Web Frameworks Practical’s

Q.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.

<!DOCTYPE html>

<html >

<head><title>Student Registration</title>

</head>

<body>

<form name="f1" onsubmit="return validateform()">

<div>

<label for="fname">First Name</label><br>

<input type="text" id="fname",name="fname">

</div>

<div>

<label for="lname">Last Name</label><br>

<input type="text" id="lname",name="lname">

</div>

<div>

<label for="mobile">Mobile Number</label><br>

<input type="text" name="mobile" id="mobile">

</div>

<div>

<label for="age">Age</label><br>

<input type="number" name="age",id="age">

</div>

<br>

<input type="submit" value="Submit">

</form>

</body>

<script type="text/javascript">

function validateform(){

var fname=document.f1.fname.value;

var lname=document.f1.lname.value;

var age=document.f1.age.value;

var age1=parseInt(age,10);

if(fname==""||fname==null){

alert("First name can't be blank");

return false;

}

if(lname==""||lname==null){

alert("Last name can't be blank");

return false;

}

if( age1>18 && age1<50);

else{

alert("Age is not in limit");

return false;

}

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

if(!isNaN(fname.charAt(i))){

alert("Name can't contain numbers");

return false;

}

}

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

if(!isNaN(lname.charAt(i))){

alert("Name can't contain numbers");

return false;

}

}

}

</script>

</html>

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

<!DOCTYPE html>

<html kang="en">

<head><title>Employee Registration</title>

</head>

<body>

<form name="f1" onsubmit="return validateform()">

<table>

<tr>

<td>First Name:</td>

<td><input type="text" name="fname" required><br>

</tr>

<tr>

<td>LastName:</td>

<td><input type="text" name="lname" required><br>

</tr>

<tr>

<td>Email:</td>

<td><input type="email" name="email" required><br>

</tr>

<tr>

<td>DOB:</td>

<td><input type="date" name="dob" required><br>

</tr>

<tr>

<td>Joining date:</td>

<td><input type="date" name="jdate"><br>

</tr>

<tr>

<td>Salary:</td>

<td><input type="number" name="sal"><br>

</tr>

<tr>

<td></td>

<td><input type="submit" name= "reg" value="Register"></td>

</tr>

</table>

</form>

<script>

function validateform(){

var fname=document.f1.fname.value;

var lname=document.f1.lname.value;

var dob=document.f1.dob.value;

var email=document.f1.email.value;

var dob = new Date(document.f1.dob.value);

var cdate = new Date();

var joindate = new Date(document.f1.jdate.value);

var sal = document.f1.sal.value;

var sal1 = parseInt(sal,10);

if(fname==""||fname==null){

alert("First name can't be blank");

return false;

}

if(lname==""||lname==null){

alert("Last name can't be blank");

return false;

}

if( cdate.getFullYear() - dob.getFullYear() > 18 )

{

if(cdate.getFullYear() - joindate.getFullYear() > 1)

{

if(sal1 <= 20000 )

{

alert("Salary can't be less than 20000..");

return false;

}

}

else

{

if(sal1 > 20000)

{

alert("fresher's salary cannot be more than 20000..");

return false;

}

}

}

/\* else

{

alert("You are not 18+..So u cannot register..Sorry");

return false;

} \*/

}

</script>

</body>

</html>

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

<!DOCTYPE html>

<html lang="en">

<head>

<title>Login </title>

</head>

<body>

<form name="f1" onsubmit="return validateform()">

<table>

<tr>

<td>Email\_Id:</td>

<td><input type="text" name = "name" required></td>

</tr>

<br>

<tr>

<td>Password:</td>

<td><input type="password" name = "pass" required></td>

</tr>

<br>

<tr>

<td></td>

<td><input type="submit" name = "login" value="Login"></td>

</tr>

<br>

</table>

</form>

<script>

function validateform(){

var email = document.f1.name.value;

var pass = document.f1.pass.value;

var pat = /^[a-zA-Z0-9.\_-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,4}$/;

if (!pat.test(email))

{

alert("Please enter a valid email address..");

}

if (email == "aish2@gmail.com"&& pass =="qwert" )

{

document.write("Logged in successfully..")

}

else{

alert("Incorrect");

}

}

</script>

</body>

</html>

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

var http = require('http');

var uc = require('upper-case');

http.createServer(function (req, res) {

res.write(uc.upperCase("Hello World!"));

res.end();

}).listen(5000);

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

var fs =require('fs');

function ReadAppend(file,appendFile)

{

fs.readFile(appendFile,function(err,data)

{

if(err) throw err;

console.log('File was read');

fs.appendFile(file,data,function(err)

{

if(err) throw err;

console.log('The "data to append" was appended to file!');

});

});

}

ReadAppend('test.txt','sample.txt');

Q.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) {

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

}

res.write(data);

return res.end();

});

}).listen(8081);

Q.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(8081);

Q.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: "password",

database: "node"

});

con.connect(function(err) {

if (err) throw err;

console.log("Connected!");

var sql = "CREATE TABLE customers (name VARCHAR(255), address VARCHAR(255))";

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

if (err) throw err;

console.log("Table created");

});

});

Q.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: "password",

database: "node"

});

con.connect(function(err) {

if (err) throw err;

var sql="SELECT \* FROM customers";

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

if (err) throw err;

console.log(result);

});

});

Q.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: "password",

database: "node"

});

con.connect(function(err) {

if (err) throw err;

console.log("Connected!");

var sql = "INSERT INTO customers (name, address) VALUES ('Sahil', 'Kurwali')";

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

if (err) throw err;

console.log("1 record inserted");

});

})

Q.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: "password",

database: "node"

});

con.connect(function(err) {

if (err) throw err;

var sql = "DELETE FROM customers WHERE address = 'Kurwali'";

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

if (err) throw err;

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

});

});

Q.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(8081); //3 - listen for any incoming requests

console.log('Node.js web server at port 8081 is running..')

Q.13: Using node js create a User Login System

Login.html

<!DOCTYPE html>

<html>

    <head>

        <meta charset="utf-8">

        <title>Login Form Tutorial</title>

        <style>

        .login-form {

            width: 300px;

            margin: 0 auto;

            font-family: Tahoma, Geneva, sans-serif;

        }

        .login-form h1 {

            text-align: center;

            color: #4d4d4d;

            font-size: 24px;

            padding: 20px 0 20px 0;

        }

        .login-form input[type="password"],

        .login-form input[type="text"] {

            width: 100%;

            padding: 15px;

            border: 1px solid #dddddd;

            margin-bottom: 15px;

            box-sizing:border-box;

        }

        .login-form input[type="submit"] {

            width: 100%;

            padding: 15px;

            background-color: #535b63;

            border: 0;

            box-sizing: border-box;

            cursor: pointer;

            font-weight: bold;

            color: #ffffff;

        }

        </style>

    </head>

    <body>

        <div class="login-form">

            <h1>Login Form</h1>

            <form action="auth" method="POST">

                <input type="text" name="username" placeholder="Username" required>

                <input type="password" name="password" placeholder="Password" required>

                <input type="submit">

            </form>

        </div>

    </body>

</html>

Login.js

var mysql = require('mysql');

var express = require('express');

var session = require('express-session');

var bodyParser = require('body-parser');

var path = require('path');

var connection = mysql.createConnection({

    host     : 'localhost',

    user     : 'root',

    password : 'password',

    database : 'node'

});

var app = express();

app.use(session({

    secret: 'secret',

    resave: true,

    saveUninitialized: true

}));

app.use(bodyParser.urlencoded({extended : true}));

app.use(bodyParser.json());

app.get('/', function(request, response) {

    response.sendFile(path.join(\_\_dirname + '/login.html'));

});

app.post('/auth', function(request, response) {

    var username = request.body.username;

    var password = request.body.password;

    if (username && password) {

        connection.query('SELECT \* FROM accounts WHERE username = ? AND password = ?', [username, password], function(error, results, fields) {

            if (results.length > 0) {

                request.session.loggedin = true;

                request.session.username = username;

                response.redirect('/home');

            } else {

                response.send('Incorrect Username and/or Password!');

            }

            response.end();

        });

    } else {

        response.send('Please enter Username and Password!');

        response.end();

    }

});

app.get('/home', function(request, response) {

    if (request.session.loggedin) {

        response.send('Welcome back, ' + request.session.username + '!');

    } else {

        response.send('Please login to view this page!');

    }

    response.end();

});

app.listen(8081);

Q.14: Using node js create a eLearning System

Html

<html>

 <head>

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

 <style>

 body{

 font-family: Arial;

 color: white;

 }

 .splitl{

 height: 100%;

 width: 30%;

 position: fixed;

 z-index:1;

 top: 0;

 overflow-x: hidden;

 padding-top: 20px;

 }

 .splitl{

 height: 100%;

 width: 30%;

 position: fixed;

 z-index:1;

 top: 0;

 overflow-x: hidden;

 padding-top: 20px;

 }

 .left{

 left: 0;

 background-color: aquamarine;

 }

 .right{

 right:0;

 background-color: brown;

 }

 .centered{

 position: absolute;

 transform: translate(-50%,-50%);

 top: 50%;

 left: 50%;

 text-align: center;

 }

 .link{

 position: absolute;

 width: 100%;

 height: 100%;

 top: 0;

 left: 0;

 z-index: 1;

 }

 </style>

 </head>

 <body>

 <div class="splitl left">

 <div class="centered">

 <h1>

 <a href="/java\_tutorial" target="a">JAVA</a><br>

 <a href="/c\_tutorial" target="a">C</a><br>

 <a href="/javascript\_tutorial" target="a">JavaScript</a><br>

 <a href="/nodejs\_tutorial" target="a">NodeJs</a><br>

 </h1>

 </div>

 </div class="split right">

 <iframe name="a" height="100%" width="100%"></iframe>

 </iframe>

 <div>

 </div>

 </body>

 </html>

JS

var fs=require('fs');

var http=require('http');

var con=http.createServer(function(req,res){

 if(req.url=='/'){

 fs.readFile('prog14.html',function(err,data){

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

 res.write(data);

 res.end();

 });

 }

 else if(req.url=='/java\_tutorial'){

 fs.readFile('java\_tutorial.pdf',function(err,data){

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

 res.write(data);

 res.end();

 });

 }

 else if(req.url=='/c\_tutorial'){

 fs.readFile('c\_tutorial.pdf',function(err,data){

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

 res.write(data);

 res.end();

 });

 }

 else if(req.url=='/javascript\_tutorial'){

 fs.readFile('javascript\_tutorial.pdf',function(err,data){

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

 res.write(data);

 res.end();

 });

 }

 else if(req.url=='/nodejs\_tutorial'){

 fs.readFile('nodejs\_tutorial.pdf',function(err,data){

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

 res.write(data);

 res.end();

 });

 }else{

 res.end("The end");

 }

}).listen(9000);

console.log("SUCCESS")

Q.15: Using node js create a Recipe Book

Html

<html>

 <head>

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

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

 <style>

 body {

 font-family: "Lato", sans-serif;

}

/\* 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>

 </head>

 <body>

 <center><h1>My Recipe Book </h1></center>

 <div class="sidenav">

 <a class="active" href="recipe.jpg" target="a">HOME</a>

 <button class="dropdown-btn"> Delicious Menu

 <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="/bakery" target="a">Bakery</a>

 </div>

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

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

 </div>

 <div class="content">

 <iframe src="cake.jpg" name="a" height="100%" width="100&"></iframe>

 </div>

 <script>

 /\* Loop through all dropdown buttons to toggle between hiding and showing its dropdown

content - This allows the user to have multiple dropdowns without any conflict \*/

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>

 </div>

 </body>

 </html>

JS

const { Console } = require('console');

var fs=require('fs');

var http=require('http');

var con=http.createServer(function(req,res){

 if(req.url=='/'){

 fs.readFile('prog15.html',function(err,data){

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

 res.write(data);

 res.end();

 });

 }

 // else if(req.url.match('.\png$')){

 //var filestream=fs.createReadStream("menu.png");

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

 // filestream.pipe(res);

 // }

 else if(req.url=='/contact'){

 fs.readFile('contact.html',function(err,data){

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

 res.write(data);

 res.end();

 });

 }

 else if(req.url=='/about'){

 fs.readFile('about.html',function(err,data){

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

 res.write(data);

 res.end();

 });

 }

 else if(req.url=='/rice'){

 fs.readFile('rice.pdf',function(err,data){

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

 res.write(data);

 res.end();

 });

 }

 else if(req.url=='/snacks'){

 fs.readFile('snacks.pdf',function(err,data){

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

 res.write(data);

 res.end();

 });

 }

 else if(req.url=='/bakery'){

 fs.readFile('bakery.pdf',function(err,data){

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

 res.write(data);

 res.end();

 });

 }

 else{

 res.end("The END");

 }

}).listen(9000);

console.log("SUCCESS")

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

let http = require('http');

let fs = require('fs');

let handleRequest = (request, response) => {

 response.writeHead(200, {

 'Content-Type': 'text/html'

 });

 fs.readFile('simple.txt', null, function (error, data) {

 if (error) {

 response.writeHead(404);

 response.write('Whoops! File not found!');

 } else {

 response.write(data);

 }

 response.end();

 });

};

http.createServer(handleRequest).listen(8000)

Q.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.

**Modules.js**

var http = require('http');

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

http.createServer(function (req, res) {

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

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

res.end();

}).listen(8000);

**CurrentDate.js**

exports.myDateTime = function () {

return Date();

};

Q.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.

const EventEmitter = require('events');

const myEmitter = new EventEmitter();

setImmediate(function(){

 myEmitter.emit('Test-Event');

 myEmitter.emit('LOADED');

});

setTimeout(function(){

 console.log('2000 milliseconds have passes')

},2000);

myEmitter.on('Test-Event', function(){

 console.log('The test event was fired');

});

myEmitter.on('Test-Event', function(){

 console.log('The test event1 was fired');

});

myEmitter.on('LOADED', function(){

 console.log('The test load was fired');

})

Q.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.

**Express.js**

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);

});

Q.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.

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

Q.22: Write and run Django code to add data to your site using relational databases with Django’s Object Relational Mapper.

Q.23: Develop a basic poll application (app).It should consist of two parts:

a) A public site in which user can pick their favourite programming language and vote.

b) An admin site that lets you add, change and delete programming languages.

Q.24: A public site in which user can pick their favourite programming language and vote.

Q.25: An admin site that lets you add, change and delete programming languages.

Q.26: Implement a simple Django application for portfolio management.

Q.27: Create your own blog using Django

Q.28: Build your own To-Do app in Django

Q.29: Create a clone of the “Hacker News” website.

Q.30: Develop Online School System using Django

Q.31: Implement your E-commerce Website using Django

Q.32: Implement Login System using Django