Roll No. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Exam No. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Dr D Y Patil Educational Enterprises Charitable Trust’s

**DR D Y PATIL SCHOOL OF MCA**

Dr. D.Y. Patil Knowledge City, Charholi (Bk.), Via Lohegaon, Pune – 412105



**CERTIFICATE**

This is to certify that

Mr./ Miss **Samruddhi Sudam Gaikwad**, Of Class **MCA – First year (Sem- II) Div. A** Roll No. **115** Has completed all the practical work in the subject **Advanced Internet Technologies (IT23L)** satisfactorily in the Department of **MCA** as prescribed by University of Pune, in the academic year 2024 – 2025.

University Seat No. \_\_\_\_\_\_\_\_\_\_\_\_\_

Staff In-charge Head of the Department Director

|  |  |
| --- | --- |
|  | Dr D Y Patil Educational Enterprises Charitable Trust’s  **DR D Y PATIL SCHOOL OF MCA**  Dr. D.Y. Patil Knowledge City, Charholi (Bk.), Via Lohegaon, Pune – 412105 |

**INDEX**

Name : **Samruddhi Sudam Gaikwad** Div : **A** Roll No. **115**

Class : **MCA I year (SEM II)** Subject : **Advanced Internet Technologies**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sr.**  **No.** | **Title of Experiment** | **Date** | **Page No.** | **Remark** | **Sign of Teacher** |
| 1 | Create Resume by using HTML and HTML5 tags |  |  |  |  |
| 2 | Program to implement Audio and Video features for your web page. |  |  |  |  |
| 3 | Write HTML Programs design different geometrical shapes using Canvas and SVG (minimum 5 Shapes). |  |  |  |  |
| 4 | Program to design form using HTML5 elements, attributes and Semantics tags and apply HTML5 validation to it |  |  |  |  |
| 5 | Programs to demonstrate external and internal styles in the web page using font, text,  background, borders, opacity and other CSS 3 properties. |  |  |  |  |
| 6 | Programs to demonstrate external and internal styles in the web page using font, text,  background, borders, opacity and other CSS 3 properties. |  |  |  |  |
| 7 | Implement Transformation using Translation, Rotation and Scaling in your web page. |  |  |  |  |
| 8 | Program to show current date and time using user defined module in AngularJs. |  |  |  |  |
| 9 | Write calculator program in AngularJs to perform basic arithmetic operations(+, -, \*, /) using angular controller function. |  |  |  |  |
| 10 | Program using NPM which will convert entered string into either case. |  |  |  |  |
| 11 | Program to demonstrate the installation of Node, NPM, Angular in web base application and Create angular project which will demonstrate the usage of component directive, structural  directive and attribute directives |  |  |  |  |
| 12 | Write Program for Form validation in Angular. |  |  |  |  |
|  | Write a program to create a calculator using Angular. |  |  |  |  |
| 13 | Write a program to create a calculator using Node JS. (Install and configure Node JS and Server) |  |  |  |  |
| 14 | What is NodeJs Architecture? Explain REPL with suitable example.   1. Write REPL code to print number from 1 to 50. 2. Write REPL code to print largest number from three given number. |  |  |  |  |
| 15 | Create angular project which has HTML template and handle the click event on click of the  button (Installation of Angular and Bootstrap 4 CSS Framework) |  |  |  |  |
| 16 | Program for basic operations, array and user interface handling. |  |  |  |  |
| 17 | Program to demonstrate session management using various techniques. |  |  |  |  |
| 18 | Design Employee Information form in PHP and perform the CRUD Operations using  PHP Script.  a) Create mydb database using PHP code  b) Create employee\_info table using PHP code  c) Write PHP program to design Employee information form which includes fields  like (Employee ID, First Name, Lastname, Contact No, Designation, Department,  Salary….) |  |  |  |  |
| 19 | Design Students Registration form (Signup form), Student Login form (signIn form) in PHP and implement session, cookies concept in it.  a) Registration form consists (Student id, Student first & Surname, mobile No, Address…)  b) Student Login form (Student Username, Password)  c) Sign-out form (session destroy implementation) |  |  |  |  |
| 20 | Write event drive program to perform operations like select, insert and delete on Employee information using PHP & MySQL. |  |  |  |  |
| 21 | Explain Selectors and Pseudo Classes of CSS3 with example. Write HTML code with CSS3 classes to design Indian Flag.(Apply Transition & Animation) |  |  |  |  |

1. **Create Resume by using HTML and HTML5 tags**

<!DOCTYPE html>

<!DOCTYPE html>

<html lang="en">

<head>

<title>Hrishikesh parab - Resume</title>

</head>

<body>

<header>

<h1>Hrishikesh Parab</h1>

<p>Business operation analyst</p>

</header>

<section>

<h2>Contact Information</h2>

<ul>

<li>Email: rushikeshparab2157@gmail.com</li>

<li>Phone: (+91) 9284115131</li>

<li>Address: Anand Nagar, Alandi Road, Shastri Chowk, Bhosari Pune - 411039</li>

</ul>

</section>

<section>

<h2>Summary</h2>

<p>Experienced web Operation Analyst with a passion for creating clean and efficient code. Skilled in HTML, CSS, and JavaScript, with a focus on front-end development. Strong communication and teamwork abilities.</p>

</section>

<section>

<h2>Skills</h2>

<ul>

<li>HTML5</li>

<li>CSS3</li>

<li>JavaScript</li>

<li>Responsive Design</li>

<li>Unix</li>

<li>SQL</li>

</ul>

</section>

<section>

<h2>Experience</h2>

<h3>Business Operation Analyst</h3>

<p>Amdocs, Magarpatta</p>

<p>June 2024 - Present</p>

<ul>

<li>Developed responsive and user-friendly web applications using HTML, CSS, and JavaScript.</li>

<li>Collaborated with the design team to implement mockups and prototypes.</li>

<li>Optimized website performance and cross-browser compatibility.</li>

</ul>

</section>

<section>

<h2>Education</h2>

<h3>Bachelor of Computer Application</h3>

<p>Pune University</p>

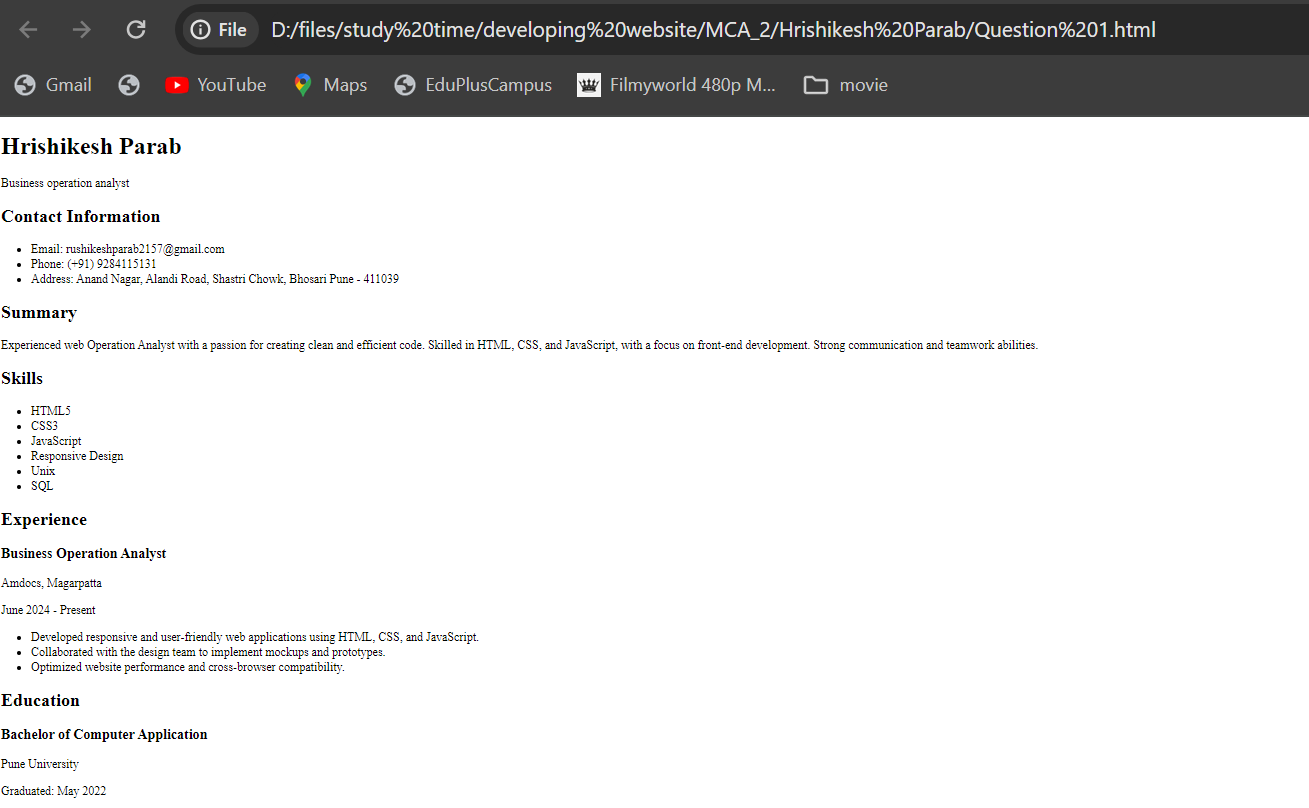
<p>Graduated: May 2022</p>

</section>

</body>

</html>

**OUTPUT**



1. **Program to implement Audio and Video features for your web page.**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Audio and Video Player</title>

</head>

<body>

<h1>Audio Player</h1>

<audio controls>

<source src="audio\_file.mp3" type="audio/mpeg">

Your browser does not support the audio element.

</audio>

<h1>Video Player</h1>

<video controls width="600">

<source src="video\_file.mp4" type="video/mp4">

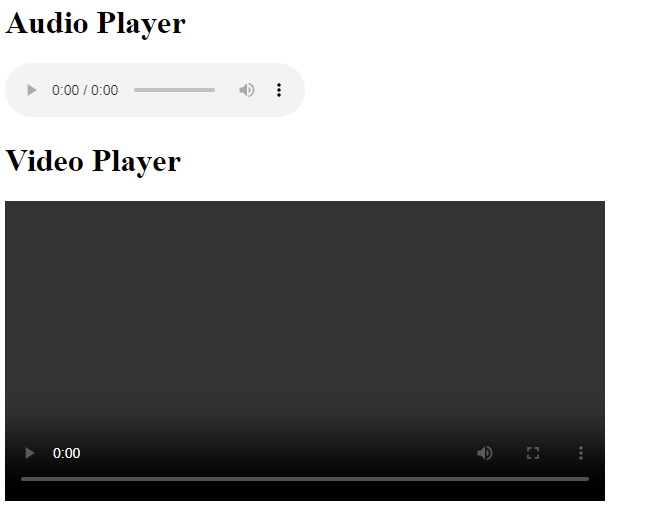
Your browser does not support the video element.

</video>

</body>

</html>

**OUTPUT**:



1. **Write HTML Programs design different geometrical shapes using Canvas and SVG (minimum 5 Shapes).**

<!DOCTYPE html>

<html>

<head>

<title>Geometrical Shapes</title>

<style>

canvas, svg { border: 1px solid black; margin-bottom: 20px; }

</style>

</head>

<body>

<h2>Canvas Shapes</h2>

<canvas id="circleCanvas" width="200" height="200"></canvas>

<canvas id="triangleCanvas" width="200" height="200"></canvas>

<canvas id="pentagonCanvas" width="200" height="200"></canvas>

<h2>SVG Shapes</h2>

<svg width="200" height="200">

<rect width="150" height="100" style="fill:rgb(0,0,255);stroke-width:3;stroke:rgb(0,0,0)" />

</svg>

<svg width="200" height="100">

<ellipse cx="100" cy="50" rx="90" ry="40" style="fill:yellow;stroke:purple;stroke-width:2" />

</svg>

<script>

// Canvas Circle

var circleCanvas = document.getElementById('circleCanvas');

var circleContext = circleCanvas.getContext('2d');

var centerX = circleCanvas.width / 2;

var centerY = circleCanvas.height / 2;

var radius = 70;

circleContext.beginPath();

circleContext.arc(centerX, centerY, radius, 0, 2 \* Math.PI, false);

circleContext.fillStyle = 'green';

circleContext.fill();

circleContext.lineWidth = 5;

circleContext.strokeStyle = '#003300';

circleContext.stroke();

// Canvas Triangle

var triangleCanvas = document.getElementById('triangleCanvas');

var triangleContext = triangleCanvas.getContext('2d');

triangleContext.beginPath();

triangleContext.moveTo(100, 20);

triangleContext.lineTo(20, 150);

triangleContext.lineTo(180, 150);

triangleContext.closePath();

triangleContext.fillStyle = 'red';

triangleContext.fill();

triangleContext.lineWidth = 5;

triangleContext.strokeStyle = '#003300';

triangleContext.stroke();

// Canvas Pentagon

var pentagonCanvas = document.getElementById('pentagonCanvas');

var pentagonContext = pentagonCanvas.getContext('2d');

var sides = 5;

var angle = (Math.PI \* 2) / sides;

pentagonContext.beginPath();

for (var i = 0; i < sides; i++) {

pentagonContext.lineTo(centerX + radius \* Math.cos(i \* angle),

centerY + radius \* Math.sin(i \* angle));

}

pentagonContext.closePath();

pentagonContext.fillStyle = 'orange';

pentagonContext.fill();

pentagonContext.lineWidth = 5;

pentagonContext.strokeStyle = '#003300';

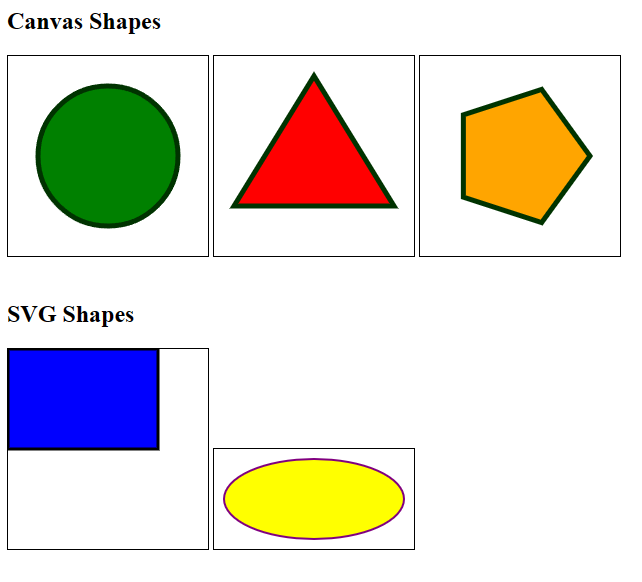
pentagonContext.stroke();

</script>

</body>

</html>

**OUTPUT:**



1. **Program to design form using HTML5 elements, attributes and Semantics tags and apply HTML5 validation to it**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>HTML5 Form Validation</title>

</head>

<body>

<h2>HTML5 Form Validation Example</h2>

<form id="myForm" action="#" method="post" novalidate>

<fieldset>

<legend>Contact Information</legend>

<div>

<label for="name">Name:</label>

<input type="text" id="name" name="name" required minlength="3" maxlength="50">

</div>

<div>

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

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

</div>

<div>

<label for="phone">Phone:</label>

<input type="tel" id="phone" name="phone" pattern="[0-9]{3}-[0-9]{3}-[0-9]{4}" required>

<small>Format: 123-456-7890</small>

</div>

</fieldset>

<fieldset>

<legend>Message</legend>

<div>

<label for="message">Message:</label>

<textarea id="message" name="message" required minlength="10" maxlength="200"></textarea>

</div>

</fieldset>

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

</form>

<script>

// Form validation

var form = document.getElementById('myForm');

form.addEventListener('submit', function(event) {

if (!form.checkValidity()) {

event.preventDefault();

alert('Please fill out all required fields correctly.');

}

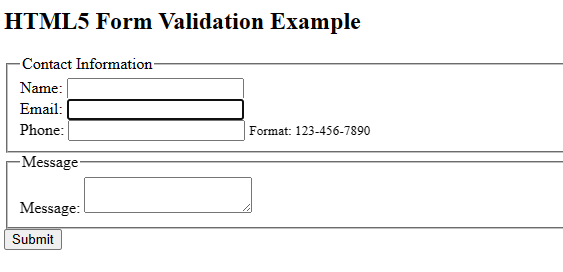
});

</script>

</body>

</html>

**OUTPUT:**



1. **Programs to demonstrate external and internal styles in the web page using font, text,**

**background, borders, opacity, and other CSS 3 properties.**

**External Style Sheet Example:**

/\* styles.css \*/

body {

font-family: Arial, sans-serif;

background-color: #f0f0f0;

color: #333;

padding: 20px;

}

h1 {

text-align: center;

}

p {

font-size: 18px;

line-height: 1.6;

}

.container {

background-color: #fff;

border: 1px solid #ccc;

padding: 20px;

border-radius: 10px;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

}

.button {

display: inline-block;

padding: 10px 20px;

background-color: #007bff;

color: #fff;

text-decoration: none;

border-radius: 5px;

transition: background-color 0.3s;

}

.button:hover {

background-color: #0056b3;

}

.opacity {

opacity: 0.6;

}

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>External Style Sheet Example</title>

<link rel="stylesheet" href="styles.css">

</head>

<body>

<div class="container">

<h1>Welcome to My Website</h1>

<p>This is a demonstration of using external styles in CSS3.</p>

<button class="button">Click Me</button>

</div>

<div class="container opacity">

<h1>Opacity Example</h1>

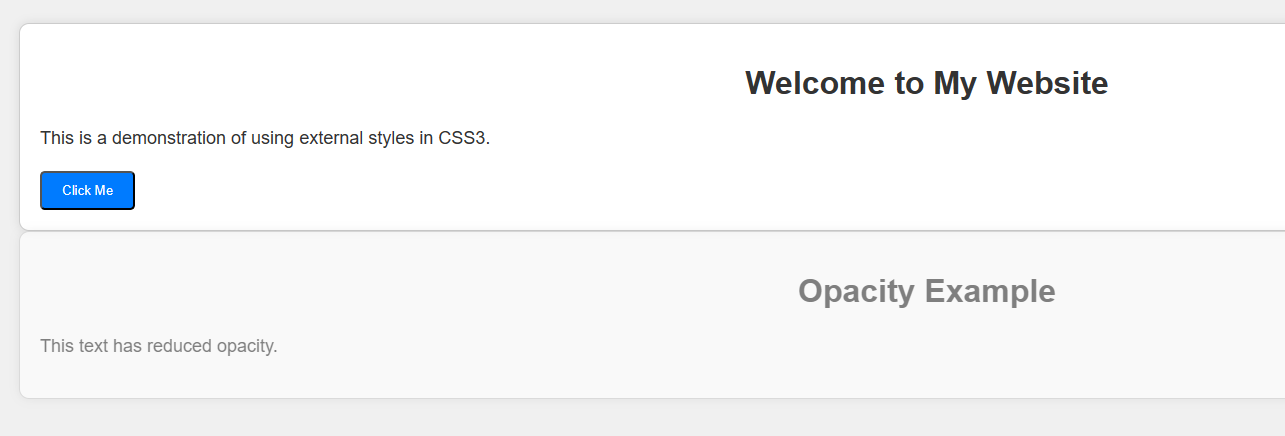
<p>This text has reduced opacity.</p>

</div>

</body>

</html>

**OUTPUT:**



**Internal Style Example:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Internal Style Example</title>

<style>

body {

font-family: Arial, sans-serif;

background-color: #f0f0f0;

color: #333;

padding: 20px;

}

h1 {

text-align: center;

}

p {

font-size: 18px;

line-height: 1.6;

}

.container {

background-color: #fff;

border: 1px solid #ccc;

padding: 20px;

border-radius: 10px;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

}

.button {

display: inline-block;

padding: 10px 20px;

background-color: #007bff;

color: #fff;

text-decoration: none;

border-radius: 5px;

transition: background-color 0.3s;

}

.button:hover {

background-color: #0056b3;

}

.opacity {

opacity: 0.6;

}

</style>

</head>

<body>

<div class="container">

<h1>Welcome to My Website</h1>

<p>This is a demonstration of using internal styles in CSS3.</p>

<button class="button">Click Me</button>

</div>

<div class="container opacity">

<h1>Opacity Example</h1>

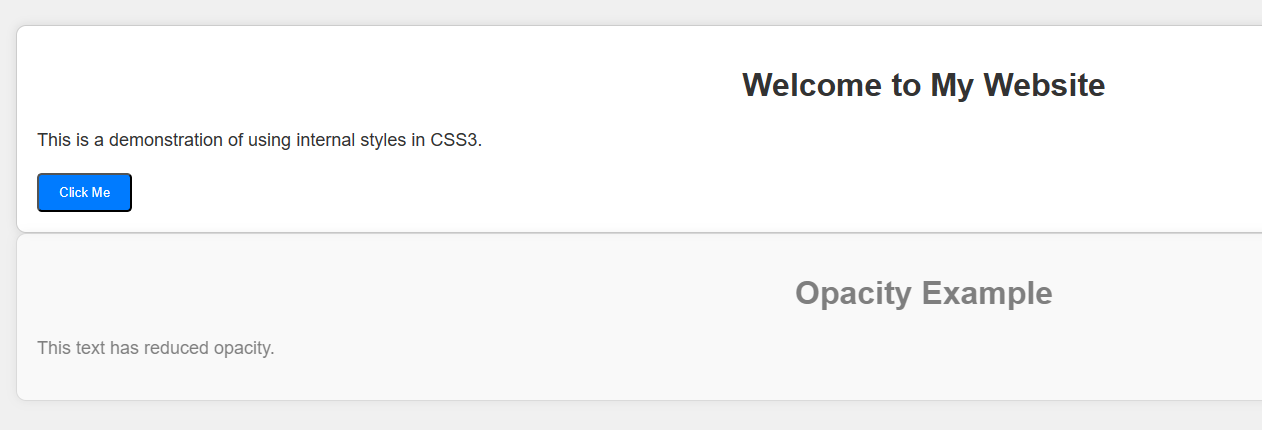
<p>This text has reduced opacity.</p>

</div>

</body>

</html>

**OUTPUT:**



1. **Implement Transformation using Translation, Rotation and Scaling in your web page.**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Transformation Example</title>

<style>

.container {

width: 300px;

height: 300px;

background-color: #f0f0f0;

margin: 50px auto;

position: relative;

}

.shape {

width: 100px;

height: 100px;

background-color: #007bff;

position: absolute;

top: 50%;

left: 50%;

transform-origin: center center;

transition: transform 0.5s;

}

.shape:hover {

transform: translate(50px, 50px) rotate(45deg) scale(1.5);

}

</style>

</head>

<body>

<div class="container">

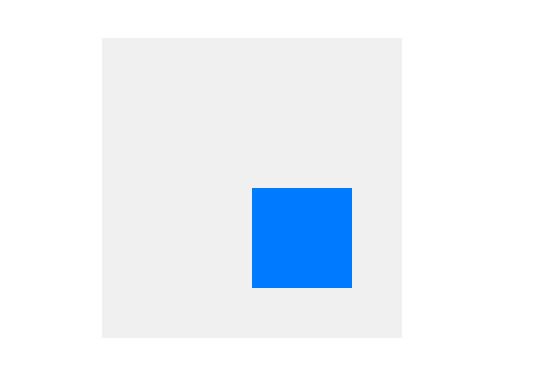
<div class="shape"></div>

</div>

</body>

</html>

**OUTPUT:**



1. **Program to show current date and time using user defined module in AngularJs.**

<!DOCTYPE html>

<html lang="en" ng-app="dateTimeApp">

<head>

<meta charset="UTF-8">

<title>AngularJS Current Date and Time</title>

<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>

<script>

var app = angular.module('dateTimeApp', []);

app.controller('DateTimeController', function($scope, $interval) {

$scope.getCurrentDateTime = function() {

var currentDate = new Date();

$scope.currentDate = currentDate.toLocaleString();

};

// Update the current date and time every second

$interval($scope.getCurrentDateTime, 1000);

// Initial call to get the current date and time

$scope.getCurrentDateTime();

});

</script>

</head>

<body>

<div ng-controller="DateTimeController">

<h1>Current Date and Time</h1>

<p>{{ currentDate }}</p>

</div>

</body>

</html>

**OUTPUT:**



1. **Write calculator program in AngularJs to perform basic arithmetic operations(+, -, \*, /) using angular controller function.**

<!DOCTYPE html>

<html lang="en" ng-app="calculatorApp">

<head>

<meta charset="UTF-8">

<title>AngularJS Calculator</title>

<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>

<script>

angular.module('calculatorApp', [])

.controller('CalculatorController', function($scope) {

$scope.num1 = 0;

$scope.num2 = 0;

$scope.result = 0;

$scope.add = function() {

$scope.result = $scope.num1 + $scope.num2;

};

$scope.subtract = function() {

$scope.result = $scope.num1 - $scope.num2;

};

$scope.multiply = function() {

$scope.result = $scope.num1 \* $scope.num2;

};

$scope.divide = function() {

if ($scope.num2 !== 0) {

$scope.result = $scope.num1 / $scope.num2;

} else {

alert("Cannot divide by zero!");

}

};

});

</script>

</head>

<body>

<div ng-controller="CalculatorController">

<h2>AngularJS Calculator</h2>

<input type="number" ng-model="num1">

<input type="number" ng-model="num2">

<br><br>

<button ng-click="add()">Addition (+)</button>

<button ng-click="subtract()">Subtraction (-)</button>

<button ng-click="multiply()">Multiplication (\*)</button>

<button ng-click="divide()">Division (/)</button>

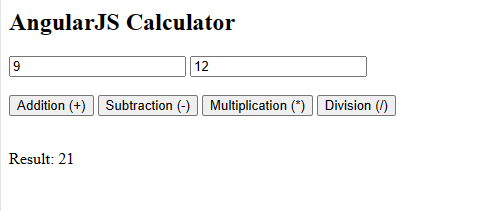
<br><br>

<p>Result: {{ result }}</p>

</div>

</body>

</html>

**OUTPUT:**

1. **Program using NPM which will convert entered string into either case.**

// index.js

const readline = require('readline-sync');

// Function to convert string to uppercase

function toUpperCase(str) {

return str.toUpperCase();

}

// Function to convert string to lowercase

function toLowerCase(str) {

return str.toLowerCase();

}

// Read user input

const inputString = readline.question('Enter a string: ');

// Ask user for choice

const choice = readline.question('Convert to (U)ppercase or (L)owercase? ');

// Convert string based on user choice

let convertedString;

if (choice.toUpperCase() === 'U') {

convertedString = toUpperCase(inputString);

} else if (choice.toUpperCase() === 'L') {

convertedString = toLowerCase(inputString);

} else {

console.log('Invalid choice! Please enter "U" for uppercase or "L" for lowercase.');

process.exit(1);

}

// Output the converted string

console.log(`Converted string: ${convertedString}`);

**OUTPUT:**

node index.js

Enter a string: test

Convert to (U)ppercase or (L)owercase? U

Converted string: TEST

node index.js

Enter a string: HELLO

Convert to (U)ppercase or (L)owercase? L

Converted string: hello

1. **Program to demonstrate the installation of Node, NPM, Angular in web base application and Create angular project which will demonstrate the usage of component directive, structural directive and attribute directives**

// app.component.ts

import { Component, Directive, ElementRef, Renderer2, OnInit } from '@angular/core';

@Directive({

selector: '[appHighlight]'

})

export class HighlightDirective implements OnInit {

constructor(private el: ElementRef, private renderer: Renderer2) { }

ngOnInit() {

this.renderer.setStyle(this.el.nativeElement, 'background-color', 'yellow');

}

}

@Directive({

selector: '[appCustomIf]'

})

export class CustomIfDirective {

constructor(private el: ElementRef, private renderer: Renderer2) {}

set appCustomIf(condition: boolean) {

if (condition) {

this.renderer.setStyle(this.el.nativeElement, 'display', 'block');

} else {

this.renderer.setStyle(this.el.nativeElement, 'display', 'none');

}

}

}

@Component({

selector: 'app-root',

template: `

<h1>Directive Demo</h1>

<app-hello-world></app-hello-world>

<div \*appCustomIf="showMessage" appHighlight>

This message is shown using custom structural directive (appCustomIf) and highlighted using appHighlight!

</div>

`,

styleUrls: ['./app.component.css']

})

export class AppComponent {

showMessage = true;

}

1. **Write Program for Form validation in Angular.**

//test.component.ts

import { Component, OnInit } from '@angular/core';

//import validator and FormBuilder

import { FormGroup, FormControl, Validators, FormBuilder } from '@angular/forms';

@Component({

selector: 'app-test',

templateUrl: './test.component.html',

styleUrls: ['./test.component.css']

})

export class TestComponent implements OnInit {

//Create FormGroup

requiredForm: FormGroup;

constructor(private fb: FormBuilder) {

this.myForm();

}

//Create required field validator for name

myForm() {

this.requiredForm = this.fb.group({

name: ['', Validators.required ]

});

}

ngOnInit()

{

}

}

//test.component.html

<div>

<h2>

Required Field validation

</h2>

<form [formGroup]="requiredForm" novalidate>

<div class="form-group">

<label class="center-block">Name:

<input class="form-control" formControlName="name">

</label>

</div>

<div \*ngIf="requiredForm.controls['name'].invalid && requiredForm.controls['name'].touched" class="alert alert-danger">

<div \*ngIf="requiredForm.controls['name'].errors.required">

Name is required.

</div>

</div>

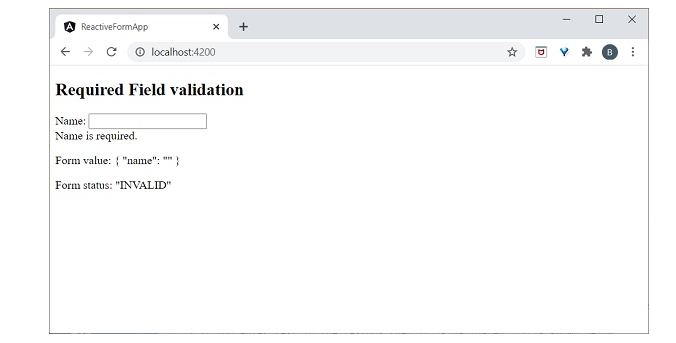
</form>

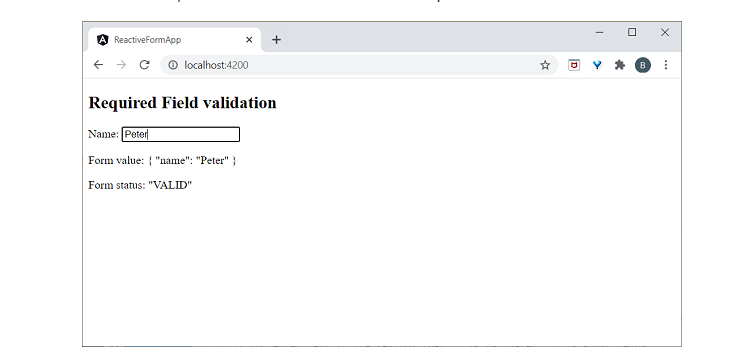
<p>Form value: {{ requiredForm.value | json }}</p>

<p>Form status: {{ requiredForm.status | json }}</p>

</div>

**OUTPUT:**





1. **Write a program to create a calculator using Angular**

// **calculator.component.html**

<!-- calculator.component.html -->

<div>

<input type="text" [(ngModel)]="result" readonly>

</div>

<div>

<button (click)="appendNumber(1)">1</button>

<button (click)="appendNumber(2)">2</button>

<button (click)="appendNumber(3)">3</button>

<button (click)="appendOperator('+')">+</button>

</div>

<div>

<button (click)="appendNumber(4)">4</button>

<button (click)="appendNumber(5)">5</button>

<button (click)="appendNumber(6)">6</button>

<button (click)="appendOperator('-')">-</button>

</div>

<div>

<button (click)="appendNumber(7)">7</button>

<button (click)="appendNumber(8)">8</button>

<button (click)="appendNumber(9)">9</button>

<button (click)="appendOperator('\*')">\*</button>

</div>

<div>

<button (click)="clear()">C</button>

<button (click)="appendNumber(0)">0</button>

<button (click)="calculate()">=</button>

<button (click)="appendOperator('/')">/</button>

</div>

// **calculator.component.ts**

// calculator.component.ts

import { Component } from '@angular/core';

@Component({

selector: 'app-calculator',

templateUrl: './calculator.component.html',

styleUrls: ['./calculator.component.css']

})

export class CalculatorComponent {

result: string = '';

appendNumber(num: number) {

this.result += num.toString();

}

appendOperator(operator: string) {

this.result += operator;

}

calculate() {

try {

this.result = eval(this.result);

} catch (error) {

this.result = 'Error';

}

}

clear() {

this.result = '';

}

}

//app.component.html

<!-- app.component.html -->

<h1>Calculator</h1>

<app-calculator></app-calculator>

1. **Write a program to create a calculator using Node JS.**

// calculator.js

const readline = require('readline');

const rl = readline.createInterface({

input: process.stdin,

output: process.stdout

});

function calculate(expression) {

try {

return eval(expression);

} catch (error) {

return 'Error';

}

}

function startCalculator() {

rl.question('Enter an expression (e.g., 2 + 3): ', (input) => {

const result = calculate(input);

console.log(`Result: ${result}`);

rl.close();

});

}

startCalculator();

**OUTPUT:**

node calculator.js

Enter an expression (e.g., 2 + 3): 2 +4

Result: 6

1. **What is NodeJs Architecture? Explain REPL with suitable example.**

Node.js architecture can be understood as a layered model that illustrates how the various components of a Node.js application interact with each other. It consists of the following main components:

V8 Engine:

Node.js uses the V8 JavaScript engine, developed by Google, which is known for its high performance in executing JavaScript code. V8 compiles JavaScript code into machine code, allowing it to run efficiently.

Libuv:

Libuv is a multi-platform support library in Node.js that provides asynchronous I/O operations and handles events. It abstracts the differences between different operating systems and provides a consistent interface for handling I/O operations, timers, and networking.

Event Loop:

Node.js follows an event-driven, non-blocking I/O model. The event loop is responsible for handling asynchronous operations, such as I/O events, timers, and callbacks. It continuously checks for new events in the event queue and executes the associated callback functions.

Node.js Core Modules:

Node.js provides a set of core modules that offer essential functionality for building applications, such as fs for file system operations, http for creating HTTP servers, util for utility functions, and events for event handling.

Node.js Binding:

Node.js includes bindings to C/C++ libraries, allowing developers to create native add-ons for extending Node.js with functionality that is not available in JavaScript.

User Modules:

Developers can create their own modules or use third-party modules from the npm registry to extend the functionality of their Node.js applications.

Node.js Runtime:

The Node.js runtime provides an environment for executing JavaScript code outside of the browser. It manages memory, handles exceptions, and provides APIs for interacting with the operating system.

The architecture of Node.js enables it to handle a large number of concurrent connections efficiently, making it suitable for building scalable and real-time applications.

REPL (Read-Eval-Print Loop):

REPL is a programming environment that allows you to interactively run JavaScript code and see the results immediately. Node.js comes with a built-in REPL environment, which provides a quick and easy way to experiment with JavaScript code.

Here's an example of using the Node.js REPL:

Open your terminal or command prompt.

Type node and press Enter to start the Node.js REPL.

You will see the > prompt, indicating that the REPL is ready to accept input.

You can now enter JavaScript expressions, statements, or even multi-line code blocks.

Press Enter after each line of code to execute it.

The REPL will evaluate the code, print the result (if any), and display the > prompt again, indicating that it's ready for more input.

Example usage:

$ node

> 2 + 3

5

> let greet = 'Hello, world!'

undefined

> console.log(greet)

Hello, world!

undefined

> for (let i = 1; i <= 5; i++) { console.log(i); }

1

2

3

4

5

undefined

>

1. **Write REPL code to print number from 1 to 50.**

// printNumbers.js

for (let i = 1; i <= 50; i++) {

console.log(i);

}

**OUTPUT:**

node printNumbers.js

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

40

41

42

43

44

45

46

47

48

49

50

1. **Write REPL code to print largest number from three given number.**

// findLargestNumber.js

function findLargestNumber(a, b, c) {

return Math.max(a, b, c);

}

const largestNumber = findLargestNumber(10, 5, 20);

console.log("The largest number is:", largestNumber);

**OUTPUT:**

node findLargestNumber.js

The largest number is: 20

1. Create angular project which has HTML template and handle the click event on click of the

Button

// **button-demo.component.html**

<!-- button-demo.component.html -->

<button (click)="handleClick()">Click me!</button>

// **button-demo.component.ts**

// button-demo.component.ts

import { Component } from '@angular/core';

@Component({

selector: 'app-button-demo',

templateUrl: './button-demo.component.html',

styleUrls: ['./button-demo.component.css']

})

export class ButtonDemoComponent {

handleClick() {

alert('Button clicked!');

}

}

// **app.component.html**

<!-- app.component.html -->

<h1>Click Event Demo</h1>

<app-button-demo></app-button-demo>

1. **Program for basic operations, array and user interface handling.**

// calculator.js

const readline = require('readline');

const rl = readline.createInterface({

input: process.stdin,

output: process.stdout

});

function performOperation(num1, num2, operator) {

switch (operator) {

case '+':

return num1 + num2;

case '-':

return num1 - num2;

case '\*':

return num1 \* num2;

case '/':

if (num2 !== 0) {

return num1 / num2;

} else {

throw new Error('Cannot divide by zero');

}

default:

throw new Error('Invalid operator');

}

}

rl.question('Enter first number: ', (num1) => {

rl.question('Enter second number: ', (num2) => {

rl.question('Enter operator (+, -, \*, /): ', (operator) => {

try {

num1 = parseFloat(num1);

num2 = parseFloat(num2);

const result = performOperation(num1, num2, operator);

console.log(`Result: ${result}`);

} catch (error) {

console.error(error.message);

} finally {

rl.close();

}

});

});

});

1. **Program to demonstrate session management using various techniques.**

// Technique 1: In-Memory Storage

// app.js

const express = require('express');

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

const app = express();

app.use(session({

secret: 'mySecretKey',

resave: false,

saveUninitialized: true

}));

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

const { session } = req;

session.username = 'John';

res.send('Session stored in memory.');

});

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

const { session } = req;

const username = session.username || 'Guest';

res.send(`Hello, ${username}!`);

});

app.listen(3000, () => {

console.log('Server is running on port 3000');

});

// Technique 2: Using Cookies

// app.js

const express = require('express');

const cookieSession = require('cookie-session');

const app = express();

app.use(cookieSession({

name: 'session',

keys: ['mySecretKey'],

maxAge: 24 \* 60 \* 60 \* 1000 // 24 hours

}));

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

req.session.username = 'John';

res.send('Session stored in cookies.');

});

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

const { session } = req;

const username = session.username || 'Guest';

res.send(`Hello, ${username}!`);

});

app.listen(3000, () => {

console.log('Server is running on port 3000');

});

// Technique 3: Using Session Store (Redis)

// app.js

const express = require('express');

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

const RedisStore = require('connect-redis')(session);

const redis = require('redis');

const client = redis.createClient();

const app = express();

app.use(session({

store: new RedisStore({ client }),

secret: 'mySecretKey',

resave: false,

saveUninitialized: true

}));

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

req.session.username = 'John';

res.send('Session stored in Redis.');

});

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

const { session } = req;

const username = session.username || 'Guest';

res.send(`Hello, ${username}!`);

});

app.listen(3000, () => {

console.log('Server is running on port 3000');

});

**19. Design Employee Information form in PHP and perform the CRUD Operations using**

**PHP Script.**

**a) Create mydb database using PHP code**

**b) Create employee\_info table using PHP code**

**c) Write PHP program to design Employee information form which includes fields**

**like (Employee ID, First Name, Lastname, Contact No, Designation, Department,**

**Salary….)**

<?php

// Function to create the database

function createDatabase($servername, $username, $password, $dbname) {

$conn = new mysqli($servername, $username, $password);

if ($conn->connect\_error) {

die("Connection failed: " . $conn->connect\_error);

}

$sql = "CREATE DATABASE $dbname";

if ($conn->query($sql) === TRUE) {

echo "Database created successfully<br>";

} else {

echo "Error creating database: " . $conn->error . "<br>";

}

$conn->close();

}

// Function to create the employee\_info table

function createTable($servername, $username, $password, $dbname) {

$conn = new mysqli($servername, $username, $password, $dbname);

if ($conn->connect\_error) {

die("Connection failed: " . $conn->connect\_error);

}

$sql = "CREATE TABLE employee\_info (

id INT(6) UNSIGNED AUTO\_INCREMENT PRIMARY KEY,

firstname VARCHAR(30) NOT NULL,

lastname VARCHAR(30) NOT NULL,

contact\_no VARCHAR(15),

designation VARCHAR(50),

department VARCHAR(50),

salary DECIMAL(10, 2)

)";

if ($conn->query($sql) === TRUE) {

echo "Table employee\_info created successfully<br>";

} else {

echo "Error creating table: " . $conn->error . "<br>";

}

$conn->close();

}

// Create mydb database

$servername = "localhost";

$username = "username";

$password = "password";

$dbname = "mydb";

createDatabase($servername, $username, $password, $dbname);

// Create employee\_info table

createTable($servername, $username, $password, $dbname);

?>

<!-- HTML form to input employee information -->

<!DOCTYPE html>

<html>

<head>

<title>Employee Information Form</title>

</head>

<body>

<h2>Employee Information Form</h2>

<form action="insert\_employee.php" method="post">

<label for="firstname">First Name:</label><br>

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

<label for="lastname">Last Name:</label><br>

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

<label for="contact\_no">Contact No:</label><br>

<input type="text" id="contact\_no" name="contact\_no"><br>

<label for="designation">Designation:</label><br>

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

<label for="department">Department:</label><br>

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

<label for="salary">Salary:</label><br>

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

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

</form>

</body>

</html>

1. **Design Students Registration form (Signup form), Student Login form (signIn form) in PHP and implement session, cookies concept in it.**

**a) Registration form consists (Student id, Student first & Surname, mobile No, Address…)**

**b) Student Login form (Student Username, Password)**

// index.php (Registration Form)

<!DOCTYPE html>

<html>

<head>

<title>Student Registration</title>

</head>

<body>

<h2>Student Registration Form</h2>

<form action="register.php" method="post">

<label for="student\_id">Student ID:</label><br>

<input type="text" id="student\_id" name="student\_id"><br>

<label for="first\_name">First Name:</label><br>

<input type="text" id="first\_name" name="first\_name"><br>

<label for="last\_name">Last Name:</label><br>

<input type="text" id="last\_name" name="last\_name"><br>

<label for="mobile\_no">Mobile No:</label><br>

<input type="text" id="mobile\_no" name="mobile\_no"><br>

<label for="address">Address:</label><br>

<textarea id="address" name="address"></textarea><br><br>

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

</form>

</body>

</html>

// register.php (PHP script to handle registration)

<?php

session\_start();

// Check if the form is submitted

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

// Retrieve form data

$student\_id = $\_POST["student\_id"];

$first\_name = $\_POST["first\_name"];

$last\_name = $\_POST["last\_name"];

$mobile\_no = $\_POST["mobile\_no"];

$address = $\_POST["address"];

// Save data to session (for demonstration purposes)

$\_SESSION["student\_id"] = $student\_id;

$\_SESSION["first\_name"] = $first\_name;

$\_SESSION["last\_name"] = $last\_name;

$\_SESSION["mobile\_no"] = $mobile\_no;

$\_SESSION["address"] = $address;

// Set a cookie for the student ID (for demonstration purposes)

setcookie("student\_id", $student\_id, time() + (86400 \* 30), "/"); // 86400 = 1 day

}

header("Location: index.php"); // Redirect back to registration form

?>

// login.php (Login Form)

<!DOCTYPE html>

<html>

<head>

<title>Student Login</title>

</head>

<body>

<h2>Student Login Form</h2>

<form action="authenticate.php" method="post">

<label for="username">Username (Student ID):</label><br>

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

<label for="password">Password:</label><br>

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

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

</form>

</body>

</html>

// authenticate.php (PHP script to handle authentication)

<?php

session\_start();

// Dummy login credentials for demonstration purposes

$valid\_username = "123456";

$valid\_password = "password";

// Check if the form is submitted

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

// Retrieve form data

$username = $\_POST["username"];

$password = $\_POST["password"];

// Check if username and password are valid

if ($username === $valid\_username && $password === $valid\_password) {

$\_SESSION["authenticated"] = true;

header("Location: profile.php"); // Redirect to profile page

} else {

echo "Invalid username or password";

}

}

?>

// profile.php (Profile Page)

<?php

session\_start();

// Check if the user is authenticated

if (!isset($\_SESSION["authenticated"]) || $\_SESSION["authenticated"] !== true) {

header("Location: login.php"); // Redirect to login page if not authenticated

exit;

}

// Display user information (retrieved from session)

echo "<h2>Welcome, " . $\_SESSION["first\_name"] . " " . $\_SESSION["last\_name"] . "</h2>";

echo "<p>Student ID: " . $\_SESSION["student\_id"] . "</p>";

echo "<p>Mobile No: " . $\_SESSION["mobile\_no"] . "</p>";

echo "<p>Address: " . $\_SESSION["address"] . "</p>";

// Display student ID retrieved from cookie (for demonstration purposes)

echo "<p>Student ID (from cookie): " . $\_COOKIE["student\_id"] . "</p>";

?>

// logout.php (Sign-out Form)

<?php

session\_start();

// Unset all session variables

$\_SESSION = array();

// Destroy the session

session\_destroy();

// Redirect to the login page

header("Location: login.php");

exit;

?>

// **logout.php**

<!DOCTYPE html>

<html>

<head>

<title>Sign Out</title>

</head>

<body>

<h2>Sign Out</h2>

<p>Are you sure you want to sign out?</p>

<form action="logout.php" method="post">

<input type="submit" value="Sign Out">

</form>

</body>

</html>

1. **Write event drive program to perform operations like select, insert and delete on Employee information using PHP & MySQL**

<?php

// MySQL database configuration

$servername = "localhost";

$username = "username";

$password = "password";

$dbname = "mydb";

// Create connection

$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection

if ($conn->connect\_error) {

die("Connection failed: " . $conn->connect\_error);

}

// SELECT operation

$sql\_select = "SELECT \* FROM employee\_info";

$result\_select = $conn->query($sql\_select);

if ($result\_select->num\_rows > 0) {

echo "<h2>Employee Information</h2>";

echo "<table border='1'>";

echo "<tr><th>Employee ID</th><th>First Name</th><th>Last Name</th><th>Contact No</th><th>Designation</th><th>Department</th><th>Salary</th></tr>";

while($row = $result\_select->fetch\_assoc()) {

echo "<tr>";

echo "<td>" . $row["id"] . "</td>";

echo "<td>" . $row["firstname"] . "</td>";

echo "<td>" . $row["lastname"] . "</td>";

echo "<td>" . $row["contact\_no"] . "</td>";

echo "<td>" . $row["designation"] . "</td>";

echo "<td>" . $row["department"] . "</td>";

echo "<td>" . $row["salary"] . "</td>";

echo "</tr>";

}

echo "</table>";

} else {

echo "0 results";

}

// INSERT operation

$sql\_insert = "INSERT INTO employee\_info (firstname, lastname, contact\_no, designation, department, salary) VALUES ('John', 'Doe', '1234567890', 'Manager', 'HR', 50000)";

if ($conn->query($sql\_insert) === TRUE) {

echo "<p>New record inserted successfully</p>";

} else {

echo "Error: " . $sql\_insert . "<br>" . $conn->error;

}

// DELETE operation

$sql\_delete = "DELETE FROM employee\_info WHERE firstname='John' AND lastname='Doe'";

if ($conn->query($sql\_delete) === TRUE) {

echo "<p>Record deleted successfully</p>";

} else {

echo "Error: " . $sql\_delete . "<br>" . $conn->error;

}

// Close connection

$conn->close();

?>

1. **Explain Selectors and Pseudo Classes of CSS3 with example. Write HTML code with CSS3 classes to design Indian Flag.(Apply Transition & Animation)**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Indian Flag</title>

<style>

.flag {

width: 450px;

height: 300px;

position: relative;

background-color: #fff;

overflow: hidden;

}

.saffron {

background-color: #FF9933;

height: 100px;

width: 100%;

}

.white {

background-color: #fff;

height: 100px;

width: 100%;

}

.green {

background-color: #138808;

height: 100px;

width: 100%;

}

.ashoka-chakra {

position: absolute;

left: 50%;

top: 50%;

transform: translate(-50%, -50%);

width: 100px;

height: 100px;

border-radius: 50%;

background-color: #000080;

animation: rotateChakra 10s linear infinite;

}

@keyframes rotateChakra {

from {

transform: translate(-50%, -50%) rotate(0deg);

}

to {

transform: translate(-50%, -50%) rotate(360deg);

}

}

</style>

</head>

<body>

<div class="flag">

<div class="saffron"></div>

<div class="white"></div>

<div class="green"></div>

<div class="ashoka-chakra"></div>

</div>

</body>

</html>

**OUTPUT:**

