UNIVERSITY OF MUMBAI

CENTER FOR DISTANCE AND OPEN EDUCATION (CDOE)

CERTIFICATE

The Experiments Duly Signed In This Journal Represent The Work Done By Mst./Ms Shilpa Bhosale Application Id.: 20672 & Seat No.: 8221054 Respectively In **Semester I** Of **First Year Of Master Of Computer Application (NEP)** In The Computer Laboratory Of Pcp Center Prahladrai Dalmia Lions College For Subject Web Technologies During The Academic Year 2025-2026.

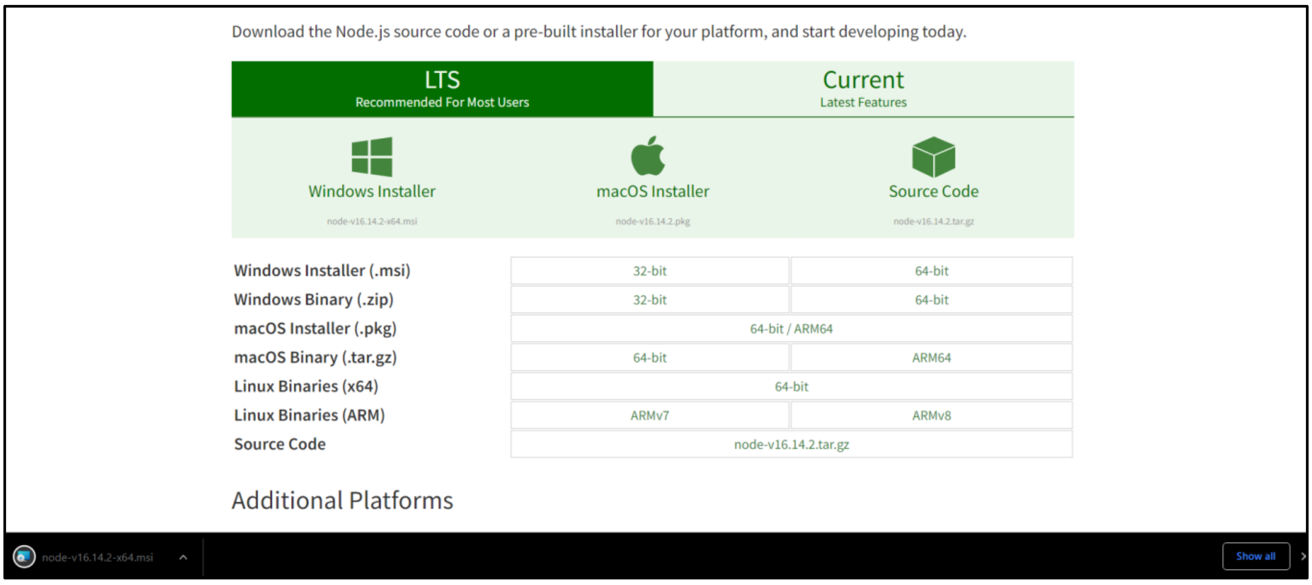
Subject Incharge	Head Of Department	External Examiner

INDEX

Module 1: Introduction To Node JS

1A. Install Node JS and verify Installation

1. Download Node.JS

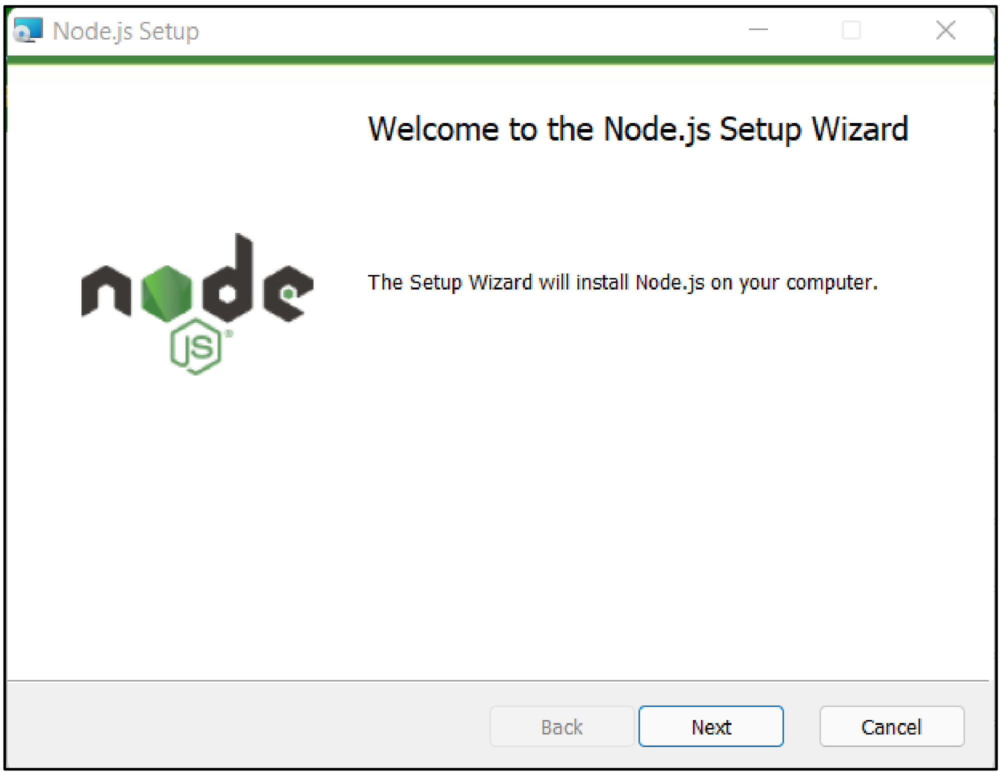


Download the Node.js source code or a pre-built installer for your platform, and start developing today.

Windows Installer (.msi)	32-bit	64-bit
Windows Binary (.zip)	32-bit	64-bit
macOS Installer (.pkg)	64-bit / ARM64	
macOS Binary (.tar.gz)	64-bit	ARM64
Linux Binaries (x64)	64-bit	
Linux Binaries (ARM)	ARMv7	ARMv8
Source Code	node-v16.14.2.tar.gz	

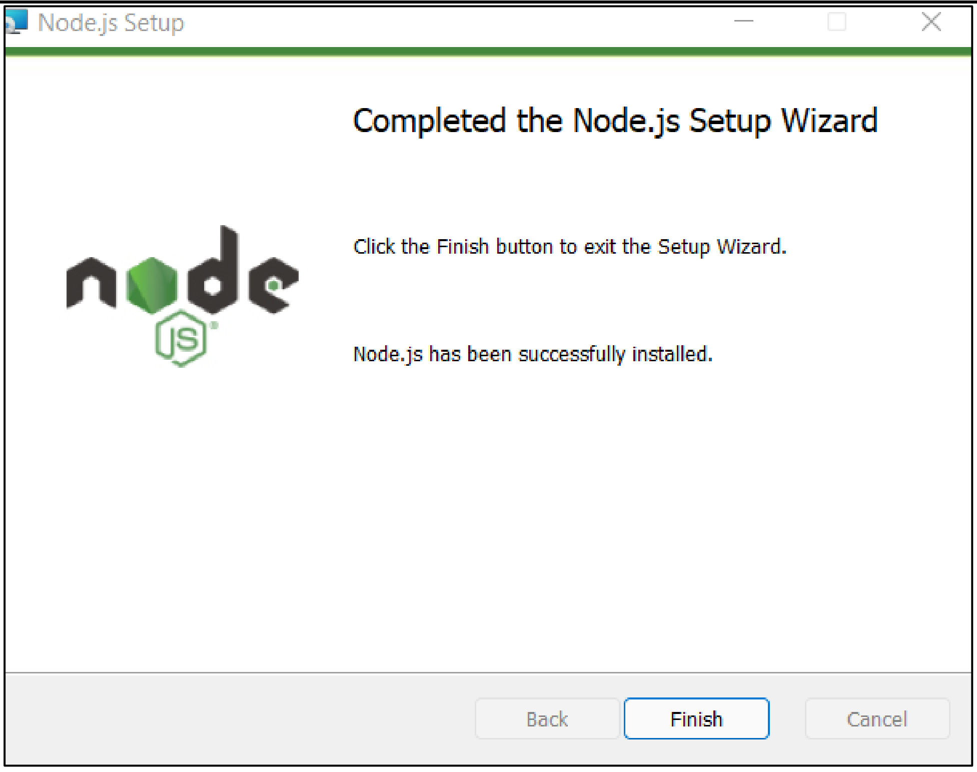
Additional Platforms

2. Install Node.JS



Welcome to the Node.js Setup Wizard

The Setup Wizard will install Node.js on your computer.



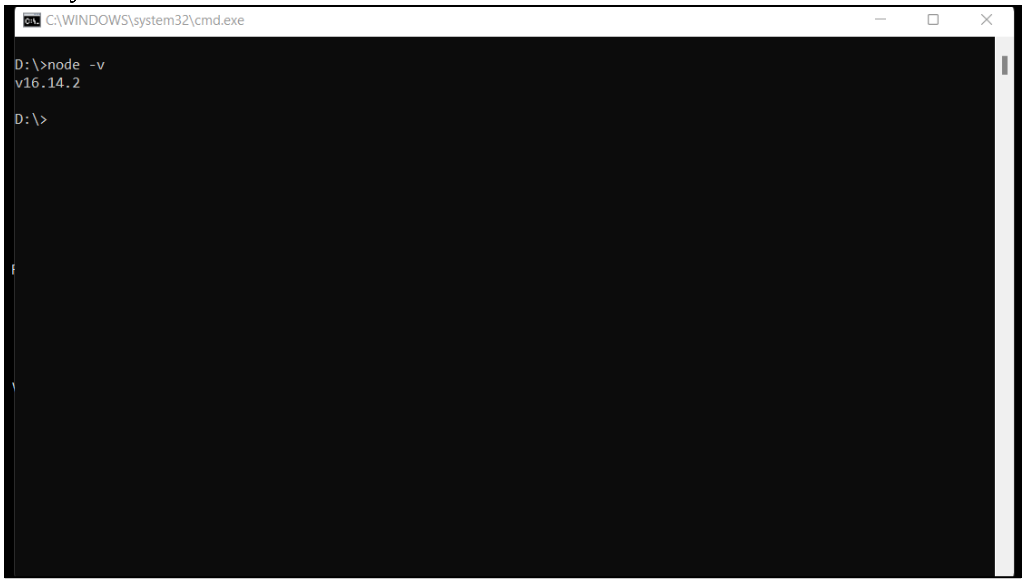
Node.js Setup

Completed the Node.js Setup Wizard

Click the Finish button to exit the Setup Wizard.

Node.js has been successfully installed.

3. Verify Node.JS Installation

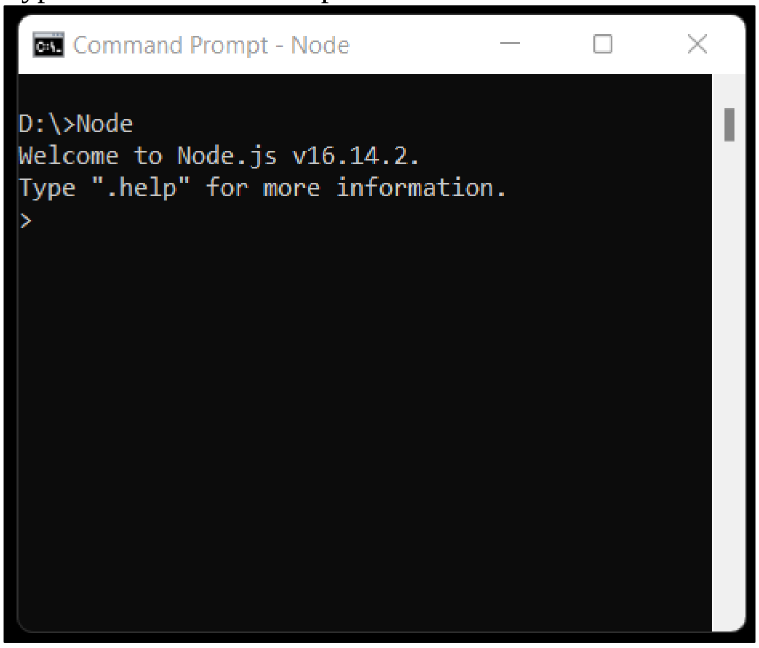


C:\WINDOWS\system32\cmd.exe

```
D:\>node -v
v16.14.2

D:\>
```

1B. Node JS REPL Terminal

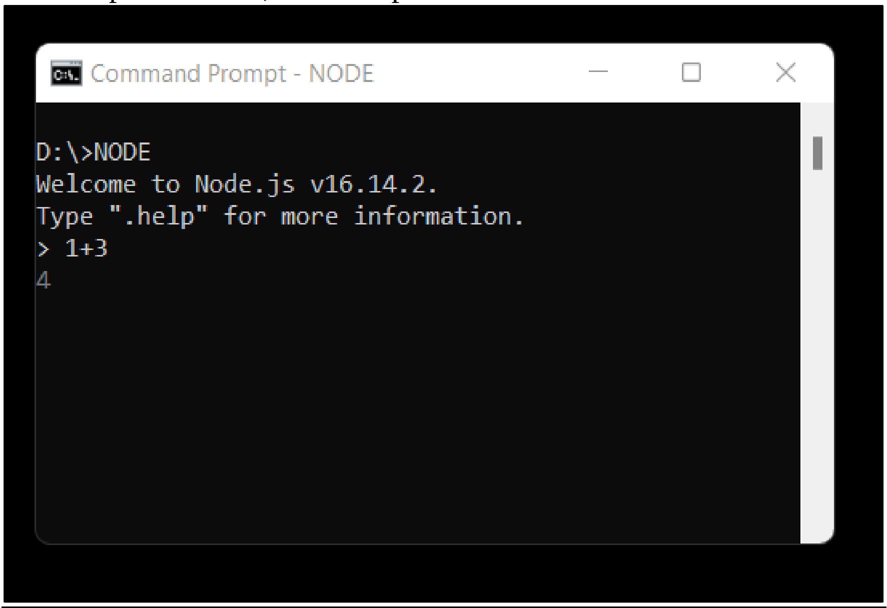
1. Type Node in CMD and press enter



Command Prompt - Node

```
D:\>Node
Welcome to Node.js v16.14.2.
Type ".help" for more information.
>
```

2. Enter expression i.e., 1+3 and press enter



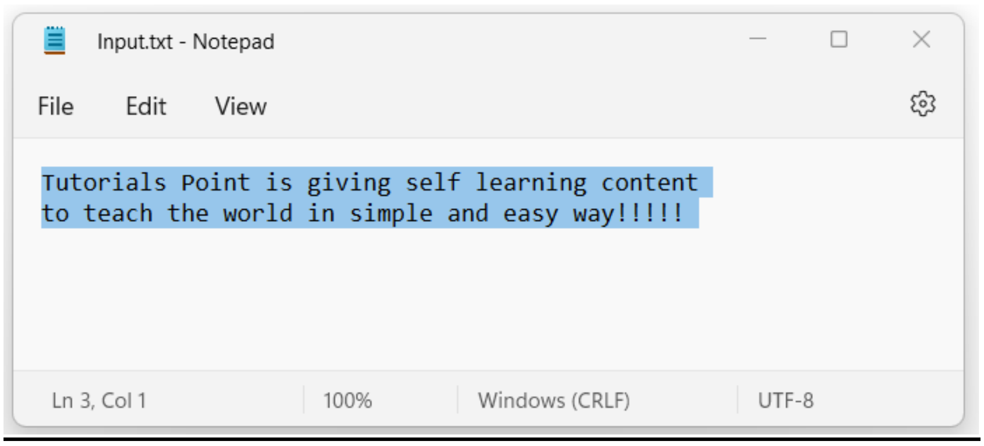
Command Prompt - NODE

```
D:\>NODE
Welcome to Node.js v16.14.2.
Type ".help" for more information.
> 1+3
4
```

Module 2: JS Node.js Modules, Events & Functions

2A. Node JS callback pattern function callback

1. Create and save text file with name input.txt with below content

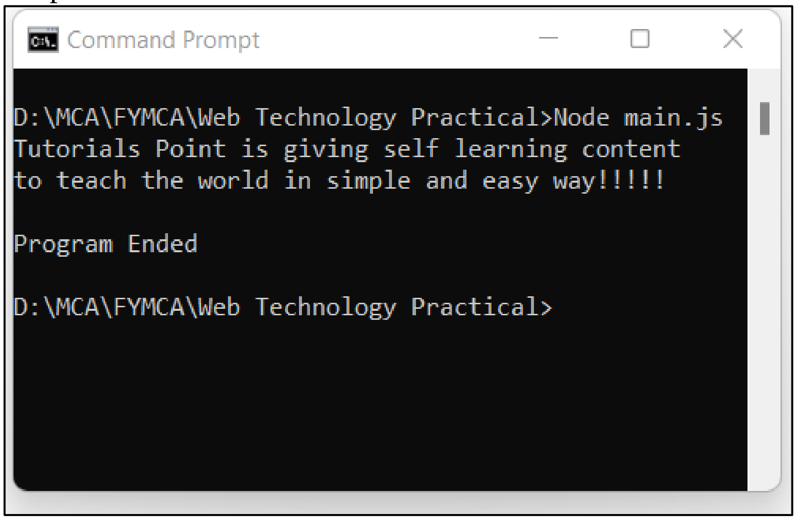


```
Tutorials Point is giving self learning content
to teach the world in simple and easy way!!!!!
```

2. Create a **main.js** file with below code and store it in same location where input.txt is stored

```
var fs = require("fs");
var data = fs.readFileSync('input.txt');
console.log(data.toString());
console.log("Program Ended");
```

3. Go to location where file is stored and run command **Node main.js** and press enter
Output-



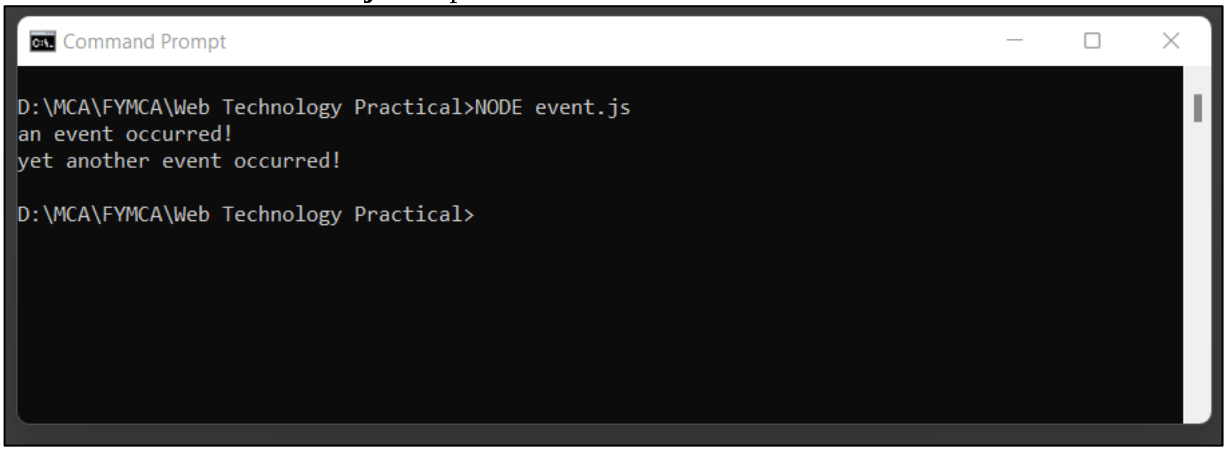
```
D:\MCA\FYMCA\Web Technology Practical>Node main.js
Tutorials Point is giving self learning content
to teach the world in simple and easy way!!!!!

Program Ended

D:\MCA\FYMCA\Web Technology Practical>
```

2B. Node JS callback pattern function callback

1. Create a new file **Event.js** and add below code

```
var EventEmitter = require('events');
const myEmitter = new EventEmitter();

function c1() {
   console.log('an event occurred!');
}
function c2() {
   console.log('yet another event occurred!');
}
myEmitter.on('eventOne', c1); // Register for eventOne
myEmitter.on('eventOne', c2); // Register for eventOne
myEmitter.emit('eventOne');
```

2. Run command **Node Event.js** and press enter



```
D:\MCA\FYMCA\Web Technology Practical>NODE event.js
an event occurred!
yet another event occurred!

D:\MCA\FYMCA\Web Technology Practical>
```

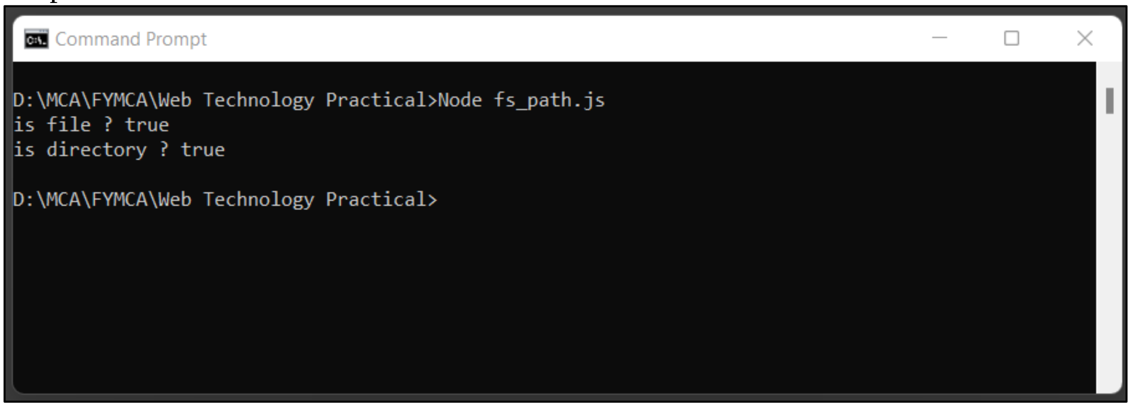
Module 3 : File Handling & HTTP Web Server

3A. FS Module File Path

1. Create a file with name **fs_path.js** and save with below code

```
// Require the given module
var fs = require('fs');
// Use statSync() method to store the returned
// instance into variable named stats
var stats = fs.statSync("D:/MCA/FYMCA/Web Technology
Practical/Students/main.js");
// Use isFile() method to log the result to screen
console.log('is file ? ' + stats.isFile());
var stats = fs.statSync("D:/MCA/FYMCA/Web Technology
Practical/Students");
// Use isDirectory() method to log the result to screen
console.log('is directory ? ' + stats.isDirectory());
```

2. In Command prompt navigate to folder where fs_path.js is stored and run command **Node Fs_path.js** and press enter
   Output-

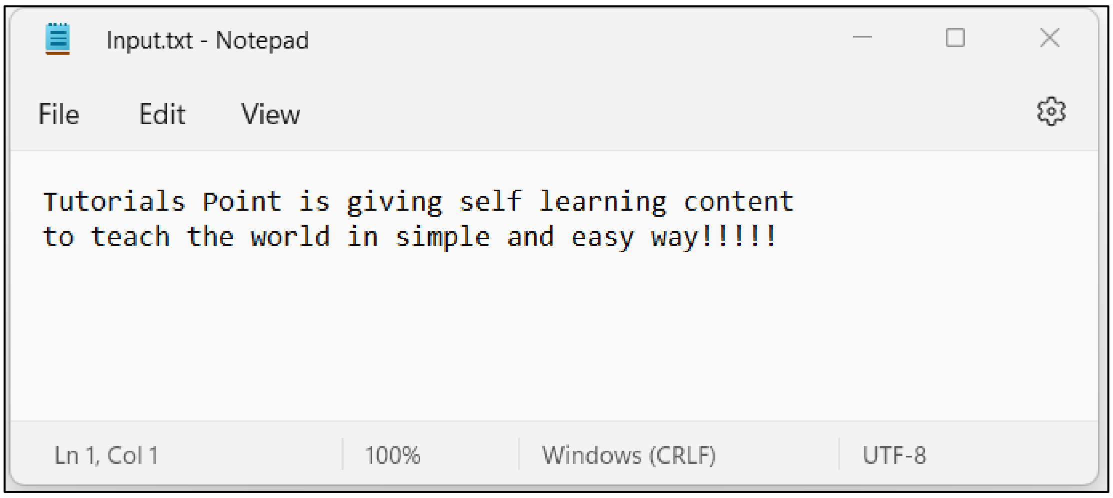


```
D:\MCA\FYMCA\Web Technology Practical>Node fs_path.js
is file ? true
is directory ? true

D:\MCA\FYMCA\Web Technology Practical>
```

3B. FS Module File Path

Read file in Node.JS

1. Create a new text file name **Input.txt** and add below content

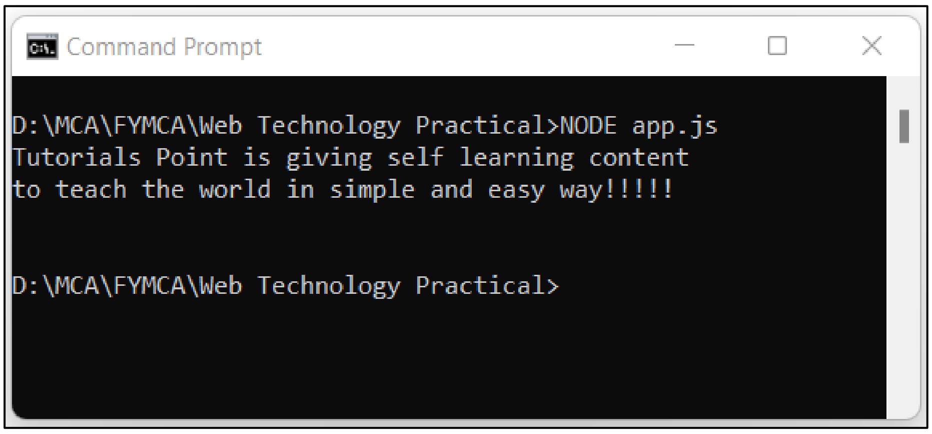


```
Tutorials Point is giving self learning content
to teach the world in simple and easy way!!!!!
```

2. Create a new file name **app.js** and add below code

```
var fs = require("fs");
fs.readFile("input.txt", function(err, buf) {
  console.log(buf.toString());
});
```

3. In Command prompt navigate to location where app.js is stored and run command **Node app.js** and press enter

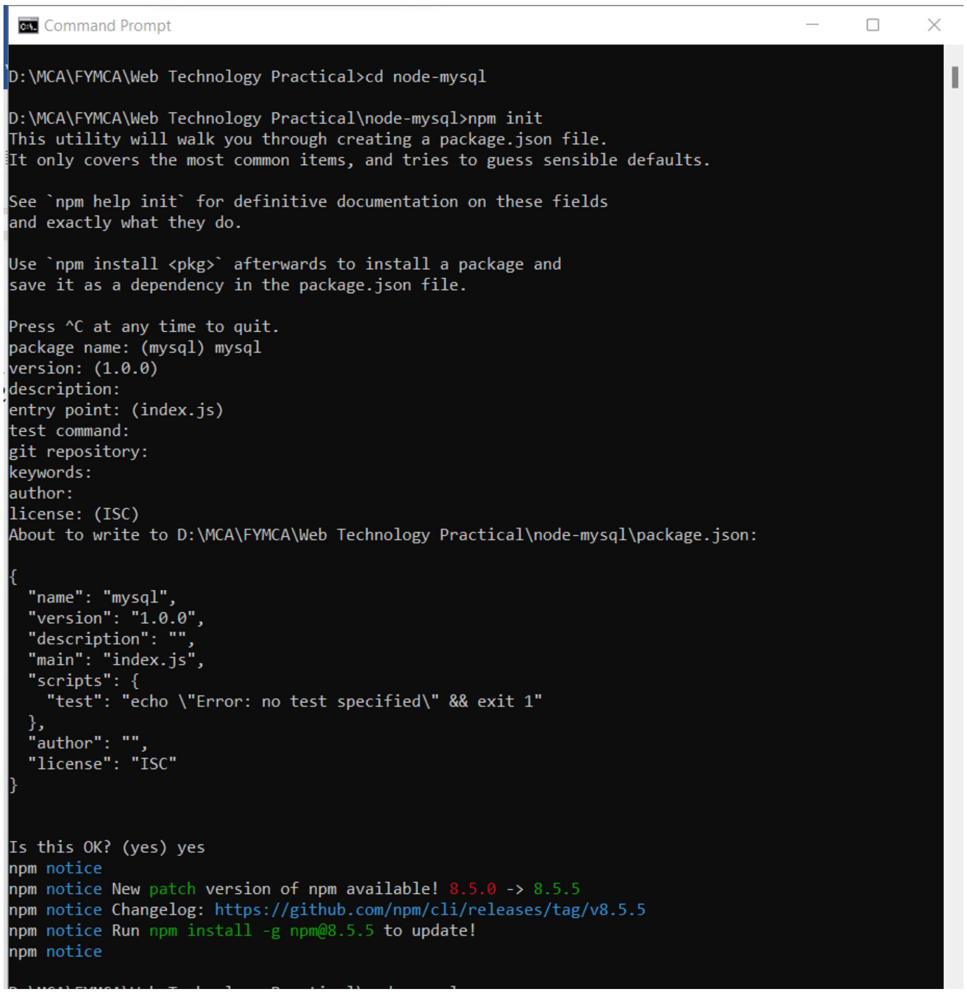


```
D:\MCA\FYMCA\Web Technology Practical>NODE app.js
Tutorials Point is giving self learning content
to teach the world in simple and easy way!!!!!


D:\MCA\FYMCA\Web Technology Practical>
```

Module 4 : Connect MySQL with Node.JS

4A. Connect MySQL with Node.JS

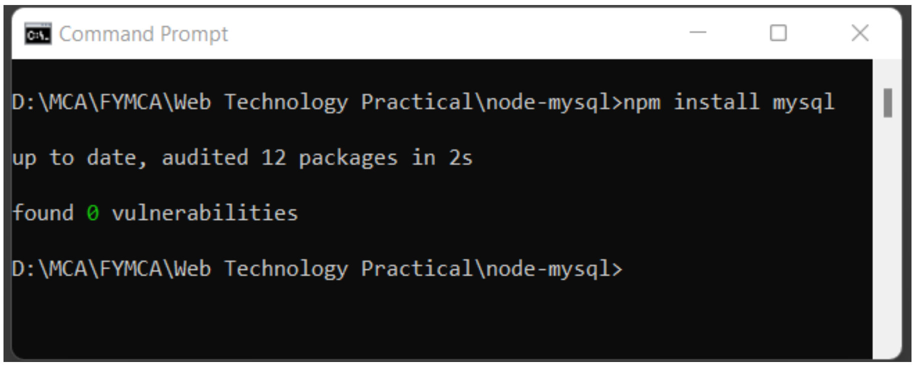
1. Add a folder with name **node-mysql**
2. In Command Prompt navigate to folder **node-mysql** folder and enter **npm init** command and press enter (for adding package.json file)



Command Prompt

```
D:\MCA\FYMCA\Web Technology Practical>cd node-mysql

D:\MCA\FYMCA\Web Technology Practical\node-mysql>npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See `npm help init` for definitive documentation on these fields
and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
package name: (mysql) mysql
version: (1.0.0)
description:
entry point: (index.js)
test command:
git repository:
keywords:
author:
license: (ISC)
About to write to D:\MCA\FYMCA\Web Technology Practical\node-mysql\package.json:

{
  "name": "mysql",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC"
}


Is this OK? (yes) yes
npm notice
npm notice New patch version of npm available! 8.5.0 -> 8.5.5
npm notice Changelog: https://github.com/npm/cli/releases/tag/v8.5.5
npm notice Run npm install -g npm@8.5.5 to update!
npm notice
```

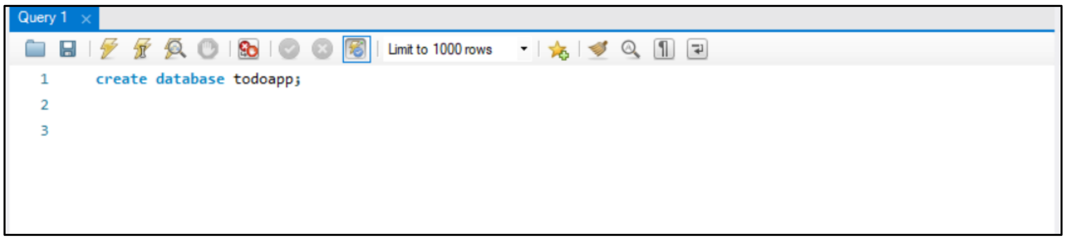
3. Enter Command **npm install mysql** and press enter (for installing mySql Package)



Command Prompt

```
D:\MCA\FYMCA\Web Technology Practical\node-mysql>npm install mysql

up to date, audited 12 packages in 2s

found 0 vulnerabilities

D:\MCA\FYMCA\Web Technology Practical\node-mysql>
```

4. Create Database **todoapp** in MySql

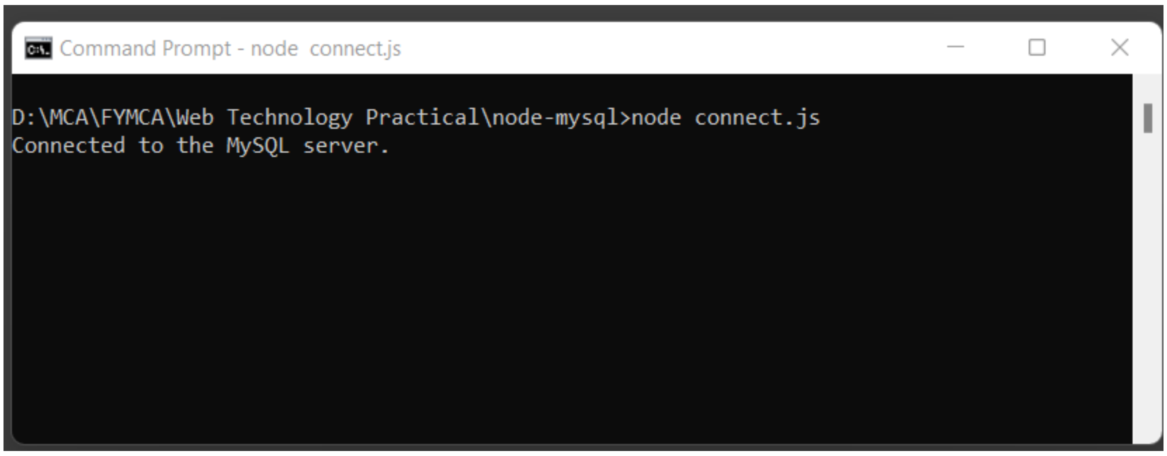


```
create database todoapp;
```

5. Create a file name **Connect.js** and add below code in it

```
let mysql = require('mysql');
let connection = mysql.createConnection({
    host: 'localhost',
    user: 'root',
    password: '1234',
    database: 'todoapp'
});

connection.connect(function(err) {
  if (err) {
    return console.error('error: ' + err.message);
  }
  console.log('Connected to the MySQL server.');
});
```

6. Open command prompt Navigate to folder where **connect.js** is located
   And run command **node connect.js** and press enter



```
D:\MCA\FYMCA\Web Technology Practical\node-mysql>node connect.js
Connected to the MySQL server.
```

4B. Insert Data in SQL using Node.JS

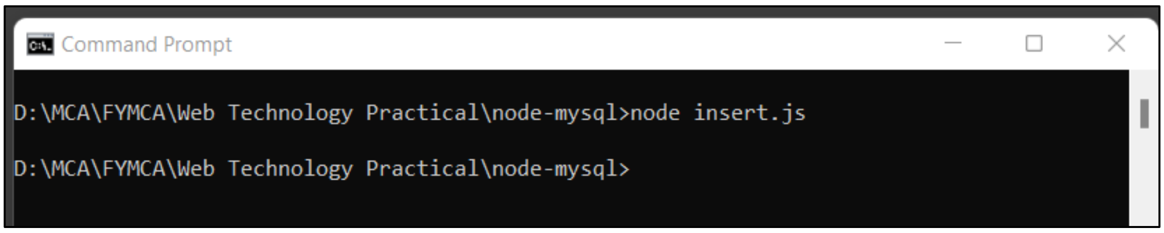
1. Create a file with name **Config.js** and add below code in it

```
 let config = {
   host     : 'localhost',
   user     : 'root',
   password: '1234',
   database: 'todoapp'
};
module.exports = config;
```

2. Create a insert.js file and add below code in it

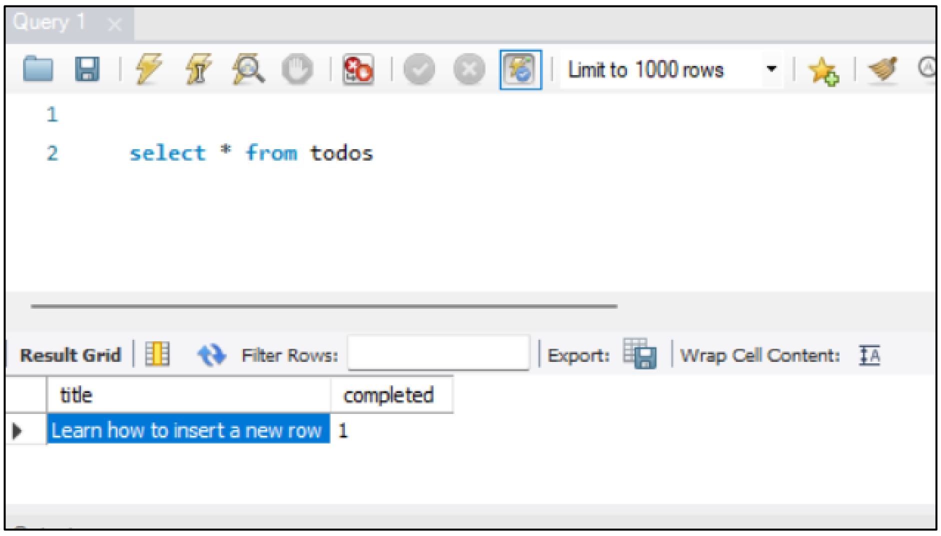
```
let mysql  = require('mysql');
let config = require('./config.js');
let connection = mysql.createConnection(config);
// insert statment
let sql = `INSERT INTO todos(title,completed)
 VALUES('Learn how to insert a new row',true)`;
// execute the insert statment
connection.query(sql);
connection.end();
```

3. Navigate to Folder where insert.js is located and run command **node insert.js** and press enter



```
D:\MCA\FYMCA\Web Technology Practical\node-mysql>node insert.js

D:\MCA\FYMCA\Web Technology Practical\node-mysql>
```

4. Verify in database



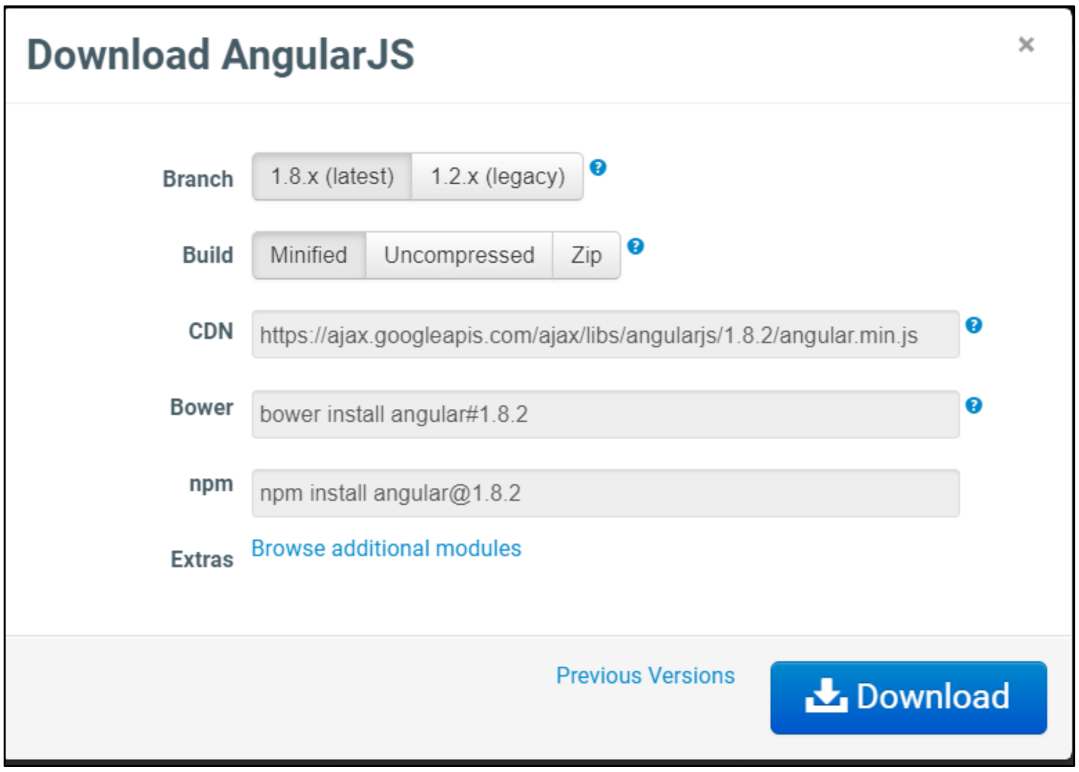
```
select * from todos
```

title	completed
Learn how to insert a new row	1

Module 5 : Angular JS Basics

5A. Setting up the environment

1. Download **Angular JS**



Download AngularJS

Branch: 1.8.x (latest) | 1.2.x (legacy)

Build: Minified | Uncompressed | Zip

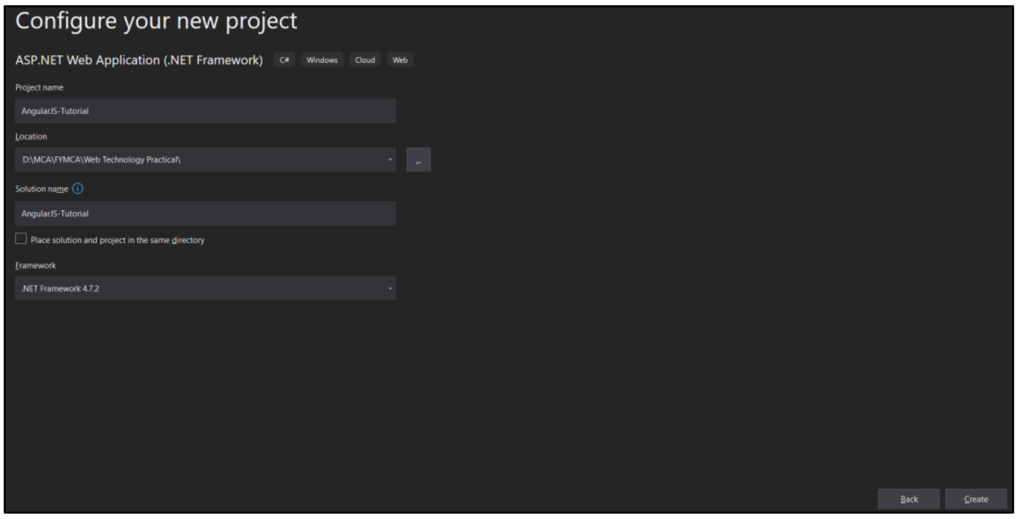
CDN: https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js

Bower: bower install angular#1.8.2

npm: npm install angular@1.8.2

Extras: Browse additional modules

2. Open Visual Studio Create new project with name **AngularJS-Tutorial** and press Create



Configure your new project

ASP.NET Web Application (.NET Framework) C# Windows Cloud Web

Project name: AngularJS-Tutorial

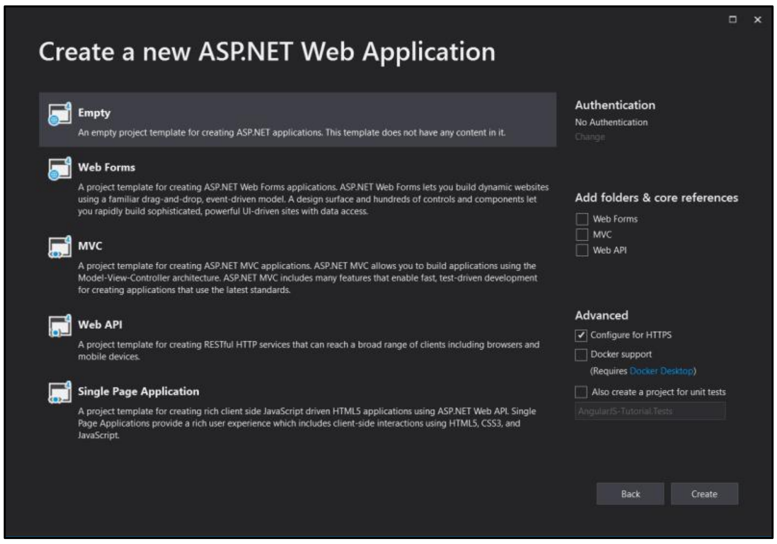
Location: D:\MCA\FYMCA\Web Technology Practical\

Solution name: AngularJS-Tutorial

☐ Place solution and project in the same directory

Framework: .NET Framework 4.7.2

3. Select Empty and press Create button



Create a new ASP.NET Web Application

Empty
An empty project template for creating ASP.NET applications. This template does not have any content in it.

Web Forms
A project template for creating ASP.NET Web Forms applications. ASP.NET Web Forms lets you build dynamic websites using a familiar drag-and-drop, event-driven model. A design surface and hundreds of controls and components let you rapidly build sophisticated, powerful UI-driven sites with data access.

MVC
A project template for creating ASP.NET MVC applications. ASP.NET MVC allows you to build applications using the Model-View-Controller architecture. ASP.NET MVC includes many features that enable fast, test-driven development for creating applications that use the latest standards.

Web API
A project template for creating RESTful HTTP services that can reach a broad range of clients including browsers and mobile devices.

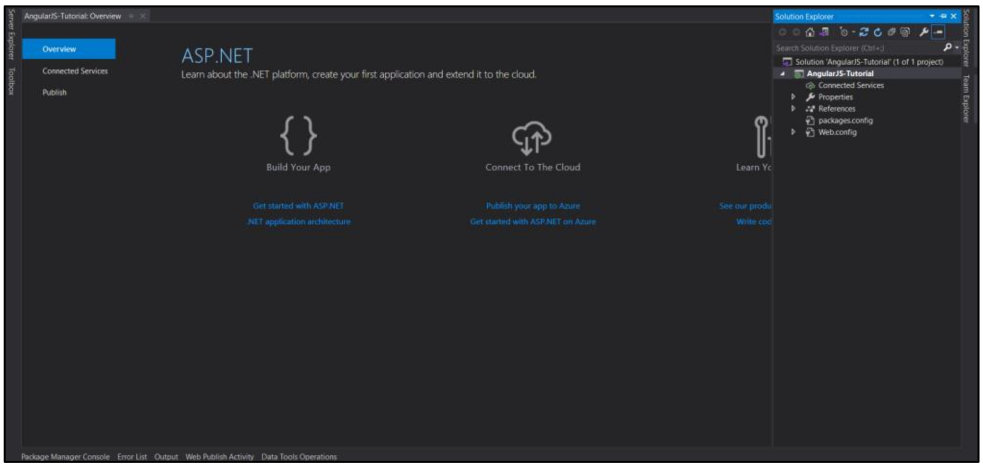
Single Page Application
A project template for creating rich client side JavaScript driven HTML5 applications using ASP.NET Web API. Single Page Applications provide a rich user experience which includes client-side interactions using HTML5, CSS3, and JavaScript.

Authentication
No Authentication
Change

Add folders & core references
- [ ] Web Forms
- [ ] MVC
- [ ] Web API

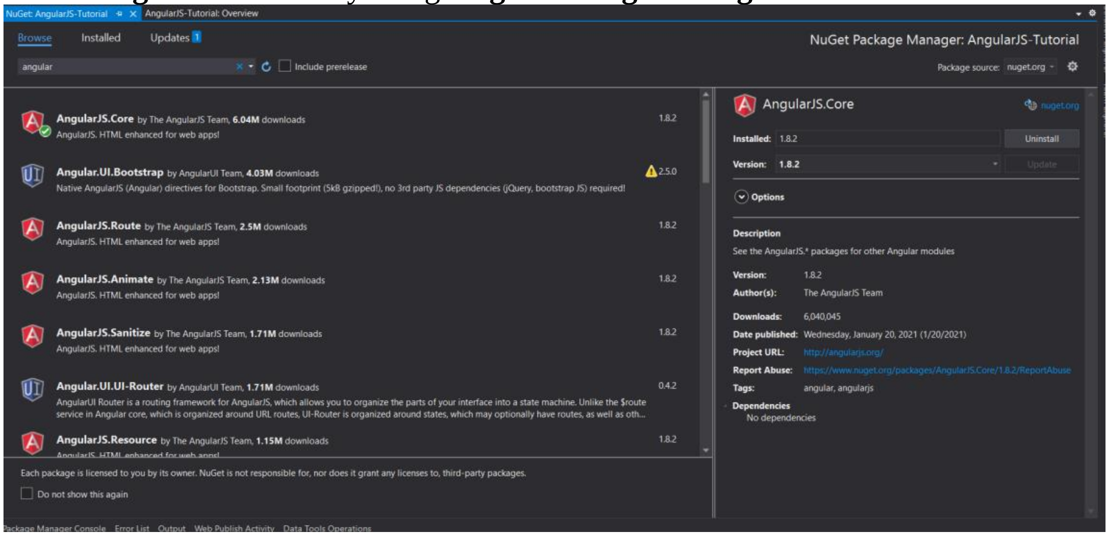
Advanced
- [✓] Configure for HTTPS
- [ ] Docker support
  (Requires Docker Desktop)
- [ ] Also create a project for unit tests
  AngularJS-Tutorial.Tests

4. This will create an empty Asp.net Project



ASP.NET
Learn about the .NET platform, create your first application and extend it to the cloud.

5. Install **Angular JS Core** by using **Nuget Package manager**



NuGet Package Manager: AngularJS-Tutorial

Browse | Installed | Updates 1

angular [ ] Include prerelease

Package source: nuget.org

AngularJS.Core by The AngularJS Team, **6.04M** downloads — 1.8.2
AngularJS. HTML enhanced for web apps!

Angular.UI.Bootstrap by AngularUI Team, **4.03M** downloads — 2.5.0
Native AngularJS (Angular) directives for Bootstrap. Small footprint (5kB gzipped!), no 3rd party JS dependencies (jQuery, bootstrap JS) required!

AngularJS.Route by The AngularJS Team, **2.5M** downloads — 1.8.2
AngularJS. HTML enhanced for web apps!

AngularJS.Animate by The AngularJS Team, **2.13M** downloads — 1.8.2
AngularJS. HTML enhanced for web apps!

AngularJS.Sanitize by The AngularJS Team, **1.71M** downloads — 1.8.2
AngularJS. HTML enhanced for web apps!

Angular.UI.UI-Router by AngularUI Team, **1.71M** downloads — 0.4.2
AngularUI Router is a routing framework for AngularJS, which allows you to organize the parts of your interface into a state machine. Unlike the $route service in Angular core, which is organized around URL routes, UI-Router is organized around states, which may optionally have routes, as well as oth...

AngularJS.Resource by The AngularJS Team, **1.15M** downloads — 1.8.2
AngularJS. HTML enhanced for web apps!

Each package is licensed to you by its owner. NuGet is not responsible for, nor does it grant any licenses to, third-party packages.
[ ] Do not show this again

AngularJS.Core — nuget.org

Installed: 1.8.2 — Uninstall
Version: 1.8.2 — Update

Options

Description
See the AngularJS.* packages for other Angular modules

Version:	1.8.2
Author(s):	The AngularJS Team
Downloads:	6,040,045
Date published:	Wednesday, January 20, 2021 (1/20/2021)
Project URL:	http://angularjs.org/
Report Abuse:	https://www.nuget.org/packages/AngularJS.Core/1.8.2/ReportAbuse
Tags:	angular, angularjs

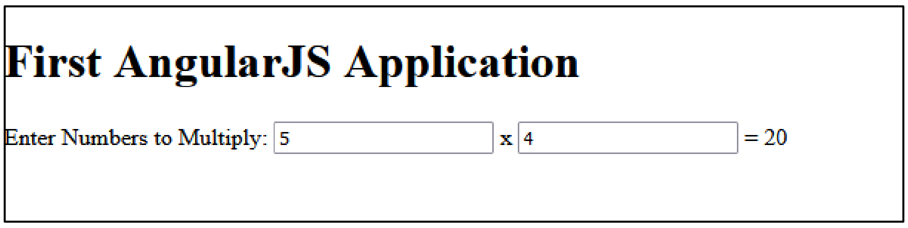
Dependencies
No dependencies

5B. First Application (Multiplier)

1. Create an HTML file in Visual Studio and add below html and **Angular.JS script path/URL** in it

```
<!DOCTYPE html>

<html>
    <head>
        <title>First AngularJS Application</title>
        <script src="Scripts/angular.js"></script>

    </head>
    <body ng-app>
        <h1>First AngularJS Application</h1>

        Enter Numbers to Multiply:
        <input type="text" ng-model="Num1" /> x <input type="text"
ng-model="Num2" />
        = <span>{{Num1 * Num2}}</span>
    </body>

</html>
```

2. Execute the code and verify the output by entering number in textbox



```
First AngularJS Application

Enter Numbers to Multiply: 5 x 4 = 20
```

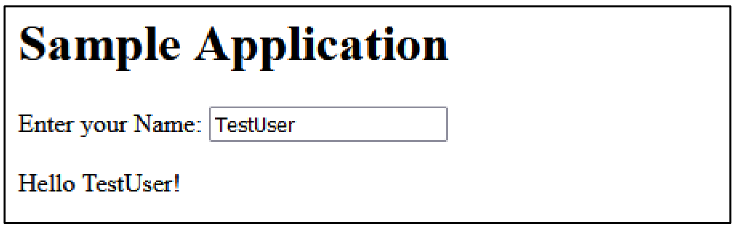
Module 6 : Filters, Directive

6A. Program to display your name with welcome note: HELLO

1. Create a html file name **WelcomeMessage.html** and write below html and **Angular.JS script path/URL** in it

```
<html>
<head>
    <title>AngularJS First Application</title>
</head>
<body>
    <h1>Sample Application</h1>

    <div ng-app="">
        <p>Enter your Name: <input type="text" ng-model="name"></p>
        <p>Hello <span ng-bind="name"></span>!</p>
    </div>

    <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.3.14/angular.
min.js">
    </script>

</body>
</html>
```

2. Run the code and verify output by entering text in textbox



Sample Application

Enter your Name: TestUser

Hello TestUser!

6B. Experiment: Create an application using Filters

1. Create an HTML file name **filter.html** and write below HTML and **Angular.JS script path/URL** in it

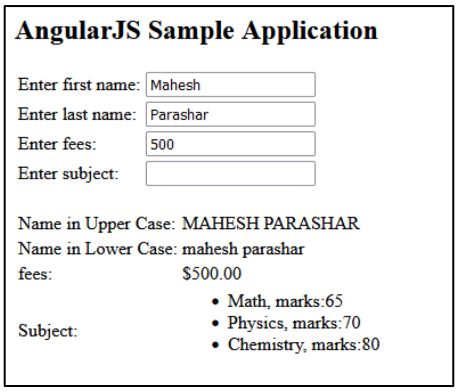
```
<html>
<head>
    <title>Angular JS Filters</title>
    <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.3.14/angular.min.js"
>
    </script>
</head>

<body>
    <h2>AngularJS Sample Application</h2>

    <div ng-app="mainApp" ng-controller="studentController">
        <table border="0">
            <tr>
                <td>Enter first name:</td>
                <td><input type="text" ng-model="student.firstName"></td>
            </tr>
            <tr>
                <td>Enter last name: </td>
                <td><input type="text" ng-model="student.lastName"></td>
            </tr>
            <tr>
                <td>Enter fees: </td>
                <td><input type="text" ng-model="student.fees"></td>
            </tr>
            <tr>
                <td>Enter subject: </td>
                <td><input type="text" ng-model="subjectName"></td>
            </tr>
        </table>
        <br />

        <table border="0">
            <tr>
                <td>Name in Upper Case: </td>
                <td>{{student.fullName() | uppercase}}</td>
            </tr>
            <tr>
                <td>Name in Lower Case: </td>
                <td>{{student.fullName() | lowercase}}</td>
            </tr>
            <tr>
                <td>fees: </td>
                <td>
                    {{student.fees | currency}}
                </td>
            </tr>
            <tr>
```

```
                    <td>Subject:</td>
                    <td>
                        <ul>
                            <li ng-repeat="subject in student.subjects |
filter: subjectName |orderBy:'marks'">
                                {{ subject.name + ', marks:' + subject.marks }}
                            </li>
                        </ul>
                    </td>
                </tr>
            </table>
        </div>
    </body>
    </html>
```

2. Add below JavaScript using script tag in html code

```
<script>
        var mainApp = angular.module("mainApp", []);

        mainApp.controller('studentController', function ($scope) {
            $scope.student = {
                firstName: "Mahesh",
                lastName: "Parashar",
                fees: 500,

                subjects: [
                    { name: 'Physics', marks: 70 },
                    { name: 'Chemistry', marks: 80 },
                    { name: 'Math', marks: 65 }
                ],
                fullName: function () {
                    var studentObject;
                    studentObject = $scope.student;
                    return studentObject.firstName + " " +
studentObject.lastName;
                }
            };
        });
    </script>
```

3. Execute the code and verify the output



AngularJS Sample Application

Enter first name: Mahesh
Enter last name: Parashar
Enter fees: 500
Enter subject:

Name in Upper Case: MAHESH PARASHAR
Name in Lower Case: mahesh parashar
fees: $500.00
Subject:
- Math, marks:65
- Physics, marks:70
- Chemistry, marks:80

Module 7 : Controllers

7A. Programming Controllers & $scope object

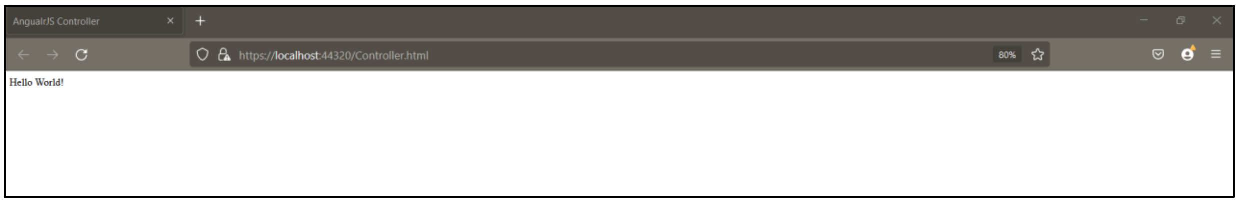
1. Add HTML file with name **Controller.html** and add below HTML and **Angular.JS script path/URL** in it

```
<!DOCTYPE html>
<html>
<head>
    <title>AngualrJS Controller</title>
    <script src="Scripts/angular.js"></script>
</head>
<body ng-app="myNgApp">
    <div ng-controller="myController">
        {{message}}
    </div>

</body>
</html>
```

2. Add Below script using script tag in HTML

```
<script>
        var ngApp = angular.module('myNgApp', []);

        ngApp.controller('myController', function ($scope) {
            $scope.message = "Hello World!";
        });
    </script>
```

3. Run the HTML file and verify the output



```
Hello World!
```

7B. Adding Behaviour to a Scope Object

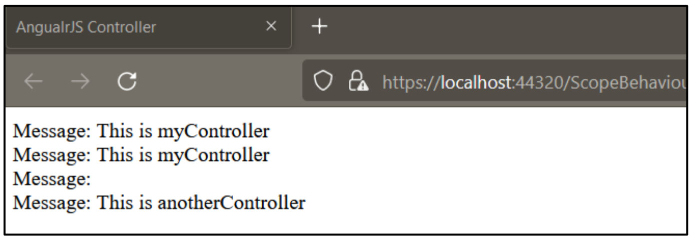
1. Add Html file with name **ScopeBehaviour.html** and add below html and **Angular.JS script path/URL** in it

```
<!DOCTYPE html>
<html>
<head>
    <title>AngualrJS Controller</title>
    <script src="Scripts/angular.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>

</body>
</html>
```

2. Add below JavaScript within script tag in HTML

```
<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>
```

3. Run above HTML file and verify the output



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

Module 8 : Forms and SPA (Single Page Application)

8A. Create Simple Angular Forms using different input controls & events

1. Add an HTML file with name **AngularForm.html** and add below HTML and **Angular.JS script path/URL** in it

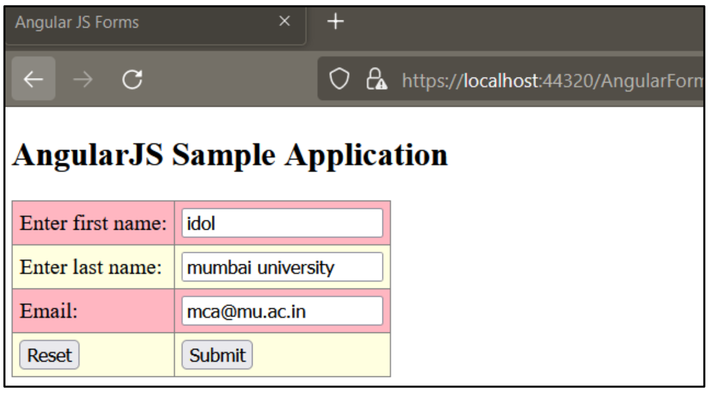
```
<!DOCTYPE html>
<html>
<head>
    <title>Angular JS Forms</title>
    <script src="Scripts/angular.js"></script>

    <style>
        table, th, td {
            border: 1px solid grey;
            border-collapse: collapse;
            padding: 5px;
        }

            table tr:nth-child(odd) {
                background-color: lightpink;
            }

            table tr:nth-child(even) {
                background-color: lightyellow;
            }
    </style>

</head>
<body>

    <h2>AngularJS Sample Application</h2>
    <div ng-app="mainApp" ng-controller="studentController">

        <form name="studentForm" novalidate>
            <table border="0">
                <tr>
                    <td>Enter first name:</td>
                    <td>
                        <input name="firstname" type="text" ng-
model="firstName" required>
                        <span style="color:red" ng-
show="studentForm.firstname.$dirty &&
studentForm.firstname.$invalid">
                            <span ng-
show="studentForm.firstname.$error.required">First Name is
required.</span>
                        </span>
                    </td>
                </tr>

                <tr>
                    <td>Enter last name: </td>
                    <td>
                        <input name="lastname" type="text" ng-
model="lastName" required>
                        <span style="color:red" ng-
show="studentForm.lastname.$dirty && studentForm.lastname.$invalid">
                            <span ng-
show="studentForm.lastname.$error.required">Last Name is
required.</span>
                        </span>
                    </td>
                </tr>

                <tr>
                    <td>Email: </td>
                    <td>
                        <input name="email" type="email" ng-
model="email" length="100" required>
                        <span style="color:red" ng-
show="studentForm.email.$dirty && studentForm.email.$invalid">
                            <span ng-
show="studentForm.email.$error.required">Email is required.</span>
                            <span ng-
show="studentForm.email.$error.email">Invalid email address.</span>
                        </span>
                    </td>
                </tr>

                <tr>
                    <td>
                        <button ng-click="reset()">Reset</button>
                    </td>
                    <td>
                        <button ng-
disabled="studentForm.firstname.$dirty &&
                        studentForm.firstname.$invalid ||
studentForm.lastname.$dirty &&
                        studentForm.lastname.$invalid ||
studentForm.email.$dirty &&
                        studentForm.email.$invalid" ng-
click="submit()">
                            Submit
                        </button>
                    </td>
                </tr>

            </table>
```

```
            </form>
        </div>



    </body>
    </html>
```

2. Add below Angular JavaScript code inside script tag in html

```
    <script>
            var mainApp = angular.module("mainApp", []);

            mainApp.controller('studentController', function ($scope) {
                $scope.reset = function () {
                    $scope.firstName = "idol";
                    $scope.lastName = "mumbai university";
                    $scope.email = "mca@mu.ac.in";
                }

                $scope.reset();
            });
        </script>
```

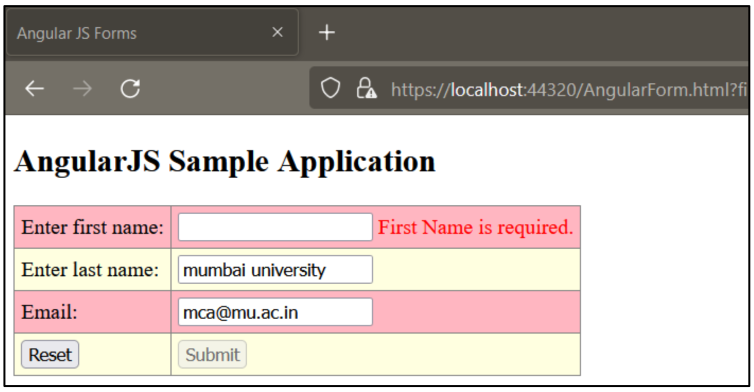
3. Run the html and verify the Output


Angular JS Forms

https://localhost:44320/AngularForm

AngularJS Sample Application

Enter first name:	idol
Enter last name:	mumbai university
Email:	mca@mu.ac.in
Reset	Submit


Angular JS Forms

https://localhost:44320/AngularForm.html?fi

AngularJS Sample Application

Enter first name:	First Name is required.
Enter last name:	mumbai university
Email:	mca@mu.ac.in
Reset	Submit

8B. Implement the concept of Single page application

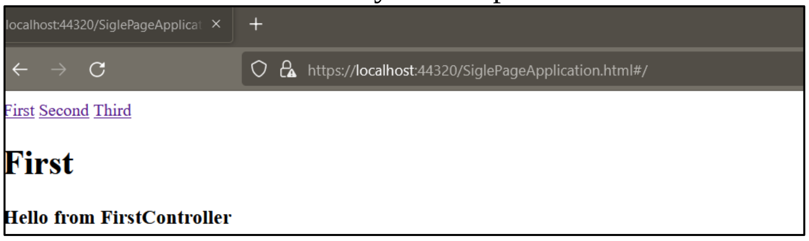
1. Add a new HTML file name **SinglePageApplication.html** and add below code with **AngularJS** and **AngularJS routing script URL**

```
<!doctype html>
<html ng-app="myApp">
<head>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.4.7/angular.m
in.js"></script>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.4.7/angular-
route.min.js"></script>
</head>
<body >
    <script type="text/ng-template" id="pages/first.html">
        <h1>First</h1>
        <h3>{{message}}</h3>
    </script>
    <script type="text/ng-template" id="pages/second.html">
        <h1>Second</h1>
        <h3>{{message}}</h3>
    </script>
    <script type="text/ng-template" id="pages/third.html">
        <h1>Third</h1>
        <h3>{{message}}</h3>
    </script>

        <a href="#/">First</a>
        <a href="#/second">Second</a>
        <a href="#/third">Third</a>

        <div ng-view></div>
    <script src="app.js"></script>
</body>
</html>
```

2. Add a new JS file name app.js and add below Angular JS code for routing

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

app.controller('FirstController', function ($scope) {
    $scope.message = 'Hello from FirstController';
});
var app = angular.module('myApp', ['ngRoute']);
app.config(function ($routeProvider) {
    $routeProvider
        .when('/', {
            templateUrl: 'pages/first.html',
            controller: 'FirstController'
        })
        .when('/second', {
            templateUrl: 'pages/second.html',
            controller: 'SecondController'
        })
        .when('/third', {
            templateUrl: 'pages/third.html',
            controller: 'ThirdController'
        })
        .otherwise({ redirectTo: '/' });
});
app.controller('FirstController', function ($scope) {
    $scope.message = 'Hello from FirstController';
});

app.controller('SecondController', function ($scope) {
    $scope.message = 'Hello from SecondController';
});

app.controller('ThirdController', function ($scope) {
    $scope.message = 'Hello from ThirdController';
});
```

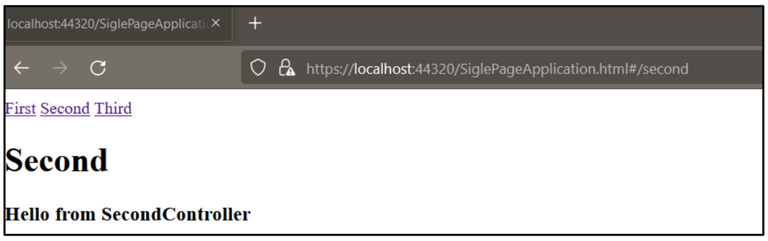
3. Run the HTML file and verify the Output


localhost:44320/SiglePageApplicat ×
https://localhost:44320/SiglePageApplication.html#/

First Second Third

First

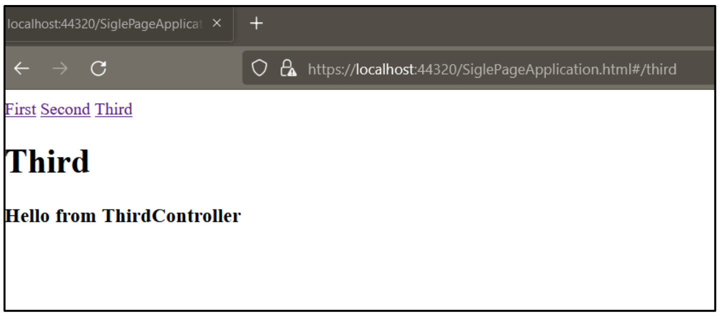
Hello from FirstController


localhost:44320/SiglePageApplicat ×
https://localhost:44320/SiglePageApplication.html#/second

First Second Third

Second

Hello from SecondController


localhost:44320/SiglePageApplicat ×
https://localhost:44320/SiglePageApplication.html#/third

First Second Third

Third

Hello from ThirdController