



## Web Frameworks

MSc(computer Science) (Savitribai Phule Pune University)

Q1) 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>
  <title>Student Registration</title>
  <script>
    function validate() {
      var Fname = document.getElementById("fname").value;
      let reF = /\d/;
      var Lname = document.getElementById("lname").value;
      let reL = /\d/;
      var Age = document.getElementById("age").value;

      if (reF.test(Fname)) {
        alert("Please use Alphabates to write first name.");
        return false;
      }
      else if ( reL.test(Lname)) {
        alert("Please use Alphabates to write last name.");
        return false;
      }

      else if(!(Age < 50 && Age > 18) )
      {
        alert("age should be between 18 to 50 ");
        return false;
      }
      alert('Registration Successful');
      return true;
    }
  </script>
</head>

<body bgcolor="yellow">

  <form onsubmit=validate(>
    <h1><b>Student Registration</b></h1>
    First Name:<input type="text" id="fname" /><br>
    Last Name:<input type="text" id="lname" /><br>
    Age:<input type="text" id="age" /><br>
    <input type="submit" >
  </form>
</body>

</html>
```

Q2) Create an HTML form that contain the Employee Registration details and write a JavaScript to validate DOB, Joining Date, and Salary

```
<html>

<head>
  <title>Employee Registration</title>
  <script>
    function validate(){
      const d = new Date();
      var DOB = document.getElementById("dob").value;
      var DOB1 =new Date(DOB);
      var jd = document.getElementById("joiningDate").value;
      var jd1 = new Date(jd);

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

      if (!(DOB1 < d )){
        alert("DOB should be less than current date ");
        return false;
      }
      else if (!(jd1 >= d)){
        alert("joining date should be greater than or equal to
current date ");
        return false;
      }
      else if (!(Salary >= 10000)){
        alert("salary should be greater than 10000");
        return false;
      }
      alert("Verification Complete");
      return true;
    }
  </script>
</head>
<body bgcolor="pink">
  <form>
    <h2>Employee registration</h2>
    Name:<input type="text" id="name" /><br>
    DOB:<input type="date" id="dob" /><br>
    Joining Date:<input type="date" id="joiningDate" /><br>
    Salary:<input type="number" id="salary"><br>
    <input type="submit" value="Submit" onclick = validate()>
  </form>
</body>
</html>
```

Q3) Create an HTML form for Login and write a JavaScript to validate email ID using Regular Expression.

```
<html>

<head>
  <title> Login Form </title>
  <script>
    function validate() {
      var username = document.getElementById("username").value;
      var password = document.getElementById("pass").value;
      let re = new RegExp('^[a-zA-Z0-9._-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,4}$')
      if (!re.test(username)) {
        alert("Please enter the username.");
        // return false;
      }
      if (password == null || password == "") {
        alert("Please enter the password.");
        // return false;
      }
      alert('Login successful');
      // return true;
    }
  </script>
</head>

<body bgcolor="sky blue">
  <form action="D:\divya\Web Frameworks\abc.html" method="get"
onsubmit=validate()>
    username:<input type="text" id="username" /><br>
    Password:<input type="password" id="pass" /><br>
    <input type="submit" value="submit">
  </form>
</body>

</html>
```

Q4) 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.writeHead(200,{ 'content-type': 'text/html' });
  res.write(uc.upperCase('hello world !'));
  res.end();
}).listen(8083)
```

Q5) Using nodejs create a web page to read two file names from user and append contents of first file into second file

→Q5.js

```
var http = require('http');
var fs = require('fs');
var formidable = require('formidable');

http.createServer(function(req,res){
  if(req.url == '/'){
    res.writeHead(200,{ 'content-type': 'text/html' });
    res.write('<form action = "fapp" method="post" enctype = "multipart/form-data">');
    res.write('<h1>SELECT TWO FILES</h1>');
    res.write('<input type = "file" name = "rf"><br>');
    res.write('<input type = "file" name = "wf"><br>');
    res.write('<input type = "submit">');
    res.end();
  }
  else if(req.url == '/fapp'){
    var form = new formidable.IncomingForm();
    form.parse(req,function(err,fields,files){
      if(!err){
        var w = fs.createWriteStream(files.wf.originalFilename,
{flags:'a'});
        var r = fs.createReadStream(files.rf.originalFilename);
        w.on('close',function(){
          console.log("Writing Done");
        });
        r.pipe(w);
        res.write(files.rf.originalFilename);
        res.end("Append Successfully");
      }
      else{res.write("error in writing");}
    });
  }
  else{
    res.end("page not found");
  }
}).listen(8001);
```

Q6) 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);

```

Q7) 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="fileupload"><br>');
  res.write('<input type="submit">');
  res.write('</form>');
  return res.end();
}).listen(8080);

```

```

var http = require('http');
var formidable = require('formidable');

http.createServer(function (req, res) {
  if (req.url == '/fileupload') {
    var form = new formidable.IncomingForm();
    form.parse(req, function (err, fields, files) {
      res.write('File uploaded');
      res.end();
    });
  } else {
    res.writeHead(200, { 'Content-Type': 'text/html' });
    res.write('<form action="fileupload" method="post" enctype="multipart/form-data">');
    res.write('<input type="file" name="fileupload"><br>');
    res.write('<input type="submit">');
  }
});

```

```

    res.write('</form>');
    return res.end();
  }
}).listen(8080);

```

```

var http = require('http');
var formidable = require('formidable');
var fs = require('fs');

http.createServer(function (req, res) {
  if (req.url == '/fileupload') {
    var form = new formidable.IncomingForm();
    form.parse(req, function (err, fields, files) {
      var oldpath = files.fileupload.filepath;
      var newpath = 'C:\Users\LAB-2\Desktop\demo' +
files.fileupload.originalFilename;
      fs.rename(oldpath, newpath, function (err) {
        if (err) throw err;
        res.write('File uploaded and moved!');
        res.end();
      });
    });
  } else {
    res.writeHead(200, {'Content-Type': 'text/html'});
    res.write('<form action="fileupload" method="post"
entype="multipart/form-data">');
    res.write('<input type="file" name="fileupload"><br>');
    res.write('<input type="submit">');
    res.write('</form>');
    return res.end();
  }
}).listen(8080);

```

Q8) Create a Node.js file that demonstrates create database and table in MySQL

```

var mysql = require('mysql');

var con = mysql.createConnection({
  host: "localhost",
  user: "root",
  password: "password"
});

con.connect(function(err) {

```

```

    if (err) throw err;
    console.log("Connected!");
    con.query("CREATE DATABASE db", function (err, result) {
        if (err) throw err;
        console.log("Database created");
    });
});

```

Q9) 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: "db"
});

con.connect(function(err) {
    if (err) throw err;
    console.log("Connected!");
    var sql = "select * from customer";
    con.query(sql, function (err, result, fields){
        if (err) throw err;
        console.log(result);
    });
});

```

Q10) 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:"db"
    });

con.connect(function(err)
{
    if (err) throw err;
    console.log("connected");

```



```

    var sql = 'insert into student values(2,"sham"),(3,"seeta"),
(4,"geeta")';
    con.query(sql, function (err, result,fields){
        if (err) throw err;
        console.log(result);
    });
    var sql1 = "select * from student";
    con.query(sql1, function (err, result,fields){
        if (err) throw err;
        console.log(result);
    });
});

```

Q11) 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:"db"
    });

con.connect(function(err)
{
    if (err) throw err;
    console.log("connected");
    var sql = 'select * from customer';
    con.query(sql, function (err, result,fields){
        if (err) throw err;
        console.log(result);
    });
    var sql1 = "delete from customer where id ='1'";
    con.query(sql1, function (err, result,fields){
        if (err) throw err;
        console.log(result);
    });
    con.query(sql, function (err, result,fields){
        if (err) throw err;
        console.log(result);
    });
});

```

Q12) Create a Simple Web Server using node js

```
var http = require('http');
http.createServer(function(req,res){
  res.writeHead(200,{ 'content-type': 'text/html' });
  res.write("Server Created");
  console.log("Server Created");
  res.end();
}).listen(8080);
```

Q13) Using node js create a User Login System

Q14) Write node js script to interact with the filesystem, and serve a web page from a file

→Q14.html

```
<html>
<body>
<h1>My Header</h1>
<p>My paragraph.</p>
</body>
</html>
```

→Q14.js

```
var http = require('http');
var fs = require('fs');
http.createServer(function (req, res) {
  fs.readFile('Q14.html', function(err, data) {
    res.writeHead(200, { 'Content-Type': 'text/html' });
    res.write(data);
    return res.end();
  });
}).listen(8080);
```

Q15) 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

```
function datetime()
{
  let dt = new Date();
  //current date
  let date = ("0"+dt.getDate()).slice(-2);

  //current month
  let month = ("0"+ (dt.getMonth()+1)).slice(-2);
```

```

    //current year
    let year = dt.getFullYear();

    //current hours
    let hours = dt.getHours();

    //current minutes
    let minutes = dt.getMinutes();

    //current seconds
    let seconds = dt.getSeconds();

    var output = year + "-" + month + "-" + date + " " + hours
+ ":" + minutes + ":" + seconds;
    return output;
}
module.exports = {datetime}

```

→Q15.js

```

var http = require('http');
var dt = require('./modules');

var server = http.createServer(function(req,res){
    res.writeHead(200,{ 'content-type': 'text/html' });
    const result = dt.datetime();
    res.write('current date and time is ');
    res.write(result);
    res.end();
});
server.listen(1234);

```

Q16) 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.

```

//import event modules
var events = require('events');

//create an EventEmitter object
var EventEmitter = new events.EventEmitter();

//create an event handler
var connectHandler = function connected(s){
    console.log('Its',s);
}

```

```
//Bind the connection event with the Handler
eventEmitter.on('data_received',function(name){
    console.log(name,"Understood event -Driven");
});
eventEmitter.emit('data_received',"Divya Meher");

eventEmitter.on('connection',connectHandler);
eventEmitter.emit('connection',"SIMPLE SOLUTION")

console.log("program Ended");
```

Q17) 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.

```
var express = require('express');
const fs = require('fs');
var app = express();
var PORT = 3000;

var bodyParser = require('body-parser');
app.use(bodyParser.urlencoded({extended:false}));
app.get('/',function(req,res){
    const files = fs.createReadStream('Q17.html');
    res.writeHead(200,{ 'content-type': 'text/html' });
    files.pipe(res);
});

app.post('/file-data',function(req,res){
    var name = req.body.id;
    res.download(name);
});

app.listen(PORT,function(err){
    if(err) console.log(err);
    console.log("server Listening port",PORT)
});
```

Q18) 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.

Q19) Design a Django application that adds web pages with views and templates

Q20) 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

Q21) Design a Django application: A public site in which user can pick their favourite programming language and vote.

Q22) Design a Django application: An admin site that lets you add, change and delete programming languages

Q23) Create your own blog using Django.

Q24) Implement Login System using Django.