Bharati Vidyapeeth's

Institute of Management & Information Technology

C.B.D. Belapur, Navi Mumbai 400614

Vision:
Providing high quality, innovative and value-based education in information technology to build competent professionals.
Mission
M1. Technical Skills:-To provide solid technical foundation theoretically as well as practically capable of providing quality services toindustry.
M2. Development: -Department caters to the needs of students through comprehensive educational programs and promotes lifelong learning in the field of computer Applications.
M3. Ethical leadership:-Department develops ethical leadership insight in the students to succeed in industry, government and academia.

CERTIFICATE

This is to certify that the journal is the work of Mr Roshan Pawar Roll No. 41 of MCA (Sem-1 Div:A) for the academic year 2022 – 2023

Subject Code: MCAL14

Subject Name: Web Technology Lab

Subject-in-charge

Principal

Date:_______________

External Examiner

Date:_______________

Bharati Vidyapeeth's Institute of Management & Information Technology

MCA Semester I AY 2022-23

MCAL14: Web Technologies

INDEX

Name : MCA Sem: I Div :

Sr No.	Date	Topic	Sign
1		**Nodejs Module**	
1.1		Create an application to demonstrate Node.js Modules.	
2		**Events**	
2.1		Create an application to demonstrate various Node.js Events.	
2.2		Implement all Methods of Event Emitter class.	
2.3		Create an application to demonstrate Node.js Functions	
3		**File System and HTTP Server**	
3.1		Create an HTTP Server and perform operations on it.	
3.2		Using File Handling demonstrate all basic file operations (Create, write, read, delete)	
4		**MySQL database connectivity.**	
4.1		Create an application to establish a connection with the MySQL database and perform basic database operations on it.	
5		**Angular JS**	
5.1		Write a program in AngularJs of expression for operators and variables .	
5.2		Write a program in AngularJs of expression contains any two data type.	
5.3		Write a program in AngularJs of expression for arithmetic operators which will produce the result based on the type of operands.	
5.4		Write a program in AngularJs which demonstrates handling click event of a button	
5.5		Write a program in AngularJs for scope object where controller available to the HTML elements and its child elements	
5.6		Write a program in AngularJs demonstrates multiple controllers.	
5.7		Write a program in AngularJs to demonstratesng-init directive for string, number, array, and object.	
5.8		Write a program in AngularJs to demonstrates ng-if, ng-readonly, and ng-disabled directives.	
5.9		Write a program in AngularJs for currency filter to person salary.	
6.0		Write a program in AngularJs demonstrates Date filter.	
6.1		Write a program in AngularJs upper case and lowercase filter	
6.2		Write a program in AngularJs to demonstrates mouse even	

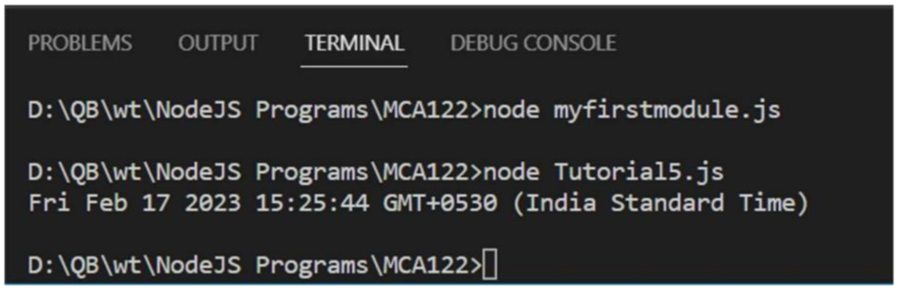
1. Node JS Module:

1.1 Create an application to demonstrate Node.js Modules.

```
//Creating own modules
exports.myDateTime=function()
{
    return Date();
}
```

```
//Implementing own modules
var dt=require('./myfirstmodule');
console.log(dt.myDateTime());
```

Output:



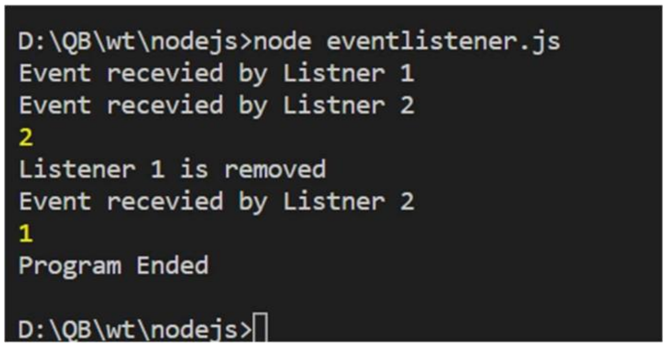
```
D:\QB\wt\NodeJS Programs\MCA122>node myfirstmodule.js

D:\QB\wt\NodeJS Programs\MCA122>node Tutorial5.js
Fri Feb 17 2023 15:25:44 GMT+0530 (India Standard Time)

D:\QB\wt\NodeJS Programs\MCA122>
```

2. Events

2.1 Create an application to demonstrate various Node.js Events.

```
// step 1 importing event
const events = require("events");

// step 2 creating an Event emitter object
const eventEmitter = new events.EventEmitter();

//write a function of event 1
function listner1() {
    console.log("Event recevied by Listner 1");
}

//write a function of event 2
function listner2() {
    console.log("Event recevied by Listner 2");
}

// step 3 adding listener through addlistener or on
eventEmitter.addListener("write", listner1);
eventEmitter.on("write", listner2);

// step 4 emiting event
eventEmitter.emit("write");
console.log(eventEmitter.listenerCount("write"));

// step 5 removing listener
eventEmitter.removeListener("write", listner1);
console.log("Listener 1 is removed");
eventEmitter.emit("write");

console.log(eventEmitter.listenerCount("write"));

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

Output:



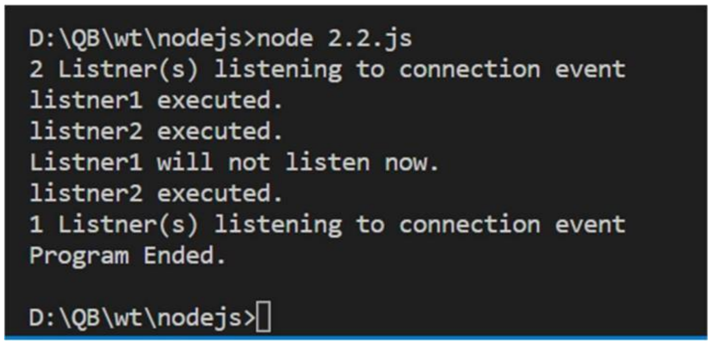
```
D:\QB\wt\nodejs>node eventlistener.js
Event recevied by Listner 1
Event recevied by Listner 2
2
Listener 1 is removed
Event recevied by Listner 2
1
Program Ended

D:\QB\wt\nodejs>
```

2.2 Implement all Methods of Event Emitter class.

```
var events = require('events');
var eventEmitter = new events.EventEmitter();

// listener #1
var listner1 = function listner1() {
   console.log('listner1 executed.');
}

// listener #2
var listner2 = function listner2() {
   console.log('listner2 executed.');
}

// Bind the connection event with the listner1 function
eventEmitter.addListener('connection', listner1);

// Bind the connection event with the listner2 function
eventEmitter.on('connection', listner2);

var eventListeners = require('events').EventEmitter.listenerCount
   (eventEmitter,'connection');
console.log(eventListeners + " Listner(s) listening to connection event");

// Fire the connection event
```

```
eventEmitter.emit('connection');

// Remove the binding of listner1 function
eventEmitter.removeListener('connection', listner1);
console.log("Listner1 will not listen now.");

// Fire the connection event
eventEmitter.emit('connection');

eventListeners =
require('events').EventEmitter.listenerCount(eventEmitter,'connection');
console.log(eventListeners + " Listner(s) listening to connection event");

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

Output:

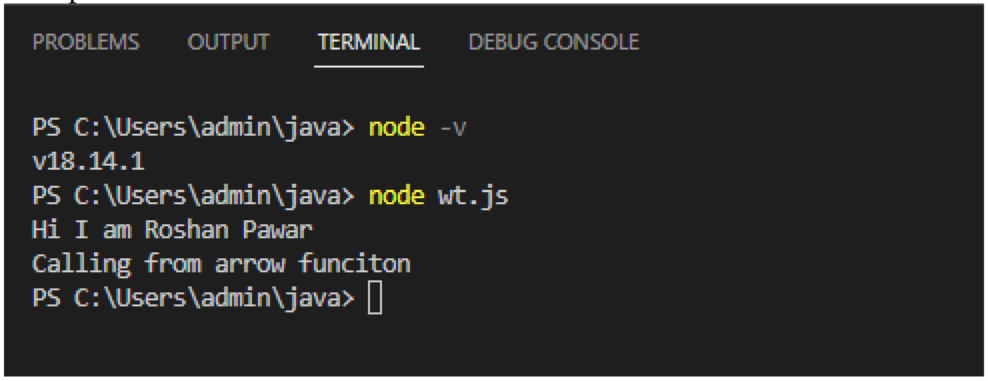


```
D:\QB\wt\nodejs>node 2.2.js
2 Listner(s) listening to connection event
listner1 executed.
listner2 executed.
Listner1 will not listen now.
listner2 executed.
1 Listner(s) listening to connection event
Program Ended.

D:\QB\wt\nodejs>
```

2.3 Create an application to demonstrate Node.js Functions

```
/* What is Call back function
A callback is a function passed as an argument to another function.

*/
//callback function - Anonymous Function
const message=function(){
    console.log("Hi I am Roshan Pawar");

}
setTimeout(message,3000);
//callback back as an Arrow function

setTimeout(()=>{
    console.log("Calling from arrow funciton");
},3000);
```

Output:



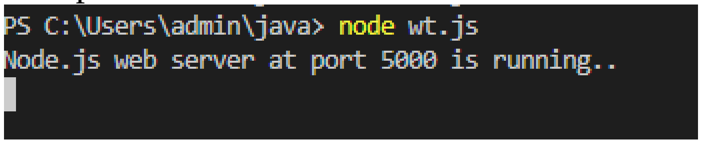
```
PS C:\Users\admin\java> node -v
v18.14.1
PS C:\Users\admin\java> node wt.js
Hi I am Roshan Pawar
Calling from arrow funciton
PS C:\Users\admin\java> 
```

3. File System and HTTP Server

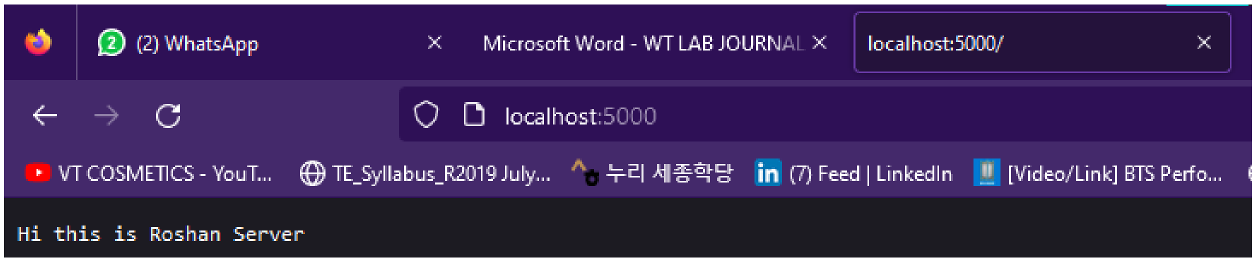
3.1 Create an HTTP Server and perform operations on it.

```
// understand http request module
var http = require('http'); // 1 - Import Node.js core module
var server = http.createServer(function (req, res) {   // 2 - creating server

    //handle incomming requests here..
    res.write("Hi this is Roshan Server");
    res.end();
});
server.listen(5000); //3 - listen for any incoming requests
console.log('Node.js web server at port 5000 is running..')
```

Output:



```
PS C:\Users\admin\java> node wt.js
Node.js web server at port 5000 is running..
```



```
Hi this is Roshan Server
```

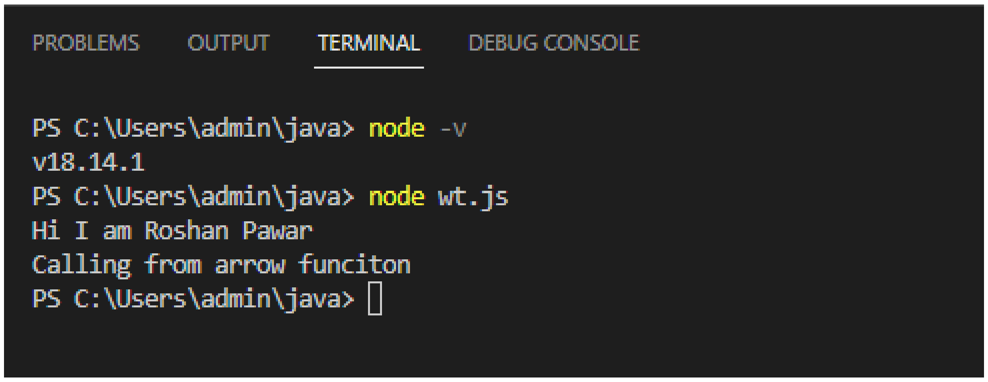
3.2 Using File Handling demonstrate all basic file operations (Create, write, read, delete).

Creating text file.
Input.txt

Hi I am Roshan Pawar!
I am from BVIMIT.
I am pursuing MCA.

```
//node js file system
//reading file

var fs = require('fs');

fs.readFile('input.txt', function (err, data) {
                    if (err) throw err;
    console.log(data.toString());
});
```

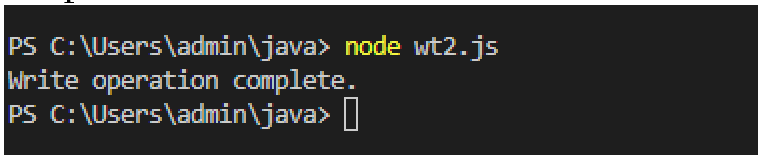
Output:



```
PS C:\Users\admin\java> node -v
v18.14.1
PS C:\Users\admin\java> node wt.js
Hi I am Roshan Pawar
Calling from arrow funciton
PS C:\Users\admin\java> 
```

```
//Writing file
var fs = require('fs');

fs.writeFile('test.txt', 'Roshan Pawar', function (err) {if
        (err)
            console.log(err);
        else
        console.log('Write operation complete.');
});
```

Output:

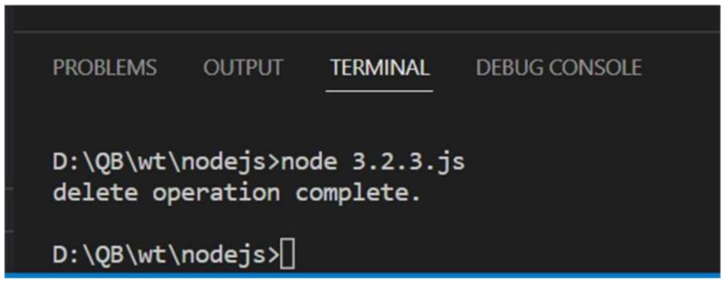


```
PS C:\Users\admin\java> node wt2.js
Write operation complete.
PS C:\Users\admin\java> 
```

New file is created by writing file.

```
//delete the file
var fs = require('fs');

fs.unlink('test.txt', function () {

    console.log('delete operation complete.');

});
```

Output:



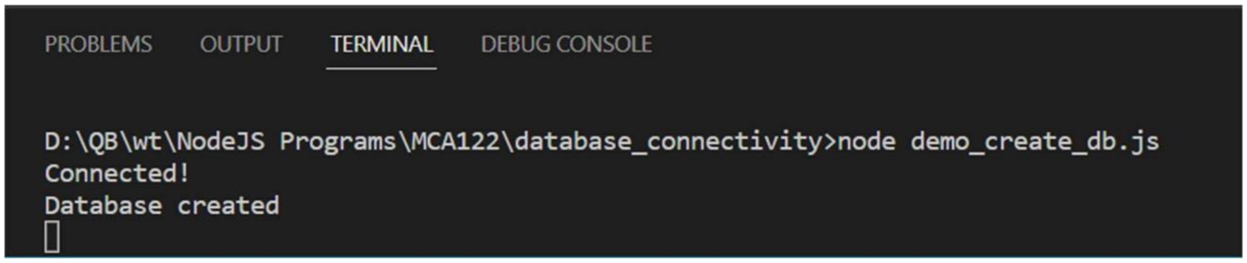
```
D:\QB\wt\nodejs>node 3.2.3.js
delete operation complete.

D:\QB\wt\nodejs>
```

4. MySQL database connectivity.

4.1 Create an application to establish a connection with the MySQL database and perform basic database operations on it.

```
//creating database
var mysql = require('mysql');

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

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

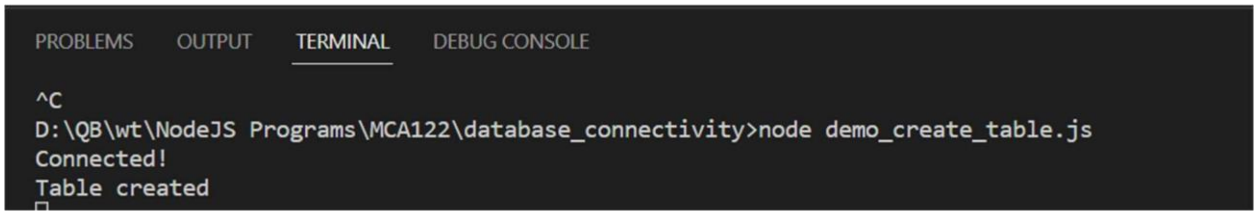
Output:



```
D:\QB\wt\NodeJS Programs\MCA122\database_connectivity>node demo_create_db.js
Connected!
Database created
```

Create Table

```
var mysql = require('mysql');

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

con.connect(function(err) {
  if (err) throw err;
  console.log("Connected!");
  var sql = "CREATE TABLE customers (id INT AUTO_INCREMENT PRIMARY KEY, name
VARCHAR(255), address VARCHAR(255))";
  con.query(sql, function (err, result) {
    if (err) throw err;
    console.log("Table created");
  });
});
```

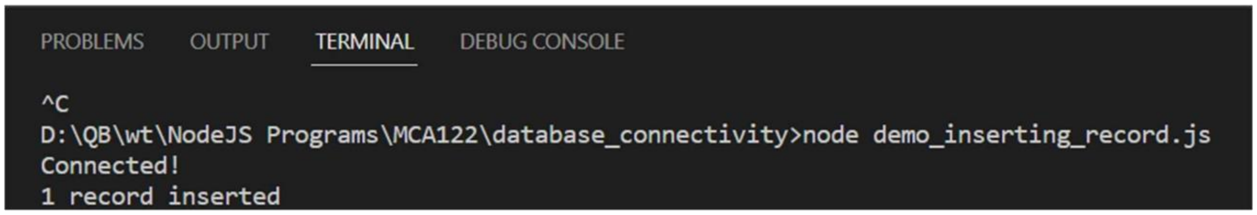


PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
^C
D:\QB\wt\NodeJS Programs\MCA122\database_connectivity>node demo_create_table.js
Connected!
Table created
```

Insertion in database

```
con.connect(function(err) {
  if (err) throw err;
  console.log("Connected!");
  var sql = "INSERT INTO customers (name, address) VALUES ('Company Inc',
'Highway 37')";
  con.query(sql, function (err, result) {
    if (err) throw err;
    console.log("1 record inserted");
  });
});
```

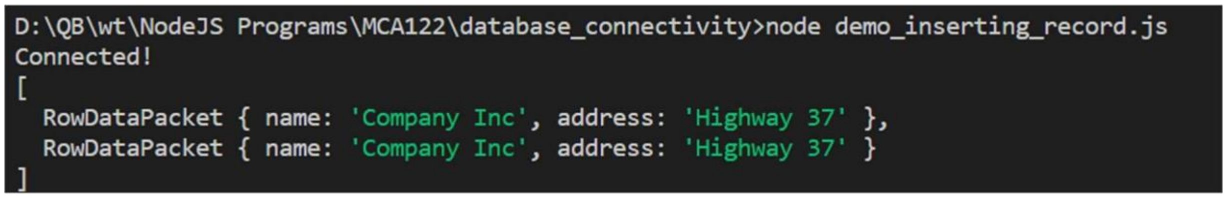


PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
^C
D:\QB\wt\NodeJS Programs\MCA122\database_connectivity>node demo_inserting_record.js
Connected!
1 record inserted
```

Reading from Data base

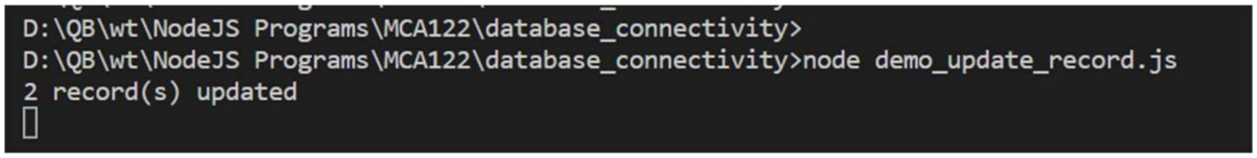
```
con.connect(function(err) {
  if (err) throw err;
  con.query("SELECT * FROM customers", function (err, result, fields) {
    if (err) throw err;
    console.log(result);
  });
});
```



```
D:\QB\wt\NodeJS Programs\MCA122\database_connectivity>node demo_inserting_record.js
Connected!
[
  RowDataPacket { name: 'Company Inc', address: 'Highway 37' },
  RowDataPacket { name: 'Company Inc', address: 'Highway 37' }
]
```

Updating Database

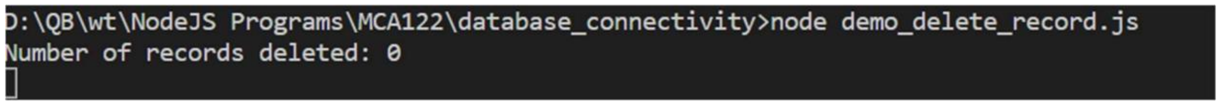
```
con.connect(function(err) {
  if (err) throw err;
  var sql = "UPDATE customers SET address = 'Canyon 123' WHERE address = 'Highway
37'";
  con.query(sql, function (err, result) {
    if (err) throw err;
    console.log(result.affectedRows + " record(s) updated");
  });
});
```



```
D:\QB\wt\NodeJS Programs\MCA122\database_connectivity>
D:\QB\wt\NodeJS Programs\MCA122\database_connectivity>node demo_update_record.js
2 record(s) updated
```

Deleting Records

```
con.connect(function(err) {
  if (err) throw err;
  var sql = "DELETE FROM customers WHERE address = 'Highway 37'";
  con.query(sql, function (err, result) {
    if (err) throw err;
    console.log("Number of records deleted: " + result.affectedRows);
  });
});
```



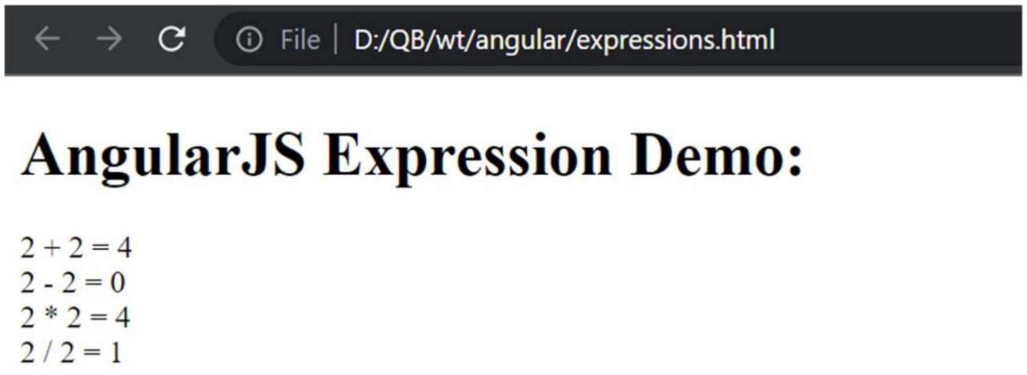
```
D:\QB\wt\NodeJS Programs\MCA122\database_connectivity>node demo_delete_record.js
Number of records deleted: 0
```

5. Angular JS:

5.1 Write a program in Angular JS of expression for operators and variables .

```
<!DOCTYPE html>
<html >
<head>
    <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/"></script>
</head>
<body >
    <h1>AngularJS Expression Demo:</h1>
    <div ng-app>
        2 + 2 = {{2 + 2}} <br/>
        2 - 2 = {{2 - 2}} <br />
        2 * 2 = {{2 * 2}} <br />
        2 / 2 = {{2 / 2}}
    </div>
</body>
</html>
```

Output:

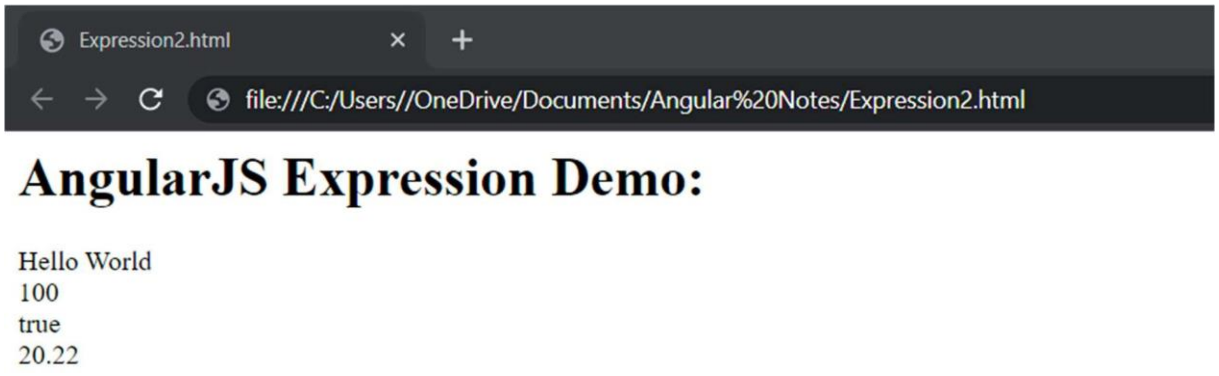


```
AngularJS Expression Demo:

2 + 2 = 4
2 - 2 = 0
2 * 2 = 4
2 / 2 = 1
```

5.2 Write a program in Angular JS of expression contains any two data type.

```
<html >
<head>
 <script                                                                           src=
"https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
</head>
<body >
    <h1>AngularJS Expression Demo:</h1>
    <div ng-app>
        {{"Hello World"}}<br />
        {{100}}<br />
        {{true}}<br />
        {{10.2}}
    </div>
</body>
</html>
```

Output:



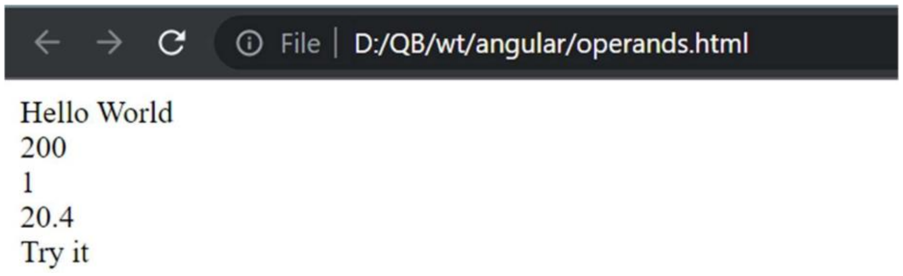
Expression2.html
file:///C:/Users//OneDrive/Documents/Angular%20Notes/Expression2.html

```
AngularJS Expression Demo:

Hello World
100
true
20.22
```

5.3 Write a program in Angular JS of expression for arithmetic operators which will produce the result based on the type of operands.

```html
<!DOCTYPE html>
<html >
<head>
    <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
</head>
<body >
    <div ng-app>
        {{"Hello" + " World"}}<br />
        {{100 + 100 }}<br />
        {{true + false}}<br />
        {{10.2 + 10.2}}<br />
    </div>
</body>
</html>
```

Output:

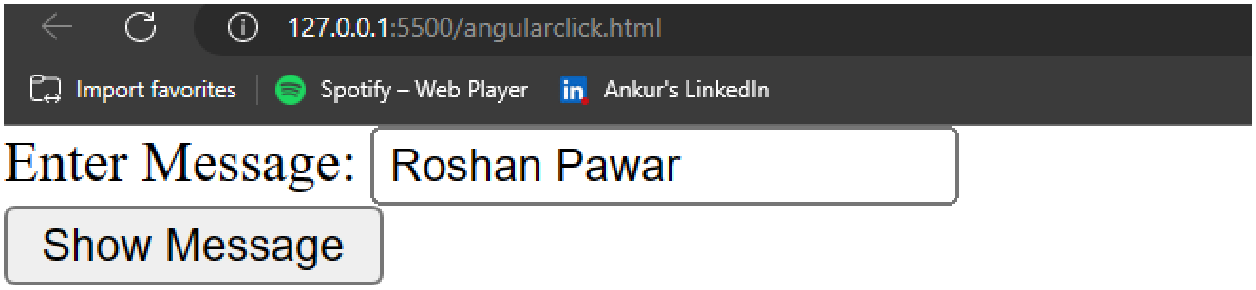


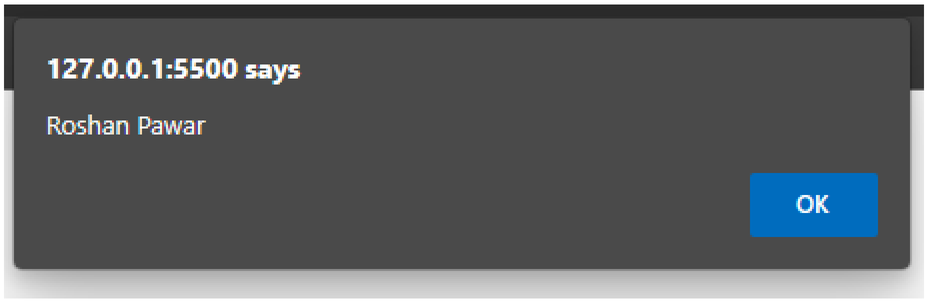
```
Hello World
200
1
20.4
Try it
```

5.4 Write a program in AngularJs which demonstrates handling click event of a button.

```
<!DOCTYPE html>
<html>
<head>
    <title>AngualrJS Controller</title>
    <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
</head>
<body ng-app="myNgApp">
    <div ng-controller="myController">
        Enter Message: <input type="text" ng-model="message" /> <br />
        <button ng-click="showMsg(message)">Show Message</button>
    </div>
    <script>
        var ngApp = angular.module('myNgApp', []);

        ngApp.controller('myController', function ($scope) {
            $scope.message = "Roshan Pawar";

            $scope.showMsg = function (msg) {
                alert(msg);
            };
        });
    </script>
</body>
</html>
```

Output:



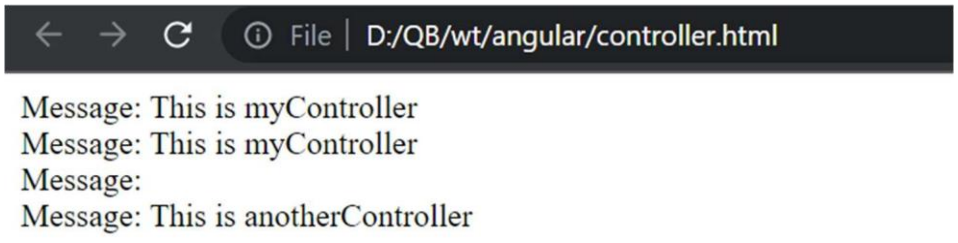

127.0.0.1:5500 says

Roshan Pawar

5.5 Write a program in Angular JS for scope object where controller available to the HTML elements and its child elements.

```
<!DOCTYPE html>
<html>
<head>
    <title>AngualrJS Controller</title>
    <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
</head>
<body ng-app="myNgApp">
    <div id="div1" ng-controller="myController">
        Message: {{message}} <br />
        <div id="div2">
            Message: {{message}}
        </div>
    </div>
    <div  id="div3">
        Message: {{message}}
    </div>
    <div id="div4" ng-controller="anotherController">
        Message: {{message}}
    </div>
    <script>
        var ngApp = angular.module('myNgApp', []);

        ngApp.controller('myController', function ($scope) {
            $scope.message = "This is myController";
        });

        ngApp.controller('anotherController', function ($scope) {
            $scope.message = "This is anotherController";
        });
    </script>
</body>
</html>
```

Output:

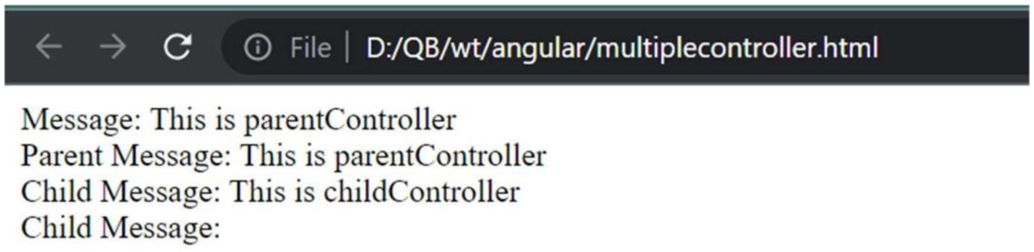


```
Message: This is myController
Message: This is myController
Message:
Message: This is anotherController
```

5.6 Write a program in AngularJs demonstrates multiple controllers.

```
<!DOCTYPE html>
<html>
<head>
    <title>AngualrJS Controller</title>
    <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
</head>
<body ng-app="myNgApp">
    <div ng-controller="parentController">
        Message: {{message1}}
        <div ng-controller="childController">
            Parent Message: {{message1}}  </br>
            Child Message: {{message2}}
        </div>
        Child Message: {{message2}}
    </div>
    <script>
         var ngApp = angular.module('myNgApp', []);

        ngApp.controller('parentController', function ($scope) {
            $scope.message1 = "This is parentController";
        });

        ngApp.controller('childController', function ($scope) {
            $scope.message2 = "This is childController";
        });
    </script>
</body>
</html>
```

Output:

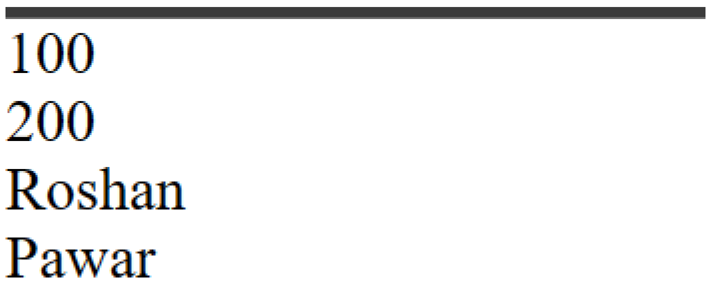


```
Message: This is parentController
Parent Message: This is parentController
Child Message: This is childController
Child Message:
```

5.7 Write a program in AngularJs to demonstrates ng-init directive for string, number, array, and object.

```
<!DOCTYPE html>
<html >
<head>
    <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
</head>
<body >
    <div ng-app ng-init="greet='Hello World!'; amount= 100; myArr = [100, 200]; person
= { firstName:'Roshan', lastName :'Pawar'}">
        {{amount}}      <br />
        {{myArr[1]}}    <br />
        {{person.firstName}}<br />
         {{person.lastName}}
    </div>
</body>
</html>
```

Output:



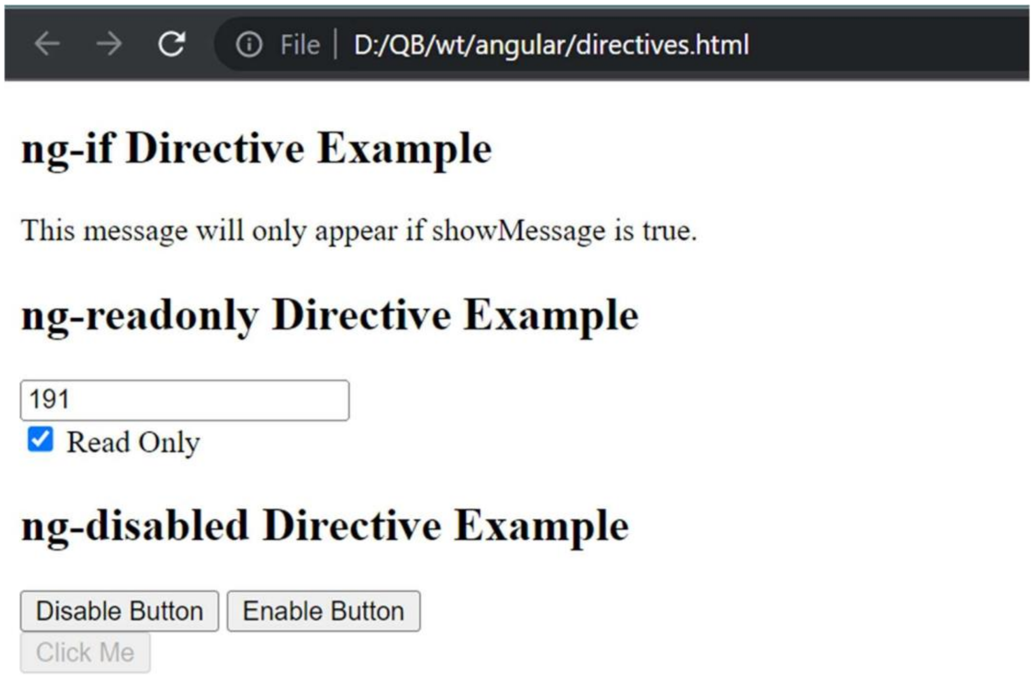
```
100
200
Roshan
Pawar
```

5.8 Write a program in AngularJs to demonstrates ng-if, ng-readonly, and ng- disabled directives.

```
<!DOCTYPE html>
<html >
<head>
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
</head>
<body >
  <div ng-app="myApp" ng-controller="myCtrl">
    <h2>ng-if Directive Example</h2>
    <div ng-if="showMessage">
      <p>This message will only appear if showMessage is true.</p>
    </div>

    <h2>ng-readonly Directive Example</h2>
    <input type="text" ng-model="readOnlyText" ng-readonly="isReadOnly">
    <br>
<label>
      <input type="checkbox" ng-model="isReadOnly"> Read Only
    </label>

    <h2>ng-disabled Directive Example</h2>
    <button ng-click="disableButton()">Disable Button</button>
    <button ng-click="enableButton()">Enable Button</button>
    <br>
    <button ng-disabled="isDisabled">Click Me</button>
   </div>
   <script>
    var app = angular.module('myApp', []);
      app.controller('myCtrl', function($scope) {
        // ng-if example
        $scope.showMessage = true;
```

```
            // ng-readonly example
            $scope.readOnlyText = "This text is editable by default.";
            $scope.isReadOnly = false;

            // ng-disabled example
            $scope.isDisabled = false;
            $scope.disableButton = function() {
              $scope.isDisabled = true;
            }
            $scope.enableButton = function() {
              $scope.isDisabled = false;
            }
            });

        </script>
          </body>
          </html>
```

Output:



D:/QB/wt/angular/directives.html

ng-if Directive Example

This message will only appear if showMessage is true.

ng-readonly Directive Example

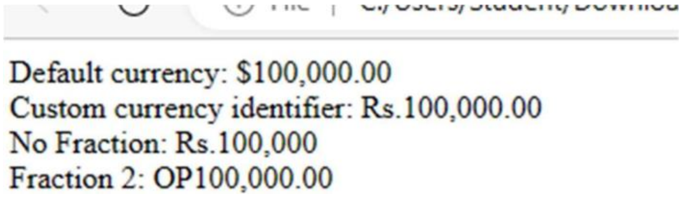
191
☑ Read Only

ng-disabled Directive Example

5.9 Write a program in AngularJs for currency filter to person salary.

```
<!DOCTYPE html>
<html >
<head>
   <script
src="https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.8.3/angular.min.js"></script>
</head>
<body ng-app="myApp">
   <div ng-controller="myController">
      Default currency: {{person.salary | currency}} <br />
      Custom currency identifier: {{person.salary | currency:'Rs.'}} <br />
      No Fraction: {{person.salary | currency:'Rs.':0}} <br />
      Fraction 2: <span ng-bind="person.salary| currency:'GBP':2"></span>
   </div>
   <script>
      var myApp = angular.module('myApp', []);

      myApp.controller("myController", function ($scope) {
         $scope.person = { firstName: 'Omkar', lastName: 'Pednekar', salary: 100000}
      });
   </script>
</body>
</html>
```

Output

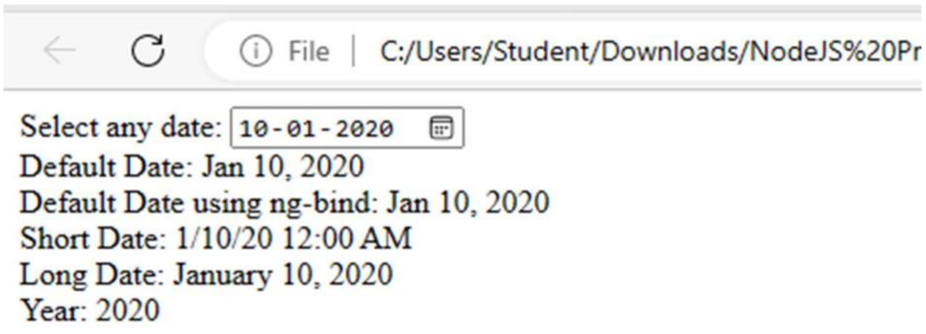


```
Default currency: $100,000.00
Custom currency identifier: Rs.100,000.00
No Fraction: Rs.100,000
Fraction 2: OP100,000.00
```

6.0 Write a program in Angular JS demonstrates Date filter.

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Document</title>
   <script src="https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.8.3/angular.min.js"></script>
</head>
<body ng-app>
   Select any date:
   <input type="date" ng-model="date"><br>
   <!-- <div ng-init = "person.DOB = 323234234898"> -->
   Default Date: {{date | date}}</br>
   Default Date using ng-bind: <span ng-bind="date | date"></span></br>
   Short Date: {{date | date:'short'}}</br>
   Long Date: {{date | date:'longDate'}}</br>
   Year: {{date | date:'yyyy'}}</br>
   </div>
</body>
</html>
```

Output:



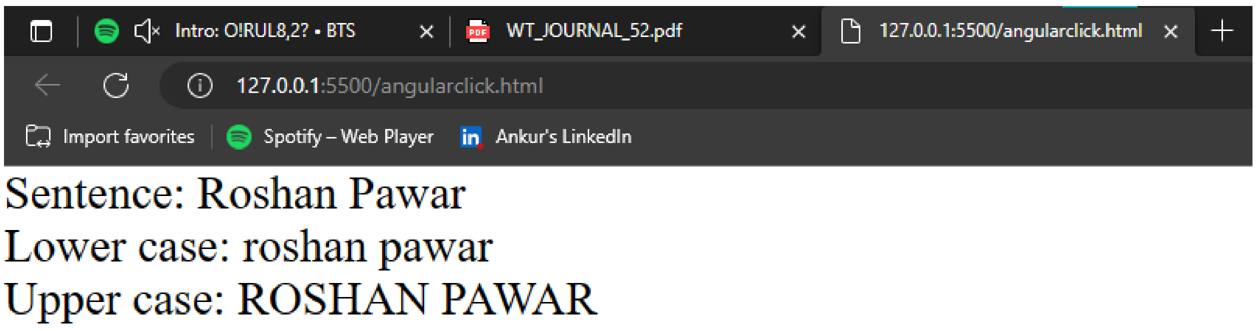
File | C:/Users/Student/Downloads/NodeJS%20Pr

```
Select any date: 10-01-2020
Default Date: Jan 10, 2020
Default Date using ng-bind: Jan 10, 2020
Short Date: 1/10/20 12:00 AM
Long Date: January 10, 2020
Year: 2020
```

6.1 Write a program in AngularJs upper case and lowercase filter.

```
<!DOCTYPE html>
<html >
<head>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.8.3/angular.min.js"></script>
</head>
<body ng-app>
    <div ng-init="person.firstName='Roshan';person.lastName='Pawar'">
        Sentence: {{person.firstName + ' ' + person.lastName}} <br />
        Lower case: {{person.firstName + ' ' + person.lastName | lowercase}} <br />
        Upper case: {{person.firstName + ' ' + person.lastName | uppercase}}
    </div>
</body>
</html>
```

Output:



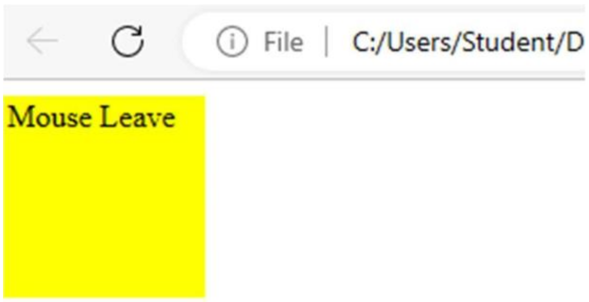
```
Sentence: Roshan Pawar
Lower case: roshan pawar
Upper case: ROSHAN PAWAR
```

6.2 Write a program in AngularJs to demonstrates mouse event.

```
<!--Mouse Event-->
<!DOCTYPE html>
<html>
<head>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.8.3/angular.min.js"></script>
    <style>
        .redDiv {
            width: 100px;
            height: 100px;
            background-color: red;
            padding:2px 2px 2px 2px;
        }

        .yellowDiv {
            width: 100px;
            height: 100px;
            background-color: yellow;
            padding:2px 2px 2px 2px;
        }
    </style>
</head>
<body ng-app>
        <div ng-class="{redDiv: enter, yellowDiv: leave}" ng-
mouseenter="enter=true;leave=false;" ng-mouseleave="leave=true;enter=false">
```

```
            Mouse <span ng-show="enter">Enter</span> <span ng-show="leave">Leave</span>
        </div>
</body>
</html>
```

Output:



Mouse Leave