

SRI GURU GOBIND SINGH COLLEGE OF COMMERCE

PRACTICAL FILE

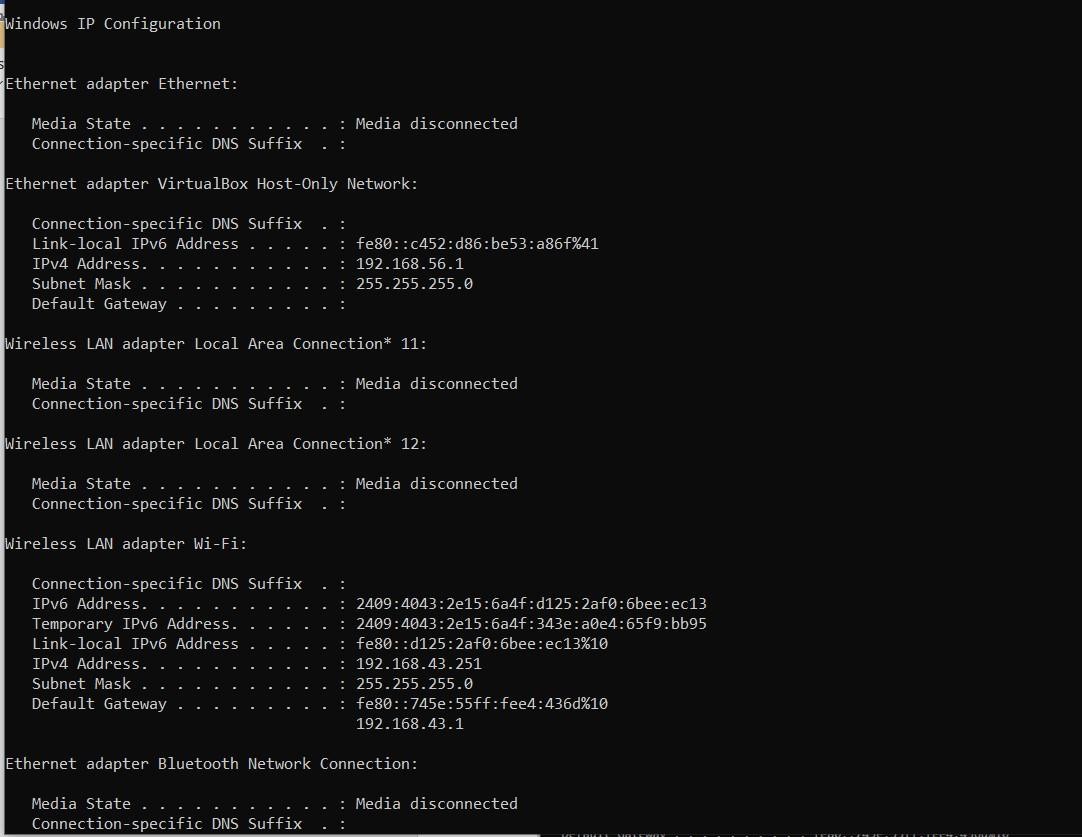
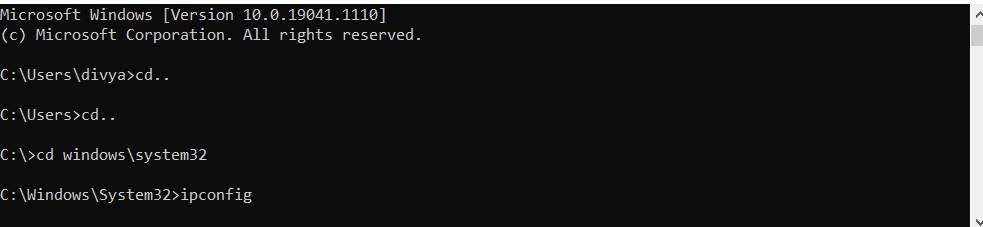
Internet Technologies

**Name- Ayush Gupta**

**Roll No. – 20078570017**

**Ques1.** Display your systems IP Address, Subnet mask using ipconfig, and find out the network address and the maximum number of systems possible on your network and range of IP addresses available to these systems.

**Ans.**

**Systems IP Address and Subnet mask using ipconfig:**

**Network address:** 192.168.43.0

**Maximum number of systems possible on network:**

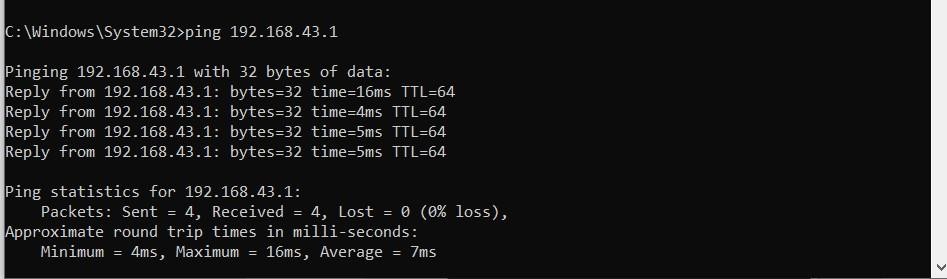
Total no of hosts = 2^8 = 256

Usable no of hosts = 254 (as 2 hosts are reserved)

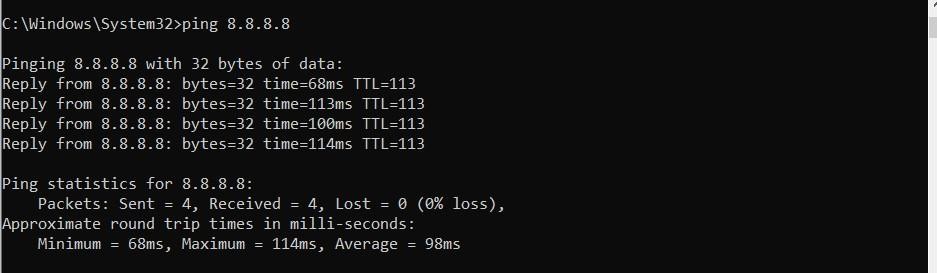
**Range of IP addresses available to these systems:** 192.168.43.1 to 192.168.43.254

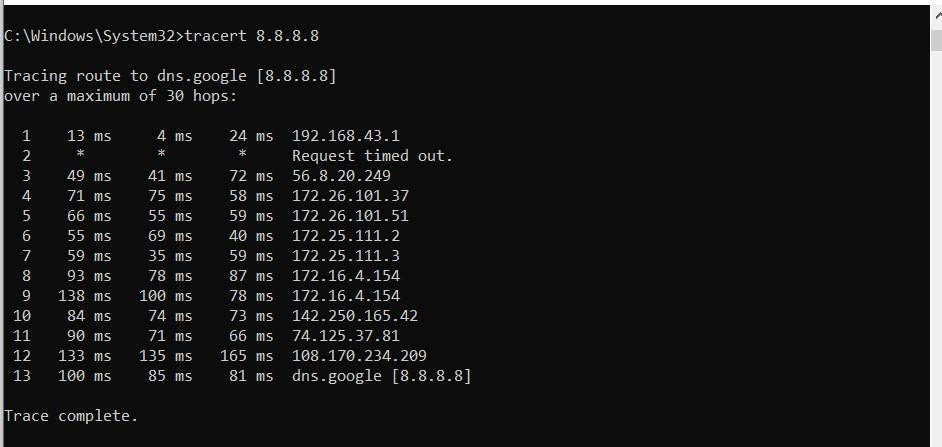
**Ques2.** With help of ping, check if you are connected to other systems of your network and find the route to connect to that system using tracert. List all the processes which are using ports for TCP protocol.

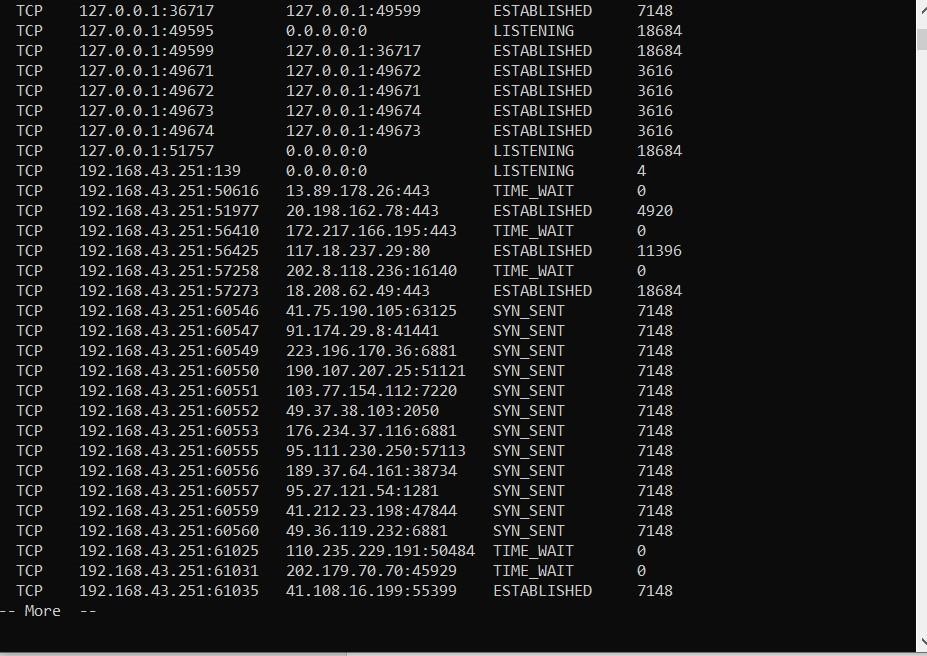
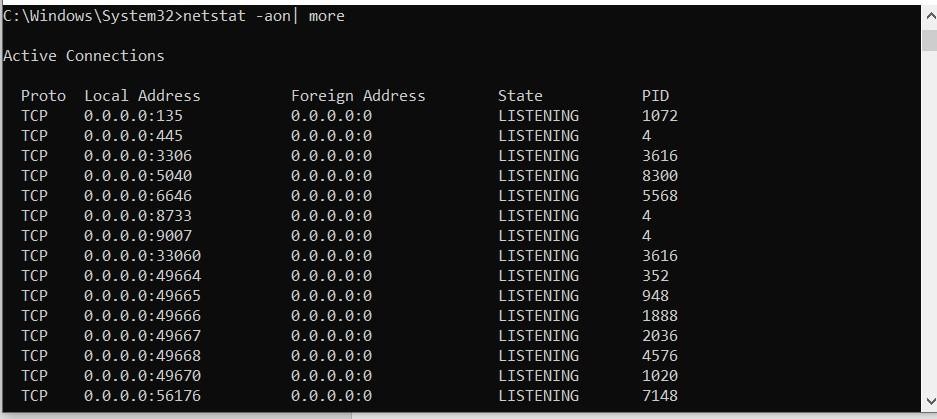
**Ans.**

**Connected to other systems of network:**

**Route to connect to that system using tracert: Example of Google**



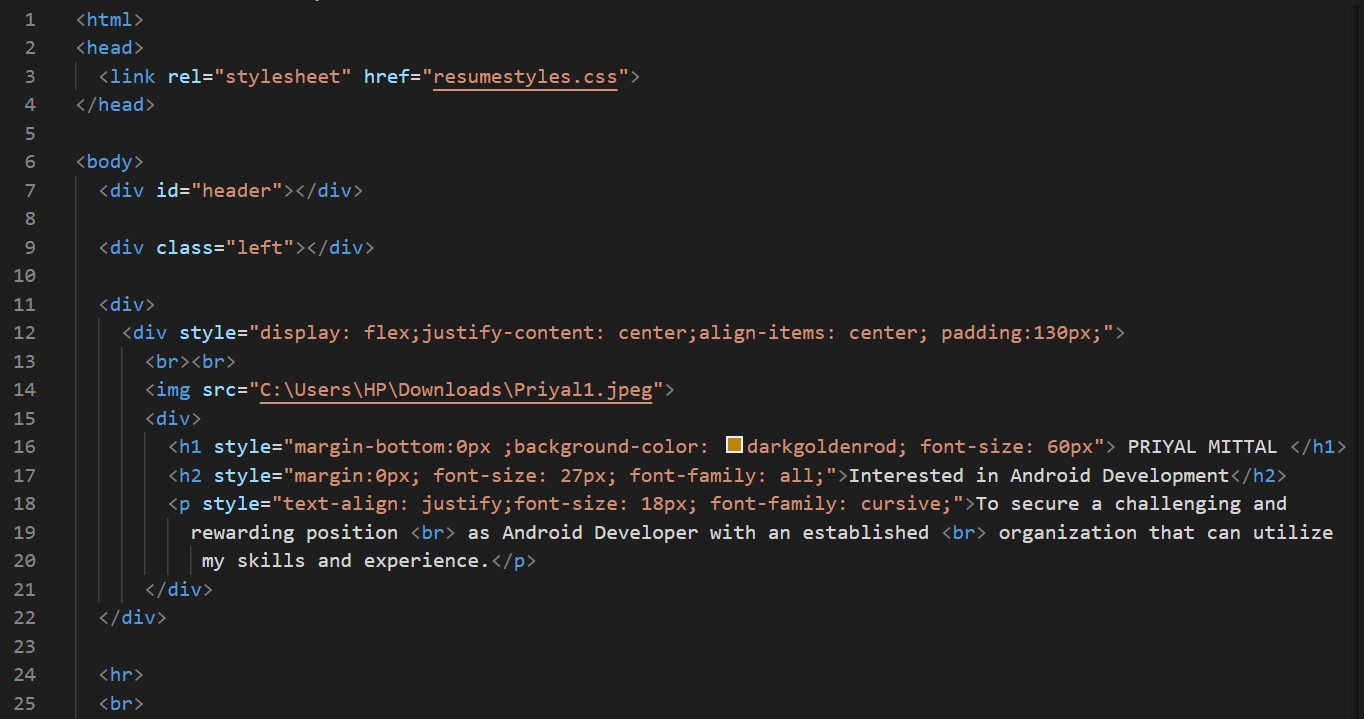


**List all the processes which are using ports for TCP protocol:**

**Ques3**. Create an HTML page that shows information about you, your course, hobbies, address, and your plans. Use CSS for styling of HTML page so that looks nice.

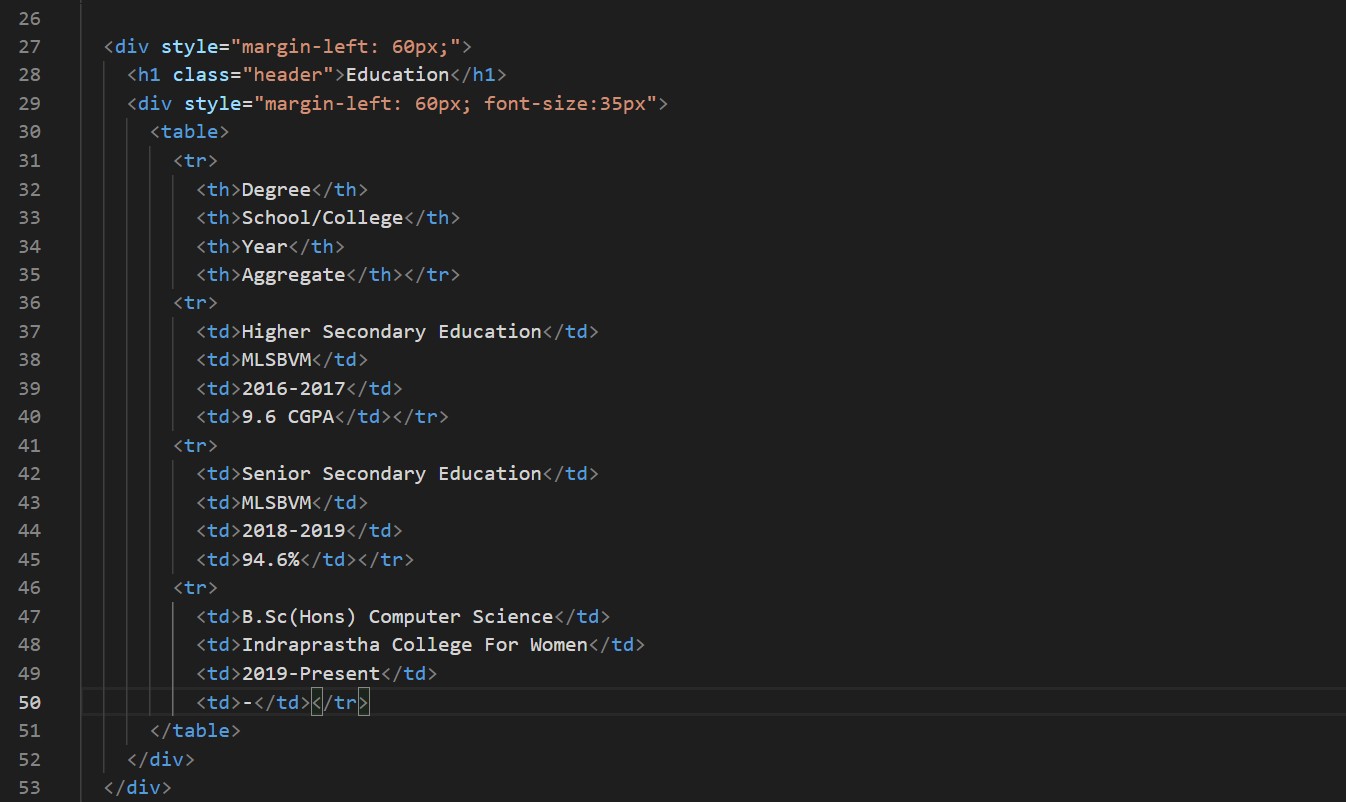
**Ans.**

**.html**



**Anjali.jpeg**

ANJALI PATHAK

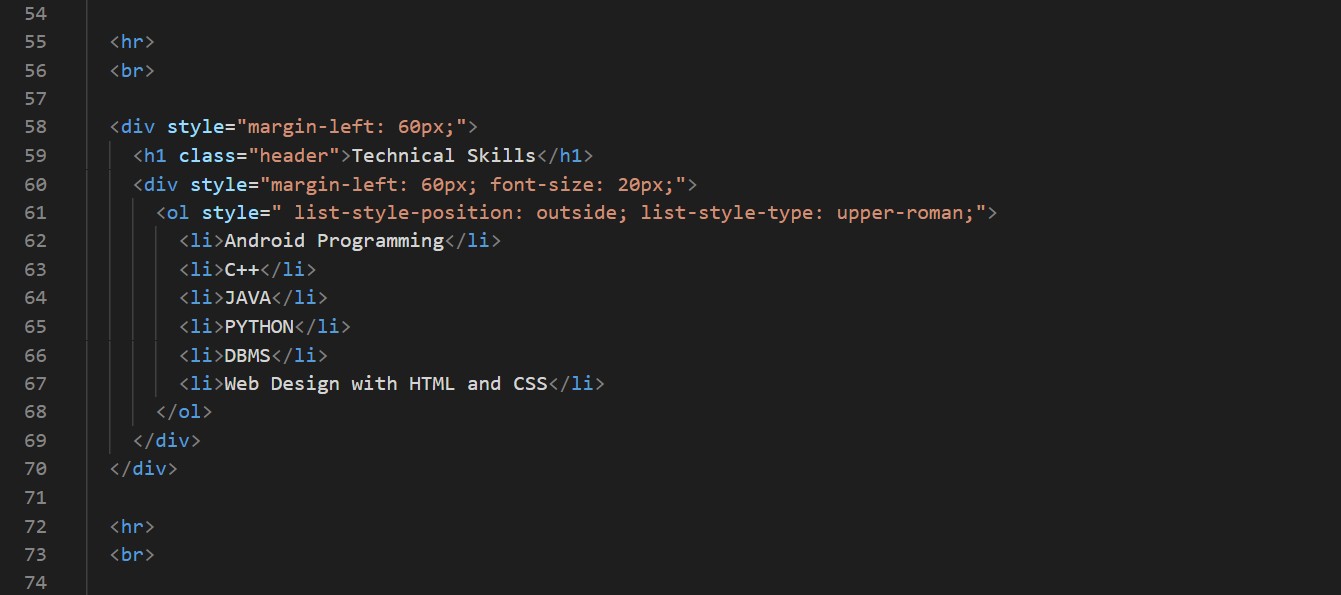


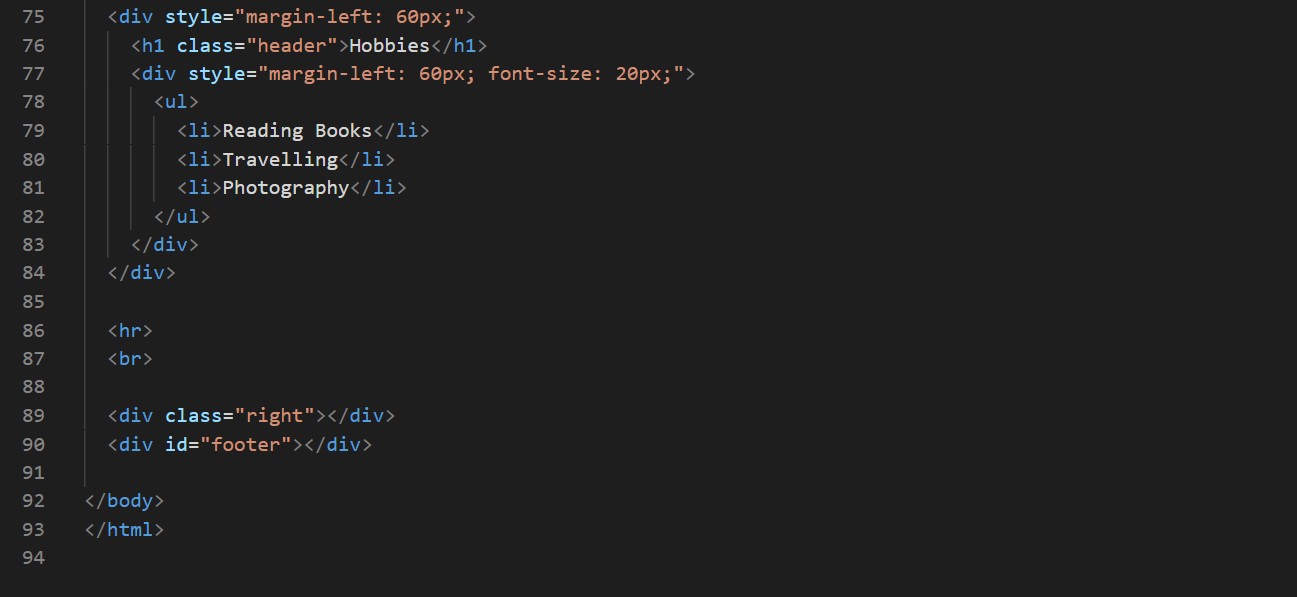
2015 - 2016

10 CGPA

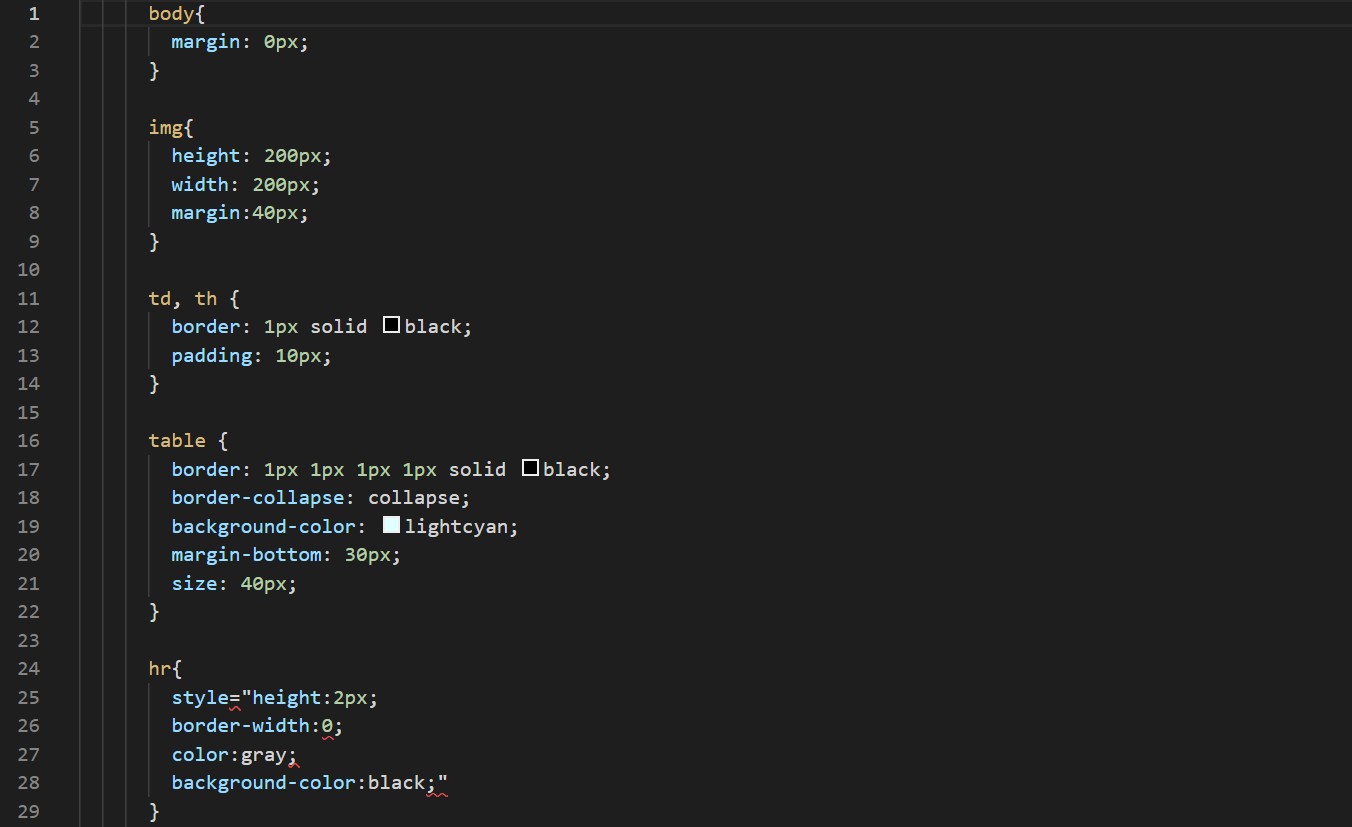
2017 - 2018

91.8%

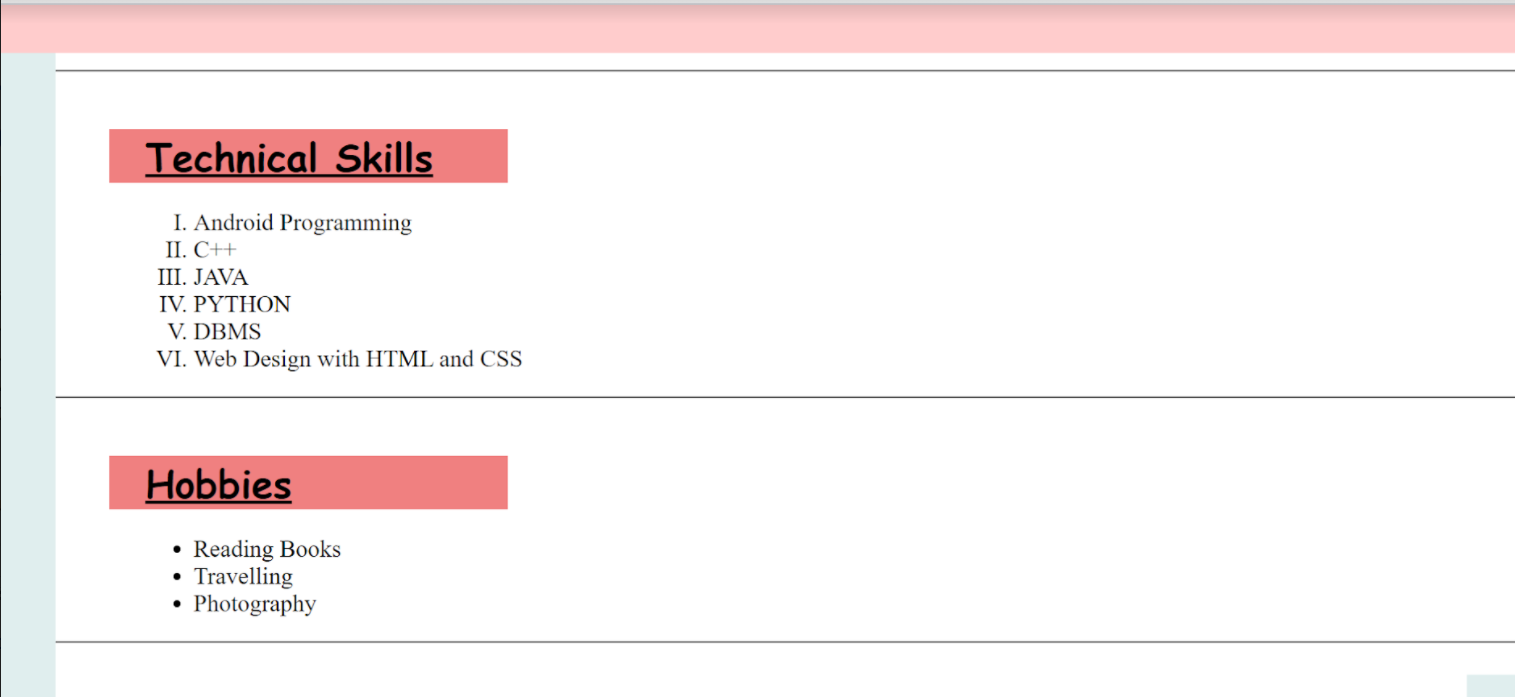




**.css**

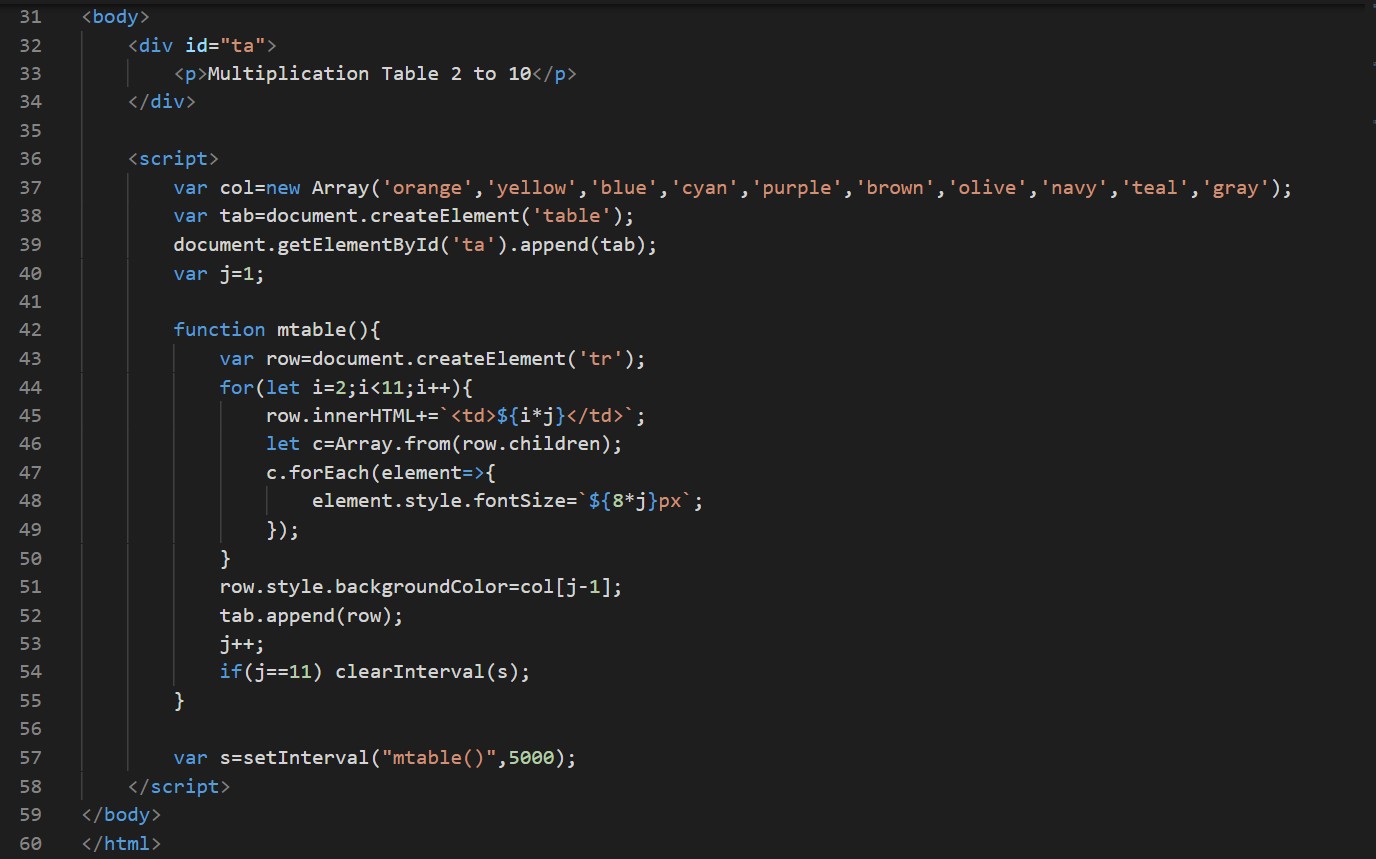
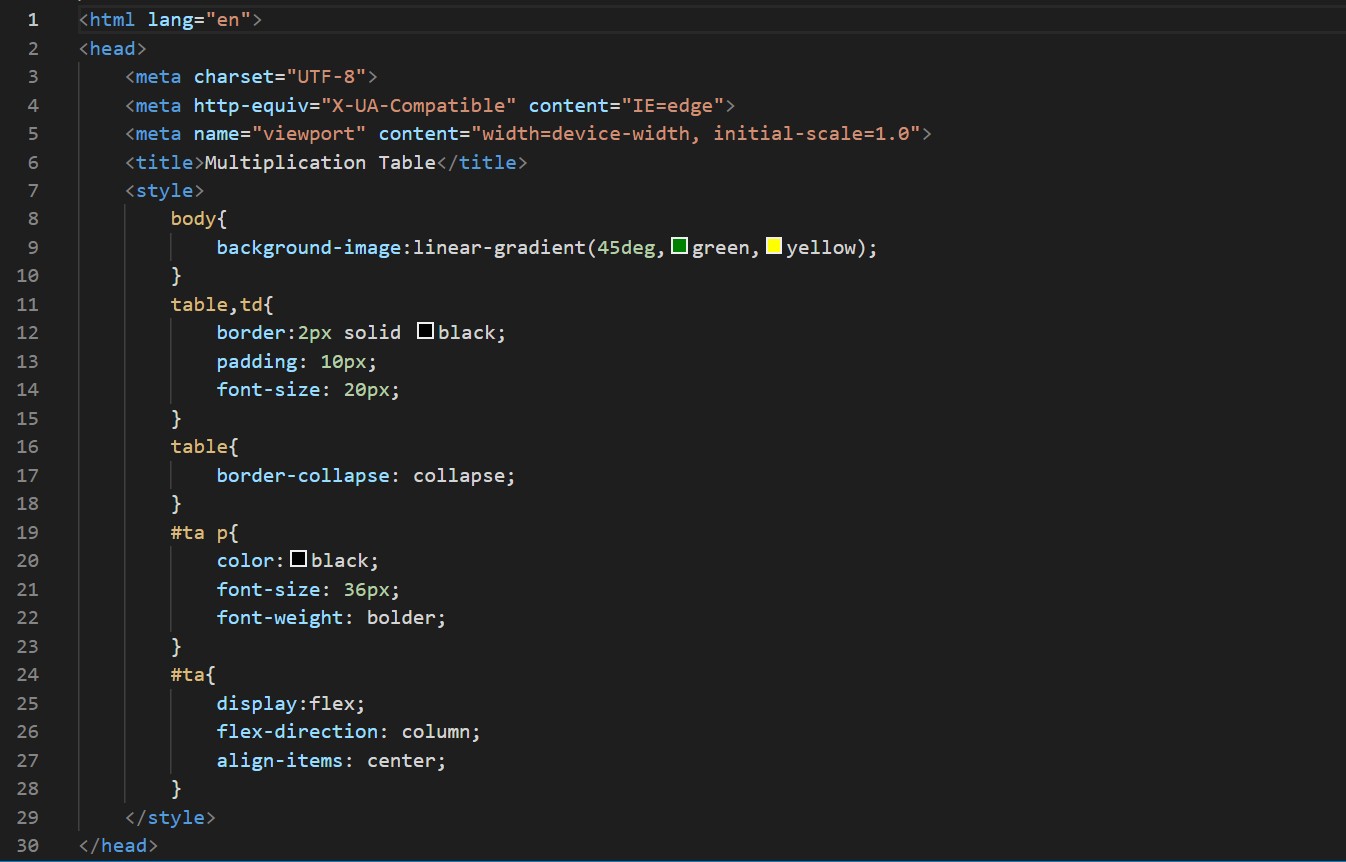




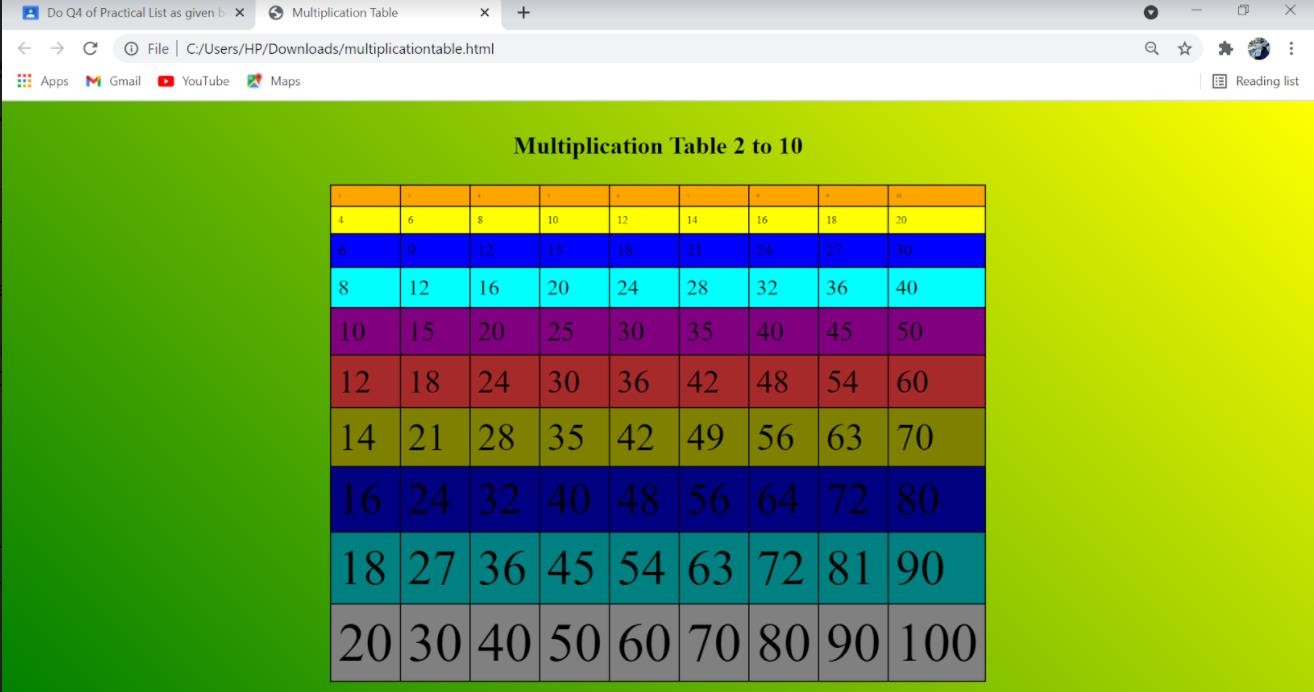
**OUTPUT**

**Ques4.** Create an HTML page with the sole purpose to show multiplication tables of 2 to 10 (row-wise) created by JavaScript. Initially, the page is blank. With help of setInterval function print a row every 5 seconds in different colors and increasing font size.

**Ans.**

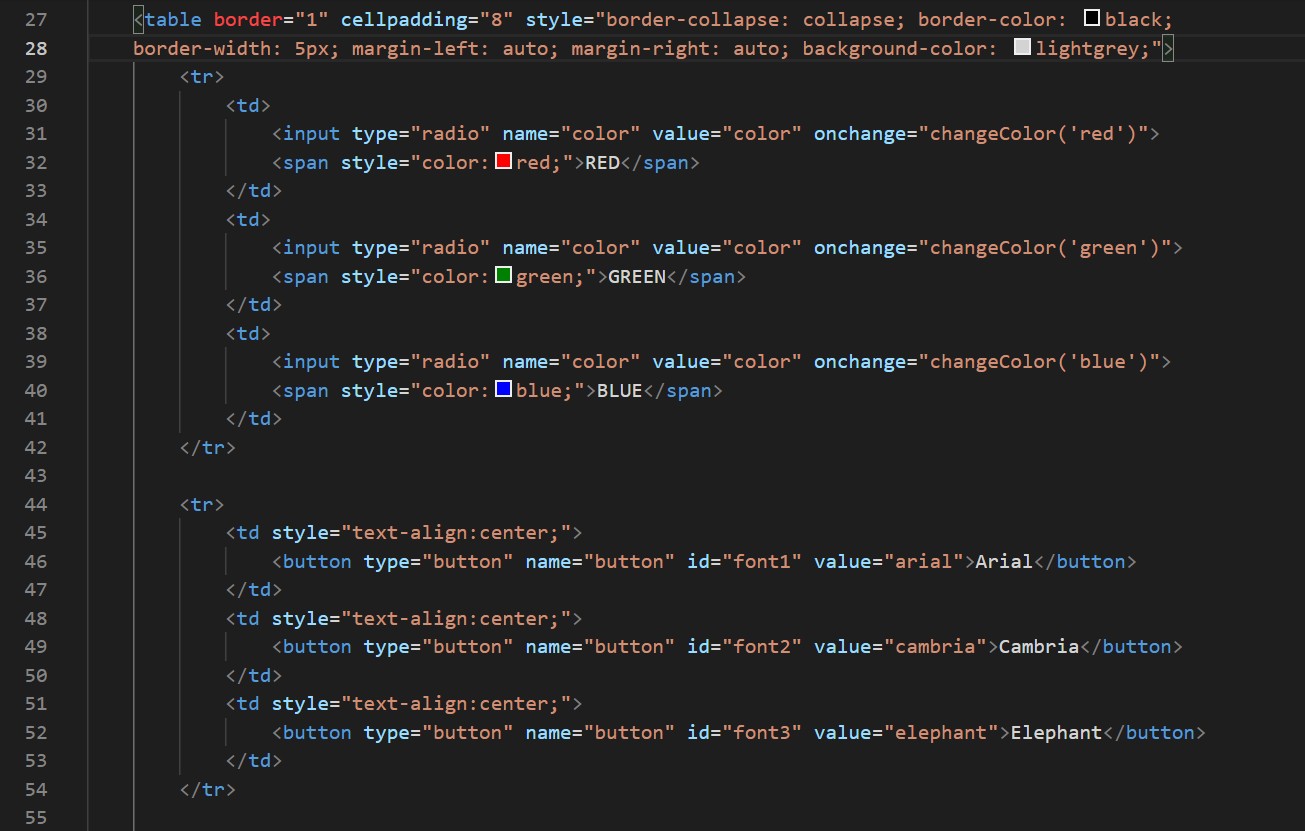
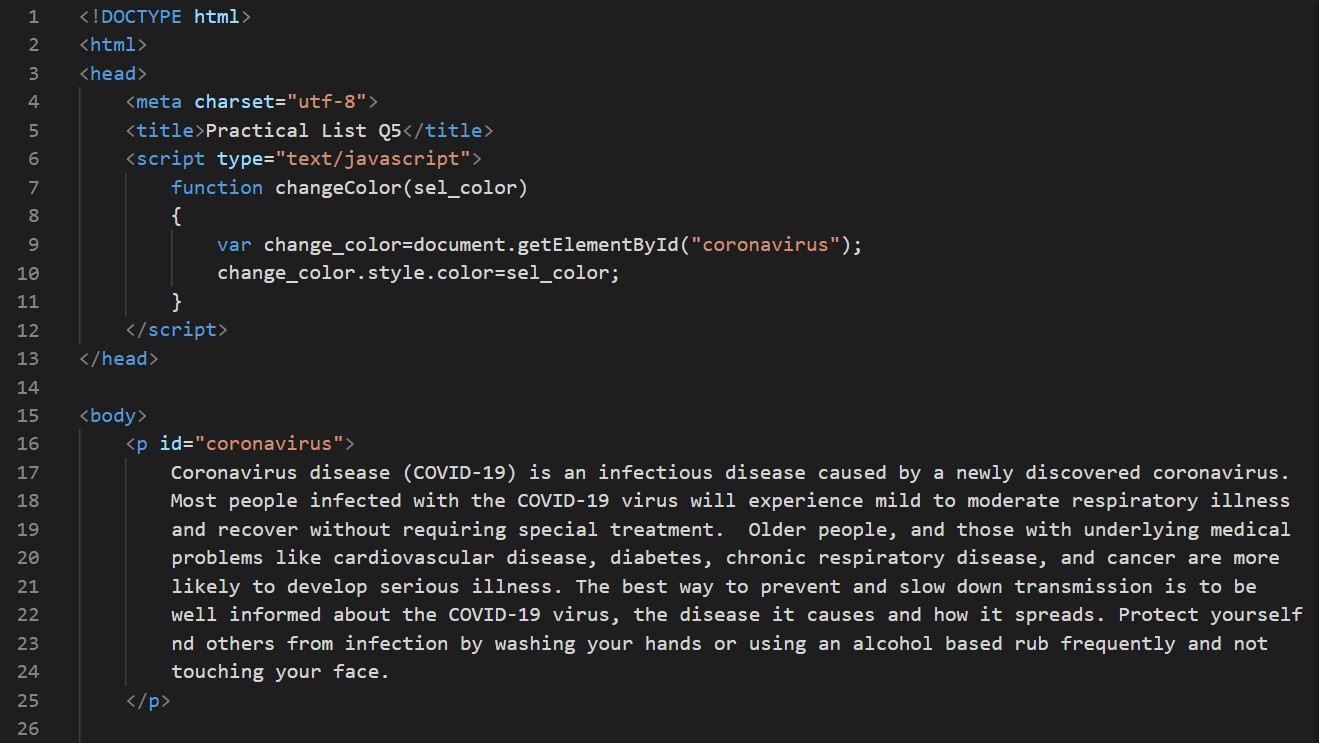


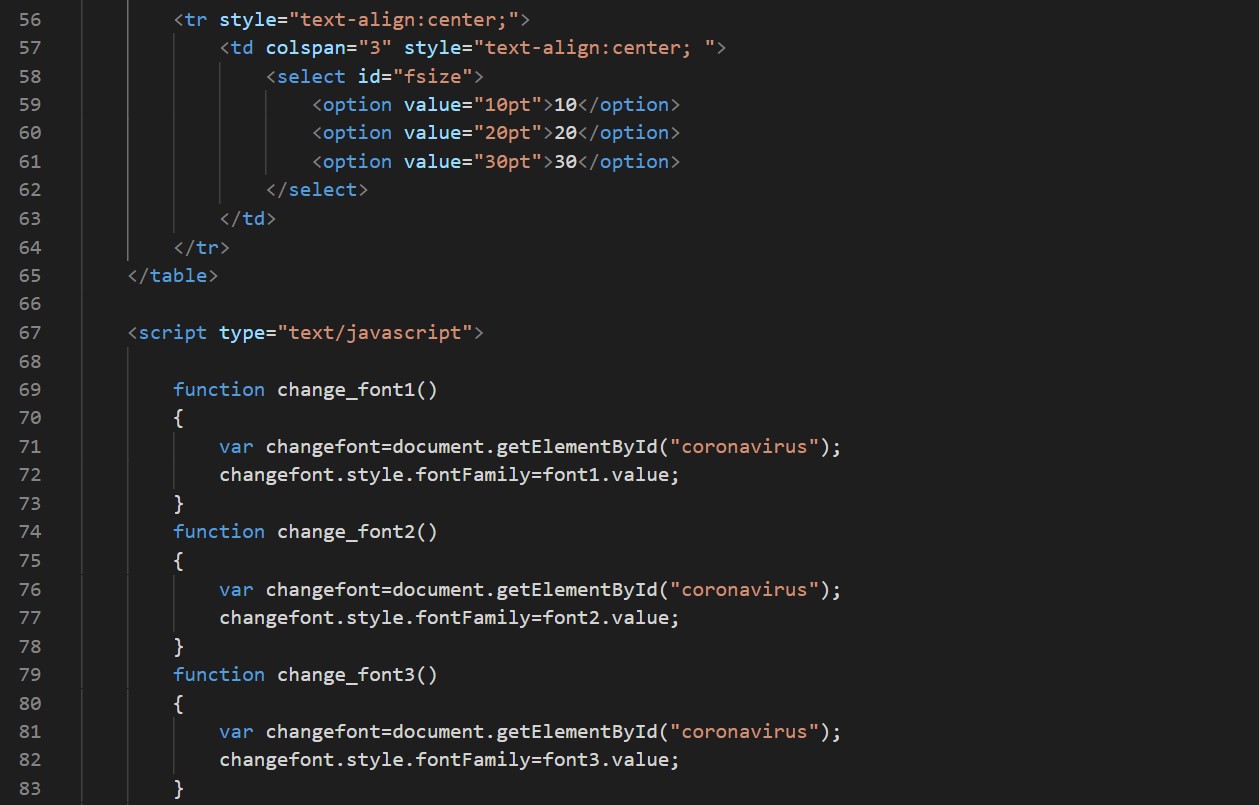
**OUTPUT**



**Ques5.** Create an HTML page with a paragraph written on it and under which 9 buttons are placed in a 3X3 grid. The first row is for buttons labeled with colors names Red, Green, and Blue, the second row with numbers 10, 20, 30, and the third row with different font names. Click event of each of the buttons should make the appropriate change in the style of paragraph.

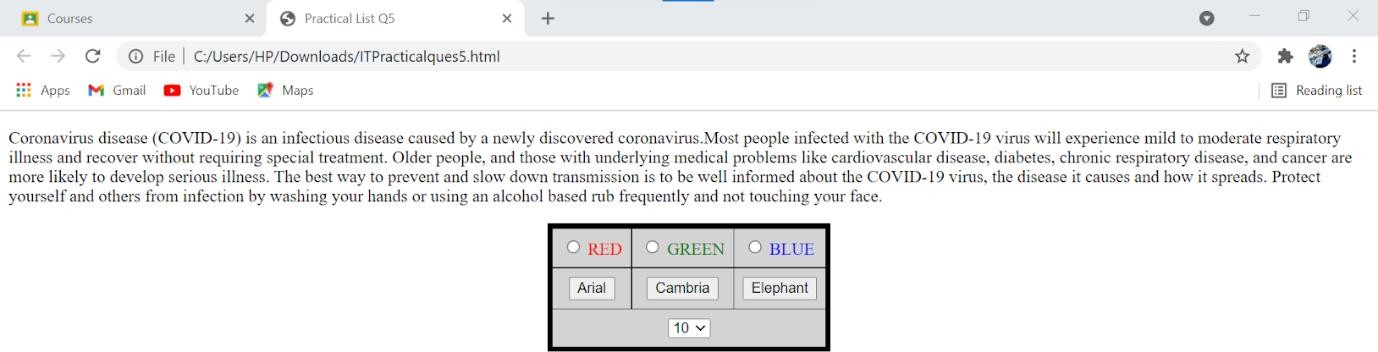
**Ans.**

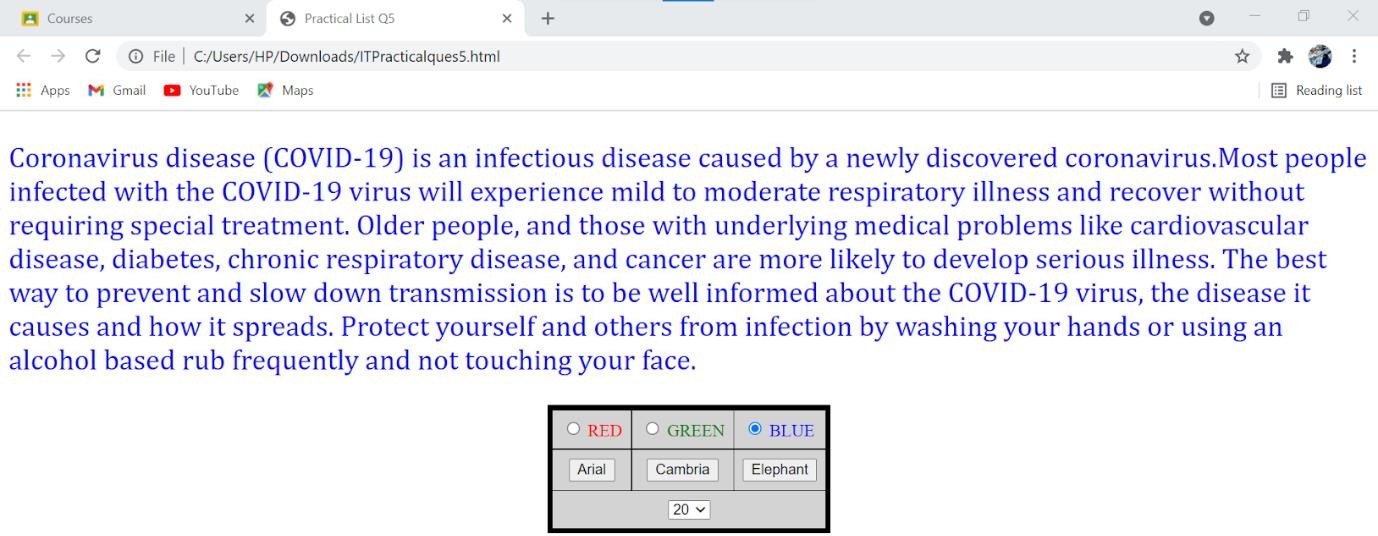


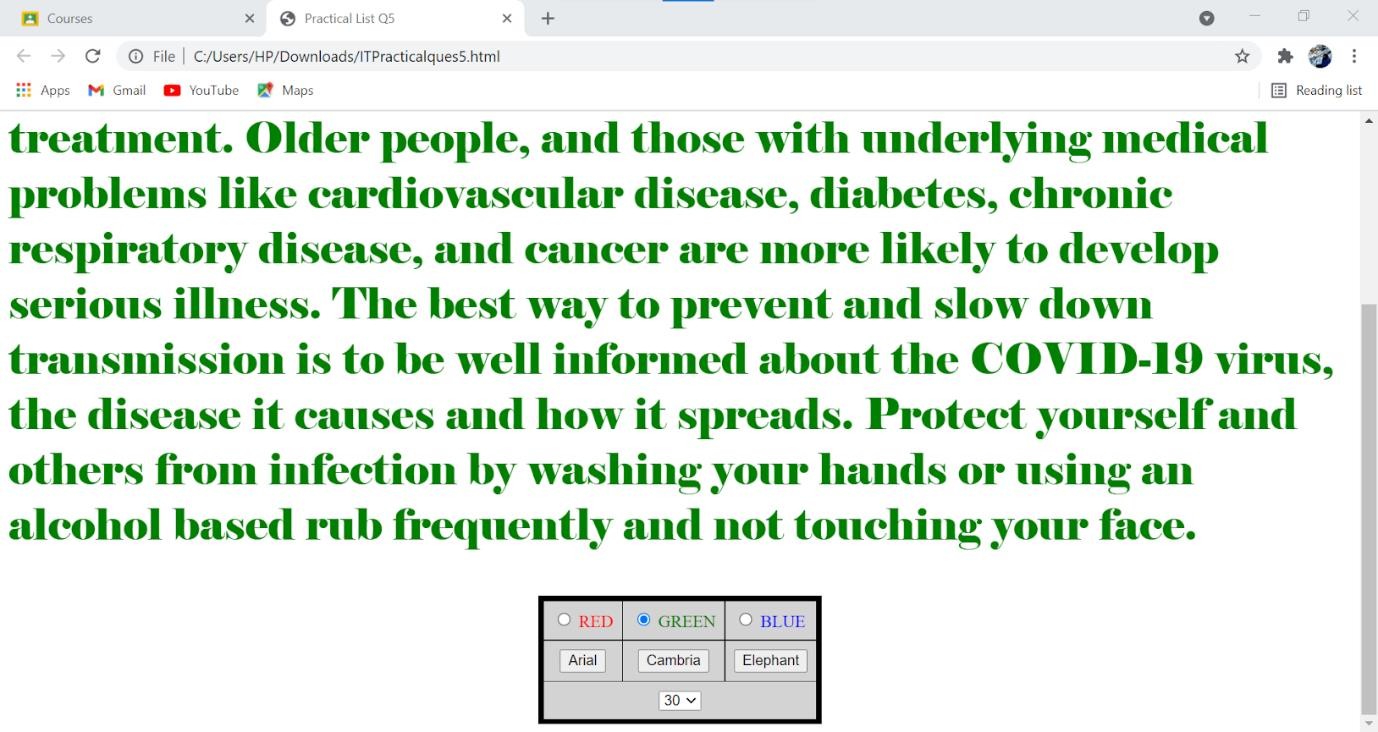


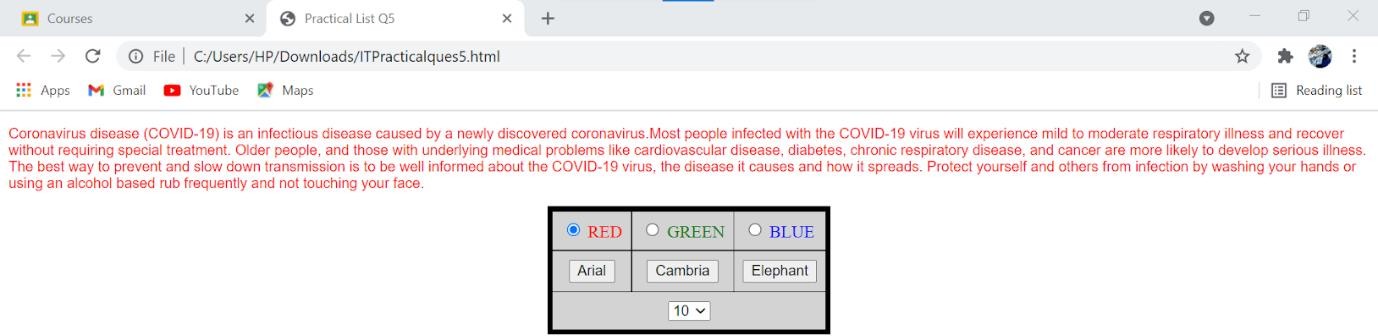


**OUTPUT**







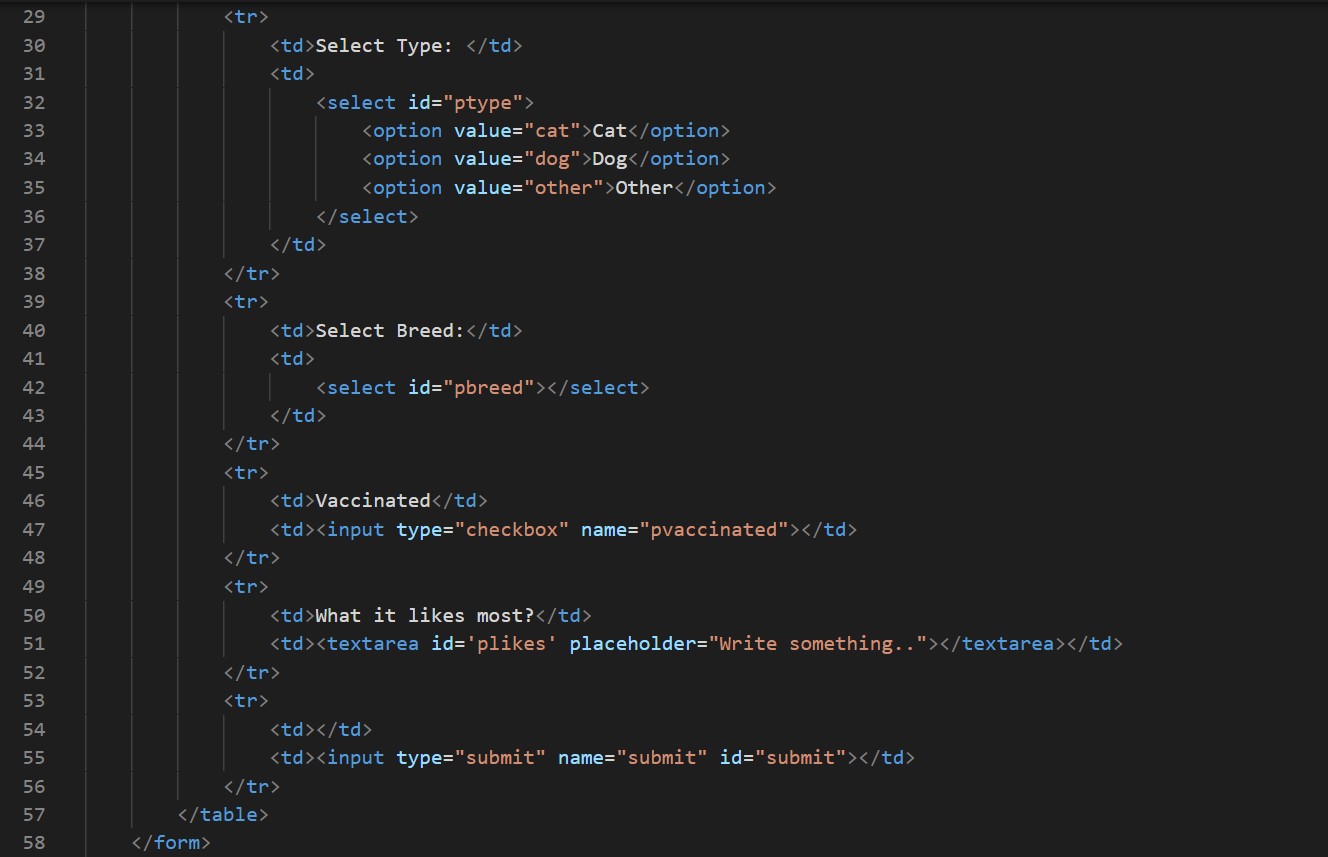


**Ques6.** Create a form that takes data about a pet. The form must be well designed and should accept the pet’s name, age, weight, type, and what it likes most. At the submission of this form create a Pet object in JavaScript filled with these values and log that object and equivalent JSON on the console.

**Ans.**

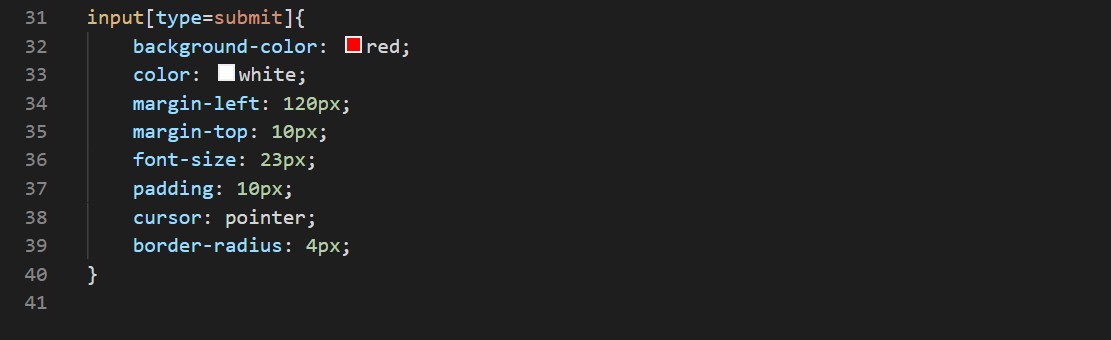
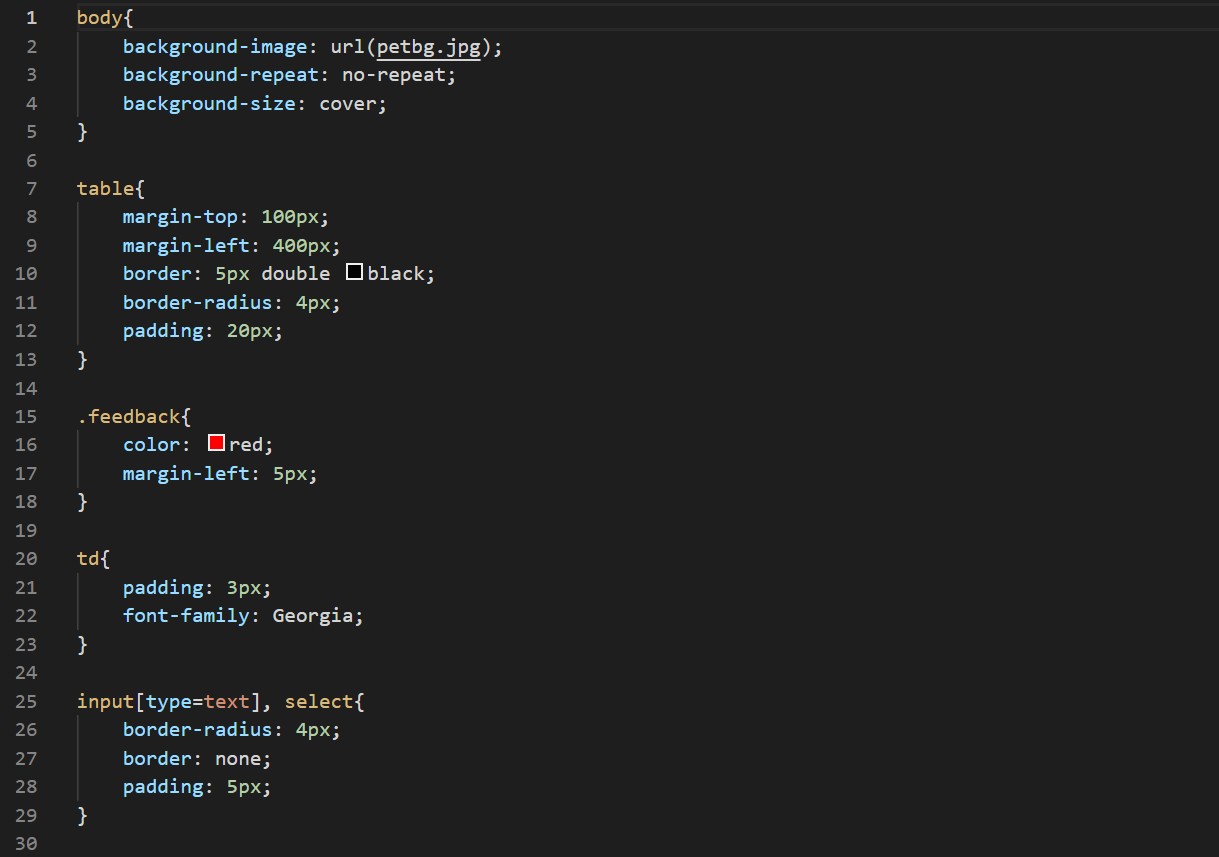
**.html**



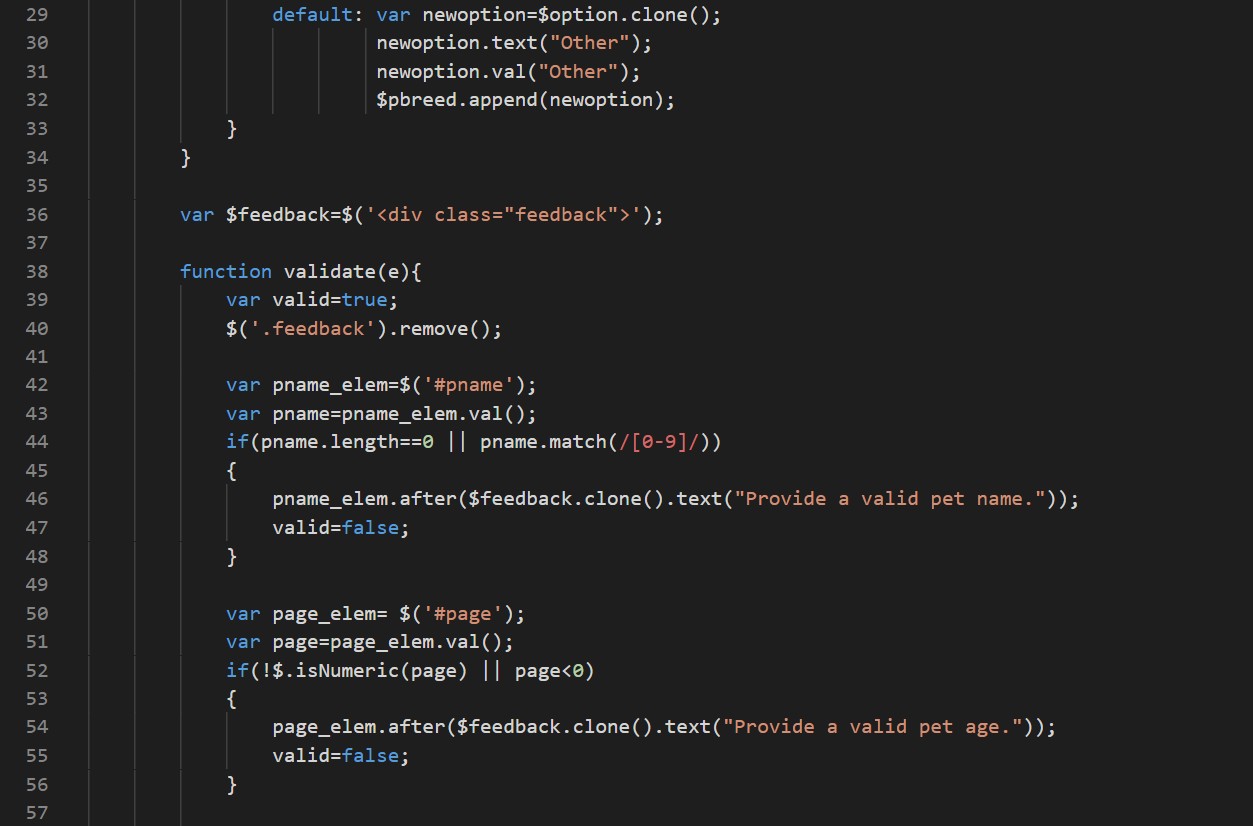
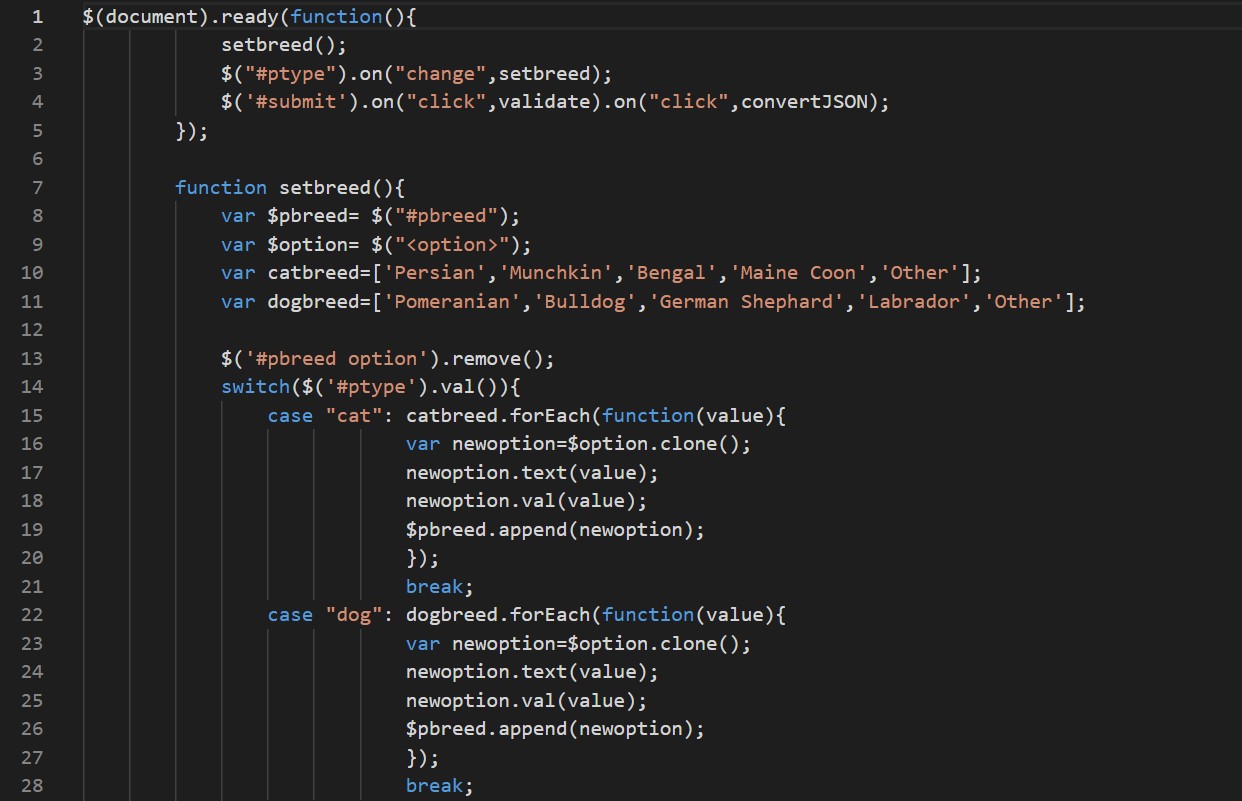




**.css**

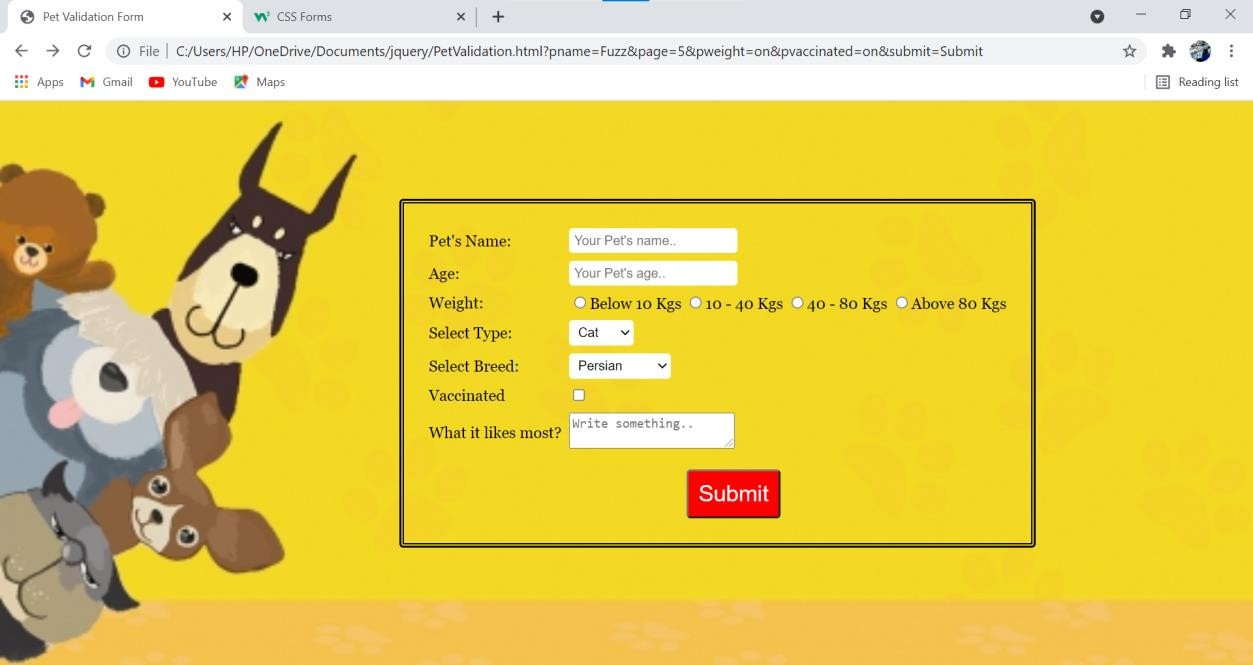


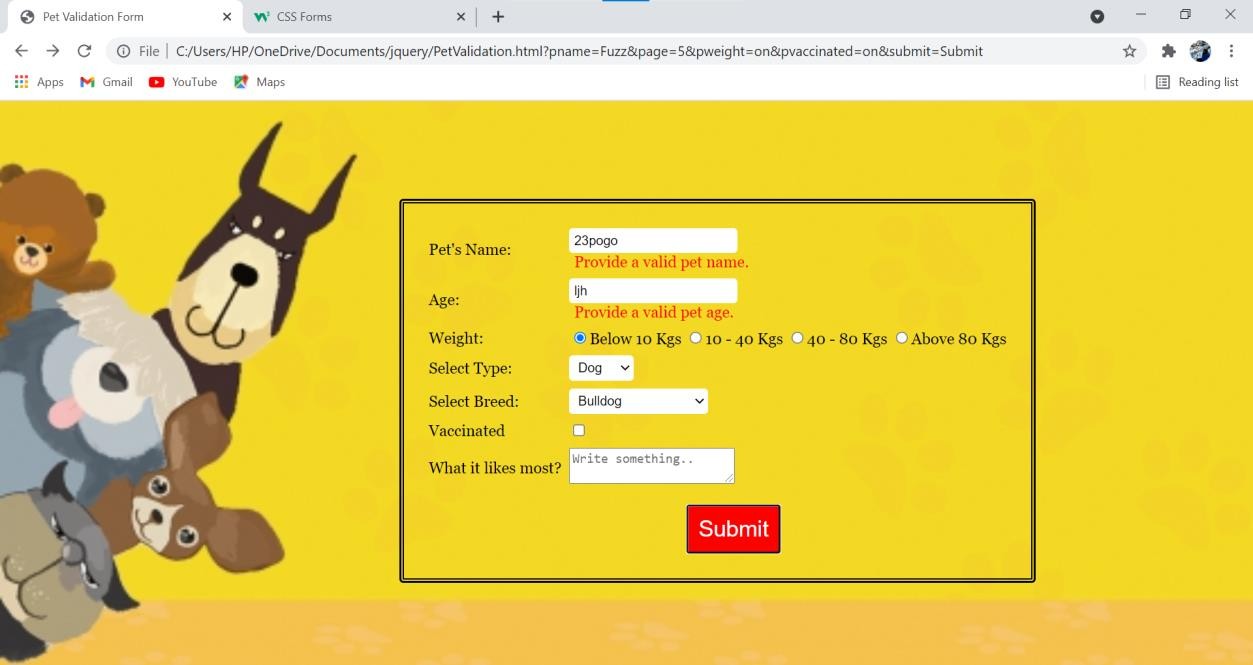
**.js**

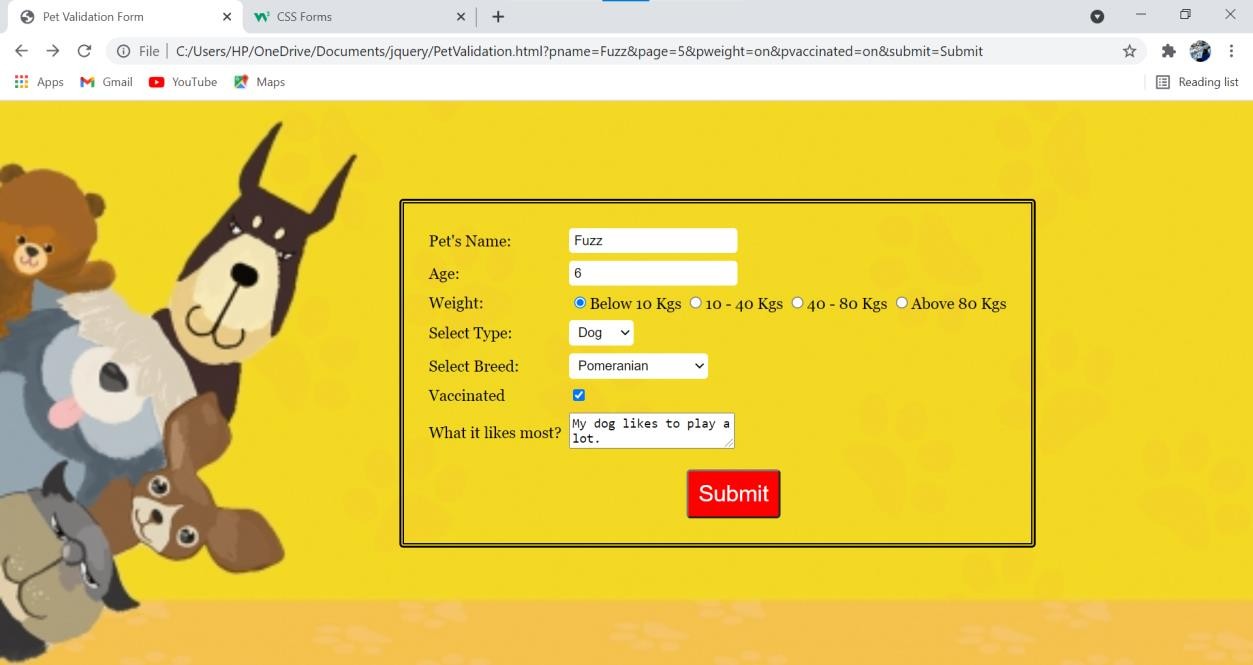


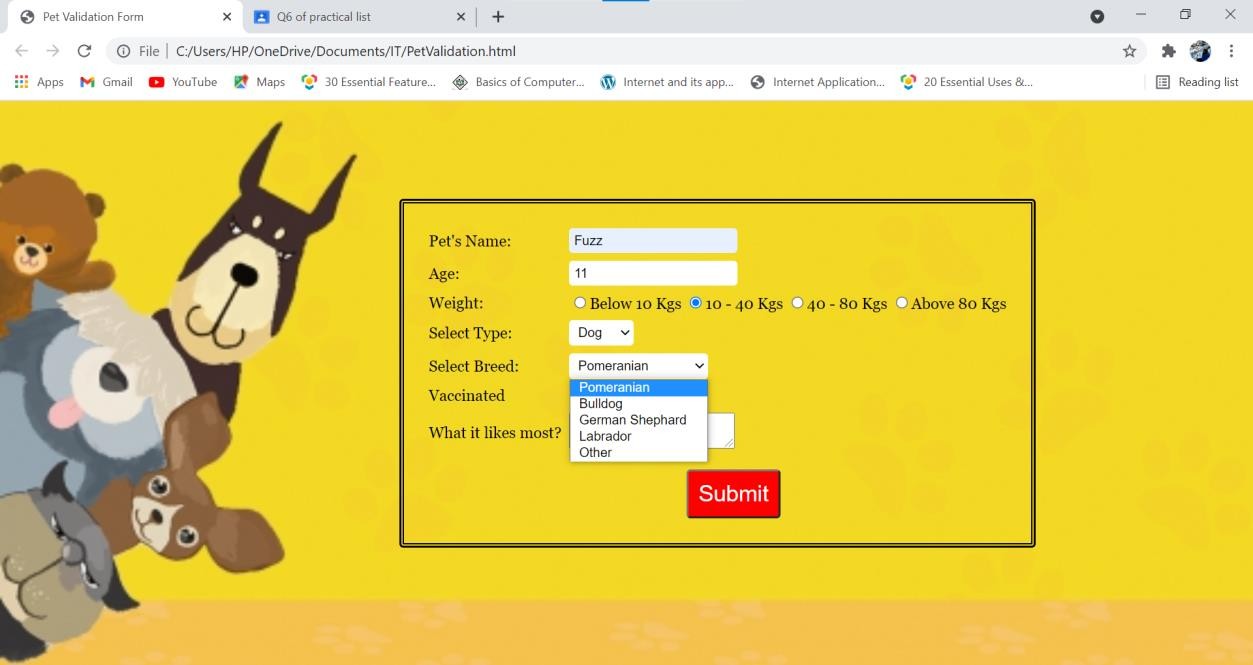


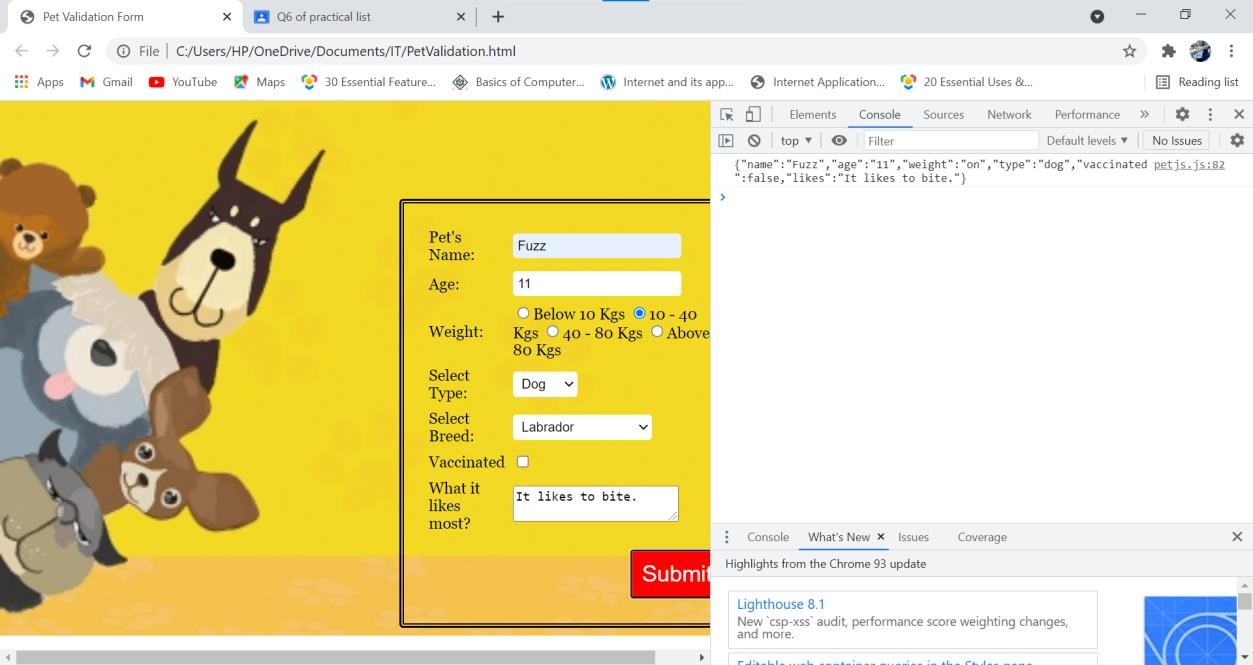
**OUTPUT**





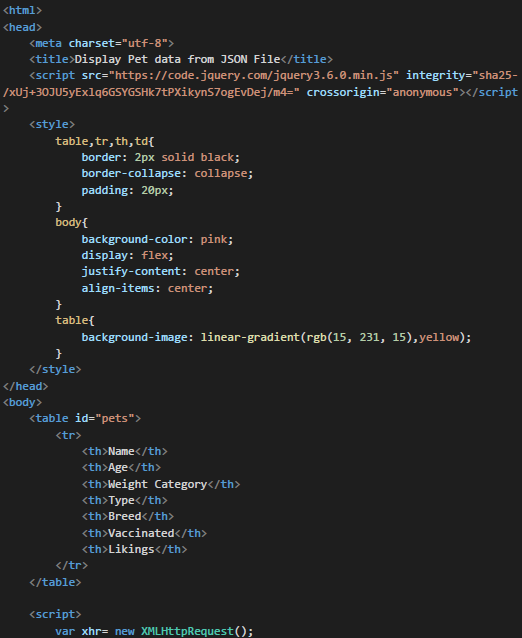






**Ques7.** Store JSON data of few pets that you created in previous practical in a JSON file (copy from console output of previous program to a .json file). Using AJAX, load data from the file and display it in a presentable way using HTML and CSS.

**Ans.**

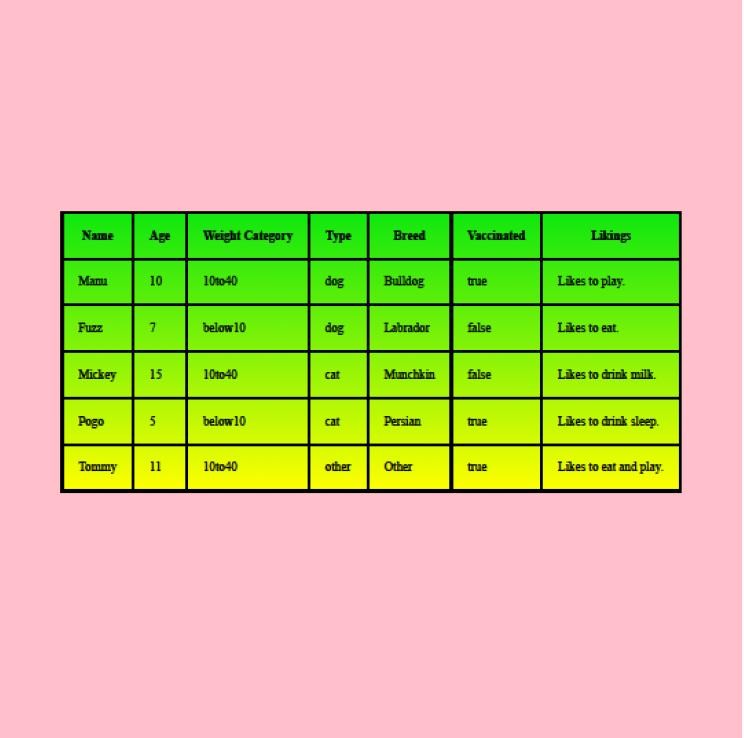




**.json**



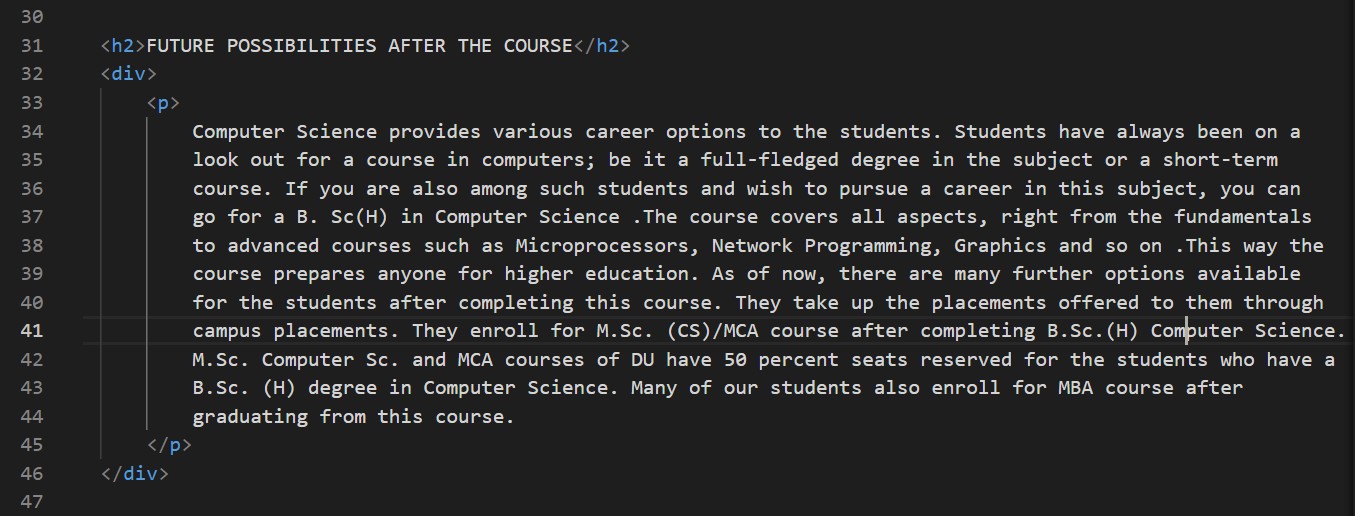
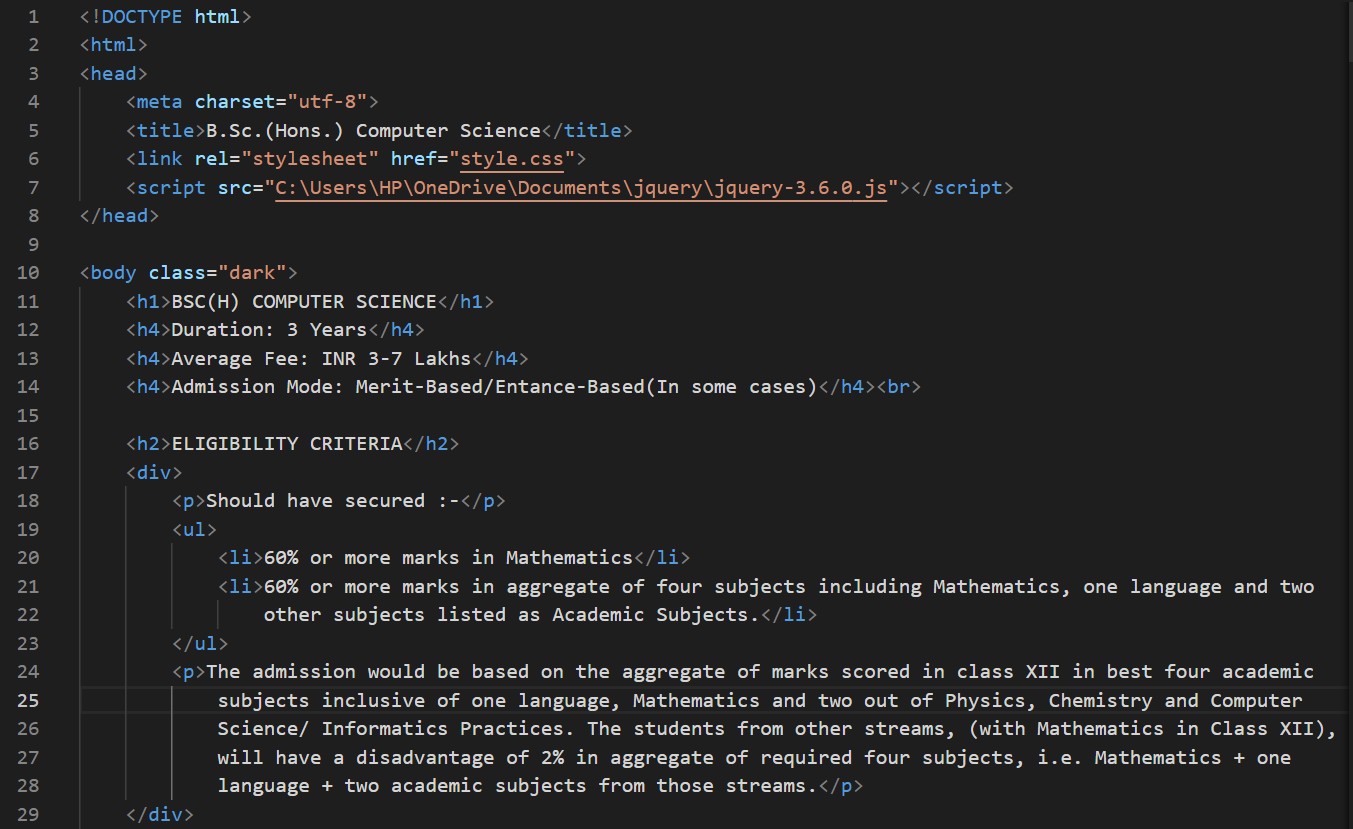
**OUTPUT**

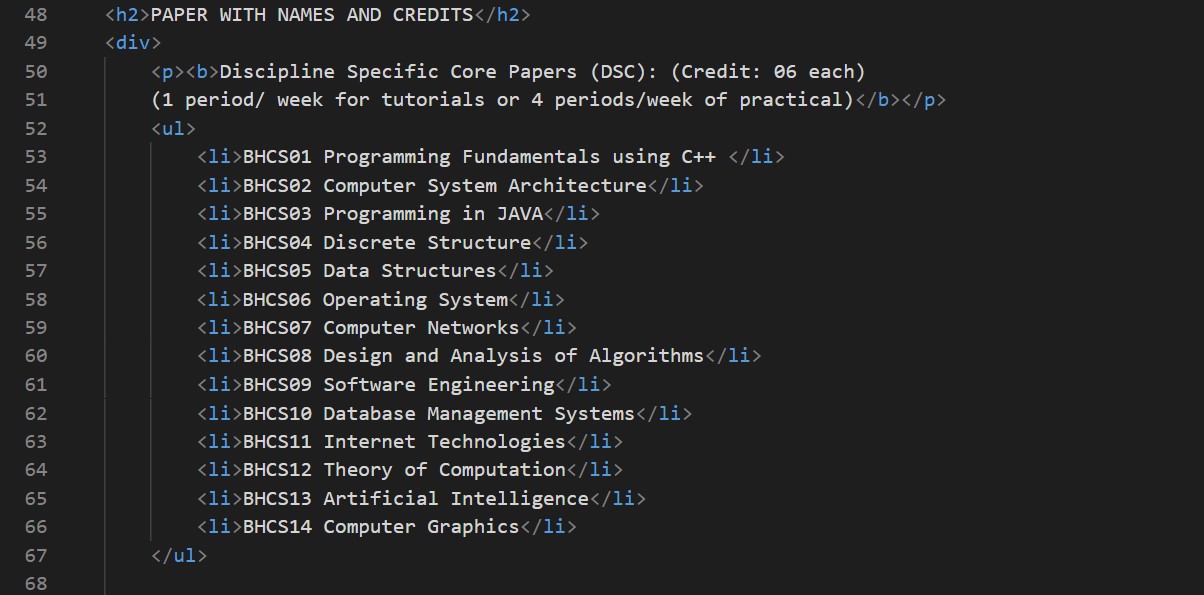


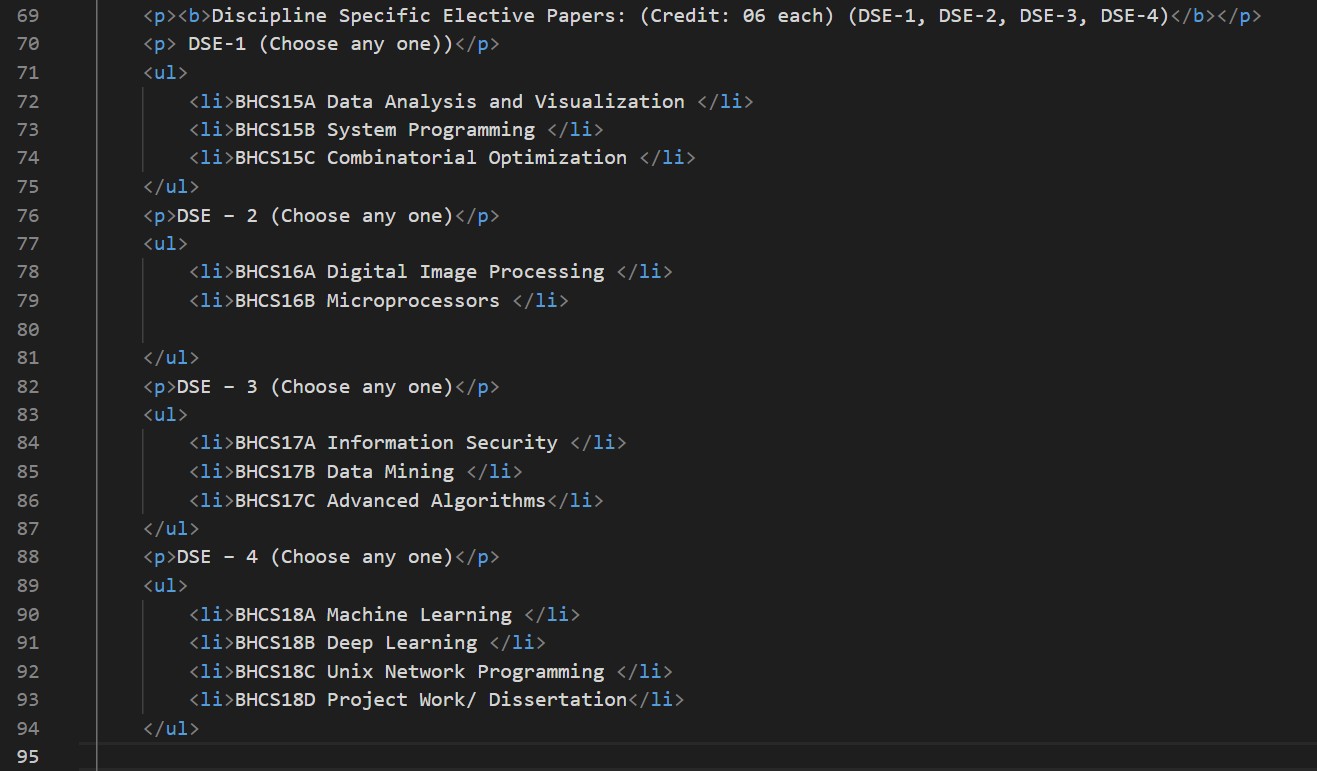
**Ques8.** Create a plain HTML page for B.Sc. Hons CS course, mentioning details like fee, eligibility criteria, papers with names and credits, and future possibilities after the course. A button for styling should be there at bottom of the page. On clicking on this button JavaScript should redesign the complete page using jQuery in a nice presentable way.

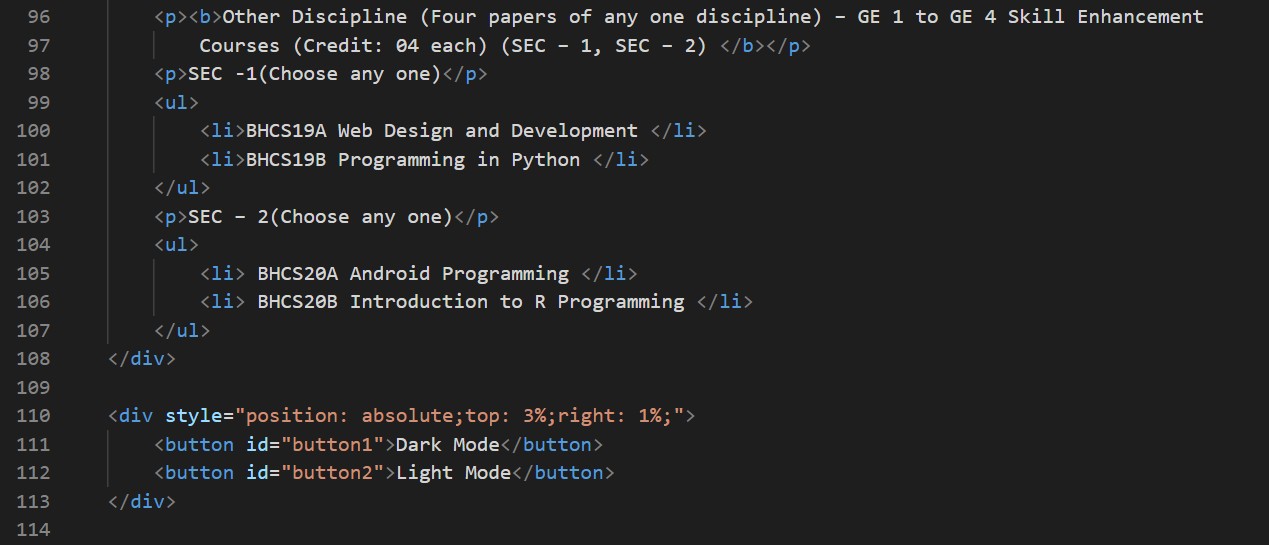
**Ans.**

**.html**



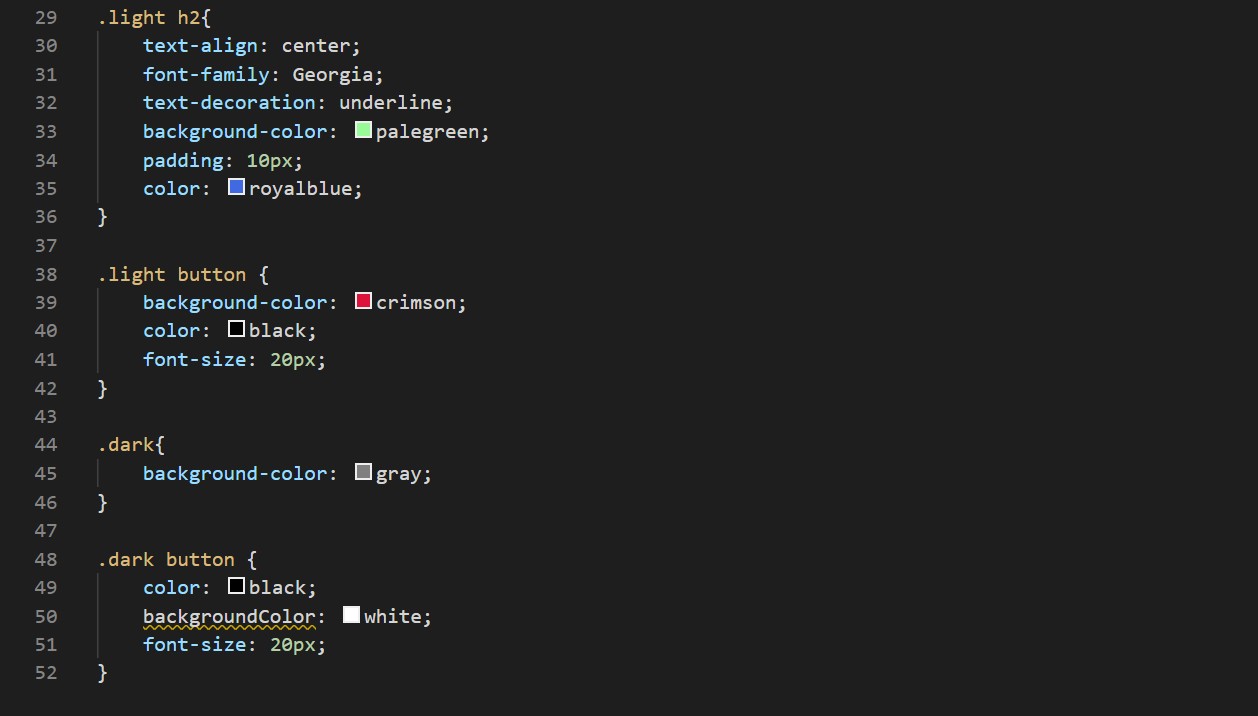




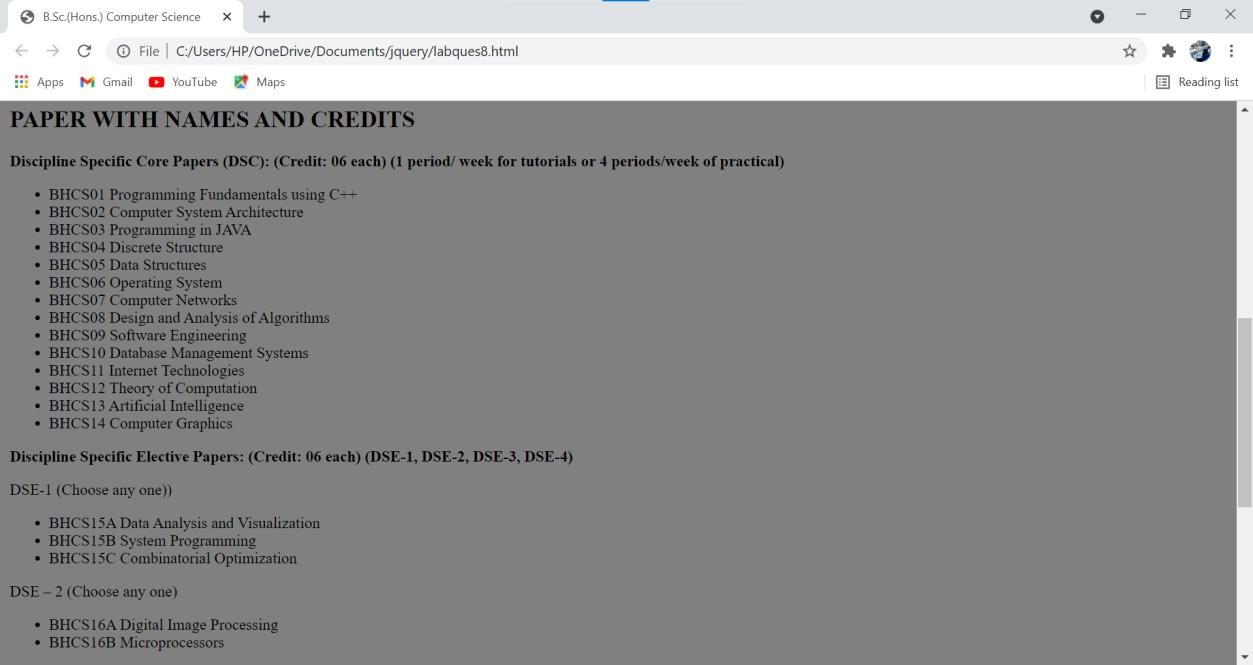
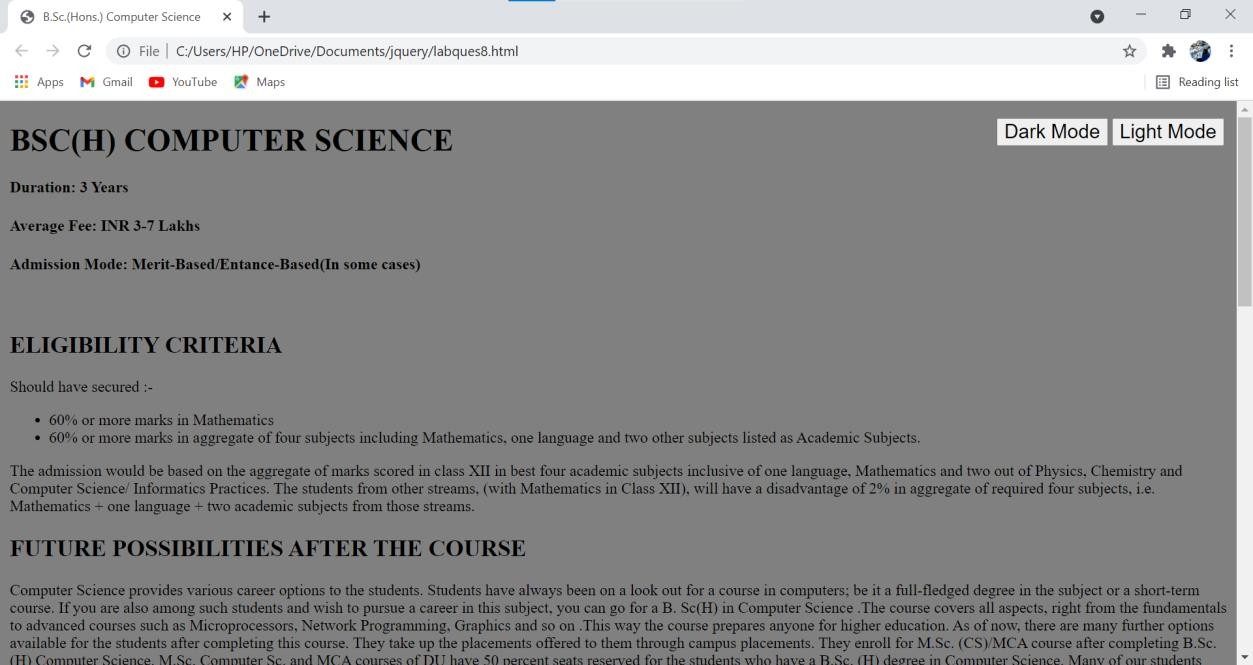


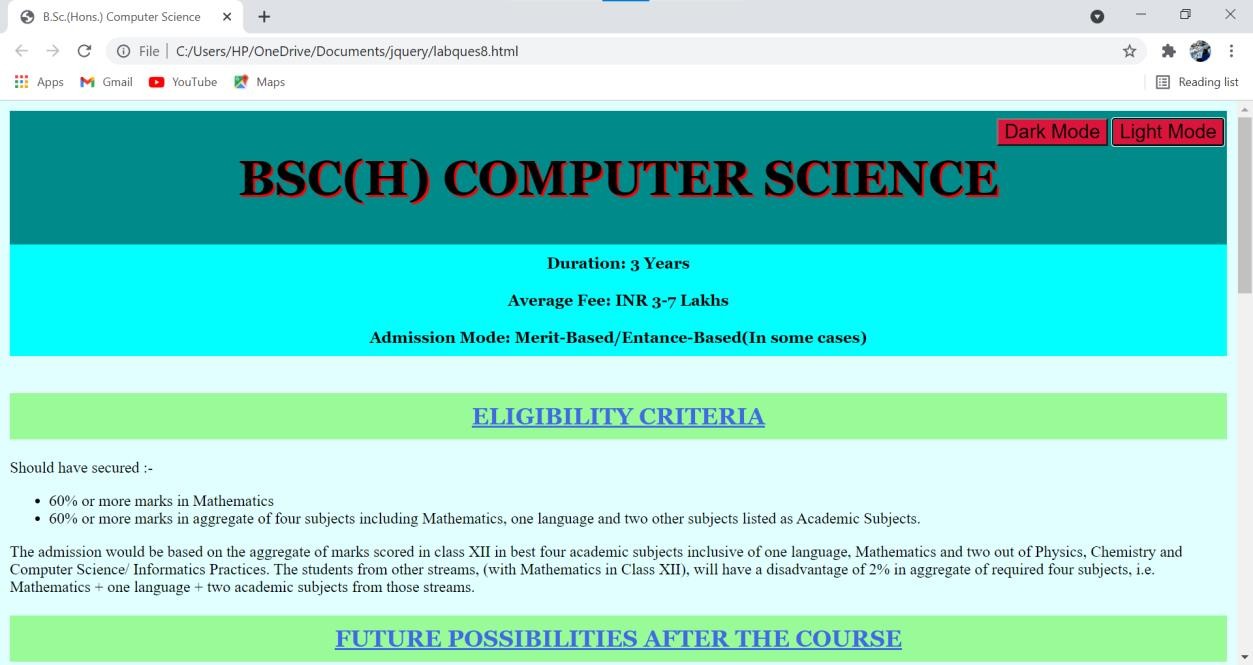


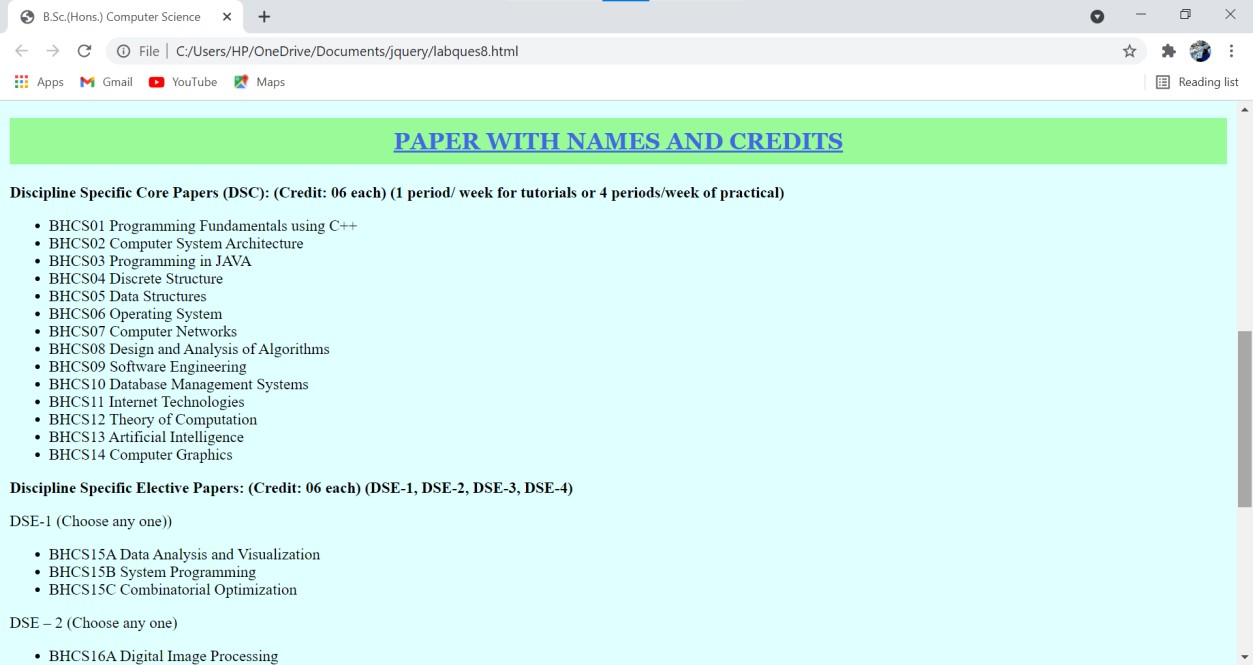
**.css**



**OUTPUT**

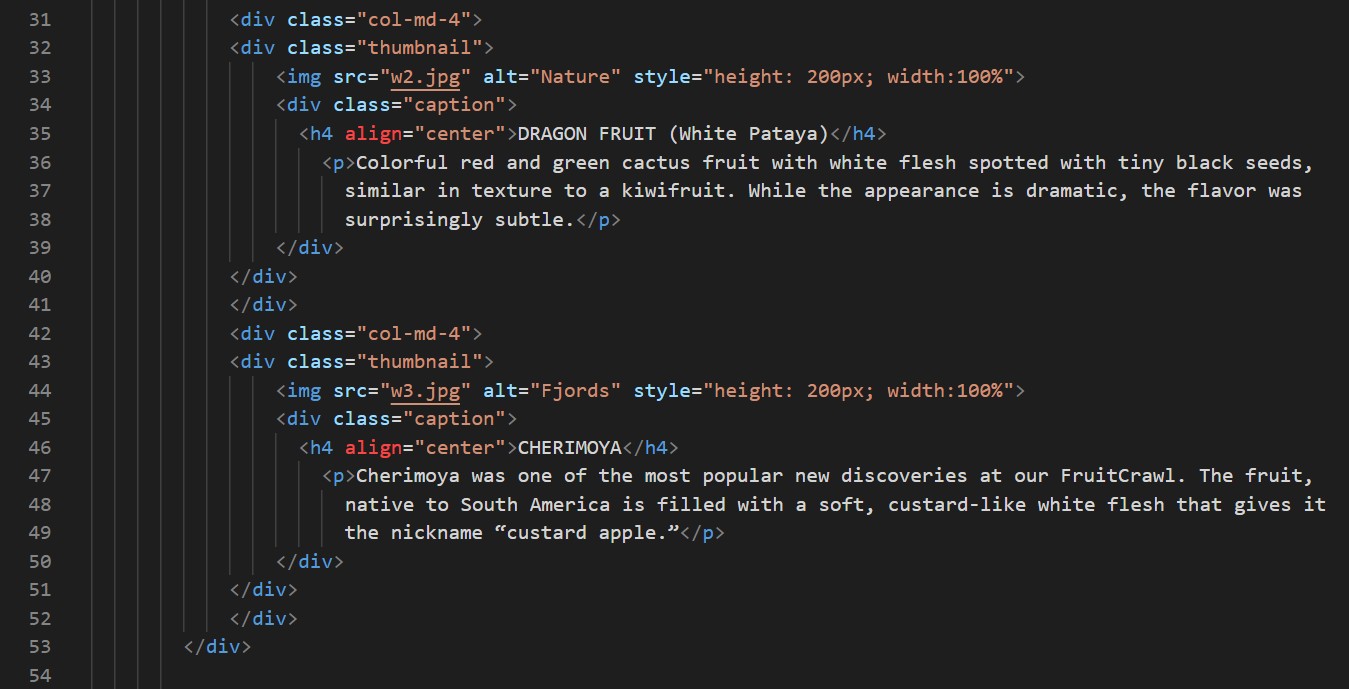
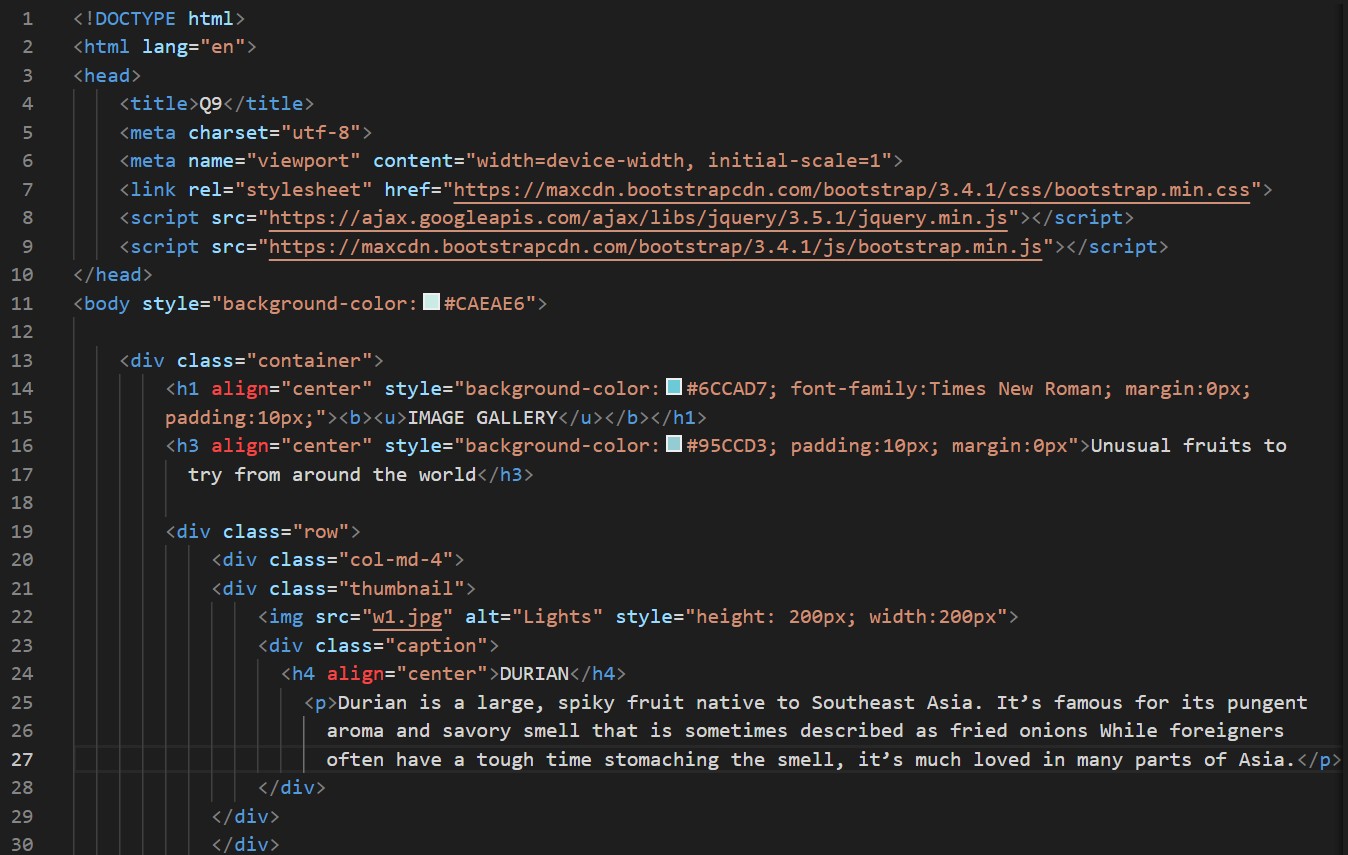


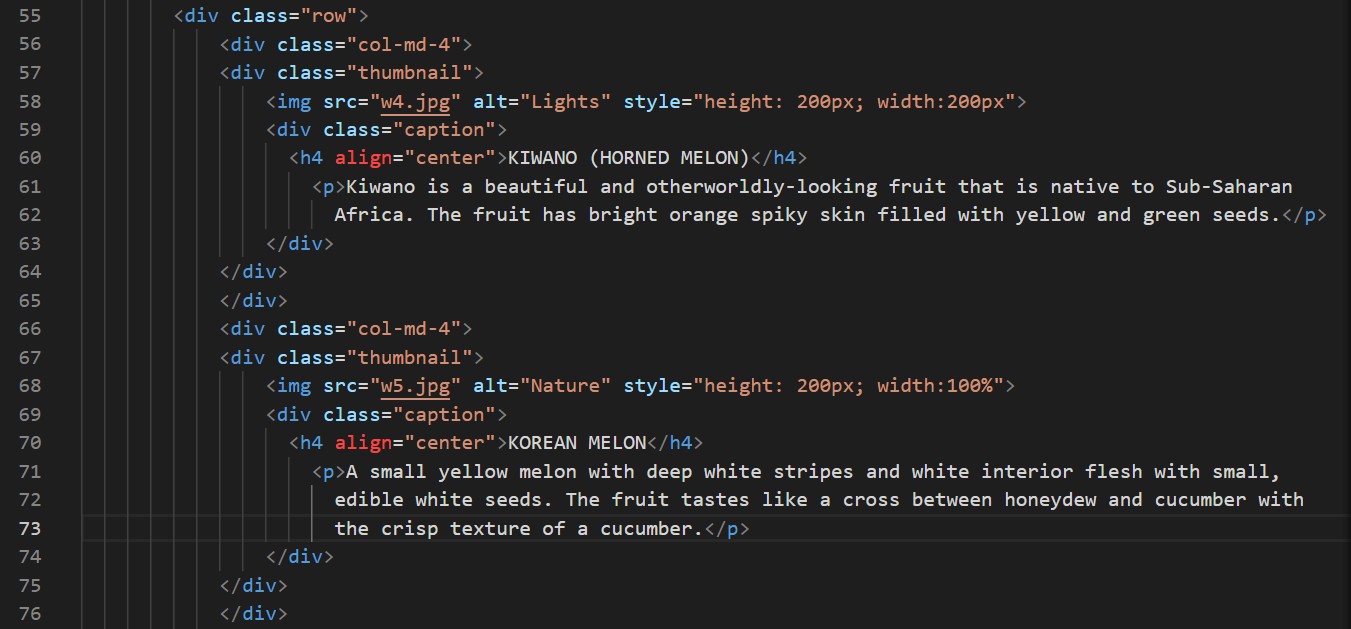


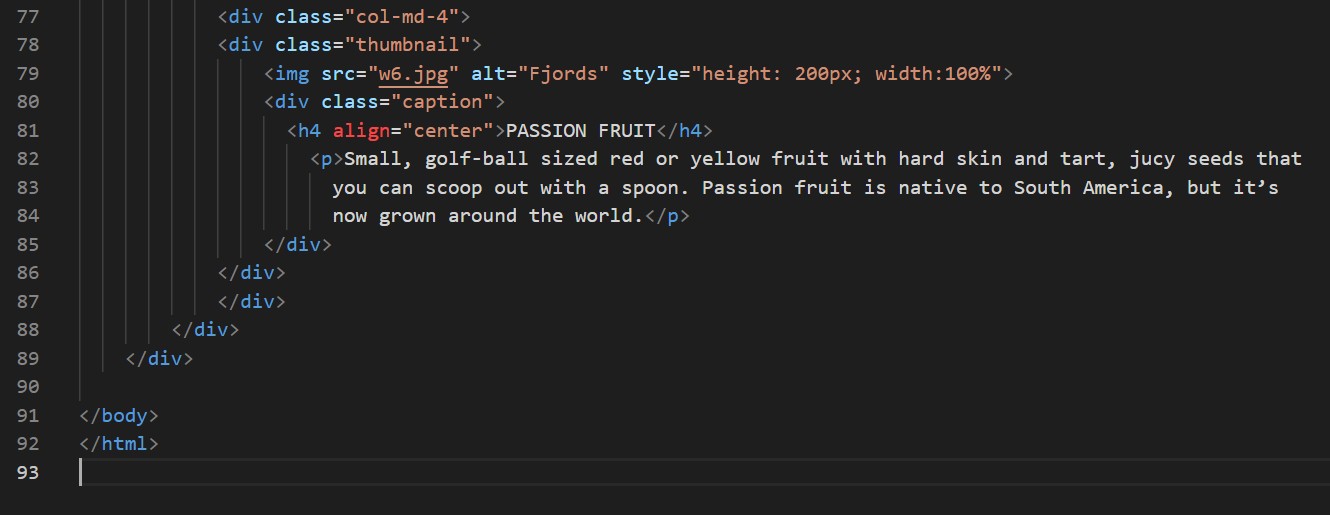


**Ques9.** Create an HTML page for an image gallery which shows the use of BOOTSTRAP to rearrange and resize its contents on resizing the browser.

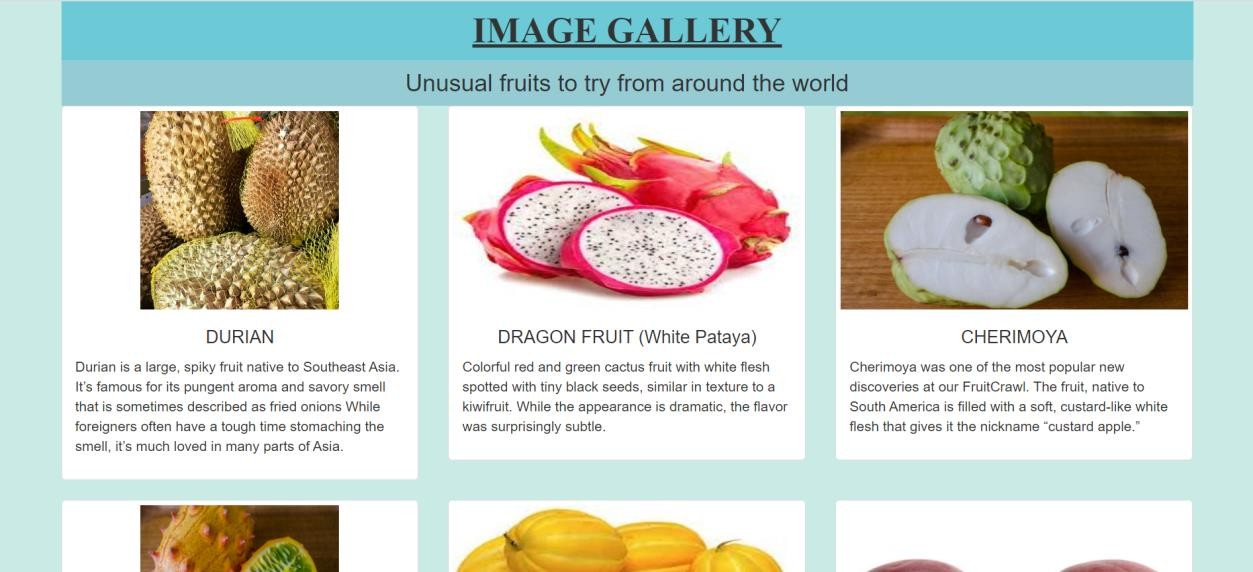
**Ans.**





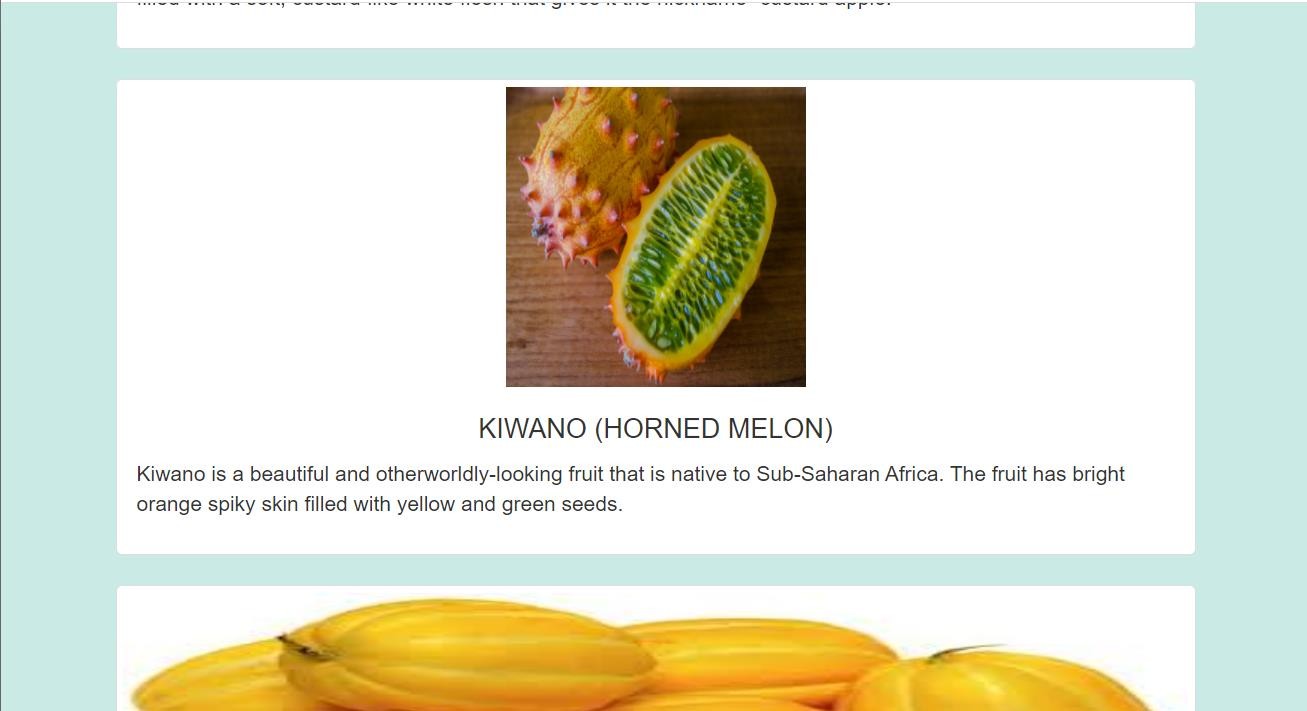
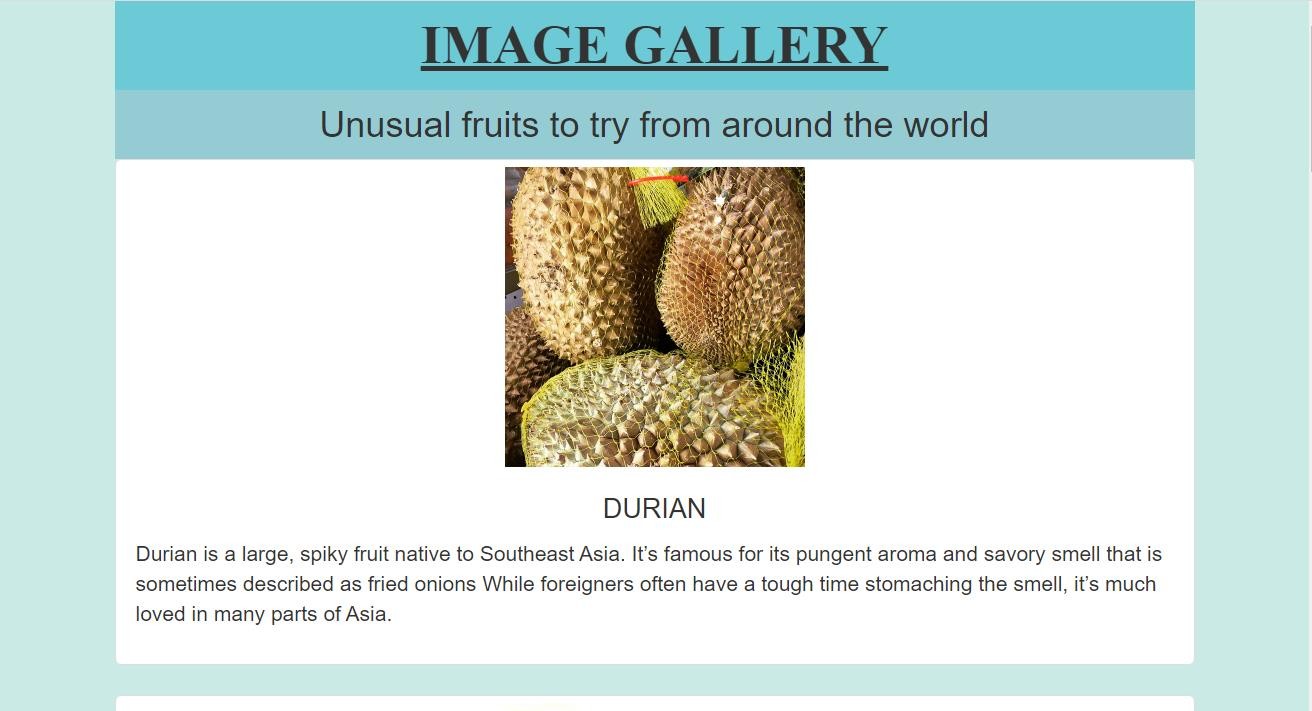


**OUTPUT**



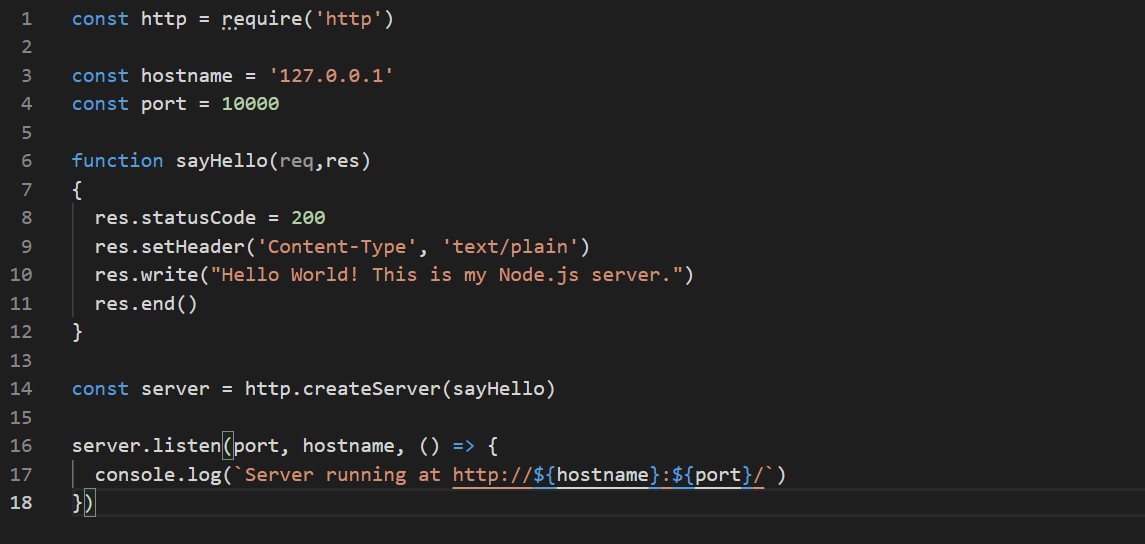


On resizing the browser window

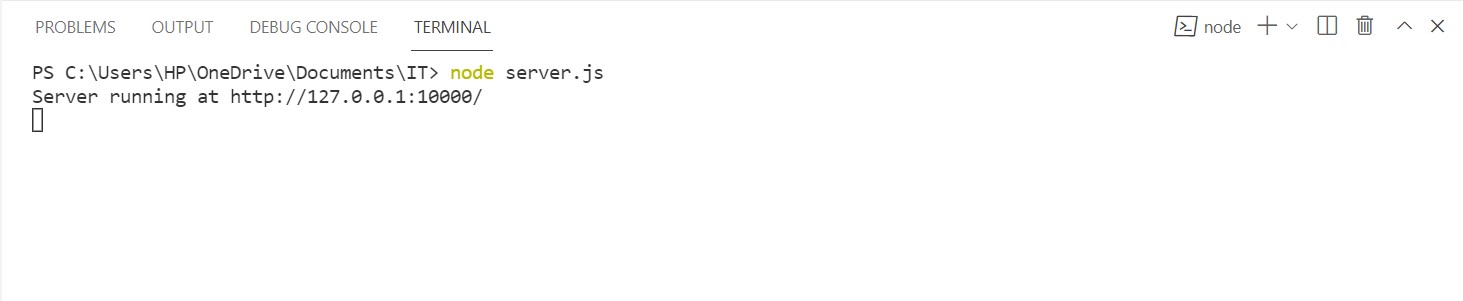


**Ques10.** Create an HTTP server using Node.js which handles requests on port 10000 or a free port beyond 10000. Modify the server in such a way that opening localhost:10000 will display “Hello world, This is my Node.js server” on browser.

**Ans.**



**OUTPUT**



**Ques11.** Create index.html file containing two forms for SignIn and SignUp. Submitting SignIn form should search for credentials in mysql database using server created in previous practical. On successful signin, a welcome page should be displayed. Submitting SignUp form should insert new entry for credentials in mysql database using server created in previous practical. On successful signup, user should be returned back to index.html.

**Ans.**

const http = require('http') const fs = require('fs')

const mysql = require('mysql'); const qs = require('querystring')

const hostname = '127.0.0.1' const port = 4000

function onRequest(req, res) {

//var baseURL = 'http://' + req.headers.host + '/';

//var myURL = new URL(req.url, baseURL);

res.statusCode = 200

res.setHeader('Content-Type', 'text/HTML') //implies response body will contain HTML

console.log(req.url) if (req.url == '/') {

index(req, res)

} else if (req.url == '/showsignin') { showsignin(req, res)

} else if (req.url == '/dosignin') { dosignin(req, res)

} else if (req.url == '/showsignup') { showsignup(req, res)

} else if (req.url == '/dosignup') { dosignup(req, res)

} else { res.writeHead(404, {

'Content-Type': 'text/html'

});

return res.end("404 Not Found");

}

}

function showsignup(req, res) { fs.readFile('signup.html', function (err, data) {

res.write(data); return res.end();

});

}

function dosignup(req, res) { var body = ''

//collect the request data req.on('data', function (data) {

body += data

console.log('Partial body: ' + body)

})

req.on('end', function () { console.log('Body: ' + body)

var qs = new URLSearchParams(body) var username = qs.get("username") var passwd = qs.get('passwd')

var confpasswd = qs.get('confpasswd')

if (passwd != confpasswd) { res.write("<h1>Password Mismatch</h1>") return res.end();

}

var con = mysql.createConnection({ host: "localhost",

user: "root", password: "1234", database: "it2021"

});

con.connect(function (err) {

if (err) throw err;

console.log("Connected!");

//query set up

var sql = "INSERT INTO user (username, passwd) VALUES (?,?)";

//executing the query

con.query(sql, [username, passwd], function (err, result) { if (err) throw err;

console.log("1 record inserted");

res.writeHead(302, {Location:"http://127.0.0.1:4000/"});; res.end()

});

});

})

}

function dosignin(req, res) { var body = ''

req.on('data', function (data) {

body += data //collecting the request data in the body variable console.log('Partial body: ' + body)

})

req.on('end', function () { console.log('Body: ' + body)

var qs = new URLSearchParams(body) var username = qs.get("username") var passwd = qs.get('passwd')

//build the connection object

var con = mysql.createConnection({

host: "localhost", //IP address of the database server user: "root",

password: "1234", database: "it2021"

});

//connect to the database con.connect(function (err) {

if (err) throw err; console.log("Connected!");

con.query("SELECT \* FROM user where username=? and passwd=?", [username, passwd],

function (err, result, fields) { if (err) throw err;

console.log(result);

if (result.length == 1) { res.write("<h1>Sign-In Successful</h1>") res.end()

} else {

res.write("<h1>Sign-in Failed</h1>") res.end()

}

});

});

})

}

function showsignin(req, res) { fs.readFile('signin.html', function (err, data) {

res.write(data); return res.end();

});

}

function index(req, res) { fs.readFile('index.html', function (err, data) {

res.write(data); return res.end();

});

}

const server = http.createServer(onRequest)

//start the server server.listen(port, hostname, () => {

console.log(`Server running at http://${hostname}:${port}/`)

})

**Index.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>Document</title>

</head>

<body>

<h1><a href="showsignin">Sign-In</a></h1>

<h1><a href="showsignup">Sign-up</a></h1>

</body>

</html>

**Signup.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>Document</title>

</head>

<body>

<h1>Welcome to Sign-up Page</h1>

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

<label for="username">Username</label>

<input type="text" name="username" id=""/>

<br>

<label for="passwd">Password</label>

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

<br>

<label for="confpasswd">Confirm Password</label>

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

<input type="submit" name="submit" id="" value="Sign-In"/>

</form>

</body>

</html>

**Signin.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>Document</title>

</head>

<body>

<h1>Welcome to Sign-In Page</h1>

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

<label for="username">Username</label>

<input type="text" name="username" id=""/>

<br>

<label for="passwd">Password</label>

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

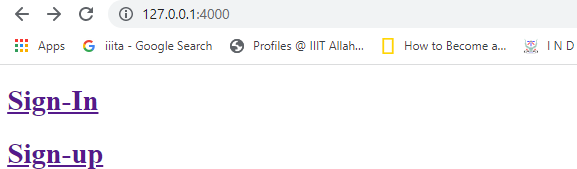
<input type="submit" name="submit" id="" value="Sign-In"/>

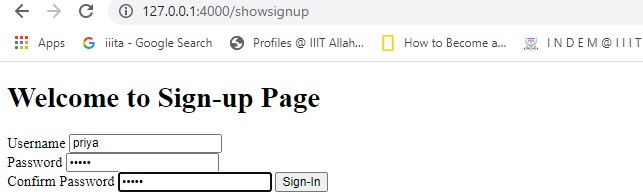
</form>

</body>

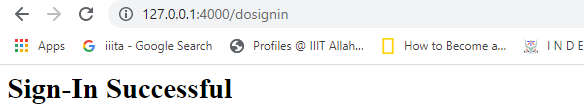
</html>

**Output:**

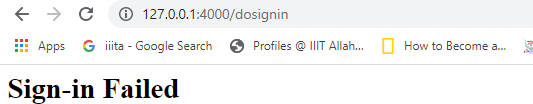


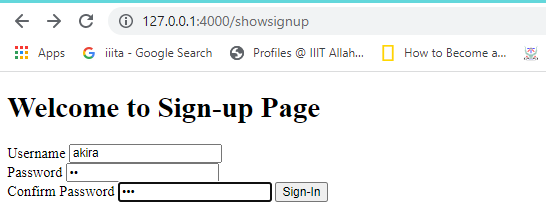


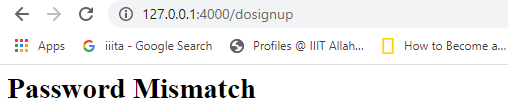












**Database:**

