****

**Practical of Web Framework**

**Submitted by**

**\_\_\_**

**of**

**S. Y. MSc(CS)**

**Year: 2022-2023**

**G.H. RAISONI COLLEGE OF ARTS, COMMERCE & SCIENCE**

**WAGHOLI, PUNE- 412207.**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr.no.** | **Practical Name** | **Page no** | **Sign** |
| 1. | Student Registration HTML Form | 3 |  |
| 2. | Employee Registration HTML Form | 5 |  |
| 3. | HTML form for Login using JavaScript to validate email ID | 7 |  |
| 4. | Create a Node.js file to convert the output "Hello World!" into upper-case | 9 |  |
| 5. | Read two file names from user and append contents of first file into second file | 10 |  |
| 6. | Node.js file that opens the requested file and returns the content to the client. If anything goes wrong, throw a 404 error | 11 |  |
| 7. | Node.js file that writes an HTML form, with an upload field | 12 |  |
| 8. | Node.js file that demonstrate create database and table in MySQL | 13 |  |
| 9. | Node.js file that Select all records from the "customers" table, and display the result object on console | 14 |  |
| 10. | Node.js file that Insert Multiple Records in "student" table, and display the result object on console | 15 |  |
| 11. | Node.js file that Select all records from the "customers" table, and delete the specified record. | 16 |  |
| 12. | Create a Simple Web Server using node js | 17 |  |
| 13. | Using node js create a User Login System | 18 |  |
| 14. | Using node js create a eLearning System | 22 |  |
| 15. | Using node js create a Recipe Book | 25 |  |
| 16. | write node js script to interact with the filesystem, and serve a web page from a file | 28 |  |
| 17. | Build Your Own Node.js Module. Use require (‘http’) module is a built-in Node module that invokes the functionality of the HTTP library to create a local server. | 32 |  |
| 18. | js file named main.js for event-driven application. There should be a main loop that listens for events, and then triggers a callback function when one of those events is detected. | 33 |  |
| 19. | Node js application that transfer a file as an attachment on web and enables browser to prompt the user to download file using express js. | 34 |  |
| 20. | Create your Django app in which after running the server, you should see on the browser, the text “Hello! I am learning Django”, which you defined in the index view. | 35 |  |
| 21. | Design a Django application that adds web pages with views and templates | 37 |  |

1.Create an HTML form that contain the Student Registration details and write a JavaScript to validate Student first and last name as it should not contain other than alphabets and age should be between 18 to 50.

<html>

<head>

<script type="text/javascript">

function validate()

{

var regName=/^[a-zA-z]+[a-zA-Z]+$/;

var fname=document.getElementById("txtfname").value;

var lname=document.getElementById("txtlname").value;

var age=document.getElementById("txtage").value;

var mobno=document.getElementById("txtmobno").value;

if(age<18||age>50)

alert("student age must be 18 to 50");

if(!regName.test(fname))

alert("invalid name is given");

else

alert("valid name is given");

}

</script>

</head>

<body>

<form>

enter student first name

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

enter student last name

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

enter student age

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

enter mobile no

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

<input type="button" value="validate" onclick="validate()">

</form>

</body>

</html>

2.Create an HTML form that contain the Employee Registration details and write a JavaScript to validate DOB, Joining Date, and Salary.

<html>

    <head>

        <script type="text/javascript">

        function validate()

        {

        var regName=/^[a-zA-z]+[a-zA-Z]+$/;

        var dateformatdob = /^(0?[1-9]|[12][0-9]|3[01])[\/\-](0?[1-9]|1[012])[\/\-]\d{4}$/;

        var dateformatjdate = /^(0?[1-9]|[12][0-9]|3[01])[\/\-](0?[1-9]|1[012])[\/\-]\d{4}$/;

     var salaryformat=/^\d{1,6}(?:\.\d{0,2})?$/

        var name=document.getElementById("txtname").value;

        var dob=document.getElementById("txtdob").value;

        var jdate=document.getElementById("txtjdate").value;

        var salary=document.getElementById("txtsalary").value;

        if(!regName.test(name))

            alert("invalid name is given");

        else

            alert("valid name is given");

        if(!dateformatjdate.test(jdate))

            alert("invalid joining date is given");

        else

            alert("valid joining date is given");

        if(!dateformatdob.test(dob))

            alert("invalid date of birth is given");

        else

            alert("valid date of birth is given is given");

        if(!salaryformat.test(salary))

            alert("invalid salary");

        else

            alert("salary is valid");

        }

        </script>

    </head>

<body>

<form>

    <h1>Employee Rsgistration Details</h1>

        enter employee first name

        <input type="text" name="txtfname" id="txtname"><br>

        enter date of birth

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

        enter joining date

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

        enter salary

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

        <input type="button" value="validate" onclick="validate()">

    </form>

    </body>

</html>

3.Create an HTML form for Login and write a JavaScript to validate email ID using Regular Expression.

<html>

<head>

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

<style>

    </style>

</head>

<body>

<script>

function validateform(){

var email = document.getElementById("email").value;

var password = document.getElementById("psw").value;

if (/^\w+([\.-]?\w+)\*@\w+([\.-]?\w+)\*(\.\w{2,3})+$/.test(email))

  {

    alert("Valid Email Id..")

    return (true)

  }

   else{ alert("You have entered an invalid email address!")

    return (false)

   }

}

</script>

<form name="myform" onsubmit="return validateform()">

  <div class="container">

    <p>Please fill in this form to Login.</p>

    <hr>

    <label for="email"><b>Email</b></label>

    <input type="text" autocomplete="off" placeholder="Enter Email" name="email" id="email" required>

    <label for="psw"><b>Password</b></label>

    <input type="password" autocomplete="off" placeholder="Enter Password" name="psw" id="psw" required>

    <hr>

    <button type="submit" class="registerbtn">Register</button>

  </div>

</form>

</body>

</html>

4.Create a Node.js file that will convert the output "Hello World!" into upper-case letters:

var http = require('http');  // includes the http module

var uc = require('upper-case'); // include the upper-case module

http.createServer(function (req, res) {

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

    res.write(uc("hello world!")); // assign the upper-case module

    res.end();

}).listen(8080);

5.Using nodejs create a web page to read two file names from user and append contents of first file into second file.

const fs = require('fs');

console.log("\nFile Contents of file before append:",

a=fs.readFileSync("file1.txt", "utf8"));

fs.appendFile("file2.txt", a, (err) => {

if (err) {

    console.log(err);

}

else {

    console.log("\nFile Contents of file after append:",

    fs.readFileSync("file2.txt", "utf8"));

}

});

6.Create a Node.js file that opens the requested file and returns the content to the client. If anything goes wrong, throw a 404 error.

var http = require('http');

var url = require('url');

var fs = require('fs');

http.createServer(function (req, res) {

  var q = url.parse(req.url, true);

  var filename = "." + q.pathname;

  fs.readFile(filename, function(err, data) {

    if (err) {

      res.writeHead(404, {'Content-Type': 'text/html'});

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

    }

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

    res.write(data);

    return res.end();

  });

}).listen(8080);

7.Create a Node.js file that writes an HTML form, with an upload field

var http = require('http');

 http.createServer(function (req, res) {

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

  res.write('<form action="fileupload" method="post" enctype="multipart/form-data">');

  res.write('<input type="file" name="filetoupload"><br>');

  res.write('<input type="submit">');

  res.write('</form>');

  return res.end();

}).listen(8080);

8.Create a Node.js file that demonstrate create database and table in MySQL

var mysql = require('mysql');

var con = mysql.createConnection({

host: "localhost",

user: "root",

password: "nikhil96",

database: "Node"

});

con.connect(function(err) {

if (err) throw err;

console.log("Connected!");

con.query("CREATE DATABASE WFN", function (err, result) {

if (err) throw err;

console.log("Database created");

});

});

var sql = "CREATE TABLE customers2(name VARCHAR(25), address VARCHAR(25))";

con.query(sql, function (err, result) {

  if (err) throw err;

  console.log("Table created");

});

9.Create a node.js file that Select all records from the "customers" table, and display the result object on console.

var mysql = require('mysql');

var con = mysql.createConnection({

  host: 'localhost',

  user: "root",

  password: "nikhil96",

  database:'employee'

});

con.connect(function(err) {

  if (err) throw err;

  console.log("Connected!");

});

con.query('SELECT \* FROM emp', (err,rows) => {

    if(err) throw err;

    console.log('Data received from Db:');

    console.log(rows);

  });

10.Create a node.js file that Insert Multiple Records in "student" table, and display the result object on console

var mysql = require('mysql');

var con = mysql.createConnection({

host: "localhost",

user: "root",

password: "nikhil96",

database: "node"

});

con.connect(function(err) {

if (err) throw err;

console.log("Connected!");

var sql = "INSERT INTO student (rollno,name, percentage) VALUES ?";

var values = [

[1,'abc', 77.6],

[2,'def', 89.6],

[3,'ghi', 91.6]

];

con.query(sql, [values], function (err, result)

 {

    if (err) throw err;

    console.log("Number of records inserted: " + result.affectedRows);

  });

con.query("SELECT \* FROM student", function (err, result, fields) {

    if (err) throw err;

    console.log(result);

  });

});

11.Create a node.js file that Select all records from the "customers" table, and delete the specified record.

var mysql = require('mysql');

var con = mysql.createConnection({

  host: "localhost",

  user: "root",

  password: "nikhil96",

  database: "employee"

});

con.connect(function(err) {

  if (err) throw err;

  var sql = "DELETE FROM emp WHERE name = 'dada'";

  con.query(sql, function (err, result) {

    if (err) throw err;

    console.log("Number of records deleted: " + result.affectedRows);

  });

});

12. Create a Simple Web Server using node js

var http = require('http'); // 1 - Import Node.js core module

var server = http.createServer(function (req, res) {   // 2 - creating server

    //handle incomming requests here..

});

server.listen(5000); //3 - listen for any incoming requests

console.log('Node.js web server at port 5000 is running..')

13.Using node js create a User Login System

INDEX HTML file

<!DOCTYPE html>

<html lang = "en">

<head>

    <meta charset = "UTF-8">

    <title> My Form </title>

    <style>

        a{

            font-size: 40px;

        }

        </style>

</head>

<body align='center'>

    <a href="./registration.html">Register</a>

    <br>

    <a href="./login.html">Login</a>

</body>

</html>

LOGIN HTML file

<!DOCTYPE html>

<html lang = "en">

<head>

    <meta charset = "UTF-8">

    <title> My Form </title>

    <style>

        #mylink{

            font-size: 25px;

        }

    </style>

</head>

<body align='center'>

        <header>

        <h1>Login</h1>

    </header>

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

        <fieldset>

            <label>Email ID</label>

            <input type ="email" id = 'email' name="email" placeholder="abc@example.com" required>

            <br><br>

            <label>Password</label>

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

            <br><br>

            <button type ="reset">Reset</button>

            <button type ="submit">Submit</button>

        </fieldset>

        </form>

        <br><br>

        <a id="mylink" href="./registration.html">register</a>

</body>

</html>

Register Html file

<!DOCTYPE html>

<html lang = "en">

<head>

    <meta charset = "UTF-8">

    <title> My Form </title>

    <style>

        #mylink{

            font-size: 25px;

        }

    </style>

</head>

<body align='center'>

        <header>

        <h1>Register</h1>

    </header>

    <form action="/register" method="POST">

        <fieldset>

            <label>Username</label>

            <input type ="text" id = 'username' name="username" placeholder="maverick" required>

            <br><br>

            <label>Email ID</label>

            <input type ="email" id = 'email' name="email" placeholder="abc@example.com" required>

            <br><br>

            <label>Password</label>

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

            <br><br>

            <button type ="reset">Reset</button>

            <button type ="submit">Submit</button>

        </fieldset>

    </form>

    <br><br>

        <a id="mylink" href="./login.html">login</a>

</body>

</html>

14.Using node js create a eLearning System

<html>

  <head>

    <title>Skills</title>

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

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">

  </head>

  <style>

  body {

    font-family: "Lato", sans-serif;

  background-color: blue;

  }

/\* Fixed sidenav, full height \*/

.sidenav {

  height: 100%;

  width: 200px;

  position: fixed;

  z-index: 1;

  top: 0;

  left: 0;

  background-color: #111;

  overflow-x: hidden;

  padding-top: 20px;

}

/\* Style the sidenav links and the dropdown button \*/

.sidenav a, .dropdown-btn {

  padding: 6px 8px 6px 16px;

  text-decoration: none;

  font-size: 20px;

  color: #818181;

  display: block;

  border: none;

  background: none;

  width: 100%;

  text-align: left;

  cursor: pointer;

  outline: none;

}

/\* On mouse-over \*/

.sidenav a:hover, .dropdown-btn:hover {

  color: #f1f1f1;

}

/\* Main content \*/

.main {

  margin-left: 200px; /\* Same as the width of the sidenav \*/

  font-size: 20px; /\* Increased text to enable scrolling \*/

  padding: 0px 10px;

}

/\* Add an active class to the active dropdown button \*/

.active {

  background-color: green;

  color: white;

}

/\* Dropdown container (hidden by default). Optional: add a lighter background color and some left padding to change the design of the dropdown content \*/

.dropdown-container {

  display: none;

  background-color: #262626;

  padding-left: 8px;

}

/\* Optional: Style the caret down icon \*/

.fa-caret-down {

  float: right;

  padding-right: 8px;

}

/\* Some media queries for responsiveness \*/

@media screen and (max-height: 450px) {

  .sidenav {padding-top: 15px;}

  .sidenav a {font-size: 18px;}

}

  </style>

  <body>

    <center><h1> BOOKS </h1></center>

    <div class="sidenav">

      <a class="active" href="menu.jpg" target="a">Home</a>

      <button class="dropdown-btn">Subject List

        <i class="fa fa-caret-down"></i>

      </button>

      <div class="dropdown-container">

        <a href="/rice" target="a">Java</a>

        <a href="/snacks" target="a">Python</a>

        <a href="/cake" target="a">javascript</a>

        <a href="/chicken" target="a">Node</a>

      </div>

      <a href="/contact" target="a">Contact Us</a>

      <a href="/about" target="a">About Us</a>

    </div>

    <div class="content">

      <iframe src="C:\Users\Admin\Desktop\Practicals\nodejs\ass4\ass4\samosa.jpg" name="a" height="100%" width="100%"></iframe>

    </div>

    <script>

var dropdown = document.getElementsByClassName("dropdown-btn");

var i;

for (i = 0; i < dropdown.length; i++) {

  dropdown[i].addEventListener("click", function() {

  this.classList.toggle("active");

  var dropdownContent = this.nextElementSibling;

  if (dropdownContent.style.display === "block") {

  dropdownContent.style.display = "none";

  } else {

  dropdownContent.style.display = "block";

  }

  });

}

    </script>

  </body>

</html>

15. Using node js create a Recipe Book

About html file

<!DOCTYPE html>

<html>

 <head>

    <!-- Required meta tags -->

    <meta charset="utf-8">

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

    <!-- Bootstrap CSS -->

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3" crossorigin="anonymous">

   <title> About Us</title>

  </head>

<style>

.container {

  position: relative;

  text-align: center;

  color: white;

}

.centered {

  position: absolute;

  top: 50%;

  left: 50%;

  transform: translate(-50%, -50%);

}

</style>

</head>

<body style="background-color:powderblue;">

    <nav class="navbar navbar-expand-lg navbar-dark bg-dark">

  <div class="container-fluid">

    <a class="navbar-brand" href="http://localhost:3000/" style="font-size:30px;">RecipeBook</a>

    <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">

      <span class="navbar-toggler-icon"></span>

    </button>

    <div class="collapse navbar-collapse" id="navbarSupportedContent">

      <ul class="navbar-nav me-auto mb-2 mb-lg-0">

        <li class="nav-item dropdown">

          <a class="nav-link dropdown-toggle" href="#" id="navbarDropdown" role="button" data-bs-toggle="dropdown" aria-expanded="false" style="font-size:20px;">

             Menu

          </a>

          <ul class="dropdown-menu" aria-labelledby="navbarDropdown">

            <li><a class="dropdown-item" href="/rice">Rice</a></li>

            <li><a class="dropdown-item" href="/cake">Cakes</a></li>

            <li><a class="dropdown-item" href="/rice">Rice</a></li>

            <li><a class="dropdown-item" href="/chicken">Chicken</a></li>

            <li><hr class="dropdown-divider"></li>

            <li><a class="dropdown-item" href="/no">Something else here</a></li>

          </ul>

        </li>

       <li class="nav-item">

          <a class="nav-link" href="/contact" style="font-size:20px;">Contact Us</a>

        </li>

  <li class="nav-item">

          <a class="nav-link" href="/about" style="font-size:20px;">About Us</a>

        </li>

      </ul>

    </div>

  </div>

</nav>

<div>

<center><BR><BR><BR><BR><BR><BR><B style="font-size:20px;">This is simple recipe book app. <BR>

</center>

</div>

   <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlRH9sENBO0LRn5q+8nbTov4+1p" crossorigin="anonymous"></script>

    <!-- Option 2: Separate Popper and Bootstrap JS -->

    <!--

    <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.10.2/dist/umd/popper.min.js" integrity="sha384-7+zCNj/IqJ95wo16oMtfsKbZ9ccEh31eOz1HGyDuCQ6wgnyJNSYdrPa03rtR1zdB" crossorigin="anonymous"></script>

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.min.js" integrity="sha384-QJHtvGhmr9XOIpI6YVutG+2QOK9T+ZnN4kzFN1RtK3zEFEIsxhlmWl5/YESvpZ13" crossorigin="anonymous"></script>

    -->

  </body>

</html>

Contact html file

<!DOCTYPE html>

<html>

 <head>

    <!-- Required meta tags -->

    <meta charset="utf-8">

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

    <!-- Bootstrap CSS -->

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3" crossorigin="anonymous">

   <title> Contact Us</title>

  </head>

<style>

.container {

  position: relative;

  text-align: center;

  color: white;

}

.centered {

  position: absolute;

  top: 50%;

  left: 50%;

  transform: translate(-50%, -50%);

}

</style>

</head>

<body style="background-color:powderblue;">

    <nav class="navbar navbar-expand-lg navbar-dark bg-dark">

  <div class="container-fluid">

    <a class="navbar-brand" href="http://localhost:3000/" style="font-size:30px;">RecipeBook</a>

    <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">

      <span class="navbar-toggler-icon"></span>

    </button>

    <div class="collapse navbar-collapse" id="navbarSupportedContent">

      <ul class="navbar-nav me-auto mb-2 mb-lg-0">

        <li class="nav-item dropdown">

          <a class="nav-link dropdown-toggle" href="#" id="navbarDropdown" role="button" data-bs-toggle="dropdown" aria-expanded="false" style="font-size:20px;">

            Make Recipe

          </a>

          <ul class="dropdown-menu" aria-labelledby="navbarDropdown">

            <li><a class="dropdown-item" href="/rice">Rice</a></li>

            <li><a class="dropdown-item" href="/cake">Cakes</a></li>

            <li><a class="dropdown-item" href="/snacks">Snacks</a></li>

            <li><a class="dropdown-item" href="/chicken">Chicken</a></li>

            <li><hr class="dropdown-divider"></li>

            <li><a class="dropdown-item" href="/no">Something else here</a></li>

          </ul>

        </li>

       <li class="nav-item">

          <a class="nav-link" href="/contact" style="font-size:20px;">Contact Us</a>

        </li>

  <li class="nav-item">

          <a class="nav-link" href="/about" style="font-size:20px;">About Us</a>

        </li>

      </ul>

    </div>

  </div>

</nav>

<div>

</div>

   <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlRH9sENBO0LRn5q+8nbTov4+1p" crossorigin="anonymous"></script>

    <!-- Option 2: Separate Popper and Bootstrap JS -->

    <!--

    <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.10.2/dist/umd/popper.min.js" integrity="sha384-7+zCNj/IqJ95wo16oMtfsKbZ9ccEh31eOz1HGyDuCQ6wgnyJNSYdrPa03rtR1zdB" crossorigin="anonymous"></script>

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.min.js" integrity="sha384-QJHtvGhmr9XOIpI6YVutG+2QOK9T+ZnN4kzFN1RtK3zEFEIsxhlmWl5/YESvpZ13" crossorigin="anonymous"></script>

    -->

  </body>

</html>

Demo html file

<html>

  <head>

    <title>Recipe Book</title>

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

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">

  </head>

  <style>

  body {

    font-family: "Lato", sans-serif;

  background-color: blue;

  }

/\* Fixed sidenav, full height \*/

.sidenav {

  height: 100%;

  width: 200px;

  position: fixed;

  z-index: 1;

  top: 0;

  left: 0;

  background-color: #111;

  overflow-x: hidden;

  padding-top: 20px;

}

/\* Style the sidenav links and the dropdown button \*/

.sidenav a, .dropdown-btn {

  padding: 6px 8px 6px 16px;

  text-decoration: none;

  font-size: 20px;

  color: #818181;

  display: block;

  border: none;

  background: none;

  width: 100%;

  text-align: left;

  cursor: pointer;

  outline: none;

}

/\* On mouse-over \*/

.sidenav a:hover, .dropdown-btn:hover {

  color: #f1f1f1;

}

/\* Main content \*/

.main {

  margin-left: 200px; /\* Same as the width of the sidenav \*/

  font-size: 20px; /\* Increased text to enable scrolling \*/

  padding: 0px 10px;

}

/\* Add an active class to the active dropdown button \*/

.active {

  background-color: lightblue;

  color: white;

}

/\* Dropdown container (hidden by default). Optional: add a lighter background color and some left padding to change the design of the dropdown content \*/

.dropdown-container {

  display: none;

  background-color: #262626;

  padding-left: 8px;

}

/\* Optional: Style the caret down icon \*/

.fa-caret-down {

  float: right;

  padding-right: 8px;

}

/\* Some media queries for responsiveness \*/

@media screen and (max-height: 450px) {

  .sidenav {padding-top: 15px;}

  .sidenav a {font-size: 18px;}

}

  </style>

  <body>

    <center><h1>RECIPE BOOK</h1></center>

    <div class="sidenav">

      <a class="active" href="menu.jpg" target="a">Home</a>

      <button class="dropdown-btn">Deliciout Menus

        <i class="fa fa-caret-down"></i>

      </button>

      <div class="dropdown-container">

        <a href="/rice" target="a">Rice</a>

        <a href="/snacks" target="a">Snacks</a>

        <a href="/cake" target="a">Cakes</a>

        <a href="/chicken" target="a">Chicken</a>

      </div>

      <a href="/contact" target="a">Contact Us</a>

      <a href="/about" target="a">About Us</a>

    </div>

    <div class="content">

      <iframe src="C:\Users\Admin\Desktop\Practicals\nodejs\ass4\ass4\samosa.jpg" name="a" height="100%" width="100%"></iframe>

    </div>

    <script>

var dropdown = document.getElementsByClassName("dropdown-btn");

var i;

for (i = 0; i < dropdown.length; i++) {

  dropdown[i].addEventListener("click", function() {

  this.classList.toggle("active");

  var dropdownContent = this.nextElementSibling;

  if (dropdownContent.style.display === "block") {

  dropdownContent.style.display = "none";

  } else {

  dropdownContent.style.display = "block";

  }

  });

}

    </script>

  </body>

</html>

16. write node js script to interact with the filesystem, and serve a web page from a file

var express = require('express');

var app = express();

var PORT = 3000;

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

    res.download('hello.txt');

});

app.listen(PORT, function(err){

    if (err) console.log(err);

    console.log("Server listening on PORT", PORT);

});

17. Write node js script to build Your Own Node.js Module. Use require (‘http’) module is a built-in Node module that invokes the functionality of the HTTP library to create a local server. Also use the export statement to make functions in your module available externally. Create a new text file to contain the functions in your module called, “modules.js” and add this function to return today’s date and time.

var http = require('http');

var dt = require('./myfirstmodule');

http.createServer(function (req, res) {

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

  res.write("The date and time are currently: " + dt.myDateTime());

  res.end();

}).listen(8080);

18. Create a js file named main.js for event-driven application. There should be a main loop that listens for events, and then triggers a callback function when one of those events is detected.

Create a js file named main.js with the following code −

[Live Demo](http://tpcg.io/sw4wtc)

// Import events module

var events = require('events');

// Create an eventEmitter object

var eventEmitter = new events.EventEmitter();

// Create an event handler as follows

var connectHandler = function connected() {

console.log('connection succesful.');

// Fire the data\_received event

eventEmitter.emit('data\_received');

}

// Bind the connection event with the handler

eventEmitter.on('connection', connectHandler);

// Bind the data\_received event with the anonymous function

eventEmitter.on('data\_received', function() {

console.log('data received succesfully.');

});

// Fire the connection event

eventEmitter.emit('connection');

console.log("Program Ended.");

Create a js file named main.js having the following code −

var fs = require("fs");

fs.readFile('input.txt', function (err, data) {

if (err) {

console.log(err.stack);

return;

}

console.log(data.toString());

});

console.log("Program Ended");

19. Write node js application that transfer a file as an attachment on web and enables browser to prompt the user to download file using express js.

Step 1: Create a package.json file and install dependencies.

$ mkdir expressjs-download

$ cd expressjs-download

$ npm init --yes

$ npm i -S express

Step 2: create a file called index.js in root expressjs-download/index.js of project. And create a express server with route, it should accept file name in the URL.

const express = require('express'); // import express js library

const app = express(); //create express js instance

const path = require('path');

// define a route to download a file

app.get('/download/:file(\*)',(req, res) => {

var file = req.params.file;

var fileLocation = path.join('./uploads',file);

console.log(fileLocation);

res.download(fileLocation, file);

});

app.listen(8000,() => {

console.log(`application is running at: http://localhost:8000`);

});

Step 3: Run the application by issuing node index.js command from your terminal, you should be able to see the output "application is running at: http://localhost:8000" in your console and if you head over to the browser with http://localhost:8000/download/{filename}, you will get download prompt.

Here {filename} = name of the file including extension ( it will look for the file in expressjs-download/uploads directory)

20. Create your Django app in which after running the server, you should see on the browser, the text “Hello! I am learning Django”, which you defined in the index view.

<html>

<head>

<script type="text/javascript">

function validate()

{

var regName=/^[a-zA-z]+[a-zA-Z]+$/;

var fname=document.getElementById("txtfname").value;

var lname=document.getElementById("txtlname").value;

var age=document.getElementById("txtage").value;

var mobno=document.getElementById("txtmobno").value;

if(age<18||age>50)

alert("student age must be 18 to 50");

if(!regName.test(fname))

alert("invalid name is given");

else

alert("valid name is given");

}

</script>

</head>

<body>

<form>

enter student first name

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

enter student last name

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

enter student age

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

enter mobile no

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

<input type="button" value="validate" onclick="validate()">

</form>

</body>

</html>

21. Design a Django application that adds web pages with views and templates.

<html>

<head>

<script type="text/javascript">

function validate()

{

var regName=/^[a-zA-z]+[a-zA-Z]+$/;

var dateformatdob = /^(0?[1-9]|[12][0-9]|3[01])[\/\-](0?[1-9]|1[012])[\/\-]\d{4}$/;

var dateformatjdate = /^(0?[1-9]|[12][0-9]|3[01])[\/\-](0?[1-9]|1[012])[\/\-]\d{4}$/;

//Max six digits, a dot, max two digits after dot

var salaryformat=/^\d{1,6}(?:\.\d{0,2})?$/

var name=document.getElementById("txtname").value;

var dob=document.getElementById("txtdob").value;

var jdate=document.getElementById("txtjdate").value;

var salary=document.getElementById("txtsalary").value;

if(!regName.test(name))

alert("invalid name is given");

else

alert("valid name is given");

if(!dateformatjdate.test(jdate))

alert("invalid joining date is given");

else

alert("valid joining date is given");

if(!dateformatdob.test(dob))

alert("invalid date of birth is given");

else

alert("valid date of birth is given is given");

if(!salaryformat.test(salary))

alert("invalid salary");

else

alert("salary is valid");

}

</script>

</head>

<body>

<form>

<h1>Employee Rsgistration Details</h1>

enter employee first name

<input type="text" name="txtfname" id="txtname"><br>

enter date of birth

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

enter joining date

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

enter salary

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

<input type="button" value="validate" onclick="validate()">

</form>

</body>

</html>