

Download node js

node –v

v10.15.3

npm –v

6.4.1

npm troubleshooting

npm is the package manager for the javascript world like maven

visual studio code download

File Explorer: ctl+shft+ E

Search Across file: ctrl+shft+F

Source code management ctrl+shft+G

Launch & Debug: ctrl+shft+D

Manage extensions ctrl+shft+X

View Errors and Warnings ctrl+shft+M

To open a file : ctrl+P

**npm init**

package name: first-package-with-npm

version

description

**npm install jquery**

node\_modules

package-lock.json

package.json files are created.

"dependencies": {

"jquery": "^3.4.1"

}

Is added in package.json file

^3.4.1: whatever the latest jquery version can download

All the jquery files are downloaded into node\_modules folder.

Whenever we give the command **npm install** then node\_modules folder will be created.

Angular uses the typescript

**Typescript= Java Script + strong typing**

Jwt for authentication

To create the enterprise application for differnet devices like Mobile, Desktop.

Javascript is loosely typed.

Ex: var val;

val= 5

val=”test”;

allows in javascript where as in java it is not allowed.

Ex: int val;

Val= 6;

Val= “test” gives the compile time error.

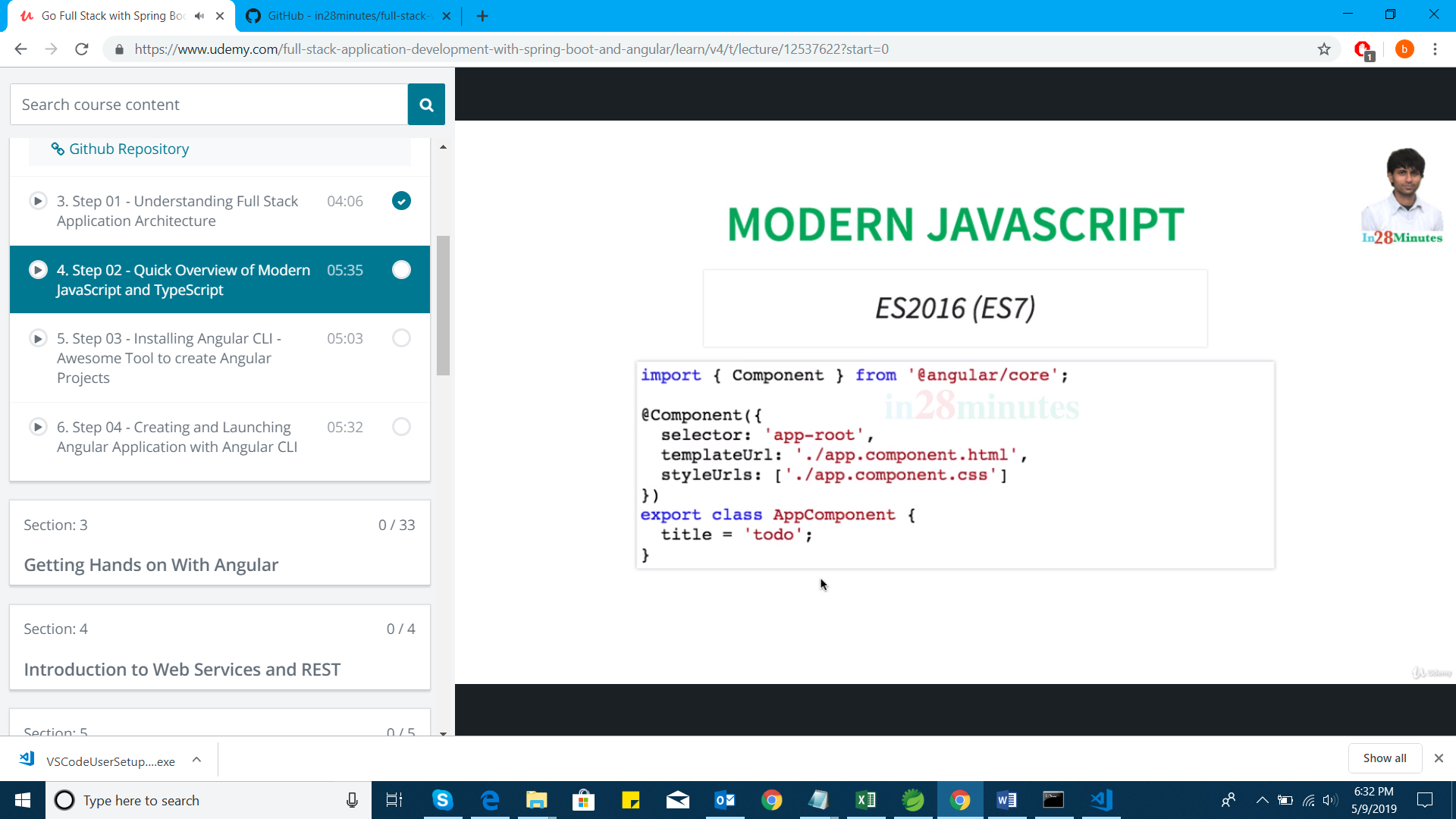
Java Script: No types.

**TypeScript Ex:**

value: number

value= 1

value=”ten” compilation errors.



Component from module

@Component is directive

EcmaScript is Standard

Javascript is implementation

ES=EcmaScript

ES6 onward will helpful to write the classes, Modularized, maintainable java script code.

TypeScript method definition

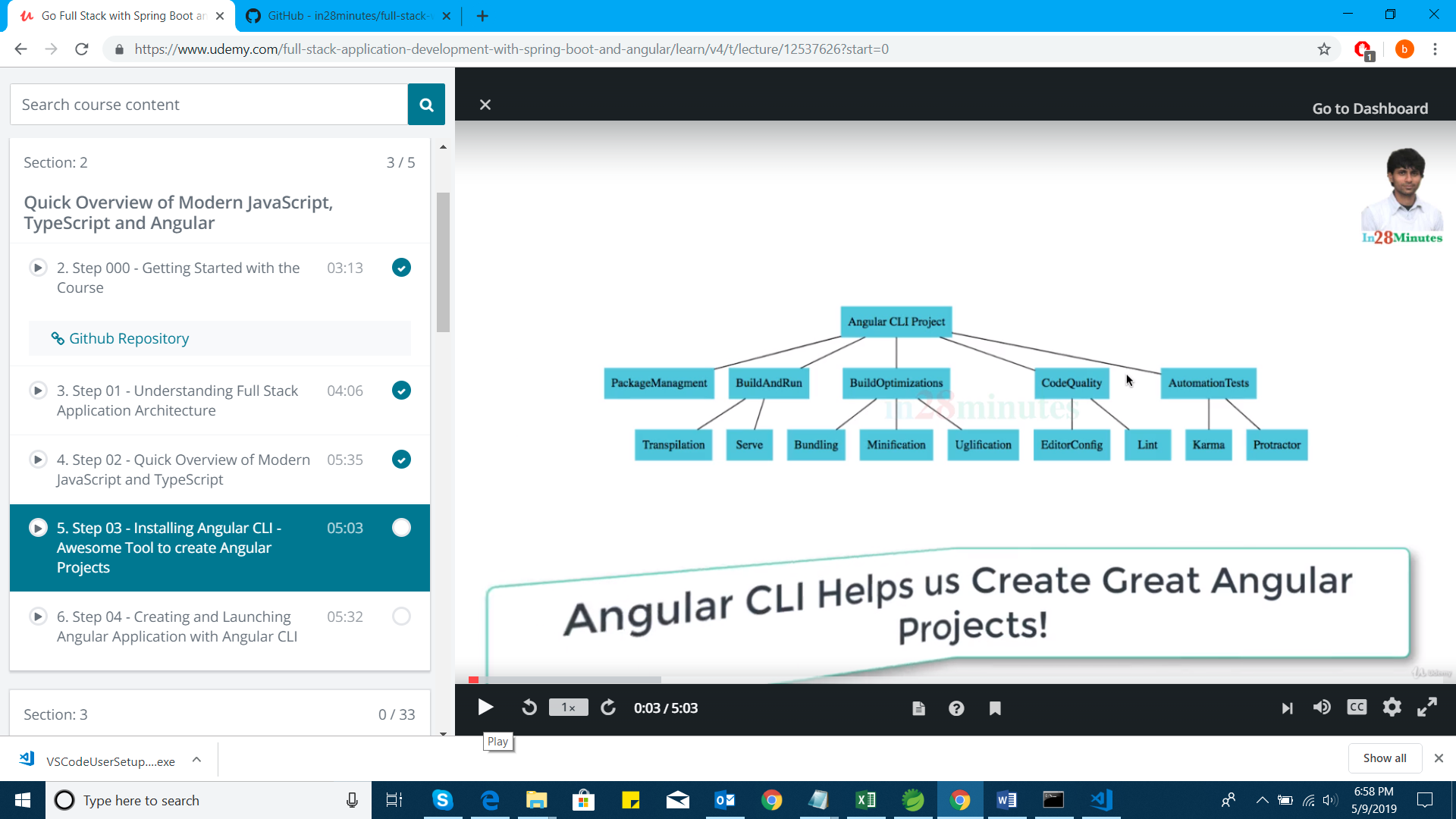
Function add (number1: number, number2: number) : number {

return number1+number2

}

interface FirstInterface {

}



**Angular cli**

Install angular cli to go to the angular page

CLI command helps to create angular application and also helps to generate the lot of code, runs the code locally and perform quality checks.

**npm install –g @angular/cli@7.0.3**

To check the version

**ng version**

To create the new project

**ng help** to get all the available commands

C:\Users\Soumya>ng help

Available Commands:

add Adds support for an external library to your project.

build (b) Compiles an Angular app into an output directory named dist/ at the given output path. Must be executed from within a workspace directory.

config Retrieves or sets Angular configuration values.

doc (d) Opens the official Angular documentation (angular.io) in a browser, and searches for a given keyword.

e2e (e) Builds and serves an Angular app, then runs end-to-end tests using Protractor.

generate (g) Generates and/or modifies files based on a schematic.

help Lists available commands and their short descriptions.

lint (l) Runs linting tools on Angular app code in a given project folder.

new (n) Creates a new workspace and an initial Angular app.

run Runs a custom target defined in your project.

serve (s) Builds and serves your app, rebuilding on file changes.

test (t) Runs unit tests in a project.

update Updates your application and its dependencies. See https://update.angular.io/

version (v) Outputs Angular CLI version.

xi18n Extracts i18n messages from source code.

New angular project creation : **ng new todo**

To execute the program : **ng serve**

<http://localhost:4200/> : to see the angular page

README.md file about the project.

(alias) Component(obj: Component): TypeDecorator  
import Component

Decorator that marks a class as an Angular component and provides configuration metadata that determines how the component should be processed, instantiated, and used at runtime.

Components are the most basic UI building block of an Angular app. An Angular app contains a tree of Angular components.

Angular components are a subset of directives, always associated with a template. Unlike other directives, only one component can be instantiated per an element in a template.

A component must belong to an NgModule in order for it to be available to another component or application. To make it a member of an NgModule, list it in the declarations field of the @NgModule metadata.

Note that, in addition to these options for configuring a directive, you can control a component's runtime behavior by

**ng serve** automatically pulls the updated code reflect the changes.

**ng lint** to check the coding standards.

For testing removed the new line at the end of the file in app.component.ts

C:\angular\frontend\todo>ng lint

Linting "todo"...

ERROR: C:/angular/frontend/todo/src/app/app.component.ts[10, 2]: file should end with a newline

Lint errors found in the listed files.

Linting "todo-e2e"...

All files pass linting.

All the coding standards are defined in the **tslint.json** file under the project folder.

"eofline": true,

"forin": true,

"import-blacklist": [

true,

"rxjs/Rx"

],

**ng build :** some files will be created **and dist** folder will be created under the project.

C:\angular\frontend\todo>ng build

Date: 2019-05-09T15:58:17.963Z

Hash: dd12c5aebb4c38cc38c4

Time: 6300ms

chunk {main} main.js, main.js.map (main) 11.5 kB [initial] [rendered]

chunk {polyfills} polyfills.js, polyfills.js.map (polyfills) 236 kB [initial] [rendered]

chunk {runtime} runtime.js, runtime.js.map (runtime) 6.08 kB [entry] [rendered]

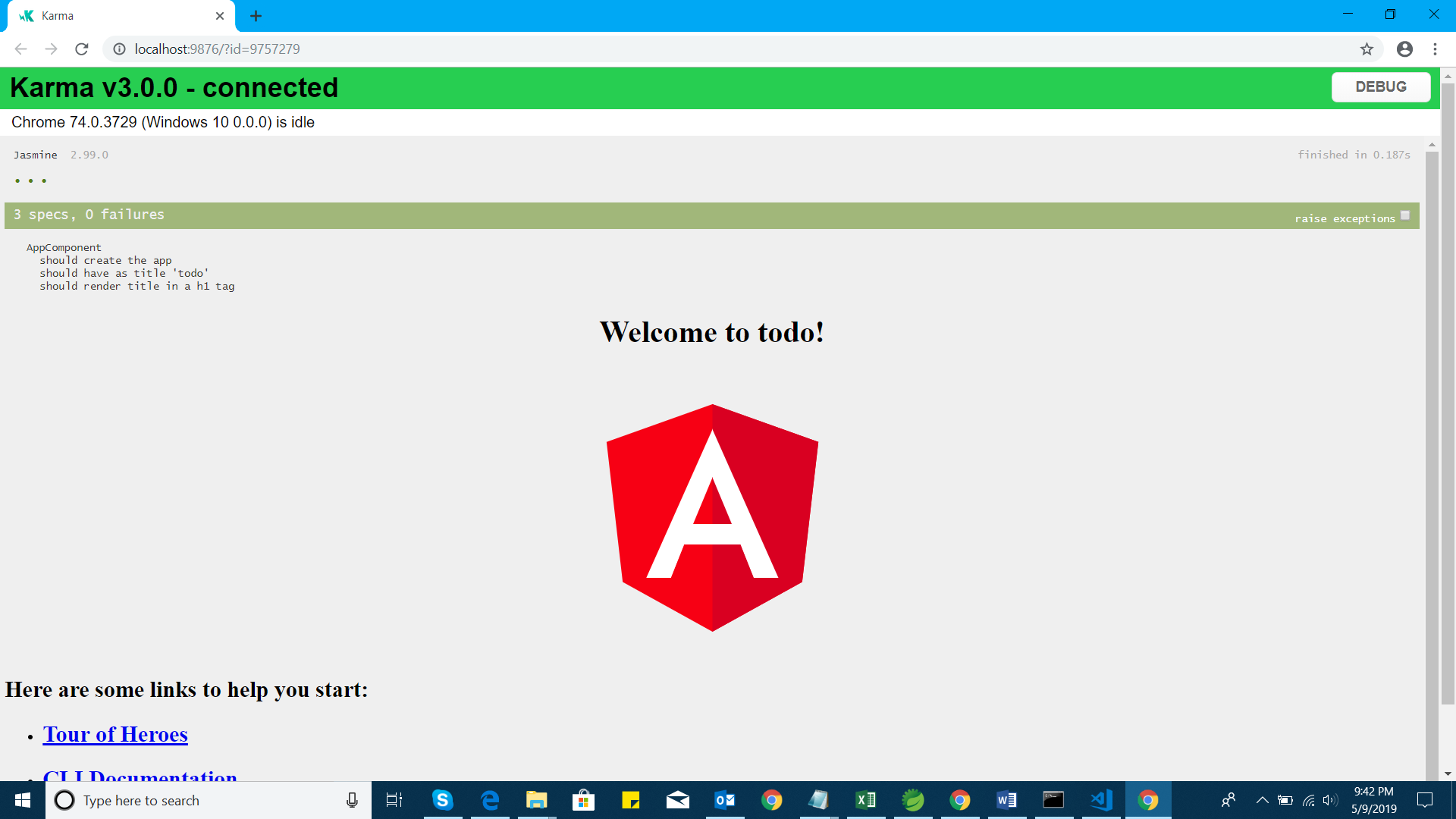
chunk {styles} styles.js, styles.js.map (styles) 16.2 kB [initial] [rendered]

chunk {vendor} vendor.js, vendor.js.map (vendor) 3.23 MB [initial] [rendered]

By copying dist folder than we can run application

To deploy into production: we run **ng dist**

**ng test** will run the test cases and it will redirect to the below page.



**karma.conf.js** is used to configure the test cases configurations.

**app.component.spec.ts** test case files

**ng e2e :** To perform the end to end testing.

**protractor framework** will help to selenium testing.

protractor.conf.js

**angular.json** file consist the information about the commands

**app folder** consists of all the components and modules.

**assets** folder consist images.

**environments folder** will be used to keep the environment specific files.

**tsconfig.json :** typescript converted to javascript which can understand the browser.

**package.json:** all the packages and tools are specified in this file.

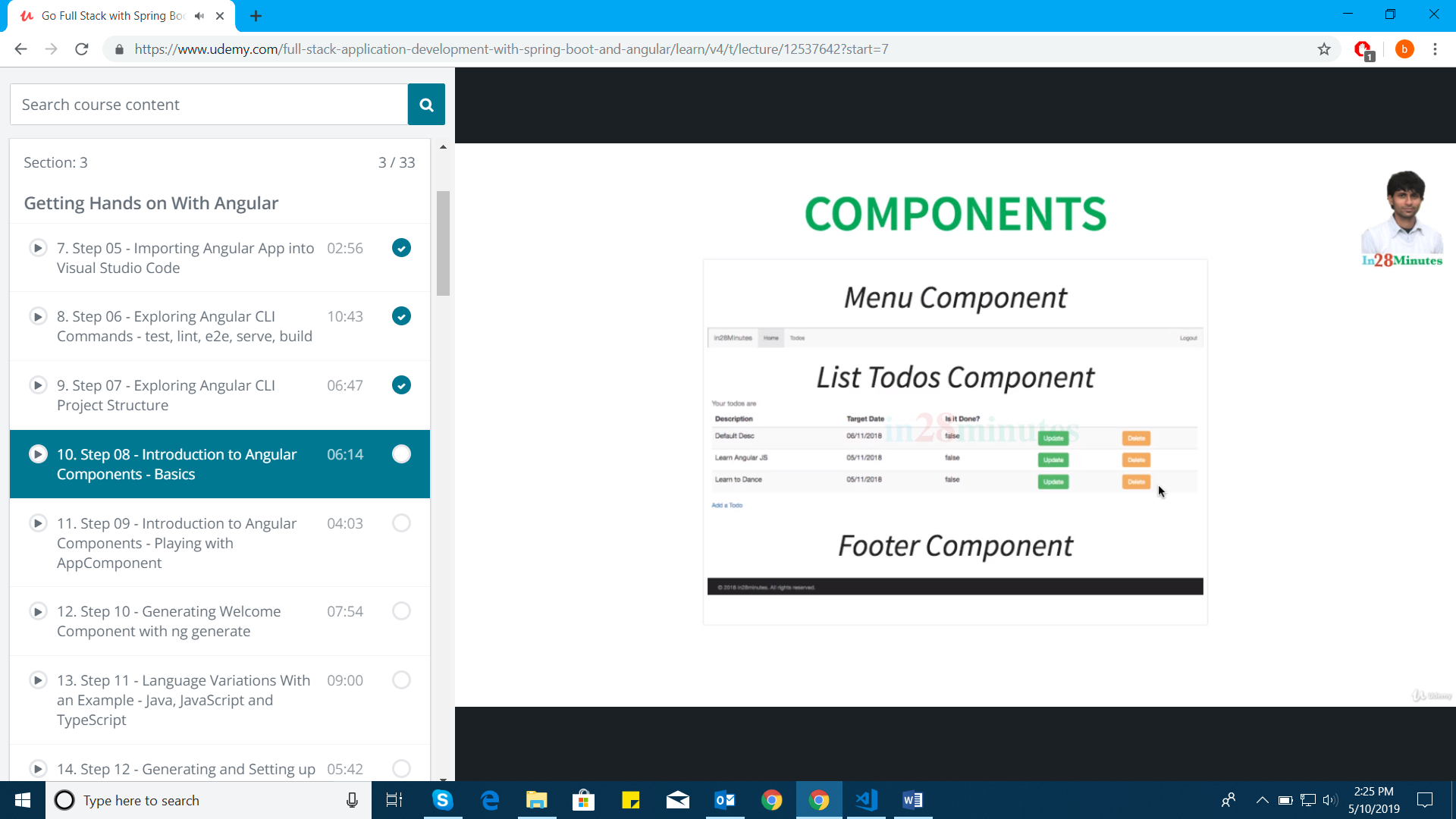
**node\_modules folder:** All the packages and tools are downloaded into this folder when we run ng new automatically executed npm install.

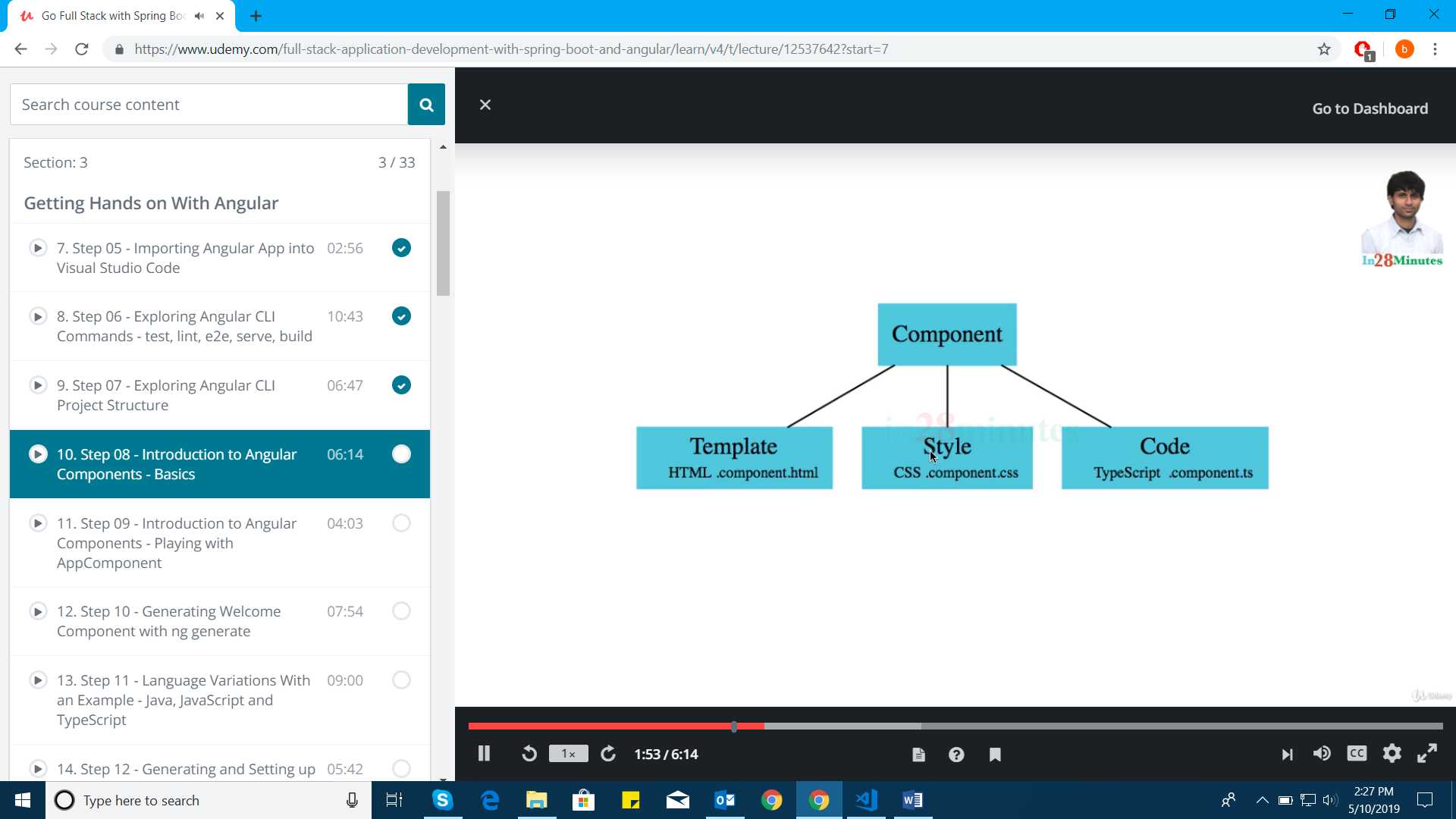
**main.ts** and **index.html** files are used to launch or bootstrap the angular applications.

**test.ts** is the starting point to run the application and is the root module that is included in the index.html.

**pollyfills.ts** file is used for browser incompatibility.

**style.css** : global style.whatever the css file is used across the application that can put in this file.





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

@Component({

selector: 'app-root',

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

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

})

export class AppComponent {

title = 'todo';

}

Selector ->**Index.html:** app-root

Entire page will be designed with the help of set of components.

Databinding can be handled with the help of interpolation in angular

Component consist of html or template, styling of the css and code to handle which is written in type script.

**@Component decorator** makes the Component class as angular component.

Creating the component:

**ng generate component** or **ng g c** welcome and once it’s created which we can’t rollback.

Every angular component (@Component) is associated with an angular module (@NgModule)

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

@Component({

selector: 'app-welcome',

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

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

})

export class WelcomeComponent implements OnInit {

constructor() { }

ngOnInit() {

}

}

Module represents one file which consists one class and function definition.

One file for one module.

export is equal to public in java

Interpolation is used for data binding **{{}}**

banana in a box to used for two way binding

Ex**: [(ngModel)]** = “username”;

**\*ngIf**

**Routing Module**

app-routing.module.ts

dependency Injection: constructor(router:Router) { }

activateroute to get the value passed to the corresponding component. Adding the routing parameter

constructor(private route:ActivatedRoute) { }

arrays and list of arrays

you can manage your todos <a routerLink="\todos" >here</a>

Coding Standards

**Pipes** areused to convert from one format to another format.

**Example**

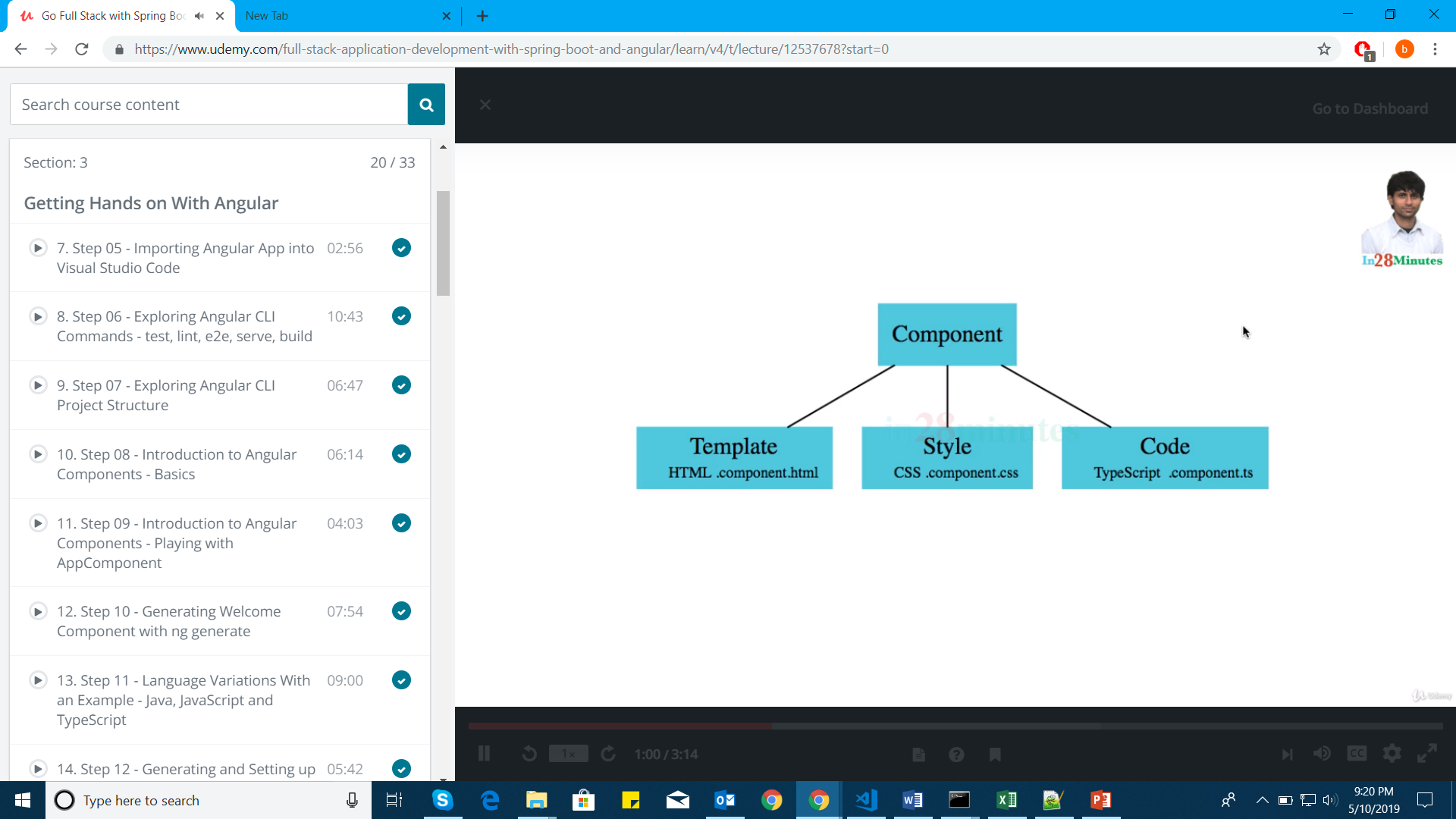
<td>{{todo.targetDate | date | uppercase | lowercase}}</td>

Ng starts with means are angular modules.

**@NgModule**

Bootstraping of Angular Application: Root module and Component

Index.html and main.ts



**unkpkg bootstrap**

style.css

**routerLink** doesn’t reload whole page. Href :loads entire page.

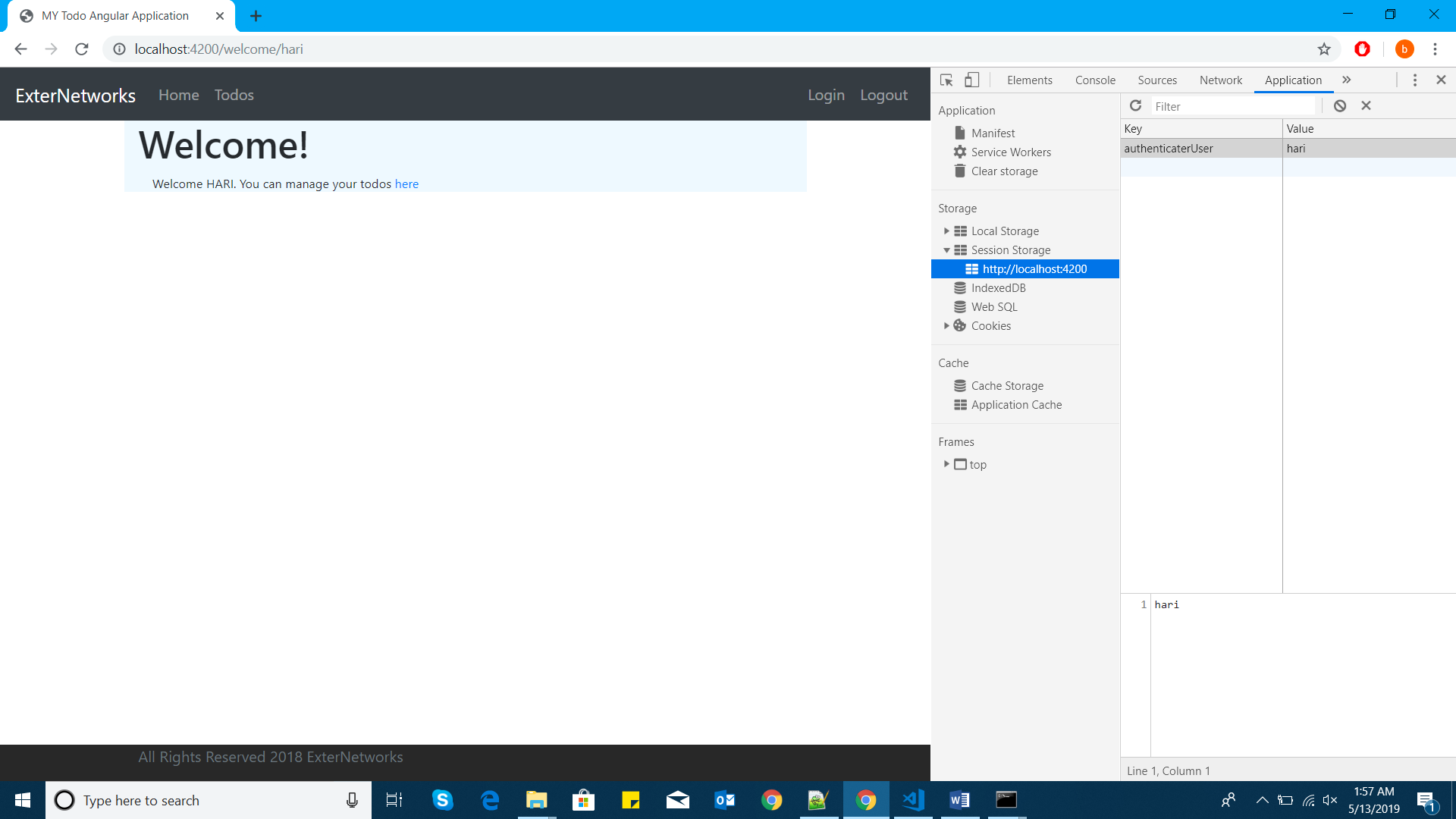
**Service**

Whenever we want to write any common logic across the components then we have to write the service.

ng generate service service/hardcodedAuthentication

**@Injectable** is a service

**sessionStorage**



**ngOnInit** works on page load page refresh

**RouteGaurdService**

ng generate service service/data/welcomeData