(c) Write a program to explain session management using cookies.

(d) Write a program to explain session management using sessions.

(e) Implement security features in myNotes application

createnote.html

```
!DOCTYPE html>

<html lang="en">

<head>

<title>W3.CSS Template</title>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">

<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Lato">

<link rel="stylesheet"

href="https://fonts.googleapis.com/css?family=Montserrat">

<link rel="stylesheet"

href="https://cdnjs.cloudflare.com/ajax/libs/fontawesome/4.7.0/css/font-awesome.min.css">

<style>

body,h1,h2,h3,h4,h5,h6 {font-family: "Lato", sans-serif}

.w3-bar,h1,button {font-family: "Montserrat", sans-serif}

.fa-anchor,.fa-coffee {font-size:200px}

</style>

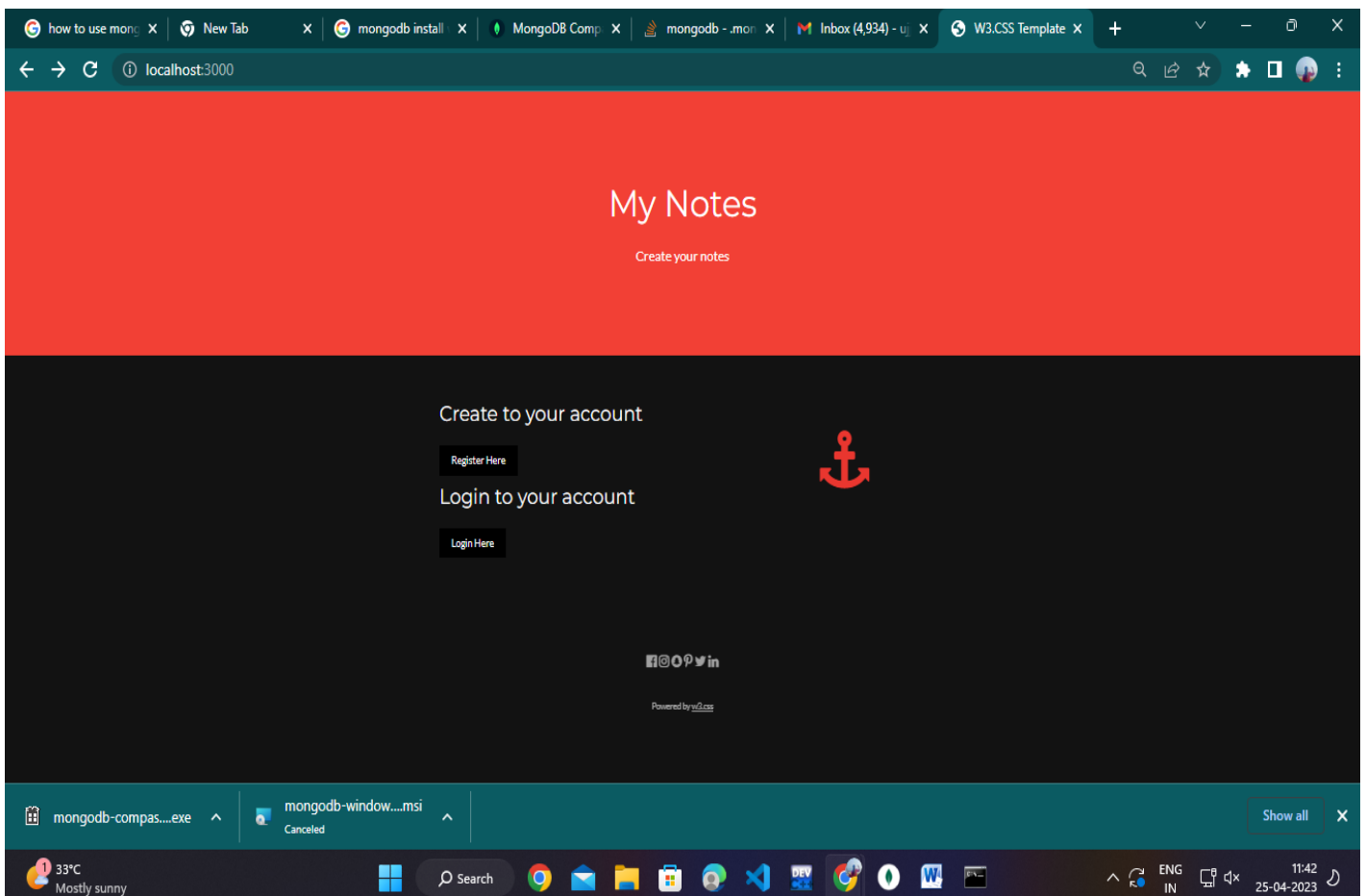
</head>

<body>

<!-- Header -->
```

Output:

```
C:\Windows\System32\cmd.exe - node index.js
Node.js v18.15.0
C:\Users\aliet\Desktop\mst\myNote>npm install express
added 34 packages, and audited 133 packages in 4s
13 packages are looking for funding
  run `npm fund` for details
found 0 vulnerabilities
C:\Users\aliet\Desktop\mst\myNote>node index.js
Time: 1682403080072
```



```

<header class="w3-container w3-red w3-center" style="padding:128px 16px">
<h1 class="w3-margin w3-jumbo">My Notes </h1>
<p class="w3-xlarge">Create your notes</p>
</header>

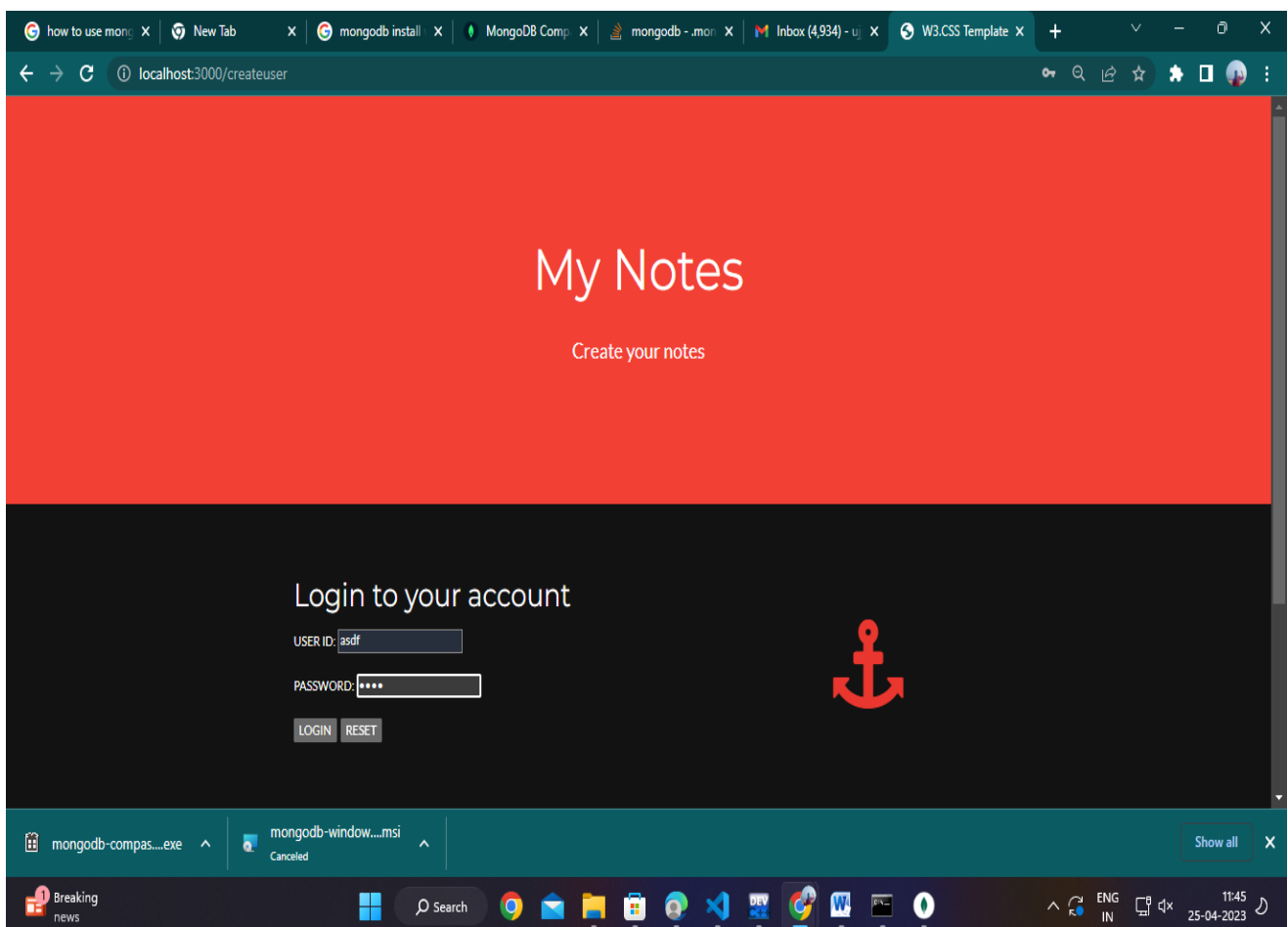
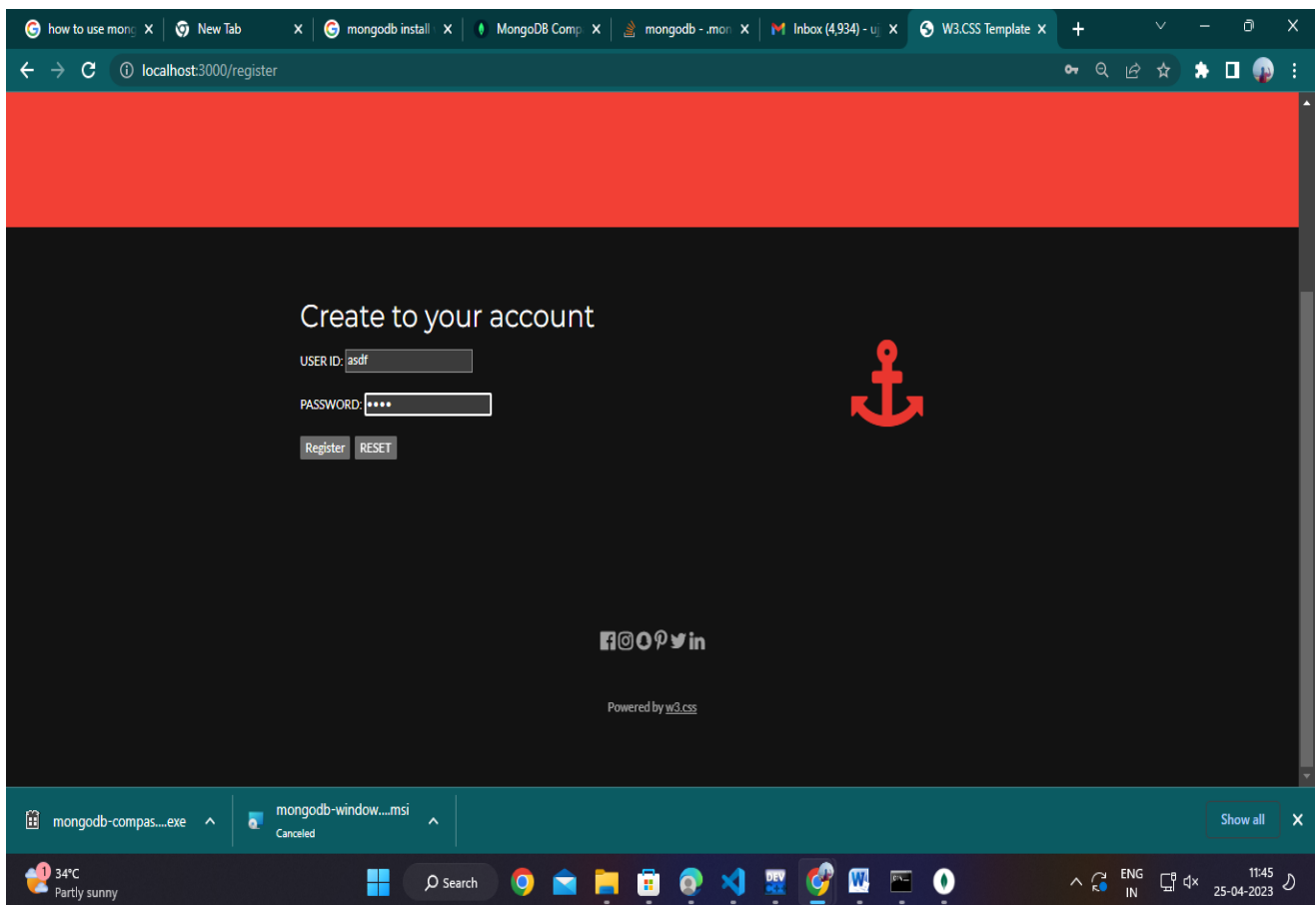
<div class="w3-row-padding w3-padding-64 w3-container">
<div class="w3-content">
<div class="w3-twothird"><form>
<input type="text" name="title" placeholder="title"><br>
<textarea name="note" class="w3-text-grey" placeholder="type
here"></textarea><br>
<input type="submit"></input>
</form></div>
<div class="w3-third w3-center">
<i class="fa fa-anchor w3-padding-64 w3-text-red"></i>
</div></div></div>

<div class="w3-container w3-black w3-center w3-opacity w3-padding-64">
<h1 class="w3-margin w3-xlarge">Quote of the day: live life</h1>
</div>

<footer class="w3-container w3-padding-64 w3-center w3-opacity">
<div class="w3-xlarge w3-padding-32">
<i class="fa fa-facebook-official w3-hover-opacity"></i>
<i class="fa fa-instagram w3-hover-opacity"></i>
<i class="fa fa-snapchat w3-hover-opacity"></i>
<i class="fa fa-pinterest-p w3-hover-opacity"></i>
<i class="fa fa-twitter w3-hover-opacity"></i>
<i class="fa fa-linkedin w3-hover-opacity"></i></div>
<p>Powered by <a href="https://www.w3schools.com/w3css/default.asp"
target="_blank">w3.css</a></p>
</footer><script></script></body></html>

```

Output:



home.html

```

<!DOCTYPE html>

<html lang="en">

<head>

<title>W3.CSS Template</title>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">

<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Lato">

<link rel="stylesheet"

href="https://fonts.googleapis.com/css?family=Montserrat">

<link rel="stylesheet"

href="https://cdnjs.cloudflare.com/ajax/libs/fontawesome/4.7.0/css/font-awesome.min.css">

<style>

body,h1,h2,h3,h4,h5,h6 {font-family: "Lato", sans-serif}

.w3-bar,h1,button {font-family: "Montserrat", sans-serif}

.fa-anchor,.fa-coffee {font-size:200px}

</style>

</head>

<body>

<header class="w3-container w3-red w3-center" style="padding:128px 16px">

<h1 class="w3-margin w3-jumbo">My Notes </h1>

<p class="w3-xlarge">Create your notes</p>

</header>

<div class="w3-row-padding w3-padding-64 w3-container">

<div class="w3-content">

<div class="w3-twothird">

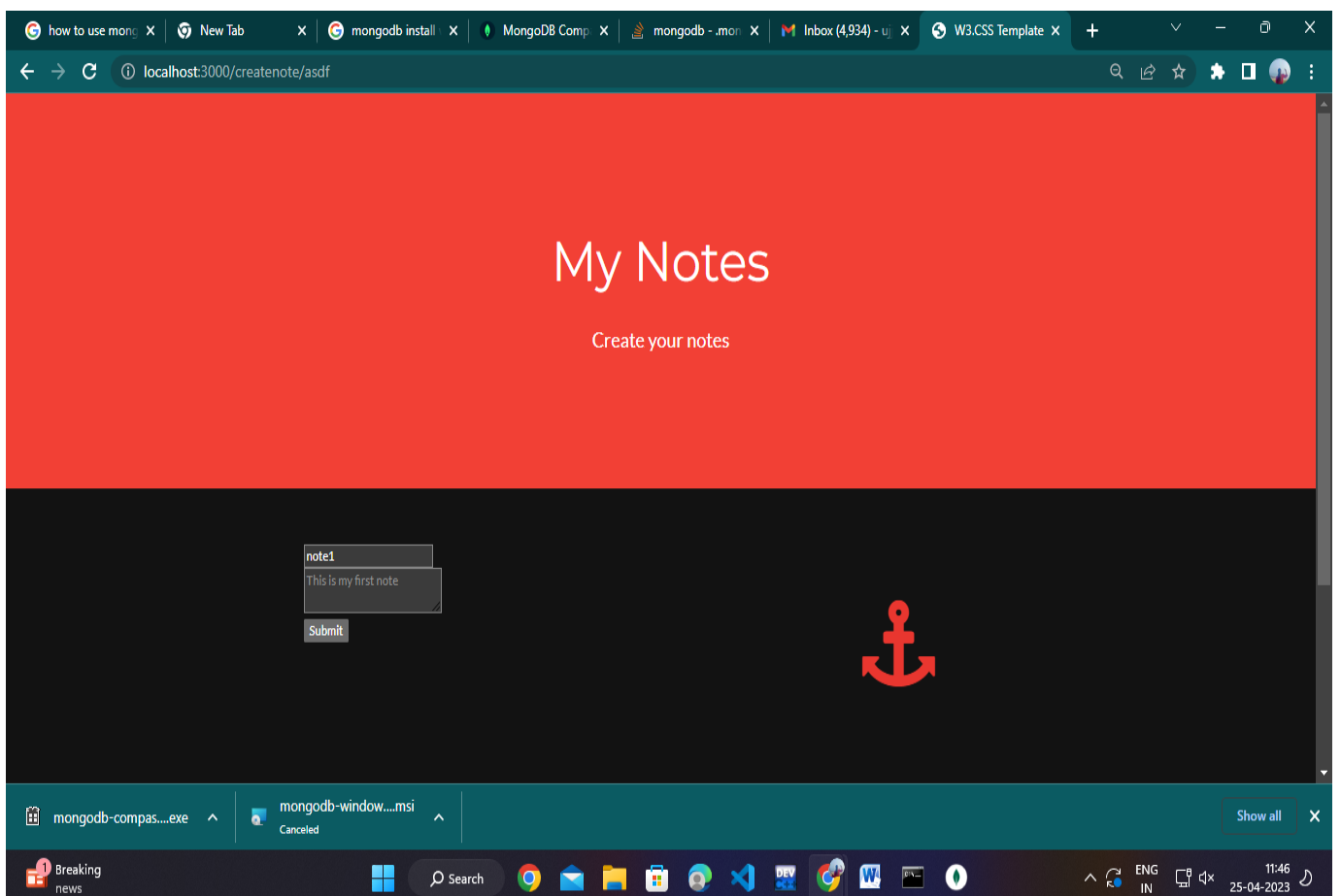
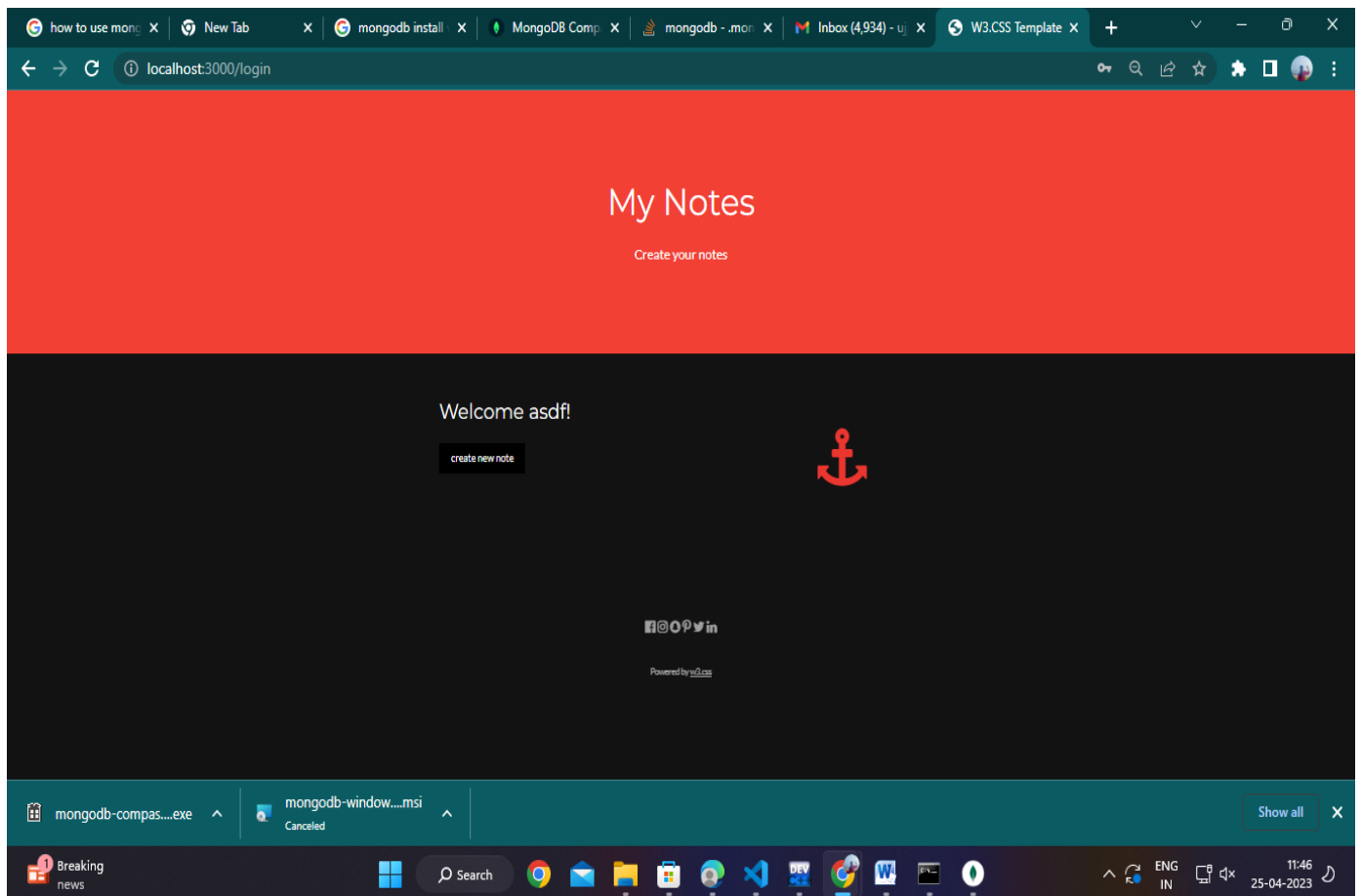
<h1>welcome !</h1>

<h5 class="w3-padding-32">note1. &nbsp;&nbsp;&nbsp;&nbsp;<i class="fa

fapencil">&nbsp;&nbsp;&nbsp;&nbsp;</i> <i class="fa fa-trash"></i></h5>

```


Output:



```

<h5 class="w3-padding-32">note2. &nbsp;&nbsp;&nbsp;<i class="fa
fapencil">&nbsp;&nbsp;&nbsp;</i> <i class="fa fa-trash"></i></h5>

<h5 class="w3-padding-32">note3. &nbsp;&nbsp;&nbsp;<i class="fa
fapencil">&nbsp;&nbsp;&nbsp;</i> <i class="fa fa-trash"></i></h5>

<button class="w3-button w3-black w3-padding-large w3-large w3-
margin-top">create new note</button>

</div>

<div class="w3-third w3-center">

<i class="fa fa-anchor w3-padding-64 w3-text-red"></i>

</div>

</div></div>

<div class="w3-container w3-black w3-center w3-opacity w3-padding-64">

<h1 class="w3-margin w3-xlarge">Quote of the day: live life</h1>

</div>

<footer class="w3-container w3-padding-64 w3-center w3-opacity">

<div class="w3-xlarge w3-padding-32">

<i class="fa fa-facebook-official w3-hover-opacity"></i>

<i class="fa fa-instagram w3-hover-opacity"></i>

<i class="fa fa-snapchat w3-hover-opacity"></i>

<i class="fa fa-pinterest-p w3-hover-opacity"></i>

<i class="fa fa-twitter w3-hover-opacity"></i>

<i class="fa fa-linkedin w3-hover-opacity"></i>

</div>

<p>Powered by <a href="https://www.w3schools.com/w3css/default.asp"
target="_blank">w3.css</a></p>

</footer>

<script>

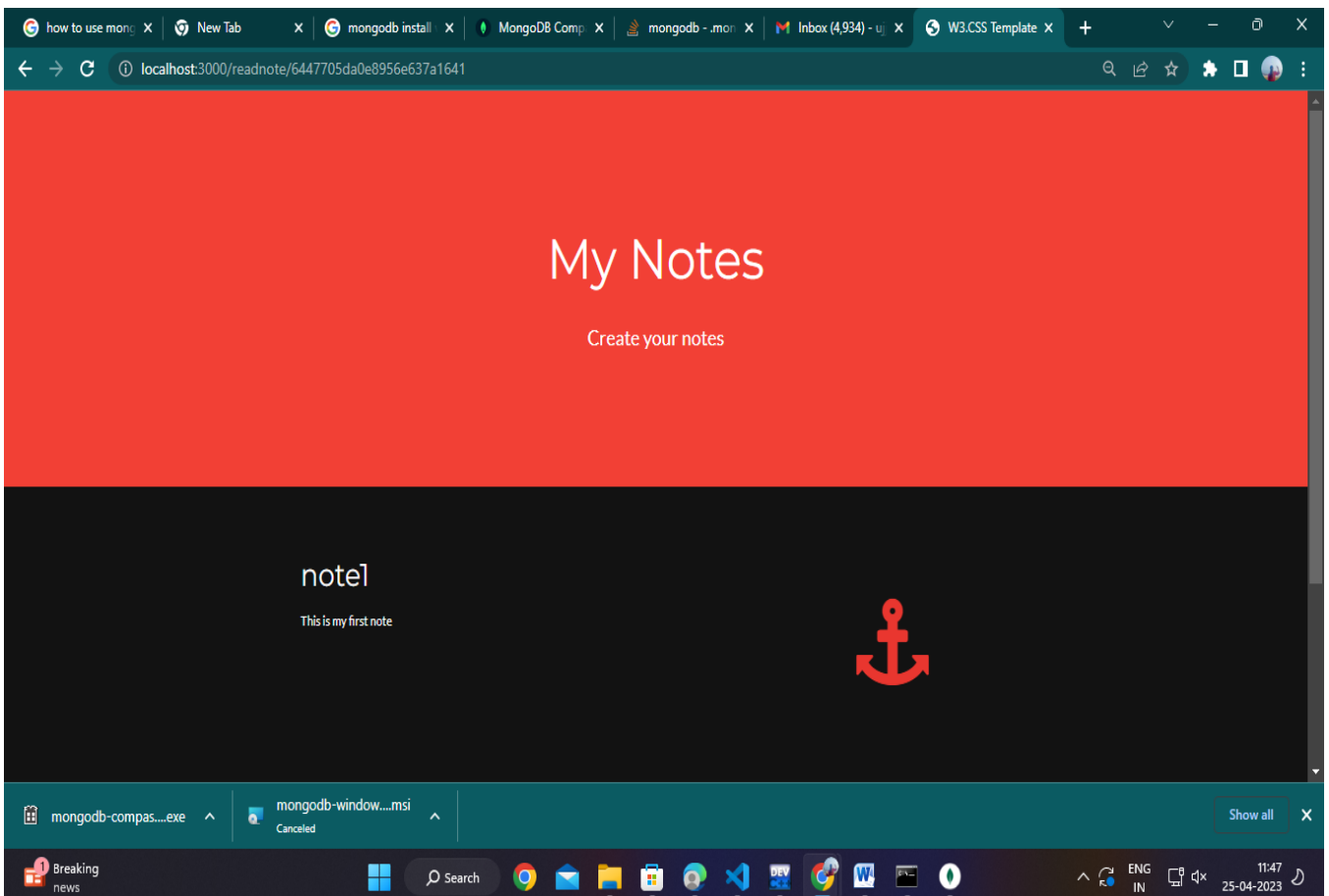
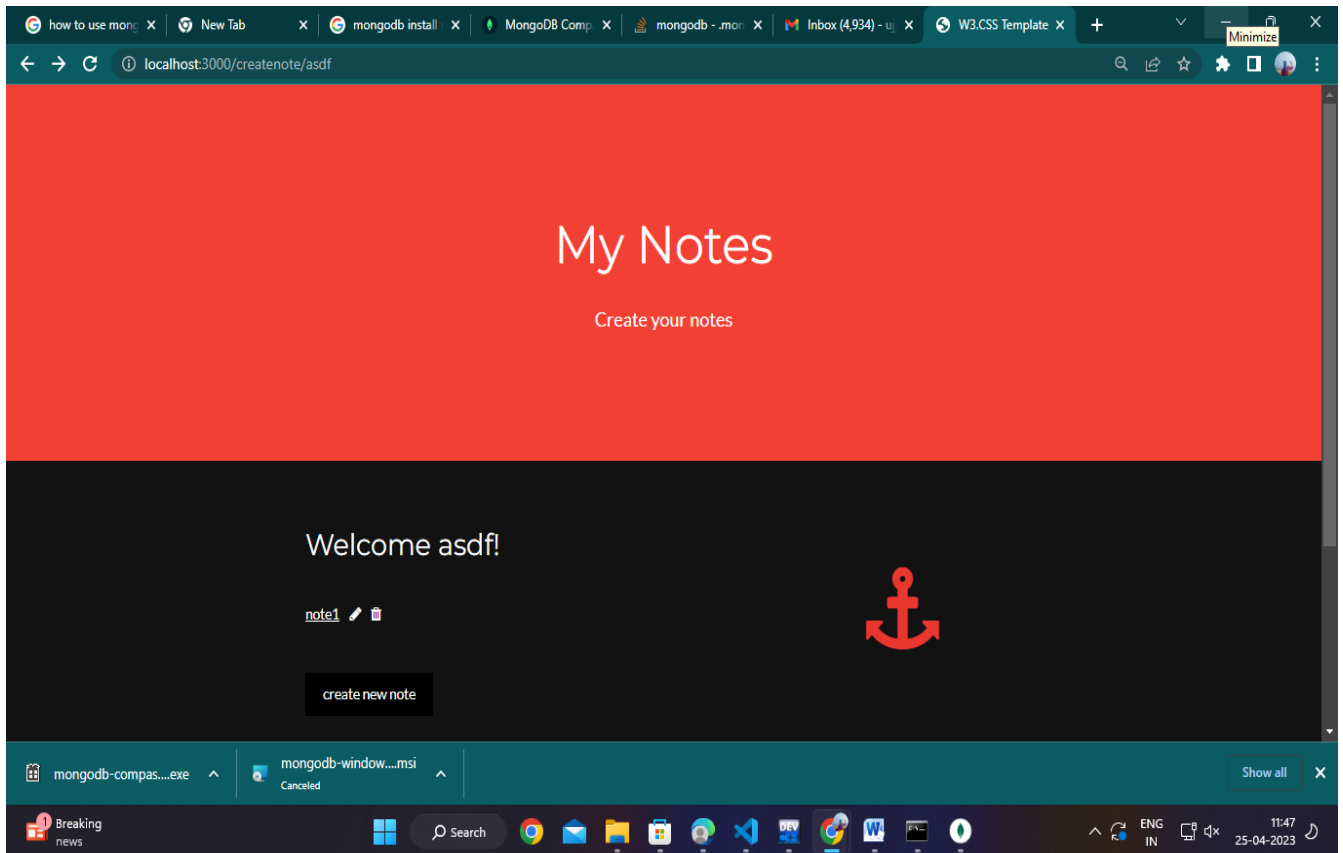
</script>

</body>

</html>

```

Output:



login.html

```

<!DOCTYPE html>

<html lang="en">

<head>

<title>W3.CSS Template</title>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">

<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Lato">

<link rel="stylesheet"

href="https://fonts.googleapis.com/css?family=Montserrat">

<link rel="stylesheet"

href="https://cdnjs.cloudflare.com/ajax/libs/fontawesome/4.7.0/css/font-awesome.min.css">

<style>

body,h1,h2,h3,h4,h5,h6 {font-family: "Lato", sans-serif}

.w3-bar,h1,button {font-family: "Montserrat", sans-serif}

.fa-anchor,.fa-coffee {font-size:200px}

</style>

</head>

<body>

<header class="w3-container w3-red w3-center" style="padding:128px 16px">

<h1 class="w3-margin w3-jumbo">My Notes </h1>

<p class="w3-xlarge">Create your notes</p>

</header>

<div class="w3-row-padding w3-padding-64 w3-container">

<div class="w3-content">

<div class="w3-twothird">

<h1>welcome !</h1>

<h5 class="w3-padding-32">note1. &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;<i class="fa

fapencil">&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;</i> <i class="fa fa-trash"></i></h5>

```

Output:

The screenshot shows the MongoDB Compass interface for the 'mynotesDatabase.users' collection. The left sidebar shows the database structure with 'users' selected. The main panel displays the 'Documents' tab with a single document. The document contains the following fields: '_id' (ObjectId), 'user' (string 'abc'), 'password' (string '1234'), and '__v' (0). The top right shows '1' document and '1' index. The bottom status bar shows the system tray with weather and taskbar icons.

MongoDB Compass - localhost:27017/mynotesDatabase.users

Connect Edit View Collection Help

localhost:27017

Documents mynotesDatabase...

My Queries

Databases

Search

admin

config

local

mynotesDatabase

notes

users

mynotesDatabase.users

1 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Explain Plan Indexes Validation

Filter Type a query: { field: 'value' }

Reset Find More Options

ADD DATA EXPORT COLLECTION

1 - 2 of 2

```
{
  "_id": ObjectId("64476b4c90f544b18288e440"),
  "user": "abc",
  "password": "1234",
  "__v": 0
}
```

```
{
  "_id": ObjectId("64477002a0e8956e637a163c"),
  "user": "asdf",
  "password": "0000",
  "__v": 0
}
```

>_MONGOSH

34°C Partly sunny

Search

ENG IN

11:47 25-04-2023

The screenshot shows the MongoDB Compass interface for the 'mynotesDatabase.notes' collection. The left sidebar shows the database structure with 'notes' selected. The main panel displays the 'Documents' tab with a single document. The document contains the following fields: '_id' (ObjectId), 'title' (string 'note1'), 'note' (string 'This is my first note'), 'user' (ObjectId), 'cdate' (timestamp), and '__v' (0). The top right shows '0' documents and '1' index. The bottom status bar shows the system tray with weather and taskbar icons.

MongoDB Compass - localhost:27017/mynotesDatabase.notes

Connect Edit View Collection Help

localhost:27017

Documents mynotesDatabase...

My Queries

Databases

Search

admin

config

local

mynotesDatabase

notes

users

mynotesDatabase.notes

0 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Explain Plan Indexes Validation

Filter Type a query: { field: 'value' }

Reset Find More Options

ADD DATA EXPORT COLLECTION

1 - 1 of 1

```
{
  "_id": ObjectId("6447705da0e8956e637a1641"),
  "title": "note1",
  "note": "This is my first note",
  "user": ObjectId("64477002a0e8956e637a163c"),
  "cdate": 2023-04-25T06:17:01.985+00:00,
  "__v": 0
}
```

>_MONGOSH

34°C Partly sunny

Search

ENG IN

11:47 25-04-2023

```

<h5 class="w3-padding-32">note2. &nbsp;&nbsp;&nbsp;<i class="fa
fapencil">&nbsp;&nbsp;&nbsp;</i> <i class="fa fa-trash"></i></h5>

<h5 class="w3-padding-32">note3. &nbsp;&nbsp;&nbsp;<i class="fa
fapencil">&nbsp;&nbsp;&nbsp;</i> <i class="fa fa-trash"></i></h5>

<button class="w3-button w3-black w3-padding-large w3-large w3-
margin-top">create new note</button>

</div>

<div class="w3-third w3-center">

<i class="fa fa-anchor w3-padding-64 w3-text-red"></i>

</div>

</div></div>

<div class="w3-container w3-black w3-center w3-opacity w3-padding-64">

<h1 class="w3-margin w3-xlarge">Quote of the day: live life</h1>

</div>

<footer class="w3-container w3-padding-64 w3-center w3-opacity">

<div class="w3-xlarge w3-padding-32">

<i class="fa fa-facebook-official w3-hover-opacity"></i>

<i class="fa fa-instagram w3-hover-opacity"></i>

<i class="fa fa-snapchat w3-hover-opacity"></i>

<i class="fa fa-pinterest-p w3-hover-opacity"></i>

<i class="fa fa-twitter w3-hover-opacity"></i>

<i class="fa fa-linkedin w3-hover-opacity"></i>

</div>

<p>Powered by <a href="https://www.w3schools.com/w3css/default.asp"
target="_blank">w3.css</a></p>

</footer>

<script>

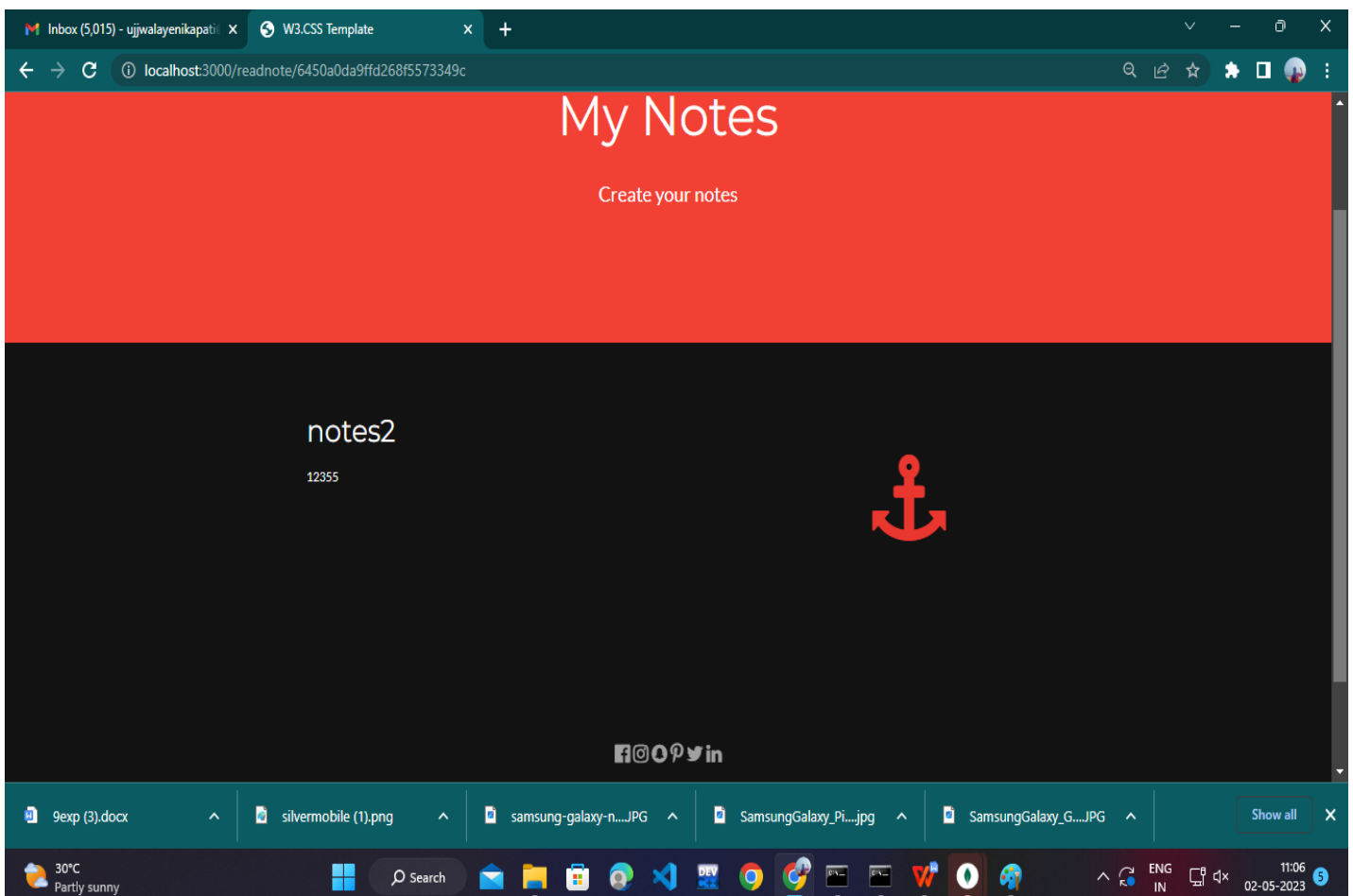
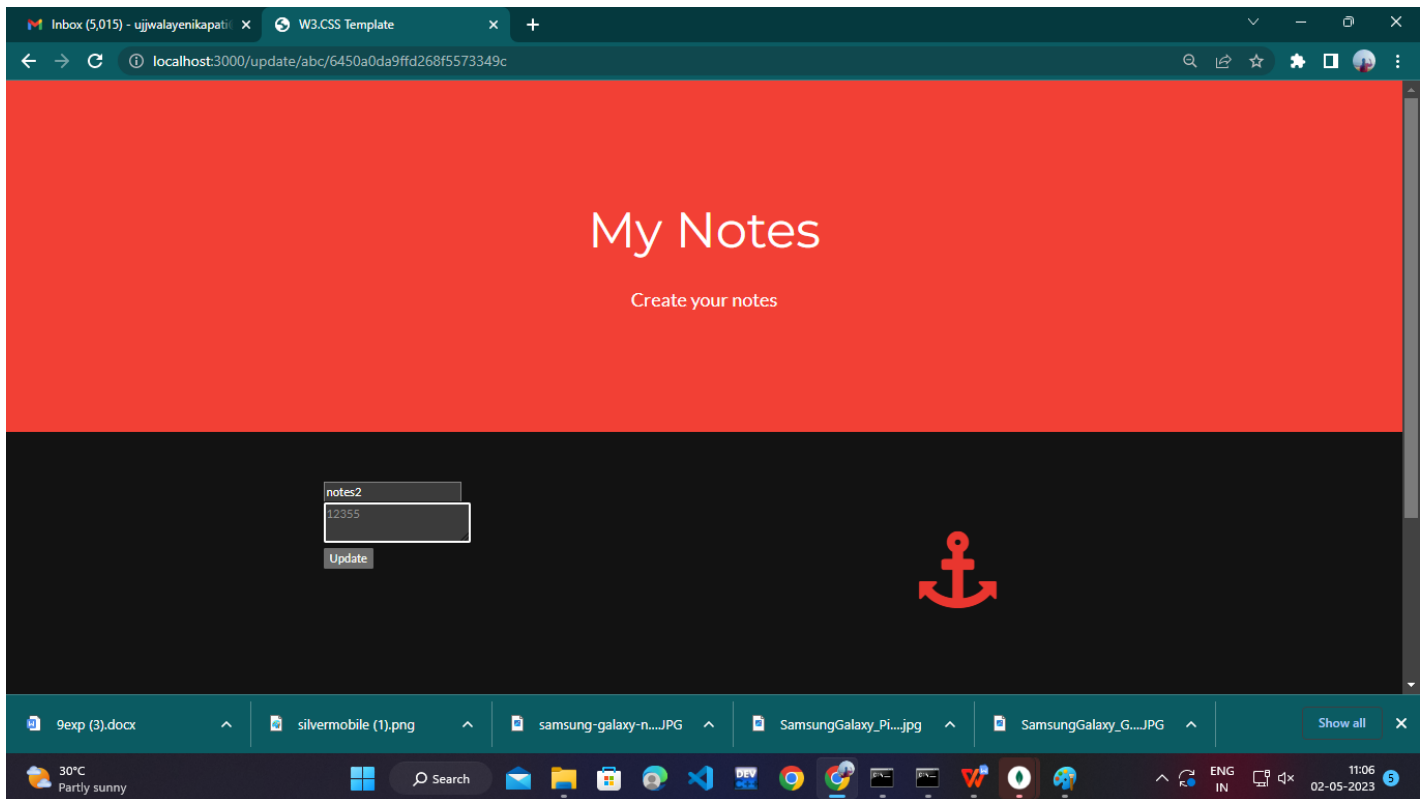
</script>

</body>

</html>

```

Output:



readnote.html

```

<!DOCTYPE html>

<html lang="en">

<head>

<title>W3.CSS Template</title>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">

<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Lato">

<link rel="stylesheet"

href="https://fonts.googleapis.com/css?family=Montserrat">

<link rel="stylesheet"

href="https://cdnjs.cloudflare.com/ajax/libs/fontawesome/4.7.0/css/font-awesome.min.css">

<style>

body,h1,h2,h3,h4,h5,h6 {font-family: "Lato", sans-serif}

.w3-bar,h1,button {font-family: "Montserrat", sans-serif}

.fa-anchor,.fa-coffee {font-size:200px}

</style>

</head>

<body>

<header class="w3-container w3-red w3-center" style="padding:128px 16px">

<h1 class="w3-margin w3-jumbo">My Notes </h1>

<p class="w3-xlarge">Create your notes</p>

</header>

<div class="w3-row-padding w3-padding-64 w3-container">

<div class="w3-content">

<div class="w3-twothird">

<h1>Note Heading</h1>

<p class="w3-text-grey">Lorem ipsum dolor sit amet, consectetur

```


adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco

laboris nisi ut aliquip ex ea commodo consequat.</p>

<button >Edit</button> <button >home</button>

</div>

<div class="w3-third w3-center">

<i class="fa fa-anchor w3-padding-64 w3-text-red"></i>

</div></div></div>

<div class="w3-container w3-black w3-center w3-opacity w3-padding-64">

<h1 class="w3-margin w3-xlarge">Quote of the day: live life</h1>

</div>

<footer class="w3-container w3-padding-64 w3-center w3-opacity">

<div class="w3-xlarge w3-padding-32">

<i class="fa fa-facebook-official w3-hover-opacity"></i>

<i class="fa fa-instagram w3-hover-opacity"></i>

<i class="fa fa-snapchat w3-hover-opacity"></i>

<i class="fa fa-pinterest-p w3-hover-opacity"></i>

<i class="fa fa-twitter w3-hover-opacity"></i>

<i class="fa fa-linkedin w3-hover-opacity"></i>

</div>

<p>Powered by <a href="https://www.w3schools.com/w3css/default.asp"

target="_blank">w3.css</p>

</footer>

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

updatenote.html

```

<!DOCTYPE html>

<html lang="en">

<head>

<title>W3.CSS Template</title>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">

<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Lato">

<link rel="stylesheet"

href="https://fonts.googleapis.com/css?family=Montserrat">

<link rel="stylesheet"

href="https://cdnjs.cloudflare.com/ajax/libs/fontawesome/4.7.0/css/font-awesome.min.css">

<style>

body,h1,h2,h3,h4,h5,h6 {font-family: "Lato", sans-serif}

.w3-bar,h1,button {font-family: "Montserrat", sans-serif}

.fa-anchor,.fa-coffee {font-size:200px}

</style>

</head>

<body>

<header class="w3-container w3-red w3-center" style="padding:128px 16px">

<h1 class="w3-margin w3-jumbo">My Notes </h1>

<p class="w3-xlarge">Create your notes</p>

</header>

<div class="w3-row-padding w3-padding-64 w3-container">

<div class="w3-content">

<div class="w3-twothird">

<form>

<input type="text" name="title" placeholder="title"><br>

```



```

<textarea name="note" class="w3-text-grey" placeholder="type
here"></textarea><br>
<input type="submit" value="update"></input>
</form>
</div>
<div class="w3-third w3-center">
<i class="fa fa-anchor w3-padding-64 w3-text-red"></i>
</div>
</div>
</div>
<div class="w3-container w3-black w3-center w3-opacity w3-padding-64">
<h1 class="w3-margin w3-xlarge">Quote of the day: live life</h1>
</div>
<footer class="w3-container w3-padding-64 w3-center w3-opacity">
<div class="w3-xlarge w3-padding-32">
<i class="fa fa-facebook-official w3-hover-opacity"></i>
<i class="fa fa-instagram w3-hover-opacity"></i>
<i class="fa fa-snapchat w3-hover-opacity"></i>
<i class="fa fa-pinterest-p w3-hover-opacity"></i>
<i class="fa fa-twitter w3-hover-opacity"></i>
<i class="fa fa-linkedin w3-hover-opacity"></i>
</div>
<p>Powered by <a href="https://www.w3schools.com/w3css/default.asp"
target="_blank">w3.css</a></p>
</footer>
<script>
</script>
</body>
</html>

```


routes.js

```

const express = require('express')
const router = express.Router()
const path = require('path')
const mongoose = require('mongoose')
const UserModel = require('../models/user')
const NoteModel = require('../models/note')

//controller function

async function get_user_notes(user){
  let user_rec = await UserModel.findOne({user:user})
  let notes = await NoteModel.find({user:user_rec._id})
  return notes;
}

async function get_user_id(user){
  let user_rec = await UserModel.findOne({user:user})
  return user_rec._id;
}

// middleware that is specific to this router
router.use((req, res, next) => {
  console.log('Time: ', Date.now())
  next()
})

// define the home page route
router.get('/', (req, res) => {
  res.render('homepage')
})

router.get('/login', (req, res) => {
  res.render('login')
})

```



```

router.get('/register', (req, res) => {
  res.render('register')
})

// define the about route
router.post('/login', (req, res) => {
  //process form data
  let userid = req.body.user;
  let pass = req.body.password;
  UserModel.findOne({user : userid}).then((user)=>{
    console.log(user)
    if(user.password == pass){
      NoteModel.find({user: user._id}).then((notes) => {
        res.cookie("user", userid, { signed : true })
        res.render('user_home',{user : user.user, notes:notes})
      })
    }
  })
  //check user exist or not get password
  //user not found
  //compare password
  //invalid user
})

router.post("/createuser",(req,res)=>{
  let userid = req.body.user;
  let password = req.body.password;
  let new_account = new UserModel({user: userid, password: password})
  new_account.save().then(()=>{console.log(`${userid} created.`)})
  res.status(200).render('login')
})

```



```

router.get("/createnote/",(req,res)=>{
  res.render('createnote', {user: req.signedCookies.user})
})

router.post("/createnote/",async (req,res)=>{
  let title = req.body.title;
  let note = req.body.note;
  let user = req.signedCookies.user;
  //let user = req.params.user;
  let user_rec = await UserModel.findOne({user: user})
  let new_note = new NoteModel({title: title, note: note, user: user_rec._id})
  await new_note.save()
  console.log(`${title} created.`)
  let user_notes = await get_user_notes(user);
  res.status(200).render('user_home',{user:req.params.user,notes:user_notes})
})

router.get("/delete/:noteid",async (req,res)=>{
  let noteid=req.params.noteid;
  let user= req.signedCookies.user;
  await NoteModel.deleteOne({_id : noteid})
  let user_notes = await get_user_notes(user);
  res.render('user_home',{user:user,notes:user_notes})
})

router.get("/update/:noteid",(req,res)=>{
  let noteid=req.params.noteid;
  let user= req.signedCookies.user;
  NoteModel.findOne({_id : noteid}).then((note)=>{
    res.render('updatenote',{user:user,note:note})
  })
})

```



```

router.post("/update/:noteid",async (req,res)=>{
  let noteid=req.params.noteid;
  let user= req.signedCookies.user;
  let title = req.body.title;
  let note= req.body.note;
  await NoteModel.updateOne({_id: noteid}, {title : title, note: note})
  let user_notes =await get_user_notes(user);
  res.render('user_home',{user:user,notes:user_notes})
})

router.get("/readnote/:noteid", async (req,res) => {
  let note_rec = await NoteModel.findOne({_id : req.params.noteid})
  res.render('readnote',{title : note_rec.title,note : note_rec.note})
})

router.get("/user_home",async (req,res)=>{
  let user= req.signedCookies.user;
  let user_notes = await get_user_notes(user);
  res.render('user_home',{user:user,notes:user_notes})
})

router.get("/logout",async (req,res)=>{
  res.clearCookie('user')
  res.render('login')
})

module.exports = router

```


note.js

```

const mongoose = require('mongoose')

const NoteSchema = new mongoose.Schema({
  title: {type: String, required: true},
  note: {type: String, required: true},
  user: {type: mongoose.Schema.Types.ObjectId, ref:'user'},
  cdate: {type: Date, default: Date.now}}})

const NoteModel = mongoose.model('Note',NoteSchema)

module.exports = NoteModel

```

user.js

```

const mongoose = require('mongoose')

const UserSchema = new mongoose.Schema({
  user: { type: String, required: true} ,
  password: { type: String, required: true}
})

const UserModel = mongoose.model('User',UserSchema)

module.exports = UserModel

```

index.js

```

const express = require('express')

const path = require('path')

const helmet = require('helmet')

const app = express()

const bodyParser = require('body-parser')

const cookieParser = require('cookie-parser')

const my_router = require("./routes/routes.js");

const mongoose = require('mongoose');

const mongoDB = "mongodb://127.0.0.1:27017/mynotesDatabase"

```



```

app.set('views', path.join(__dirname, 'views'));
app.set('view engine', 'ejs');
//app.use(helmet())
app.use(cookieParser('secret5654'))
app.use(bodyParser.urlencoded({ extended: false }));
app.use(express.static('public'));
app.use(express.json())
const conn =mongoose.connect(mongoDB, {})
app.use("/", my_router);
app.listen(3000);

```

EXP-8 .ejs files

FOOTER.ejs

```

<footer class="w3-container w3-padding-64 w3-center w3-opacity">
<div class="w3-xlarge w3-padding-32">
<i class="fa fa-facebook-official w3-hover-opacity"></i>
<i class="fa fa-instagram w3-hover-opacity"></i>
<i class="fa fa-snapchat w3-hover-opacity"></i>
<i class="fa fa-pinterest-p w3-hover-opacity"></i>
<i class="fa fa-twitter w3-hover-opacity"></i>
<i class="fa fa-linkedin w3-hover-opacity"></i>
</div>
<p>Powered by <a href="https://www.w3schools.com/w3css/default.asp"
target="_blank">w3.css</a></p>
</footer>
<script>
</script>
</body>
</html>

```


header.ejs

```

<!DOCTYPE html>

<html lang="en">

<head>

<title>W3.CSS Template</title>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">

<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Lato">

<link rel="stylesheet"

href="https://fonts.googleapis.com/css?family=Montserrat">

<link rel="stylesheet"

href="https://cdnjs.cloudflare.com/ajax/libs/fontawesome/4.7.0/css/font-awesome.min.css">

<style>

body,h1,h2,h3,h4,h5,h6 {font-family: "Lato", sans-serif}

.w3-bar,h1,button {font-family: "Montserrat", sans-serif}

.fa-anchor,.fa-coffee {font-size:100px}

</style>

</head>

<body>

<!-- Header -->

<header class="w3-container w3-red w3-center" style="padding:128px 16px">

<h1 class="w3-margin w3-jumbo">My Notes </h1>

<p class="w3-xlarge">Create your notes</p>

</header>

```


createnote.ejs

```

<%- include('./partials/header.ejs') %>

<!-- First Grid -->

<div class="w3-row-padding w3-padding-64 w3-container">

<div class="w3-content">

<div class="w3-twothird">

<form action="/createnote/" method="post">

<input type="text" name="title" placeholder="title"><br>

<textarea name="note" class="w3-text-grey" placeholder="type
here"></textarea><br>

<input type="submit"></input>

</form>

</div>

<div class="w3-third w3-center">

<i class="fa fa-anchor w3-padding-64 w3-text-red"></i>

</div>

</div>

</div>

<%- include('./partials/footer.ejs') %>

```


homepage.ejs

```

<%- include('./partials/header.ejs') %>

<!-- First Grid -->

<div class="w3-row-padding w3-padding-64 w3-container">

<div class="w3-content">

<div class="w3-twothird">

<h1>Create to your account</h1>

<a class="w3-button w3-black w3-padding-large w3-large w3-margin-top"
href="/register">Register Here</a>

<h1>Login to your account</h1>

<a class="w3-button w3-black w3-padding-large w3-large w3-margin-top"
href="/login">Login Here</a>

</div>

<div class="w3-third w3-center">

<i class="fa fa-anchor w3-padding-64 w3-text-red"></i>

</div>

</div>

</div>

<%- include('./partials/footer.ejs') %>

```


login.ejs

```

<%- include('./partials/header.ejs') %>

<!-- First Grid -->

<div class="w3-row-padding w3-padding-64 w3-container">

<div class="w3-content">

<div class="w3-twothird">

<h1>Login to your account</h1>

<form action="/login" method="post">

<label for="user">USER ID:</label>

<input type="text" name="user" id="user"><br /><br />

<label for="password">PASSWORD:</label>

<input type="password" name="password" id="password"><br /><br />

<input type="submit" value="LOGIN" >&nbsp;

<input type="reset" value="RESET">

</form>

</div>

<div class="w3-third w3-center">

<i class="fa fa-anchor w3-padding-64 w3-text-red"></i>

</div>

</div>

</div>

<%- include('./partials/footer.ejs') %>

```


readnote.ejs

```

<%- include('./partials/header.ejs') %>

<!-- First Grid -->

<div class="w3-row-padding w3-padding-64 w3-container">

<div class="w3-content">

<div class="w3-twothird">

<h1><%= title %></h1>

<p><%= note %></p>

<a class = "w3-button w3-black w3-padding-large w3-large w3-margintop" href =
"/user_home">go to home</a>

</div>

<div class="w3-third w3-center">

<i class="fa fa-anchor w3-padding-64 w3-text-red"></i>

</div>

</div>

</div>

</div>

<%- include('./partials/footer.ejs') %>

```


register.ejs

```

<%- include('./partials/header.ejs') %>

<!-- First Grid -->

<div class="w3-row-padding w3-padding-64 w3-container">

<div class="w3-content">

<div class="w3-twothird">

<h1>Create to your account</h1>

<form action="/createuser" method="post">

<label for="user">USER ID:</label>

<input type="text" name="user" id="user"><br /><br />

<label for="password">PASSWORD:</label>

<input type="password" name="password" id="password"><br /><br />

<input type="submit" value="Register" >&nbsp;

<input type="reset" value="RESET">

</form>

</div>

<div class="w3-third w3-center">

<i class="fa fa-anchor w3-padding-64 w3-text-red"></i>

</div>

</div>

</div>

<div>

<%- include('./partials/footer.ejs') %>

```


updatenote.ejs

```

<%- include('./partials/header.ejs') %>

<!-- First Grid -->

<div class="w3-row-padding w3-padding-64 w3-container">

<div class="w3-content">

<div class="w3-twothird">

<form action="/update/<%= note._id%>" method="post">

<input type="text" name="title" placeholder="title" value="<%=
note.title%>"><br>

<textarea name="note" class="w3-text-grey" placeholder="type here"><%=
note.note %></textarea><br>

<input type="submit" value="Update"></input>

</form>

</div>

<div class="w3-third w3-center">

<i class="fa fa-anchor w3-padding-64 w3-text-red"></i>

</div>

</div>

</div>

<%- include('./partials/footer.ejs') %>

```


9. (a) 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.

mobile.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">

<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;

}

</style>

</head>

<body>
```

Output:

The screenshot shows a web browser window with the address bar displaying the URL `C:/Users/aliet/Desktop/mstlab/exp-9/mobile.html`. The page title is "Mobile Cart". The main content area features a Samsung Galaxy Note 7 smartphone on the left. To its right are three color selection circles: GoldPlatinum, PinkGold, and SilverTitanium, each with a corresponding price: \$250, \$280, and \$300 respectively. Below the phone image, the text "Price: \$", "Status:", and "Discount:" is visible. A green "Add to Cart" button and a blue "Back" link are positioned below the pricing information. The description text reads: "Samsung Galaxy Note 7 is a stylish mobile you can ever have. It has 64GB memory." The Windows taskbar at the bottom shows the date as 02-05-2023 and the time as 11:06.

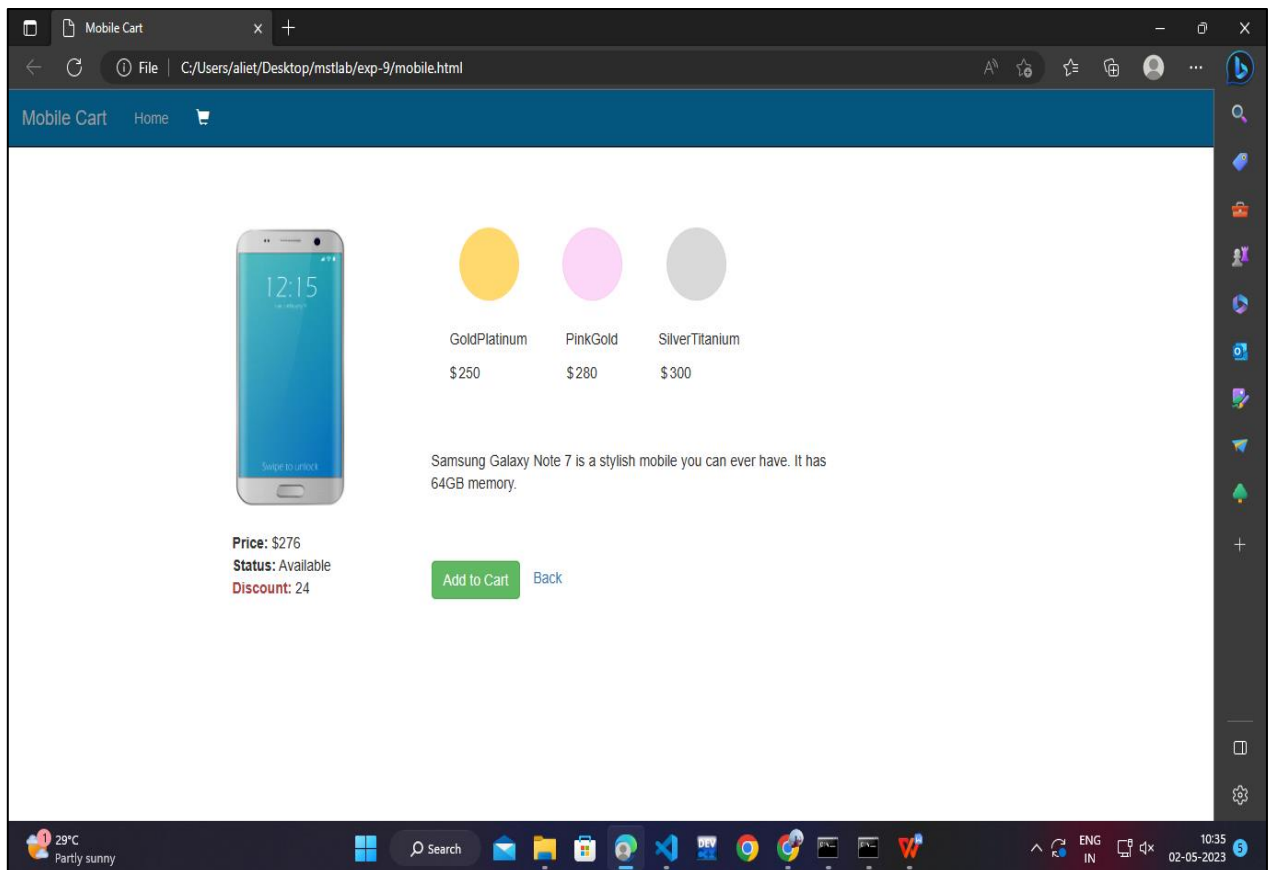
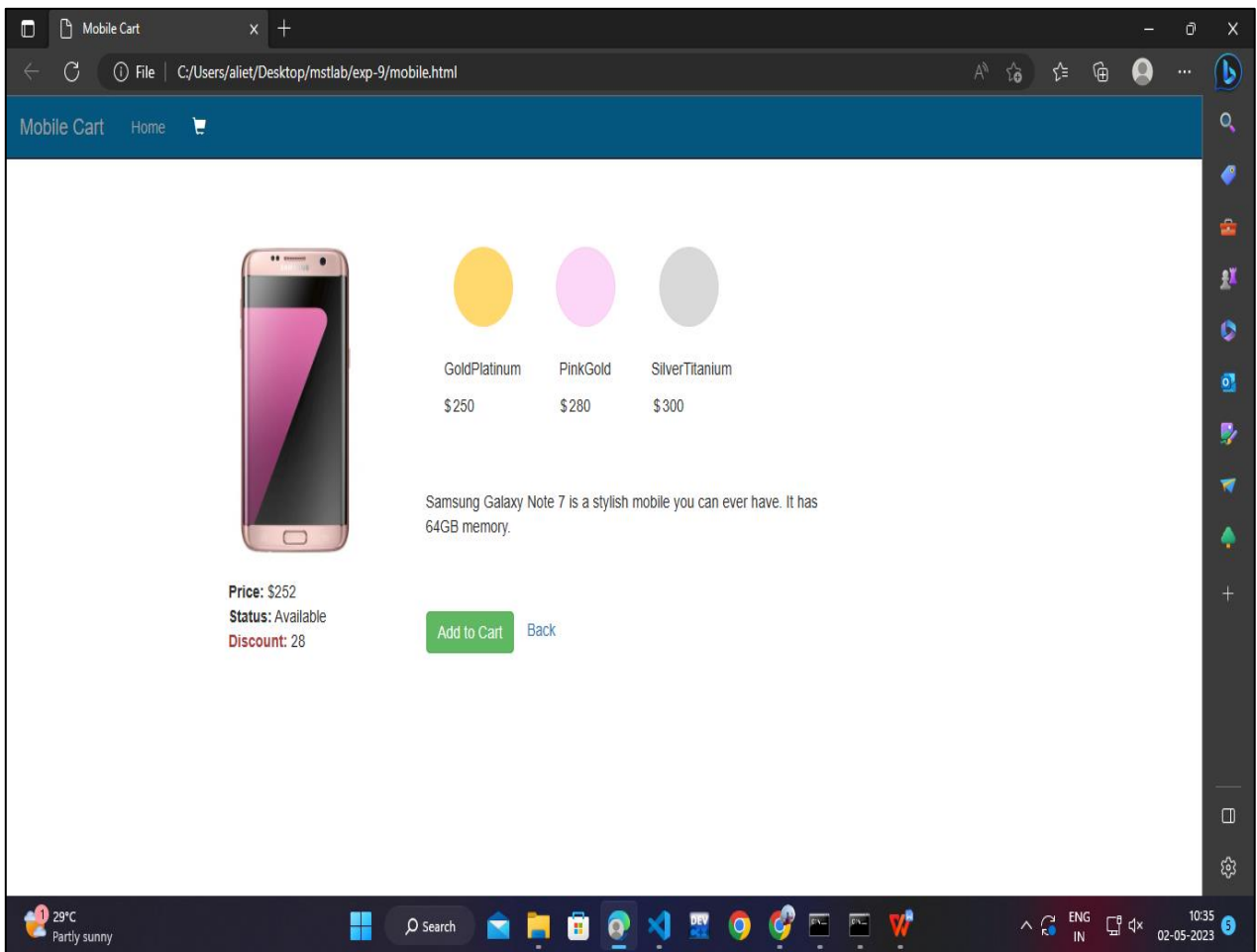
The screenshot shows a web browser window with the address bar displaying the URL `C:/Users/aliet/Desktop/mstlab/exp-9/mobile.html`. The page title is "Mobile Cart". The main content area features a Samsung Galaxy Note 7 smartphone on the left. To its right are three color selection circles: GoldPlatinum, PinkGold, and SilverTitanium, each with a corresponding price: \$250, \$280, and \$300 respectively. Below the phone image, the text "Price: \$237.5", "Status: Available", and "Discount: 12.5" is visible. A green "Add to Cart" button and a blue "Back" link are positioned below the pricing information. The description text reads: "Samsung Galaxy Note 7 is a stylish mobile you can ever have. It has 64GB memory." The Windows taskbar at the bottom shows the date as 02-05-2023 and the time as 10:35.

```

<nav class="navbar navbar-inverse navbar-fixed-top">
<div class="navbar-header">
<button type="button" class="navbar-toggle collapsed" datatoggle="collapse" data-
target="#navbar" aria-expanded="false" ariacontrols="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="Index.html">Home</a></li>
</ul>
<!--/.nav-collapse -->
<ul class="nav navbar-nav navbar-middle" style="color:white; marginright:30px;">
<li><a href="Cart.html"><span class="glyphicon glyphicon-shoppingcart"
style="color:white"></span></a></li>
</ul>
</div>
</nav>
<div class="container" style="margin-top:7%">
<div class="row">
<div class="col-sm-4">
<div style="margin-left:40%;padding-top:15px;">
<div style="padding:15px;">
<div>
<img id="phoneImage" src ="Images/Part
1/samsung_edge_silver.jpg" height="250px" >

```

Output:



```

</div>

<div style="padding-top:10px;">
<div><b><span id="pName"></span></b></div>

<div style="padding-top:10px;"><b>Price:</b>&nbsp;&dollar;<span
id="pPrice"></span></div>

<div><b>Status:</b>&nbsp;<span
id="pAvailable"></span></div>

<div><b class="text-danger">Discount:</b>&nbsp;<span
id="pDiscount"></span></div>

</div>

</div>

</div>

</div>

<div></div>

<div style="padding-top:15px;">
<div>

<img src ="Images/Part 1/goldmobile.png" style="paddingleft:15px;cursor:pointer;"
onclick="getMobileByColor('GoldPlatinum');" >

<img src ="Images/Part 1/pinkmobile.png" style="cursor:pointer;paddingleft:15px;"
onclick="getMobileByColor('PinkGold');" >

<img src ="Images/Part 1/silvermobile.png" style="cursor:pointer;paddingleft:15px;"
onclick="getMobileByColor('SilverTitanium');" >

</div>

<div style="padding-top:10px;">
<span style="padding-left:20px;">
GoldPlatinum
</span>

<span style="padding-left:38px;">
PinkGold
</span>

```



```

<span style="padding-left:40px;">
SilverTitanium
</span>
</div>

<div style="padding-top:10px;">
<span style="padding-left: 20px;">&dollar;</span><span id="GoldPlatinum"
style="padding-left: 2px;"></span>
<span style="padding-left: 90px;">&dollar;</span><span id="PinkGold"
style="padding-left: 2px;"></span>
<span style="padding-left: 65px;">&dollar;</span><span
id="SilverTitanium" style="padding-left: 2px;"></span>
<div id="productDescription" style="padding-top:5%;width:70%;textalign:justify">
Samsung Galaxy Note 7 is a stylish mobile you can ever have. It has 64GB
memory.
</div>

<div style="padding-top:5%;padding-right:10%">
<button class="btn btn-success" onclick="addtoCart();">Add to
Cart</button>

<a style="padding-left:10px;" onclick="backHome();">Back</a>
</body>

<script
src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.2.1/jquery.min.js"></scri
pt>

<script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></sc
ript>

<!-- Adding converted js code-->
<script src="productdetails.js"></script>
</html>

```


productdetails.ts

```
// declaring enum variable and assigning default values
enum MobilePrice {GoldPlatinum=250, PinkGold=280, SilverTitanium=300}

document.getElementById("GoldPlatinum").innerHTML=MobilePrice.GoldPlatinum.toStri
ng();

document.getElementById("PinkGold").innerHTML=MobilePrice.PinkGold.toString();

document.getElementById("SilverTitanium").innerHTML=MobilePrice.SilverTitanium.to
String();

function getMobileByColor(color: string): void{
  if(color == "GoldPlatinum"){
    let price: number;
    price = calculateAmount(MobilePrice.GoldPlatinum);
    //$("#phoneImage").attr('src', '/images/Part 1/SamsungGalaxy_Gold.JPG');
    document.getElementById("phoneImage")?.setAttribute("src","Images/Part
1/SamsungGalaxy_Gold.JPG");
    document.getElementById("pPrice").innerHTML=price.toString();
    document.getElementById("pAvailable").innerHTML="Available";
    document.getElementById("pDiscount").innerHTML=(MobilePrice.GoldPlatinumprice).toSt
ring();
  }
  else if(color == "PinkGold"){
    let price: number;
    price = calculateAmount(MobilePrice.PinkGold);
    document.getElementById("phoneImage")?.setAttribute("src","Images/Part
1/SamsungGalaxy_Pink.jpg");
    document.getElementById("pPrice").innerHTML=price.toString();
    document.getElementById("pAvailable").innerHTML="Available";
    document.getElementById("pDiscount").innerHTML=(MobilePrice.PinkGoldprice).toString(
);
  }
}
```



```

else {
    let price: number;
    price = calculateAmount(MobilePrice.SilverTitanium);
    document.getElementById("phoneImage")?.setAttribute("src", "Images/Part
1/samsung_edge_silver.JPG");
    document.getElementById("pPrice").innerHTML=price.toString();
    document.getElementById("pAvailable").innerHTML="Available";
    document.getElementById("pDiscount").innerHTML=(MobilePrice.SilverTitaniumprice).toS
tring();
}}
// functon to calculate final amount
function calculateAmount(price: number): number {
    let discount: number;
    let totalAmount: number;
    if (price === MobilePrice.GoldPlatinum) {
        discount = 5;
    } else if (price === MobilePrice.SilverTitanium) {
        discount = 8;
    }
    else {
        discount = 10;
    }
    totalAmount = price - price * discount / 100;
    return totalAmount;
}
// lines to populate the Actual and Final price of Black color Mobile
console.log('Actual price of the Mobile is $' + MobilePrice.GoldPlatinum);
console.log('The final price after discount is $' +
calculateAmount(MobilePrice.GoldPlatinum));

```

