إعداد ا/ احمد صبري

ANGULAR II

01158421798

Angular

عبارة عن منصة لبناء تطبيقات الويب للجوال وسطح المكتب يستخدمها الملايين من المطورين ويعتبر Angular بيئة عمل لتصميم الواجهات

☜يعتمد انجولار تقنية SPA

Single **P**age **A**pplications

جميع الصفح التى تريد انشائها سوف سوف تكون فى صفحة واحدة هى الماستر وباقى الصفح التى نقوم بانشائها تسمى components اى اجزاء تابعة للصفحة الام (الماستر)

وعن طريق ال router نستطيع الانتقال الى components الذى نريده اى نستطيع تغير الصفحة والانتقال الى صفحة ولكن فى الحقيقة نحن ننتقل الى components المربوط بصفحة الماستر فكل الصفحات هى صفحة واحدة .

ما المميز فى anguler

تستطيع بناء تطبيقات سطح المكتب Mac و Windows و Linux

استخدام نفس الأساليب Angular التي تتعلمتها فى الويب

انشاء اكواد NativeScript

كتابة الاكواد وانشاء المكونات والمكتبات بسهولة عن طريق ال CLI

إنشاء عروض واجهة المستخدم بسرعة باستخدام بنية قالب بسيطة

Integrated development environment : IDE

بمعنه انه بيئة تطوير متكاملة مثل الفيجول استديو على سبيل المثال

الكود يتم تكملته تلقائيا بالاضافة الى انه يخبرك بالاخطاء فى حالة وجود اخطاء

الفائدة العملية من SPA

السرعة/ تفادى اخطاء اسماء الصفح /سهولة تجميع الملفات فى بيئة العمل /سهولة النشر / ترتيب جيد فى محركات البحث يمكن استخدامه فى مشروعات c# مثل asp.netcor or mvc

ونستطيع ان نلاحظ ذلك بوضوح فى المشاريع الكبيرة

إذا بدأت في التنقل ، فسترى أن الصفحة لا يتم إعادة تحميلها بالكامل - يتم إرسال البيانات الجديدة فقط عبر التوصيل أثناء تنقل المستخدم عبر التطبيق ويتم تمرير المتغيرات الى الصفحة الواحدة لذلك لسنا بحاجة الى الانتظار لحين الانتقال الى تحميل الصفحة الاخرى كما كان فى الماضى.

الادوات التى سنحتاجها فى البداية

ويندوز 10

فيجول استديو كود يفضل النسخة الحديثة

انجولار النسخة التى اعمل عليها هى 11 لكن اذا كانت النسخة المثبتة عندك 7 يمكنك البدء

اصدارnode v14.15.1

<u>اهم الاضافات التى يمكن اضافتها الى الفيجول استديو كود حاليا</u>

Community Material Theme

Material Theme

Material Theme Icons

HTML CSS Support

Auto Rename Tag

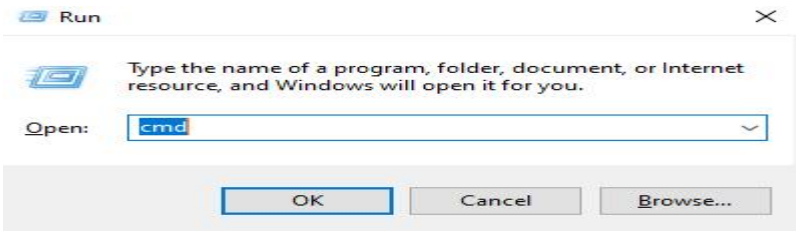
Css peek

قبل قل شئ عليك ان نضغط فى وقت واحد علي

علامة الويندوز + R

سوف يفتح لنا نافذة تدعى run

ثم نقوم بكتابة cmd ثم نضغط ok



سوف تظهر لنا شاشة سوداء تدعى **command line**

وهى التى نقوم من خلالها بكتابة الاكواد الخاصة بتثبت الادوات ونستطيع ايضا من خلالها التحقق من الاصدارات المثبتة

Stup anguler

```
npm install -g @angular/cli@11.0.5
```

OR

```
npm install -g @angular/cli@11.0.6
```

setup last version anguler

```
npm install --save-dev @angular/cli@latest
```

Uninstall

```
npm uninstall -g @angular/cli
```

```
npm uninstall --save-dev angular-cli
```

```
npm cache clean
```

install npm

```
npm  install  npm@latest -g
```

☜ نقوم بتحميل وتستطيب node v14.15.1 من الموقع الرسمى

للتحقق من الاصدارات

```
npm –v
```

6.14.8

```
node -v
```

v14.15.1

```
angular version
```

11.0.6

طريقة انشاء مشروع جديد

❶ الدخول الى مجلد المشروع

نذهب الى سطح المكتب ونقوم بانشاء فولدر جديد ونقوم بتسميته على سبيل المثال

blog-anguler

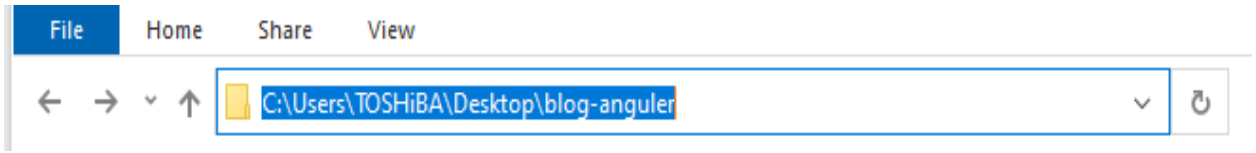
الان نريد الدخول الى الملف عن طريق موجه الاوامر يوجد طريقتين

الطريقة الاولى

علامة الويندوز + R ثم نكتب cmd ثم نضغط Enter

ثم نكتب cd ثمئ مسافة ثم مسار الملف

ولسهولة كتابة الكود نفتح الملف بالطريقة العادية open



❶ثم نقوم بنسخ مسار المجلد
❷ثم نذهب الى الشاشة السوداء cmd ونكتب cd ثم مسافة ثم نقوم بلصق المسار او نضغط على ctrl+v

✓ فيصبح شكل الكود النهائى هكذا

```
cd C:\Users\TOSHiBA\Desktop\blog-anguler
```

علما بان

TOSHiBA هو اسم الجهاز على الويندوز وهو متغير من جهاز لاخر
blog-anguler هو اسم المجلد الذى قمنا بانشائه

الطريقة الثانية

نفتح المجلد (نقوم بالدخول اليه) قم نضغط shift من الكيبورد ثم نضغط كليك لمين بالماوس ونختارopen powershell سيقوم هو تلقائيا بمعرفة المسار وكتابته

بدلاً من نسخ ولصق المسار ولكن انا افضل الطريقة الاولى

❷ بعد تعرف موجه الاوامر على معرفة مكان المشروع

نقوم بانشاء مشروع جديد داخل الفولدر ثم نكتب الكود التالى

ثم نضغط على enter

```
ng new myangulerpro
```

ng اختصار لكلمة anguler

New اى مشروع جديد

Myangulerpro هو اسم المشروع

☜ **سوف يقوم موجه الاوامر بالسؤال هل تريد تضمين ال routing**

نقوم بالضغط على حرف y ثم نضغط Enter

☜ **بعد ذلك سوف يسألنا عن نوع النمط الذى نريده نختار اول خيار css**

سوف يقوم تلقائى بفك الحزم وسوف يخبرك بانه قد تم اضافة ال package

☜نذهب الى المسار الذى قمنا بتحديده

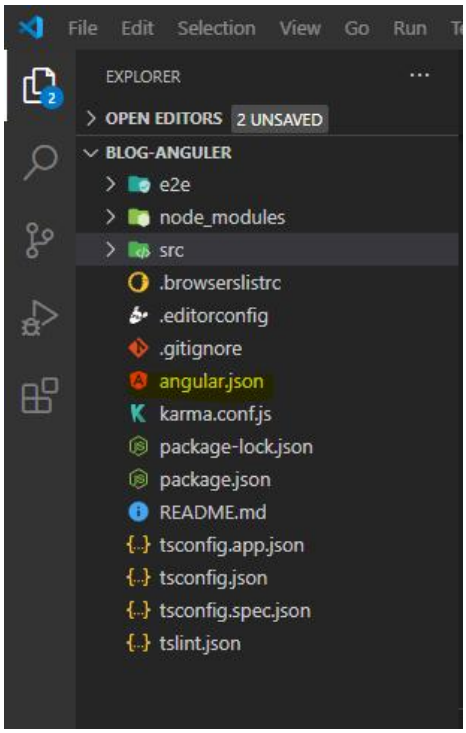
☜ثم نشغل الفولد الناتج عن عملية extract عن طريق فتح الفيجول استديو واختيار open file ثم نختار

او

عن طريق ال **power shell** نذهب الى المجلد ونقوم بفتحه كما تعلمنا shefit+ كليك لمين بالماوس واختيار power shell

بعد التعرف على المسار وتحديده نقوم بكتابة . cod

وسوف يقوم بفتح المشروع فى الفيجول استديو كود



ان لم تظهر قائمة ال
explorer
نضغط ctrl+b

هى مثل ال solution explorer

فى الفجيول استديو لتنظيم ملفات العمل

الملفات التى سنستعملها فى البداية

node_modules

ووفى هذا الفولدر يتم تستطيب المكتبات مثل البوت ستراب والجى كويرى

angular.json

فى هذا الملف يوجد كلاس لل css و لل script سنقوم فيه بوضع مسار ملفات المكتبات التى قمنا بتسطيبها حتى يتعرف عليها ال json

☜ وال json بصفة عامة هو صيغة لتنسيق و تمثيل البيانات يسمح بتخزين البيانات ومشاركتها بسهولة

☜ تمثيل البيانات بمعنى تجزئة البيانات فى كلاس الObject يحتوى على المتغيرات فى شكل مصفوفات Array لسهولة قراتها وهو يشبه الXML

✖ ولسنا فى حاجة الى التطرق الى لغة الجسون حاليا

favicon.ico

☜هى الايقونة التى تظهر فى شريط ال url لتعطى هوية للصفحة

assets

☜هو مثل المستدودع يسهل علينا التنظيم فى المشروع يمكن ان نضع فيه ملفات الصور

ويمكن ان نضع فيه ايضا ملف جافا سكربت خاص ولكن يفضل ان نعمل على ts

اختصار ل TypeScript وسنتعرف على هذا لاحقا

package.json

فى هذا الملف يمكن ان نعرف اخر تحديث اصدار يدعم المشروع الخاص بى اى يمكن معرفة ما احدث اصدار يمكن ان يعمل على البروجيكت بدون مشاكل

package-lock.json

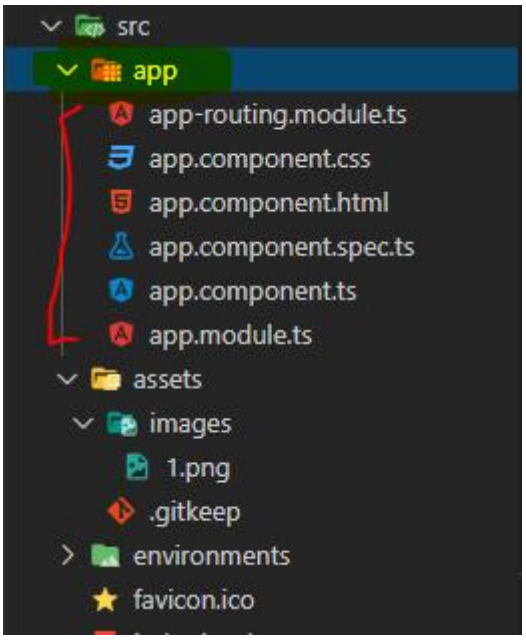
فى هذا الملف يوجد جميع المكتبات التى تم تستطيبها ورقم الاصدار الخاص بها

Scr

☜هذا الملف الذى يوجد به
الصفحات (الصفحة الواحدة التى تتكون من اجزاء تدعىcomponents بدل الصفحات فمفهوم الصفحات غير موجود فى ال anguler) ووجد به ايضا الصور

عند الضغط على ال scr سوف يظهر لنا الصفحة او الصفح وتحت كل صفحة مكوناتها

☜وبما اننا لم نقوم حتى الان باضافة صفحات اخرى فالصفحة الرئيسية للمشروع تدعى app وهى التى سوف تظهر لنا



فى هذه الصورة يظهر لنا صفحة app وتحتها مكوناتها

app-routing.module.ts

للتنقل بين الصفحات ويوجد فى مكونات صفحة ال app فقط لانها الجذر وهى فقط التى يوجد بها مفهوم ال routing

app.component.css : لكتابة ال css

app.component.html: لكتابة كود ال html

app.component.spec.ts: خاص بالاختبارات

app.component.ts:

- **لكتابى كود typescript بدلا من الجافا سكربت**
- **يحتوى ايضا على الدوال Methods والخصائص**

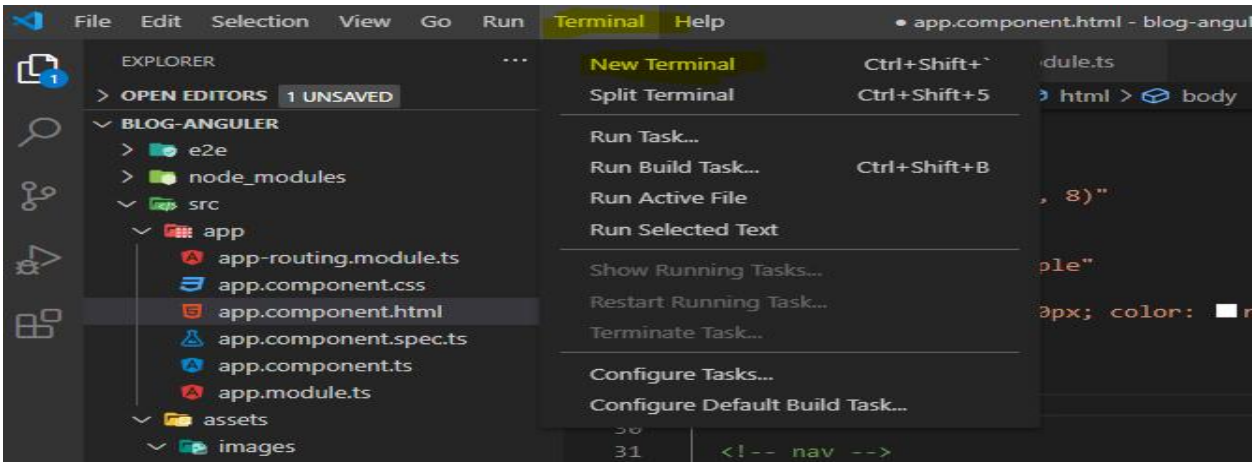
app.module.ts: مديول ts – سنتعرف عليه لاحقا فى الربط الثنائى

الان نجرب فتح المشروع

نقوم بتشغيل vs code اختصار لفيجول استديو كود

ثم نقوم بفتح terminal جديد عن طريق الضغط على (ذ+ctrl+shift)

او عن طريق شريط الاعلى واختيار new terminal



ثم نكتب السطر التالى

```
ng  serve -o
```

ng هى اختصار anguler

Serve هو السيرفر المحلى

o- لفتح اللينك فى المتصفح يمكن الاستغناء عن كتابته لكن سيتم كتابته يدويا فى المتصفح

انتظر الى ان يتم فك الضغط وسوف يقوم الفيجول استديو تلقائيا بفتح الموقع

هاااااااام جدا

لو ظهرت مشكلة فى اثناء التشغيل

"الاسكربت قيد الاستخدام" ندخل الى المسارات الاتية ونحذف الملفات المذكورة

```
C:\Users\sheko\AppData\Roaming\npm
```

نحذف ng.ps1

```
C:\Users\sheko\AppData\Roaming\npm-cache
```

نحذف كل الملفات التى بداخله

Anguler- ADD FRONR End

المرحلة الاولى

عمل setup للأدوات الازمة لتصميم الواجهات

اضافة الخطوط (awesome)

```
npm install --save font-awesome angular-font-awesome
```

اضافة الخطوط (awesome) – (الكود مقتبس من الموقع نفسه)

```
npm install --save @fortawesome/fontawesome-free
```

اضافة الخطوط (awesome) الىanguler

```
npm install font-awesome --save
```

اضافة مكتبة ال bootstrap

```
npm  install bootstrap --save
```

اضافة مكتبة ال bootstrap باصدار معين

```
npm  i  bootstrap@4.5.3
```

اضافة مكتبة الـ jquery

```
npm install jquery --save
```

اضافة مكتبة ال popper الخاصة بوضع العناصر وتجاوب الصفحة

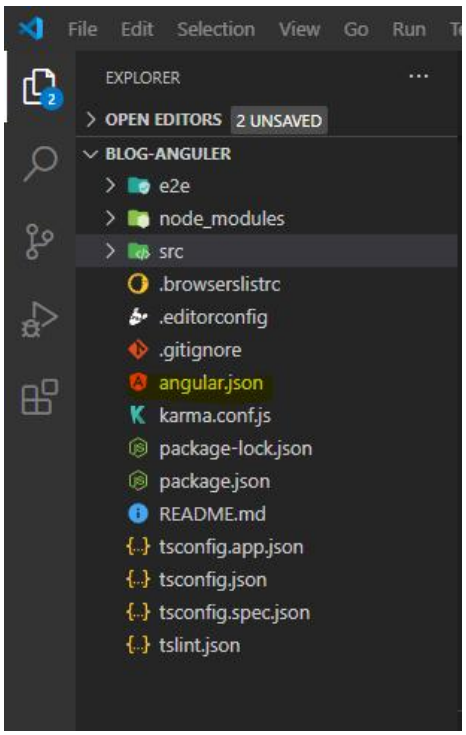
```
npm install popper.js --save
```

لاضافة الاكود باختصار فى كود واحد

```
npm  i  bootstrap@4.5.3  jquery --save
```

المرحلة الثانية

استدعاء المكتبات او بالاحرى عمل لينك للمكتبات التى تم عمل تستطيب لها فى ملف ال node_modules ونجلبها عن طريق ملف ال angular.json



نضيف مسار المكتبات التى قمنا بتثبتها (font-awesome ومكتبة bootstrap) الى كلا س ال style

ونضيف مسار المكتبات التى قمنا بتثبتها (jquery- ومكتبة popper و ملف bootstrap.min.js) الى كلا س ال scripts

```
],
            "styles": [
              "src/styles.css",
              "node_modules/bootstrap/dist/css/bootstrap.min.css",
              "node_modules/font-awesome/css/font-awesome.css"
            ],
            "scripts": [
              "node_modules/jquery/dist/jquery.min.js",
              "node_modules/popper.js/dist/umd/popper.min.js",
              "node_modules/bootstrap/dist/js/bootstrap.min.js"
                      ]
          },
```

الطريقة الثانية

طريقة استدعاء المكتبات من الموقع اى لسنا بحاجة الى الخطوات السابقة فى حالة استخدامنا الطريقة الثانية ولكن الطريقة الاولى من وجه نظرى افضل

❶ نفتح ملف html ثم نضغط shift+1

❷ نذهب الى موقع بوت ستراب

/https://getbootstrap.com/docs/4.0/getting-started/introduction

❸ نقوم بنسخ الاكواد الخاصة بالمكاتب "المكتوبة تحت التعليق" ونضع الاكواد فى صفحة ال **html** مع مراعاة وضع اكواد ال **css** تحت منطقة ال **head**

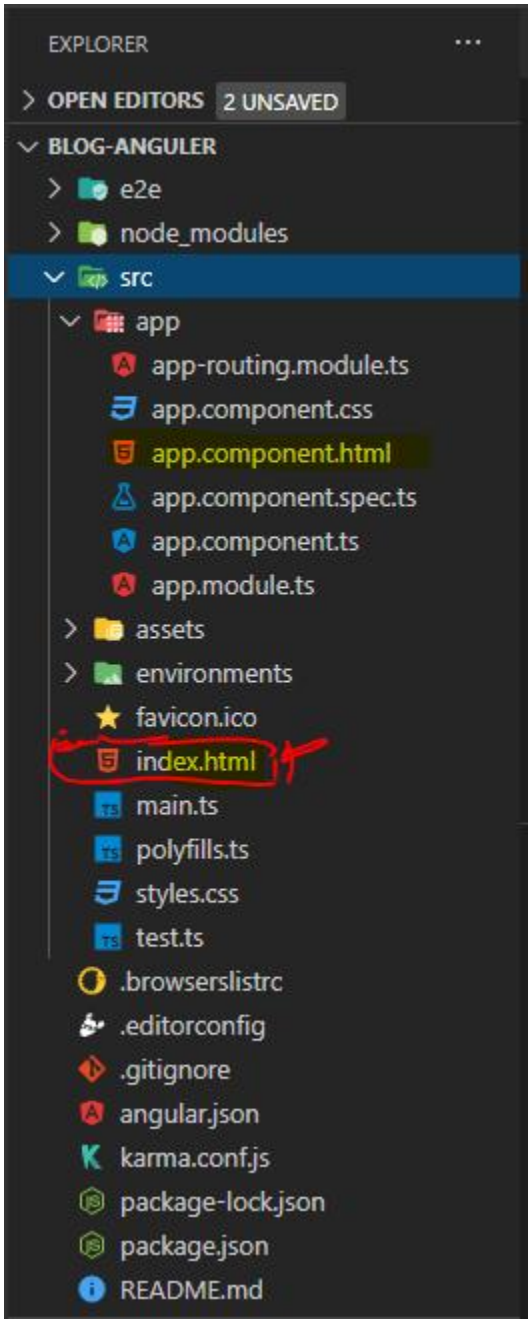
❹ نضع المكاتب الخاصة بالجافا سكربت (مثل الجى كويرى - **min.js**) فى منطقة ال **body**

السطر الملون باللون الاخضر هو تعليق للتذكر فقط لن يعمل اثناء تنفيذ الكود

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta name="viewport" content="width=device-width, initial-
scale=1, shrink-to-fit=no">
   <!-- Bootstrap CSS --> تعليق //
  <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.5.
3/dist/css/bootstrap.min.css" integrity="sha384-
TX8t27EcRE3e/ihU7zmQxVncDAy5uIKz4rEkgIXeMed4M0jlfIDPvg6uqKI2xXr2" crossori
gin="anonymous">
    <!-- Font Awesome -->تعليق//
  <link rel="stylesheet" type="text/css" href="https://maxcdn.bootstrapcdn
.com/font-awesome/4.7.0/css/font-awesome.min.css" />
  <title>Document</title>
</head>
<body>
  <style>
   <!-- Option 1: jQuery and Bootstrap Bundle (includes Popper) --
   <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity="sha384-
DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj" crossorigin="anonymous"
></script>
   <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/js/bootstrap.bundle.min.js" integr
ity="sha384-
ho+j7jyWK8fNQe+A12Hb8AhRq26LrZ/JpcUGGOn+Y7RsweNrtN/tE3MoK7ZeZDyx" crossorigin="anonymous"></script>
                                    جرب استخدام الخطوط للتاكد من تضمين المكتبة واستدعائها بنجاح
   <!--Font Awesome -->
  <i class="fa fa-home fa 3x"></i>
```

بدأ اول مشروع

anguler تشبه ال : mvc فى كونها نظام معمارى منفصل



يوجد صفحة index.html

هذه الصفحة تحتوى على صفحة ال

app.component.html

وصفحة ال

app.component.html

تحتوى على كل الصفح

بمعنى ان

index.html هى صفحة واحدة

تحتوى على كل الصفح

لان بداخلها صفحة ال index

طريقة تغير شكل الاسكرول

نذهب الى صفحة(index.html) التى تحتوى على الصفحة الرئسية

تحت وسم ال body نقوم بفتح وسم style ونكتب الكود التالى بداخله

```
<style>
  /*scrollbar*/
  ::-webkit-scrollbar {
    width: 5px; }
  ::-webkit-scrollbar-track {
    background: #f1f1f1;  }
  ::-webkit-scrollbar-thumb {
    background: #007771;  }
  ::-webkit-scrollbar-thumb:hover {
    background: #555;  }
</style>
```

Webkit : بريفكس لجعل الخاصية تعمل على المتصفحات القديمة

width : لعرض الاسكرول

ال hover كما نعلم لتغير اللون عن الضغط عليه

الان نريد ان نضع صفحة app.component.html داخل صفحة ال index.html

☜ نضغط على src ونختار app

ثم نختار app.component.ts ونضغط double click

```
@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
```

سوف نجد فى كلا س ال Component annotation
موجود ملفات ال html والcss و لهم selector هذا ال
selectorله اسم 'app-root'

هذا الاسم موضوع بين apostrophe

☜ ونلاحظ عند انتهاء كل سطر نوضع comma,

☜, تدعى comma وهى تشير الى ان هناك سطر سوف يكتب تحت هذا السطر

☜ وفى اخر سطر لا نضع comma بمعنى انه السطر الاخير

☜ الان ناخذ اسم ال selector كوبى

☜ثم نذهب الى صفحة ال index.html ومسح كل ما بداخلها

☜ونضغط shift+!

exclamation mark ! علامة تعجب

☜ونضغط enter

سوف يقوم بانشاء صفحة html فارغة

نذهب الى وسم ال body

☜ ونضع اسم ال selector فى وسم ونعلق هذا الوسم

```
<app-root></app-root>
```

هذه السطر يحتوى على صفحة ال html وال css اى اننا بهذا السطر اخذنا صفحة ال app.component.html بجمع ال component (html وال css) وقمنا بسحبها فى صفحة ال html

وهذا شبيه بالوراثة oop

✍اختصار ل object oriented programming

اى ان الصفحة الرئسية ترث من صفحة الثانية وهذه الاخيرة لو ورثت من صفحة ثالثة فان الصفحة الاولى سوف ترث الصفحة الثانية والثالثة

spiner

لتستطيب ال spiner يدويا

نذهب الى الموقع التالى

https://www.npmjs.com/package/ngx-spinner

نضع الكود التالى فى terminal

```
npm i ngx-spinner
```

بعد الانتهاء نضع الكود التالى فى الterminal

```
npm install ngx-spinner --save
```

ال i اختصار ل install

نذهب الى app.module.ts ونضع الكود التالى ☜

```
import { NgxSpinnerModule } from "ngx-spinner";//
```

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { NgxSpinnerModule } from "ngx-spinner";// ☜

import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
import { BrowserAnimationsModule } from '@angular/platform-
browser/animations';
```

وفى ال imports

☜نضيف السطر التالى

```
    NgxSpinnerModule,//اضافتها تم☜
```

```
imports: [
   BrowserModule,
   NgxSpinnerModule,//اضافتها تم☜
   AppRoutingModule, BrowserAnimationsModule
 ],
```

☜ نذهب الى app.component.ts ونضع الكود التالى

```
import { NgxSpinnerService } from "ngx-spinner";//
```

```
import { Component } from '@angular/core';
import { NgxSpinnerService } from "ngx-spinner";//اضافتها تم
```

وفى كلا س ال AppComponent - تحت ال title

نضيف الكود التالى☜

```
export class AppComponent {
  title = 'blog-anguler';
  constructor(private spinner: NgxSpinnerService) {}//اضافتها تم
  myimage1:string ="assets/images/1.png";
  ngOnInit() {  //
    /** spinner starts on init */
    this.spinner.show();

    setTimeout(() => {
      /** spinner ends after 3 seconds */
      this.spinner.hide();
    }, 1000);
  }//

}//الكبير الكلاس غلق
```

الان نذهب الى صفحة app.component.html

ونكتب تحت وسم ال body الكود التالى

ويمكن ان نعدل البرجراف او لون الاسكرين كما نشاء

```
  <ngx-spinner
  bdColor=" rgb(231, 153, 8)"
  size="medium"
  color="#fff"
  type="ball-scale-multiple"
>
  <p style="font-size: 60px; color: rgb(255, 255, 255)">Loading...</p>
</ngx-spinner>
```

نشغل السرفر المحلى ونجرب

```
ng serve -o
```

☜فى حالة ظهور خطأ متعلق بالانميشن فى الكونسول نقوم بالطريقة بالاتى

❶نذهب الى جوجل ونكتب Angular Material

❷من الشرط العلوى نختار Angular Material

❸ثم نختار Schematics

❹ننسخ الكود التالى

```
ng add @angular/material
```

❺ ونضعه فى ال terminal ثم نضغط enter

☜بعد الانتهاء سوف يسألنا عن الثيم الذى نريده سوف نجعله على الخيار الاول

☜بعد الانتهاء سيظهر سؤال تانى هل تريد تستطيب مكتبة ال hammer js ؟

نضغط y بمنى موافق

☜هيسالنى سؤال تانى هل تريد تستطيب animation for angular material ☜ نضغط y

طريقة عرض الصور فى صفحة ال anguler

قى فولدر المستودع

assets

☜نضغط كلك لمين ونختار newfolder ونقوم بتسميته على سبيل المثال images ثم نذهب الى المسار الذى قمنا فيه بوضع ملف الimages ونقوم بفتح الملف ونضع الصورة التى نريد عرضها

☜ونقوم بتسميه الصورة ونعرف الامتداد الخاص بها png او Jpj

☜يمكنك من الويندوز اظهار امتداد الملفات من foler option وتقوم بعرض امتدادات الملفات لتسهيل الامر

ونقوم بنسخ اسم الصورة + امتداد الصورة

نذهب الى المكون

app.component.ts

فى كلاس export AppComponent

```
export class AppComponent {
```

تحت العنوان title

نقوم بتعريف متغير نصى ونجعل قيمته تساوى المسار الذى يوجد به الصورة

```
myimage1:string ="assets/images/1.png";
```

ولعرضه فى صفحة ال html يوجد ثلاث طرق

الطريقة الاولى نذهب الى app.component.html

وهذه الطريقة تدعى attribute binding

```
<img class="float_left" [src]=myimage1>
```

هذا الكلاس "float_left"=img class افترضى يمكن ازالته

وسم الصورة img

Src بين square brackets تشير الى مصدر الصورة =

☜myimage1 هى قيمة المتغير النصى string التى تساوى مسار الصورة

الخلاصة قمنا بانشاء متغير نصى فى مكون ts وجلنا قيمة المتغير تساوى مسار الصورة ثم قمنا بانشاء وسم img فى ال html وجعلنا قيمته تساوى المتغير .

الطريقة الثانية لتمرير الصورة من ال ts الى ال html

وتسمى هذه الطريقة ب string interpolation

نضع اسم المتغير بين curly brackets ونجعل ال src بدون [ ]

```
<img class="float_left" src={{myimage1}}>
```

الطريقة الثالثة نكتب= bind-src ثم اسم المتغير بدون اقواس

```
<img class="float_left" bind-src=myimage1>
```

طريقة تغير شكل الايقون التى تظهر فى اعلى الصفحة

☜ نذهب الى مسار favicon.ico

وهو موجود فى ملف src

يمكن تحميل اى ايقونة بصيغة ico واستبدالها بنفس الاسم favicon وسوف تظهر بدل من شعار anguler الذى تم انشائه افتراضيا

☜ طريقة ايقاف السيرفر

نضغط بالماوس على اى جزى داخل ال terminal ثم نضغط ctrl+c

ثم يظهر لنا رسالة / هل تريد ايقاف السرفر نضغط y

ولتفريع الكلام الموجود دخل ال terminal نكتب cls ثم enter

☜ <u>طريقة انشاء مكون جديد</u>

Ng g c home

Ng اختصار anguler

g اختصار Generate

c اختصار ل component

books اختيارى يمكن تغيره الى الاسم الذى تريده

الان يمكنا الذهاب الى books

☜ثم الذهاب الى صفحة ال books.component.html وكتابة اى شى مثلا نكتب برجراف او span ونضع بداخله كلام

وفى ملف ال books.component.css نقوم بعمل التسنيقات

☜ثم نضغط على books.component.ts

☜ونسخ اسم ال selector

☜ندخل الى صفحة الapp.component.html(الجذر)

☜ونقوم بعمل وسم app-books ثم نغلق التاج

مثال

```
<app- books ></app- books >
```

عند تشغيل السرفر سنلاحظ ان صفحة الكتب بالكلام الذى وضعناه بداخلها والتنسيقات اتلى قمنا بها قد تم تضمينها الى صفحة ال app وصفحة ال app تم عرضها فى ال index.html

routing

بعد ان تعلمنا كيفية انشاء مكون جديد (صفحة جديدة)

الان نريد ان ننشأ اكثر من صفحة ونستعرض على كل قسم على حدة فى نفس الصفحة بمعنى كل قسم على توحده

التاكد من ان مفهوم ال routes قد تم تضمينه فى المشروع

نذهب الى المكون لنتتحق من ان مفهوم الروت قد تم تضمينه app-routing.module.ts

```
const routes: Routes = []
```

سنجد السطر التالى لاننا فعلنا الrouting اثناء التسطيب عندما قام بالسؤال هل تريد تضمين مفهوم التنقل ففلنا له y اى نعم

وفى مكون ال app.module.ts سوف ينشأ تلقائى هذا السطر

```
import { AppRoutingModule } from './app-routing.module';
```

وفى ال import

```
imports: [
   BrowserModule,
    NgxSpinnerModule,// spiner ☜ لن تجد هذا السطر اذا لم تكن قد قمت بتسطيب ال
   AppRoutingModule, BrowserAnimationsModule
 ],
```

سنجد السطر التالى AppRoutingModule

اما عن BrowserAnimationsModule فلن تجده لانه خاص بال spiner ان لم تقم بتستطيبه فلن تجده ما يهمنا الان هو ال AppRout

فى حالة النسيان او الخطأ يمكن ان نضيف السطر يدويا وتعدل الاعدادات يدويا ولكن الافضل ان ناخذ الحذر عن تنسيط المشروع لاول مرة لتوفير الوقت والجهد

كل ما سبق يمكن الاستغناء عنه اذا كنت تعلم انك قد فعلت مفهوم الروت اثناء التسطيب

<u>الان نذهب الى</u>

app.component.html

نقوم بعمل روابط وليكن مثلا نريد عمل اربع صفح للتنقل بينهم

يمكن كتابتهم بالطريقة العادية ولكن هناك طريقة تشهل علينا كتابة جميع الروابط بسطر واحد ليفر علينا الوقت

```
ul>Li*5>a[href="#"]{$}
```

ul: the unordered list

li قائمة عناصر مرتبة

4a* لاننا نريد انشاء اربع روابط

"#" شباك بين علامة quotation mark الشباك يمكن استبدالة بالينك مثل جوجل مثلا للانتقال اليه

{$} علامة الدولار بين curly brackets يشير الى رقم الاندكس

الان سنضع مكان ارقام الاندكس 1,2,3,4 سنضع اسم الروابط كما نريدها ان تظهر فى المتصفح

الان نذهب الى

app-routing.module.ts

ونقوم بفتح الاقواس المربعة كما هو موضح

```
const routes: Routes = [


]
```

نضع المسار فارغ لان صفحة ال app هى الصفحة الرئسية ولا تظهر فى الget فى ال url / ونجعل التوجيه ينقلنا الى نفس الصفحة redirectTo

```
  {
    path:'',
    redirectTo:'',
   pathMatch :'full'
  },
```

ثم ننقل نفس الكود ونعدل فيه فنجعل المسار يساوى اسم المكون الذى نريد الانتقال اليه

وال component يكون اسم المكون الذى نريد الانتقال اليه ثم component

```
{
    path:'about',
   component :AboutComponent,
   pathMatch :'full'
  },
```

الكود التالى

فى حالة وضع– مسار غير موجود سوف ينقلنا الى المكون الذى قمنا بانشائه والذى يحتوى على برجراف يفيد ان الصفحة غير موجودة

```
{
 path:'**',
 component :PageNotfoundComponent
},
```

الان نقوم بانشاء مكون جديد ونعطيه اسم وليكن PageNotfound

```
Ng g c PageNotfound
```

ونقوم بتكرار هذه العملية مع تغير اسم المكون و 'pathMatch :'full تظل موجودة بنفس الاسم فى جميع ال component التى تريد الانتقال اليها

☜ ندخل الى صفحة ال index.html

ونضع تحت العنوان السطر التالى

```
<base href="/">
```

☜ وفى صفحة(app.component.html)

تحت الليبات الاربعة التى قمنا بانشائها بعد اغلاق ال ul نضع السطر التالى

نضع السطر التالى لتشغيل مفهوم ال router

```
<router-outlet></router-outlet>لضبط توجيه الصفحات
```

النقطة الثانية نخبره ان href هو رابط داخلى للتنقل الى ال component

```
<ul>
  <Li><a href="#">1</a></Li>
  <Li><a href="#">2</a></Li>
  <Li><a href="#">3</a></Li>
  <Li><a href="#">4</a></Li>
</ul>
```

☜ فنقوم باستبدال كلمة href ب routerLink

ومكان الاندكس لن نضع url
بل سنقوم بوضع اسم ال component (about) التى وضعناها فى مكون ال app-routing.module.ts
فى المسار , path:'about'

ونفس الامر بالنسبة لباقى اللينكات

Property Binding

طريقة تمرير قيمة من ال ts الى ال html

كما فعلنا مع الصورة ولكن سقوم بمثال اخر بتمرير القيم
نذهب الى app.component.ts

```
export class AppComponent {
```

نضع تحته متغير ونعطى له قيمة

```
  name:string =’ahmedsabry’;
```

✍ملحوظة لو حذفنا ال string: سوف يتعرف تلقائيا من نوع المتغير

☜الان سنقوم بعملية تسمى " string interpolation"

☜ندخل الى مكون ال app.component.html

ونكتب على سبيل المثال وسم عنوان رئيسى h1

وفى داخله نضع القيمة بداخل الاقواس المتعرجة curly
brackets {{ }}

فيصبح الشكل النهائى

```
<h1> {{name}} </h1>
```

☜والنتيجة التى سوف تعرض على المتصفح هى قيمة ال name وليس كلمة name

☑مثال اخر للتوضيح

نريد ان نعمل كومبو بوكس فى ال html ونجعله ياخذ القيم التى بداخله من ال ts

نذهب الى app.component.ts

وننزل الى كلا س التصدير ان صح التعبير لل AppComponent

```
export class AppComponent {
```

نعرف متغير جديد من نوع مصفوفة بين علامات مربعة

```
  colors=['Red','Green','Purple','Yellow','White','Black']
```

لاحظ اننا لم نعرف نوع المتغير الذى أسميناه على سبيل المثال colors وبين علامات تنصيص مفردة وضعنا الاسماء التى نريدها وبعد كل اسم وضعنا comma واختصارها حرف (و) من الكيبورد باللغة الانجليزية وفى اخر السطر نضع semicolon مثل اكواد السى شارب واختصارها حرف ال ك باللغة الانجليزية

الان نذهب الى صفحة ال app اى المكون app.component.html

ونقوم بعمل وسم select بداخله وسم option

```
  <select>
   <option *ngFor="let color of colors">
     {{color}}
   </option>
 </select>
```

تحليل الكود

***ngFor : ng اختصار ل anguler / و for مثل السى شارب تكرار عشان يلف على كل القيم**

let color of color : هو متغير فارغ

let color of colors : colors : هو المتعير فى ال ts الذى قمنا بوضع مصفوفة من الالوان بداخله

وهذا السطر مع جملة التكرار سوف يجعل المتغير الفاضى ياخذ جميع القيم of من المتغير colors الموجود فى ال ts

الان نعرض المتغير الفارغ الذى تم تعبئته من المتغير الموجود فى ال ts

نضغط shift+ج باللغة الانجليزية مرتين لعمل قوسين متعرجين ونضع بداخلهم اسم المتغير الذى تم تعبئته

نفتح السيرفر ونشاهد العملية لنتاكد

Event binding

هذا التعبير يعبر على العملية التى نمرر فيها كود من html ال ts ولتوضيح الفكرة ناخذ مثلا حدث click

☜نذهب الى الصفحة الرئيسية app.component.html

ونكتب بداخلها حقل ادخال من نوع زرار قيمتها كالتالى ثم نضع حدث مثل clik بين قوسين = ونعطى له اسم بين " "

ونفتح قوسين وكما هو معلوم فان وسم ال input لا يغلق

```
<input type="button" value="click here" (click)="myfunction()" >
```

هذه الفاكشن يجب سوف نشتغل عليها فى ال ts

☜نفتح المكون app.component.ts

```
export class AppComponent {
```

☜ثم نضع هذه الفاكشن ثم نفتح قوسين متعرجين وفى المنتصف نقوم بكتابة السطر التالتى مثل الجافا سكربت ولا ننسى ال; لنخبره ان السطر قد تم كتابته

```
myfunction(){
    alert('click event');
  }
```

☜نفتح السيرفر ونجرب نصغط على الزرار وسوف نرى ال alert تعمل بنجاح وهذا يشبه الى حد كبير الجافا سكربت

☜ الان تعلمنا تمرير قيم من ال ts الى الhtml بطريقتين

وتعلمنا طريقة تمرير حدث من ال html الى ال ts عن طريق حدث يساوى اكشن ووضعنا الاكشن فى ال ts وجعلناه يقوم بعمل alert "اظهار اشعار او رسالة "

امثلة اخرى

سنقوم بعمل متغيرين(age/name) فى ال ts ونعطى لهم قيم

```
export class AppComponent {
  title = 'blog-anguler';
 name='ahmed';
```

نذهب الى مكون ال app.component.html

نقوم بفتح وسم حقل ادخال ونحدد نوعه وقيمته

```
<input type="text" value="name">
```

الان نريد ان نجعل قمة حقل الادخال تاخذ من الproperty

name='ahmedsabry'; لو تركنا الكود كما هو مكتوب سوف تكون القيمة name وليس ahmedsabry

الحل :

نضع كلمة value بين قوسين مربعين [ ] ليفهم ان القيمة لا تساوى النص المكتوب داخل ال quotation mark ولكن تساوى القيمة الموجودة فى ال ts

```
<input type="text" [value]="name">
```

وهذه الطريقة قد شرحناها مسبقا فى التعامل مع الصور

تحت مفهوم attribute binding

وسنقوم ايضا بعمل عنوان h1 للاسم والسن فى مكون ال

نحضر الاول (المتغيرات) يمكن ان نصطلح مؤقتا على لفظ برامتر لكن هذا اللفظ غير صحيح فهذا يدعى متغير لكن بما اننا سوف نقوم بتمريره يمكن ان نطلق عليه مصطلح برامتر لكن هذا اللفظ محدث من عندى لتقريب المعنى ليس اكثر

```
export class AppComponent {
  title = 'blog-anguler';
 name='ahmed';
 age=24;
```

☜وفى مكون الapp.component.html

☜نضع h1 ونمرر القيم كما تعلمنا بطريقة ال string interpolation

```
<h1> Hello, {{name}} , age:{{age}} </h1>
```

2way binding or ng model

☜الان قيمة ال name تم تمريرها الى ال html من ال ts فى حقل الادخال وفى ال h1

☜الان سنربط قيم ts التى تم تمريرها الى ال html ببعضها بحيث ان اى تعديل فى اى منها سوف ياثر على الاخر تلقائيا ويتم تعديله مثل الاخر

☜بمعنى لو غيرنا فى حقل الادخال قيمة ال name سوف تتعير قيمة الname فى ال h1

☜ بمعنى اخر ان القيمة التى تم تمريرها من ال ts ال html سوف نعدلها من ال html وبالتالى سوف تاثر على الh1 الذى يستمد قيمته من نفس ال ts التى تم تعديلها بواسطة html من حقل الادخال

ملحوظة هامة

لن نستطيع حاليا عمل ال 2way binding لان هذا المفهوم يجب ان نضيفه اولا الى المديول

الطريقة نذهب الي مكون ال

app.module.ts

وبما اننا دخلنا الى هذا المكون سنتحدث عنه باختصار

```
  declarations: [
```

فى ال declarations سنجد جميع المكونات التى قمنا بانشائها

على سبيل المثال الصفحة (المكونBooks) الذى قمنا بانشاءه سنجده مكتوب كالاتى, BooksComponent وجميع المكونات التى قمنا بانشائها سنجدها مرتبة تحت بعض هكذا

```
@NgModule({
  declarations: [
    AppComponent,
    BooksComponent,
    VideoComponent,
    ContactmeComponent,
    AboutComponent,
    PageNotfoundComponent//url مكون يظهر فى حالة وجود عنوان غير متوقع فى ال
  ],
```

☜وفى ال imports نسجل المديول

```
imports: [
    BrowserModule,
    NgxSpinnerModule,//اضافتها تم
    AppRoutingModule, BrowserAnimationsModule
  ],
```

الان نرجع لموضوعنا نريد تضمين مفهوم ال 2way binding

نذهب الى مكون ال app.module.ts فوق ال @NgModule ({

فى اعلى الصفحة فى منطقة الاستيراد

☜ونستورد ال FormsModule من المسار التالى كما هو موضح

```
import {FormsModule} from '@angular/forms';//اضافتها تم
```

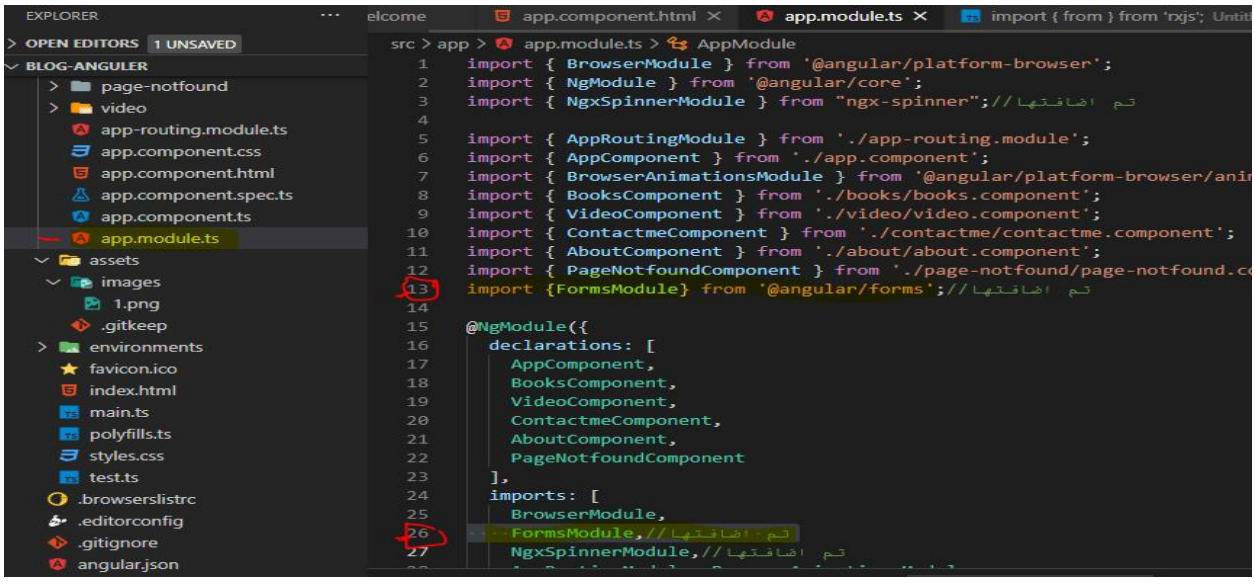
وننسخ الاسم المكتوب داخل الاقواس المتعرجة FormsModule

Ctrl+c

وفى ال imports نقوم بلصق الاسم الذى قمنا بنسخه ctrl+v

```
  imports: [

    FormsModule,//اضافتها تم
```

صورة توضحية



نعود الى مكون ال app.component.html

ونكتب بدلا هذا السطر

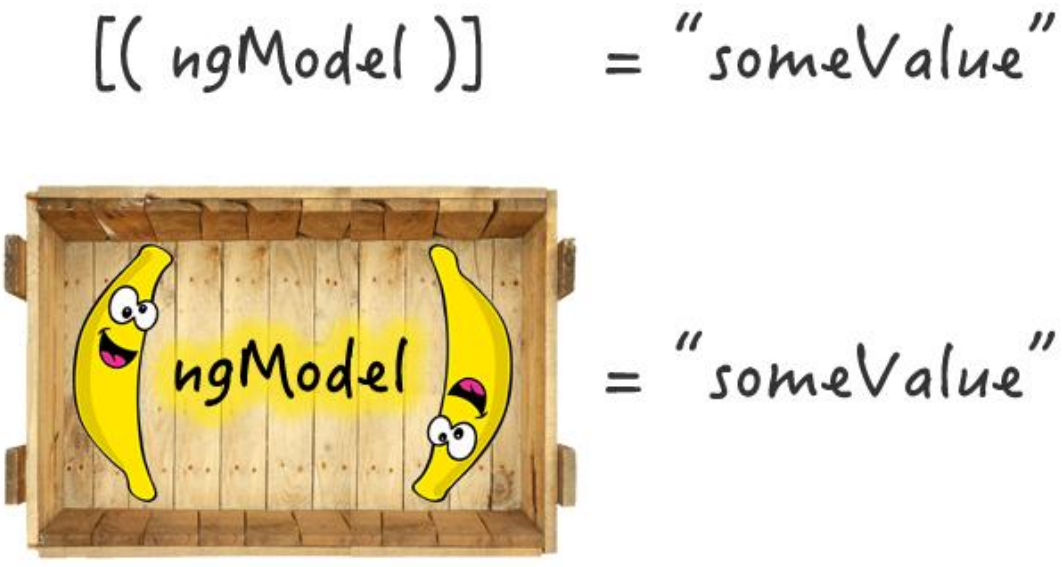
```
<input type="text" [value]="name">
```

نكتب السطر التالى☜

```
<input type="text" [(ngModel)]="name">
```

اى قمنا باستبدال كلمة value بـngModel مع مراعاة حالة الاحرف فحرف ال M كبير ثم وضعنا هذا الكلمة بين قوسين ()

وهذا الشكل فى ال ANGULER [()] يدعى banana in the box



الان عند تغير قيمة حقل الادخال name المربوط بالمتغير name فى ال ts سوف يتم تغير ال h1 المربوط بالمتغير ts

لاننا بعد ان قمنا باستيراد ال FormsModule تم تضمين ال ngModel وبالتالى يمكنا بواسطة ال html ان نعدل فى قيمة ال ts وبالتالى اى جزء فى ال html مربوط بنفس قيمة المتغير الموجود فى الts سوف يتم تغيره الى نفس القيمة التى تم تمريرها من ال html

الوحدات Module

عند انشاء مشروع جديد نجد فى المكونات التى يتم انشاءها مكون يدعى app.module.ts وهو المديول الافتراضى

فما فائدة ال module

يسمح لنا بتجميع جميع المكونات والعناصر التى نستعملها فى ال anguler

✓ فى اعلى الصفحة نجد انه قام بعمل import تلقائى

لجميع المكونات التى قمت بانشائها ولو نتذكر قمنا بفتحه مسبقاعندما قمنا بتضمين ال spiner و مفهوم الربط الثنائى – عن طريق FormsModule

فى اخر سطر اسفل الصفحة نجد انه يستعمل على انه كلاس يمكن استدعاءه فى اى مكان اخر

```
export class AppModule { }
```

هذا السطر يقوم بجلب ال NgModule
ليعرف ال{})@NgModule

```
import { NgModule } from '@angular/core';
```

```
@NgModule({
  declarations: [
```

وفى الdeclarations يتم التعرف على جميع المكونات التى قمنا بانشاؤها

وفى هذا السطر جلبنا ال ***AppRoutingModule***

```
import { AppRoutingModule } from './app-routing.module';
```

وهو خاص بالتنقل بين المكونات وقد تم شرح مفهوم ال Routing

```
import {FormsModule} from '@angular/forms';//
```

وهو الذى يمكنا من استيراد المديول الخاص بمفهوم ال 2way binding وقد تم شرحه

وفى ال imports نجد انه قد تم استيراد ال BrowserModule تلقائى وهو بمثابة البنية التحتية وهو ضرورى لتشغيل المشروع

اما الباقى فنضيفه نحن وهى الاضافات التى قمنا باستيرادها وقمنا بتعريف المسار الخاص بها للتعرف عليها مثل FormsModule/ NgxSpinnerModule

```
imports: [
   BrowserModule,
   FormsModule,//اضافتها تم
   NgxSpinnerModule,//اضافتها تم
   AppRoutingModule, BrowserAnimationsModule
 ],
```

```
    AppRoutingModule, BrowserAnimationsModule
```

اما السطر الاخير فقط تم انشاءه تلقائى عندما قمنا بتستيطب ال NgxSpinner

ال bootstrap يستقبل مصفوفة بداخلها ال AppComponent

```
  providers: [],
  bootstrap: [AppComponent]
})
```

لو رجعنا لل declarations سنجد المكون AppComponent

هو الجذر واسفله يوجد باقى المكونات

فى البوت ستراب عند جلب الAppComponent فهو يقوم بجلب الجذر الذى بداخله جميع المكونات الاخرى (الصفحات كما نراها) لكن علينا ان نعلم انها مكونات وليست صفح

☜ والسطر الاخير هو تصدير لكلا س ال AppModule

```
export class AppModule { }
```

لكى نستطيع جلبه فى اى مكان اخر

لو ذهبنا الى مكون ال main.ts

main.ts

سنجد انه عند تشغيل المشورع فانه يقوم باستيراد ال AppModule

```
import { AppModule } from './app/app.module';
```

☜ثم يستدعى هذا المديول #

```
platformBrowserDynamic().bootstrapModule(AppModule)
```

وهذا المديول الذى تم جلبه كما ذكرنا كان يحتوى على

```
  bootstrap: [AppComponent]
```

اى اننا جلبنا الجذر بجميع مكوناته الاخرى

التوجيهات Directives

يمكن ان نعرفها بانها تراكيب او مفاهيم او keywords عندما نقوم باستدعاءها فى element تقوم بتنفيذ اوامر اخرى

ال element نقصد به المحتوى الموجود داخل الوسم

✌هناك نوعين من ال Directives

❶Structural Directives

❷Non Structural Directives

Structural Directives

*ngFor مثل for فى السى شارب

*ngiF مثل if فى السى شارب (الجملة الشرطية)

*ngswith مثل ال swith فى السى شارب (الجمل الشرطية)

Non Structural Directives

*ngclass: يمكننا من اتشاء او حذف كلاسات

*ngstyle: لاضافة تنسيقات بشكل مباشر

*ngmodel : قمنا بشرحها مسبقا فى الربط الثنائى

تطبيق: (الفكرة العامة)

سنقوم بعمل زرار تسجيل دخول عند الضغط عليه يظهر رسالة ترحيبية وعندما نضغط عليه مرة اخرى يتحول الى تسجيل خروج يتغير نص اللاسالة الى رسالة اخرى تفيد بان على المستخدم اعادة تسجيل الدخول

<u>الان لدينا اربعة متغيرات</u>

المتغير الاول و الثانى وهو label ال button

والمتغير الثالث والرابع هو الرسالتين التى تتغير قيمتهما بتغير حالة الlabel / تسجيل دخول / ام تسجيل خروج

الفكرة البرمجية " على المتغير الاول والثانى "

فى مكون ال app.component.ts

عرفنا متغير منطقى من نوع boolean ياخذ قيمة نعم او لا
عرفنا متغير نصى له قيمة افتراضية على سبيل المثال "تسجيل دخول"

فى مكون ال app.component.html

☜وفى ال html قمنا بعمل زرار قيمته تاخذ من المتغير النصى الموجود فى ال ts وعملنا له حدث مع ال (click) واعطينا لهذا الحدث اسم
☜نعود الى ال ts

☜ونعمل method لحدث الclick

☜ونكتب بداخله الاتى

☜نعكس اشارة المتغير المنطقى من نوع Boolean فى كل ضغطة على الزرار (مع كل click)اذا كان نعم يصبح لا والعكس صحيح

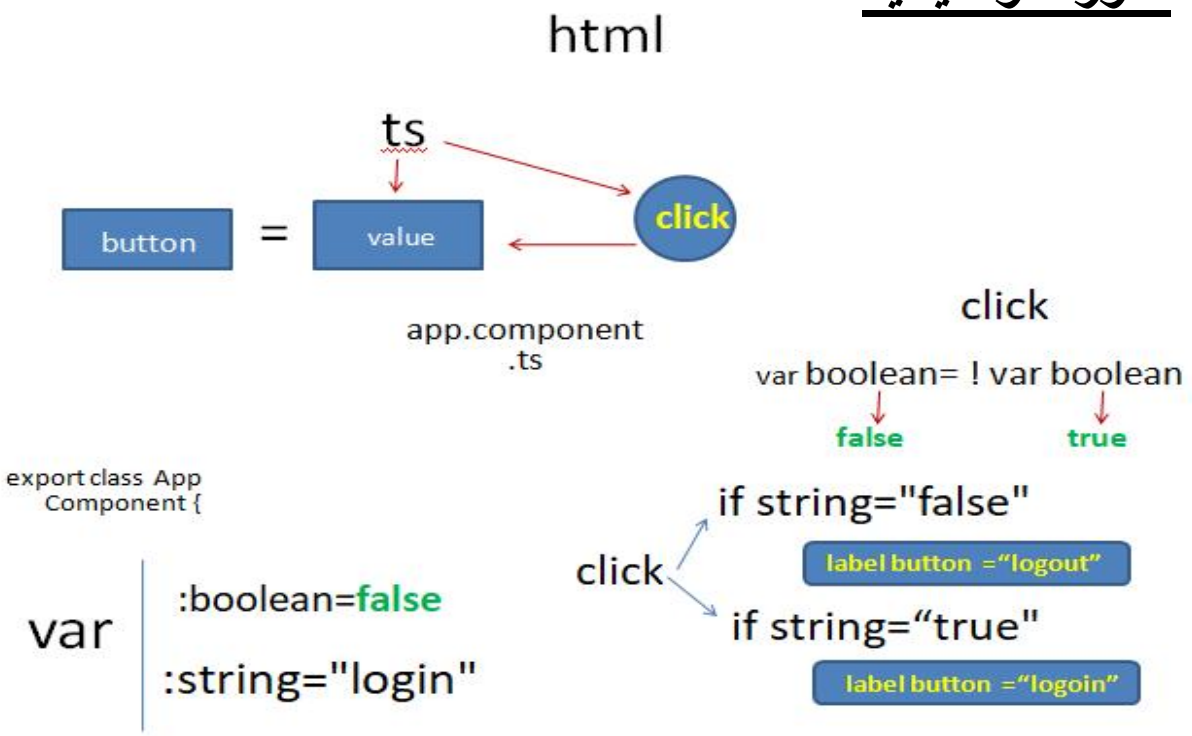
☜ثم نعطى رسالة فى حالة كان المتغير النص يساوى قيمة المتغير المنطقى الافتراضية

☜ ونعطى رسالة اخرى فى حالة اذا كان المتغير النصى يساوى القيمة المنطقية المعكوس اشارتها عن طريق الmthod التى جعلناها تعكس قيمة نعم الى لا والعس

اى ان الرسالة فيها جملة شرط اذا كانت تساوى نعم اظهر الرسالة الاولى واذا كانت تساوى عكس الاشارة لا اظهر الرسالة الثانية

تماما مثل الجملة الشرطية if "الربط الثلاثى" فى السى شارب من ناحية المفهوم

صورة توضيحية



تنفيذ الكود فى الفيجول استديو

app.component.html

```
<input type="button" [value]="logstate" (click)="toggle()" >
```

```
export class AppComponent {
  title = 'blog-anguler';
 name='ahmed';
 age=24;
 logstate:string="login"
 isAuth:boolean=false;
 toggle(){
   this.isAuth =!this.isAuth;
   this.logstate=this.isAuth? "logout":"login";
 }
```

نقوم بالاعلان عن متغيرين احدهما نصى والاخر منطقى

```
 logstate:string="login"
 isAuth:boolean=false;
```

ال toggle هى event click name

```
 toggle(){
```

قمنا بعكس المتغير اذا كان نعم يصبح لا والعكس صحيح

```
   this.isAuth =!this.isAuth;
```

نتحقق من حدث المتغير النصى لو قيمته ى القيمة الاصلية التى وضعناها فى ال ts يكتب على سبيل المثال كلمة logout وان كانت تغيرت الى القيمة المخالفة لها يظهر لنا نص login

```
   this.logstate=this.isAuth? "logout":"login";
```

حتى الان لم نتحدث عن التغير الثالث والرابع وهما الرسالة التى تتغير مع غير حالة المستخدم ومع تغير label button

الان سنستعمل Structural Directives

بالتحديد سنستخدم ngif*

الفكرة البرمجية

نقوم بعمل div ونتتحقق من قيمة المتغير المنطقى عن طريق الجملة الشرطية اذا كانت القيمة نعم يظهر رسالة نصية ونعبر عنها بكلمة مثلا yes وان كانت القيمة "لا "يظهر رسالة نصية اخرى نعبر عنها مثلا no

ولكن يفضل الابتعاد عن اختيارات كلمات محجوزة حتى لا يحدث خطأ

ثم نقوم بعمل فورم او قالب للكلمة مربوط بكلمة yes ونضع فيه محتوى الرسالة ونكرر نفس الامر بالنسبة لكلمة no

التنفيذ عبر الفيجول استديو كود

نذهب الى مكون ال

app.component.html

نقوم بعمل **div** ونتحقق من قيمة المتغير المنطقى من ال **ts** عن طريق **nglf ***

اذا كانت قيمته كما عرفناها يظهر لنا رسالة الwelcome

اذا كانت القيمة مغايرة يظهر لنا فورم ال **error**

```
<div *ngIf="isAuth; then welcome else error" >

</div>
```

لاتنسى ال ; **semicolon** بعد اسم المتغير المنطقى

ولا تنسى كذالك حالة الاحرف عند كتابة ***nglf***

نقوم بعمل قالب لل **wlecome**

```
<ng-template #welcome>
   <p>
       wlecom ahmed
   </p>
</ng-template>
```

ونقوم بعمل قالب اخر لل error

```
<ng-template #error>
     <p>
       please login to continue
     </p>
  </ng-template>
```

ويمكن ان نغير لون البرجراف وحجم الخط عبر ال css

Services

وظيفة او دور ال Services هو مشاركة الكود مع جميع المكونات التى قمنا بانشائها ولا نقصد بالمكونات الاربع مكونات للمكون الواحد بل نقصد جميع المكونات # جميع المشروعات التى قمنا بانشائها وتدعى مكون

طريقة انشاء سيرفس جديد

نفتح ال terminal ونكتب ثم نضغط enter

```
ng g s myServices
```

S اختصار ل Services

يمكن استبدال كلمة myServices بالكلمة التى تريدها

(CREATE src/app/myservices.service.spec.ts (377 bytes

CREATE src/app/myservices.service.ts (139 by

☜نلاحظ انه قام بالفعل بانشاء ملفين

نذهب الى المكون التالى الذى تم انشاءه

myservices.service.ts

☜نلاحظ وجود كلاس بالاسم الذى قمنا باختياره هذا الكلاس يمكن مشاركته

```
export class MyservicesService {

  constructor() { }
}
```

```
import { Injectable } from '@angular/core';
```

☜ونجد Injectable تفيد باننا يمكن ان نعمل حقن للكلاس فى اى مكون اخر

```
@Injectable({
  providedIn: 'root'
})
```

✍سنقوم الان بعمل function فى الكلاس الموجودة فى ال service ونقوم باستخدامه فى باقى المكونات

✓هذه الفاكشن تستقبل بارمتر من نوع متغير نصى

✓ونسجل البارمتر فى الكونسول

```
export class MyservicesService {
  constructor() { }

  Hellow(componentnam:string){
console.log("Hellow my  component is:" +componentnam )
  }

}
```

Hellow : هو اسم الوظيفة

Componentnam : هو اسم البارمتر

النص المكتوب بين علامتين تنصيص " " سيظهر كما هو

+ اسم البارمتر ليضم النص المكتوب الى نص البارمتر

☜ اسم الوظيفة واسم البارمتر يمكن تغيره الى الاسم الذى نريده

الان نذهب الى المكون الذى نريد استدعاء الفاكشن فيه

انا قمت بانشاء مكون اسمه فديز عندما كنت احضر مكونات للروابط فى ال routing

video.component.ts

اذا دخلنا اليه سنجده هكذا

```
export class VideoComponent implements OnInit {

  constructor() { }

  ngOnInit(): void {
  }

}
```

ندخل الى ال constructor

ونكتب داخل القوسين

```
private service:MyservicesService
```

private بمعنى خاص

Service غلى سبيل المثال يمكن كتابة اى كلمة اخرى وهذه الكلمة صبحت object سوف يرث خصائص ال MyservicesService

ويمكن ان نعتبره property يرث خصائص سيرفيس

MyservicesService: هو اسم السيرفيس الذى قمنا بانشاءه مسبقا

```
ng g s myServices
```

اصبح الشكل كالتالى

```
export class VideoComponent implements OnInit {

  constructor(private service :MyservicesService) { }

  ngOnInit(): void {
  }

}
```

ngOnInit

```
  ngOnInit(): void {

}
```

ngOnInit : هو اسلوب ربط او طريقة تم تضمينها فى مكتبة ال Angular core

وهو اول ما ينفذ عند استدعاء المكون

```
  ngOnInit(): void {
    this.service. Hellow ("nam parameter ");
  }
```

Service بعد اضافة . سيظهر لى اسم الفاكشن Hellow الموجودة فى السيرفيس لانها ترث من ال MyservicesService

service :MyservicesService

ووضعنا اسم للبارمتر لان الفاكشن كانت تنتظر منا بارمتر

مراجعة سريعة

```
export class MyservicesService {
  constructor() { }

  Hellow(componentnam:string){
console.log("Hellow from the component:" +componentnam )
  }

}
```

```
export class VideoComponent implements OnInit {

  constructor(private service :MyservicesService) { }

  ngOnInit(): void {
    this.service.Hellow ("namparameter");
  }

}
```

عملنا فاكشن فى السيرفيس اسمها Hellow تنتظر قيمة متغير نصى وتطبع لنا رسالة + المتغير

ثم عملنا اوبجيكت يرث من السيرفيس

وجغلنا الاوبيجكت يمرر قيمة نصية الى الفاكشن

الان الفاكشن هتشتغل فى هذا المكون المربوط بالسيرفيس

Pipes

تستحدم لتغير شكل عرض البيانات

وهى تشبه دوال التحويل فى السى شارب

مثل تحويل حالة الحروف الكبيرة الى صغيرة وتحويل طريقة عرض التاريخ والوقت

Uppercase, lowercase,

decimal pipe

currency pip

date pipe

طريقة الاستخدام

بعد كتابة اسم المتغير نضغط \+shift ثم اسم دالة التحويل

```
<h1> Hello, {{name |uppercase}}  </h1>
```

مثال على التاريخ والوقت

نذهب الى المكون

app.component.ts

نعرف متغير جديد للتاريخ

```
export class AppComponent {
  title = 'blog-anguler';
 name='ahmed';
 currentDate=new Date();
 age=24;
```

نذهب الى مكون ال

app.component.html

```
<h2> {{currentDate}}</h2>
```

نشغل السيرفر ونشاهد النتيجة سنرى ان نمط كتابة التاريخ يحتاج الى تعديل

ننعدل طريقة العرض كالتالى

يوم / شهر / سنة

```
<h2> {{currentDate |date:"dd/MM/yyyy"}}</h2>
```

لعرض عملة يوروEUR/ دولار USD/

```
<h3> {{151515.33 |currency:"EUR"}} </h3>
```

forms

هناك طريقتين للتعامل مع الفورم

Template Driven forms

Reactive forms model Driven form

الطريقة الاولى

Template Driven forms

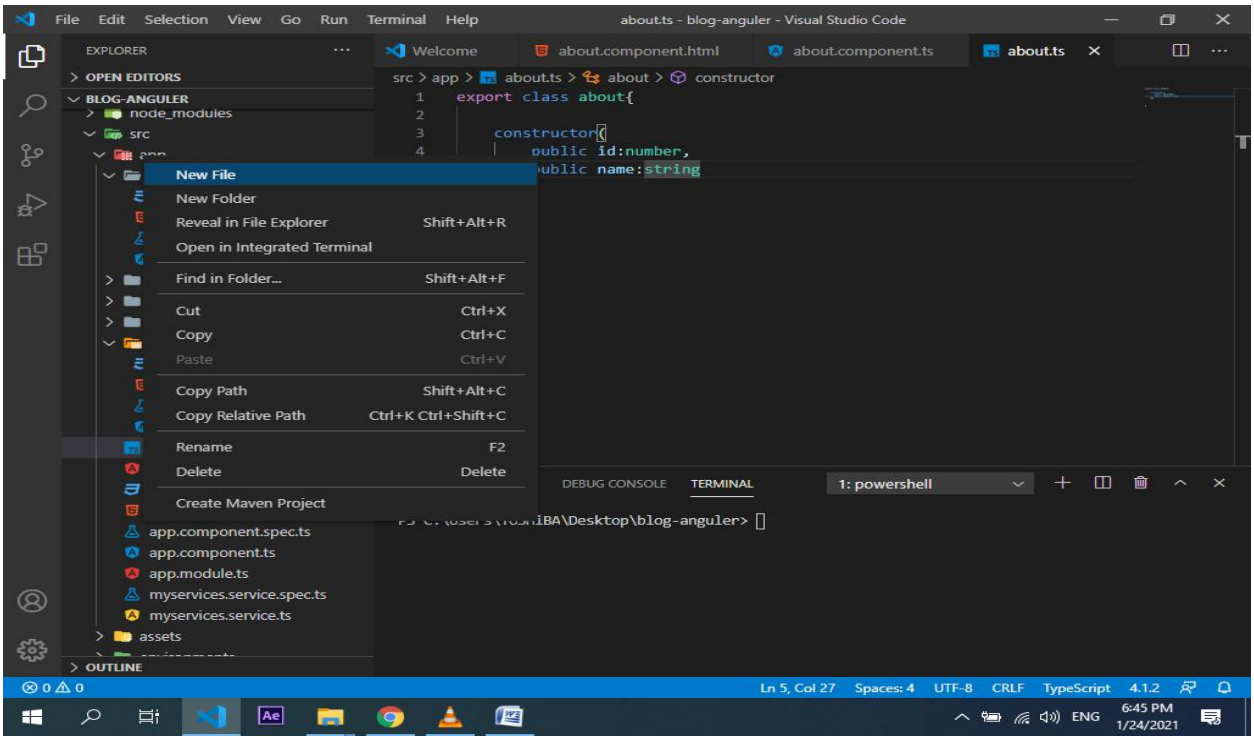
الفكرة العامة

نريد عمل 2 textbox و button

قيم ال textbox مربوطة بالـ ts بعلاقة تساوى للقيم وترسل القيم الى الكونسول على سبيل المثال عبر زرار له فاكشن يسجل القيم الى الكونسول

الطريقة

نذهب الى app ونقوم بعمل فولدر جديد



بفرض ان لدينا مكون اسمه about

<u>نقوم بتسمية الفولدر الى about.ts سيكون بمثابة كلاس خارجي</u>

نعمل كلاس اسمه about

ثم نعمل constructor

ثم نعمل متغيرين ونجعل المتغير عام عن طريق وضع كلمة public

Id: من نوع رقمي

Name: من نوع نص

```
export class about{
    constructor(
        public id:number,
        public name:string
    ){}
}
```

نذهب الى

about.component.ts

```
export class AboutComponent implements OnInit {
cat:about=new about(0,"Unknown");
```

ننشأ اوبجيكت وليكن اسمه cat ثم نضع : ليرث ثم نكتب اسم الكلاس الذى نريد ان نرث منه.

ثم نضغط enter سنجد انه عمل import لل about

```
import { about } from '../about';
```

new about= الان cat هو ابجيكت لديه خصائص id وname قمنا بتمرير قيم افتراضية عى سبيل المثال 0 للid وUnknown للاسم

```
export class AboutComponent implements OnInit {
cat:about=new about(0,"Unknown");
  constructor() { }
  ngOnInit(): void {
  }
}
```

نذهب الى مكون ال

about.component.html

ونعمل فورم

```
<form #myForm="ngForm" (ngSubmit)="onSubmit()">
    <label >
        Id:
    </label>
    <input type="text" name="id" [(ngModel)]="cat.id">

    <label >
        Name:
    </label>
<input type="text" name="name" [(ngModel)]="cat.name">
<input type="submit" value="send Data">
</form>
```

```
<form #myForm="ngForm" (ngSubmit)="onSubmit()">
```

فى بداية الفورم نعطى له اسم ويجب ان نضع شباك قبل الاسم
على سبيل المثال #myForm

ثم نكتب يساوى ثم "ngForm" لنخبره اننا سنتعامل مع الفورم عن طريق ال anguler

[(ngModel)] لنجعل القيم فى التكست بوكس تتاثر وتؤثر فى الts
بمعنى اى تغير فى ال html سوف يتغير فى ال ts

(ngSubmit) ليعرف اننا لدينا زرار ارسال يقوم بارسال البيانات

"onSubmit()"= زرا الارسال يساوى الفاكشن

نذهب الى

about.component.ts

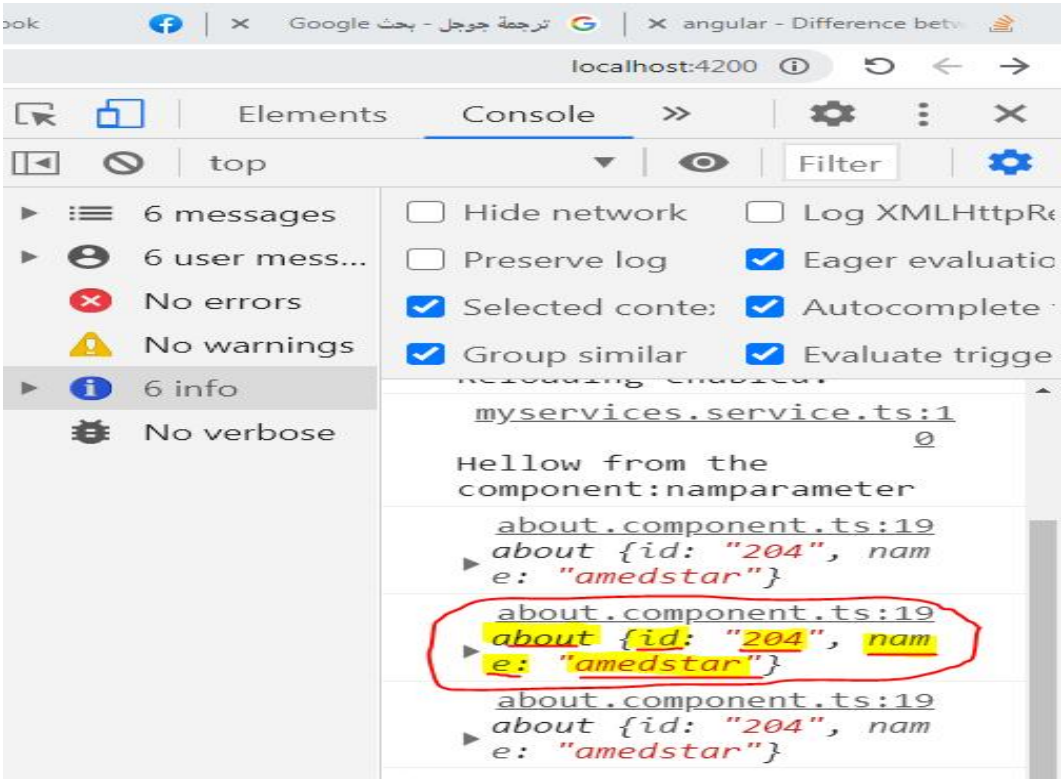
نعمل حدث للفاكشن وهو طباعة الcat فى الكونسول

```
export class AboutComponent implements OnInit {
cat:about=new about(0,"Unknown");


  constructor() { }

  ngOnInit(): void {
  }

  onSubmit(){
    console.log(this.cat);
  }

}
```

this.cat : قيمته الافتراضية هى 0 – Unknown يتم طباعتهم فى الكونسول

وعن طريق [(ngModel)] يتم تغير القيم الافتراضية فى الـ ts بنفس القيم الجديدة فى ال html

نذهب ونتاكد

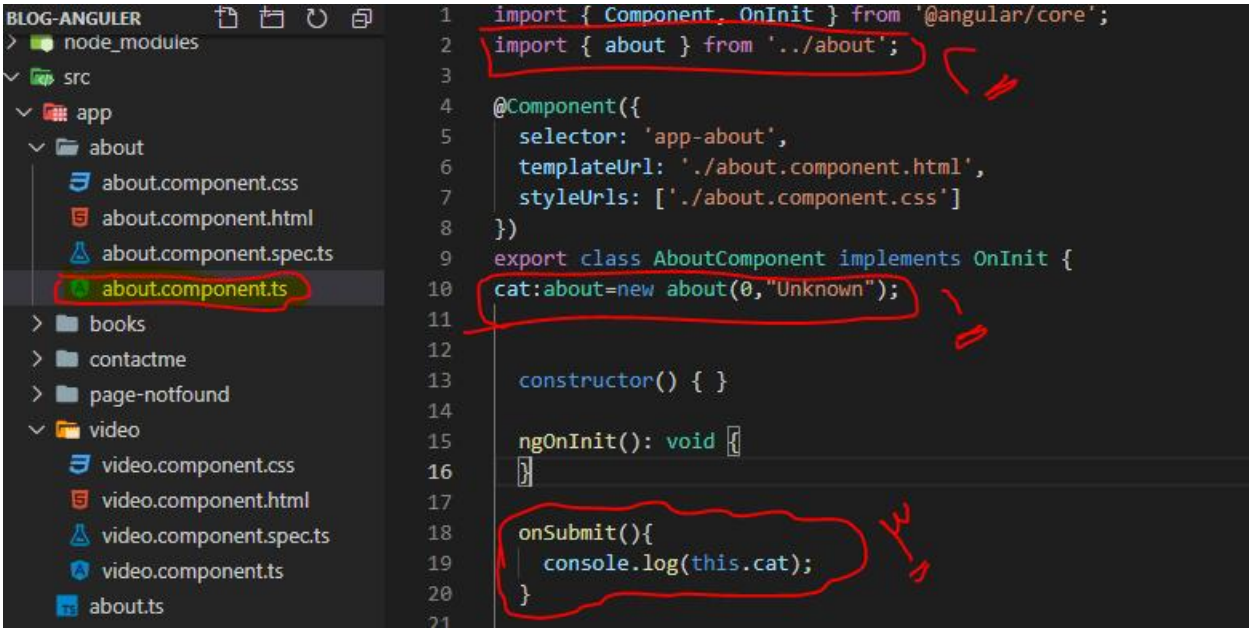


مراجعة سريعة

كلاس خارجى بمتغيرين وقمت بتعريف نوع ال data types

```
export class about{

    constructor(
        public id:number,
        public name:string
    ){}
}
```

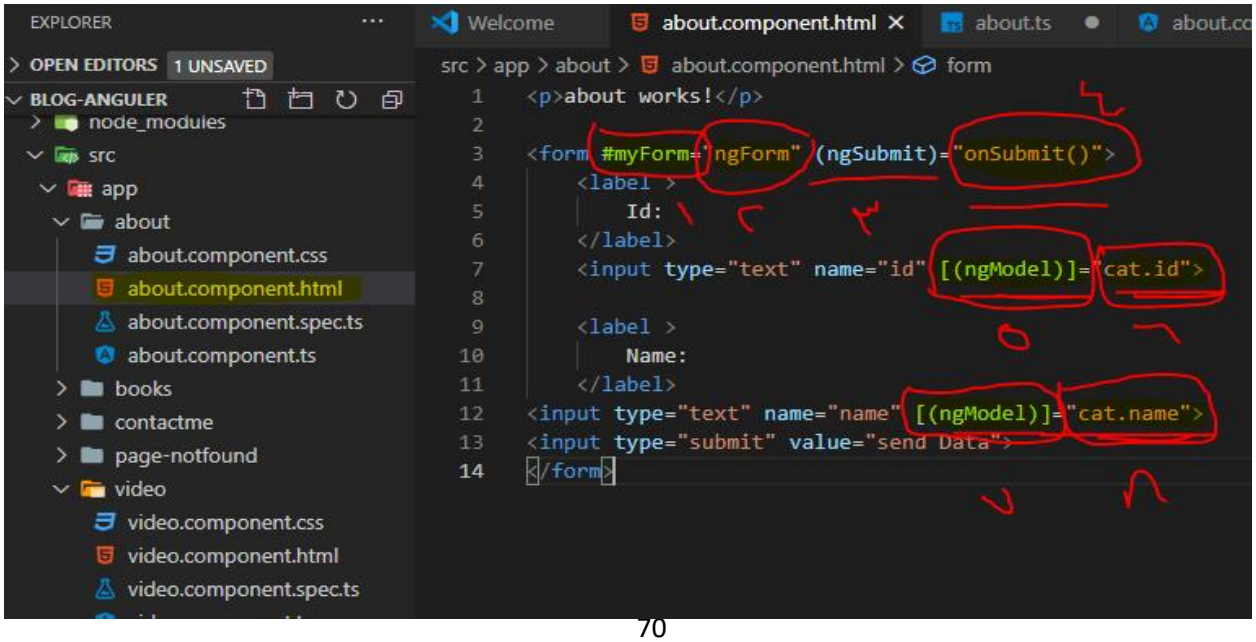
فى الts هعمل باس بمعنى تمرير للقيم "قيم افتراضية موقتا لان البارمتر يجب ان لا تكون قيمته Null"



1- عملنا اوبجيكت مستنسخ (cat) ورمينا القيم بداخله القيمة الاولى لل id والقيمة الثانية لل name

2- <u>ال import من المسار تم انشاءه تلقائى</u>

3- <u>هى الفاكشن وthis.cat هم القيمتين المتغيرين عن طريق ال 2 way binding</u>



1- هو اسم افتراضى للفورم مسبوق ب شباك

2- تعريف للفورم بصيغة anguler

3 - هو نوع حدث زر الارسال

4- هى الفاكشن التى سيتم تنفيذها

7/5 هى طريقة الربط الثنائى بين ال html وال ts

6- هى قيمة ال id الجديدة عن طريق [(ngModel)]

8- هى قيمة ال name الجديدة عن طريق [(ngModel)]

الطريقة الثانية للتعامل مع الفورم

Reactive forms model Driven form

لكى نضمن هذا المفهوم وهى انشاء فورم عن طريق المديول وليس عن طريق قالب مثل الطريقة السابقة

نذهب الى المكون

app.module.ts

فى ال imports كتبنا ثم حرف و ReactiveFormsModule

```
imports: [
  BrowserModule,
  FormsModule,//اضافتها تم
  NgxSpinnerModule,//اضافتها تم
  ReactiveFormsModule,
  AppRoutingModule, BrowserAnimationsModule
],
```

لو نظرنا فى الاعلى سنرى انه قام بالاستيراد وسنتطيع رؤية المسار الذى يجلب منه المديول

```
import {FormsModule, ReactiveFormsModule} from '@angular/forms';//اضافتها تم
```

الان نذهب الى المكون فى مكون ال ts الخاص الذى نريد اضافة الفورم اليه

على سبيل المثال انا لدى مكون اسمه

books.component.ts

نذهب الى الكلاس الخاص به

```
export class ContactmeComponent implements OnInit {

  constructor() { }

  ngOnInit(): void {
  }

}
```

نقوم عمل فورم نقوم بتسميته ونستنسخه من FormGroup

```
myForm=new FormGroup({});
```

ندخل داخل الفورم ونضيف ال property

```
myForm=new FormGroup({

});
```

```
export class BooksComponent implements OnInit {
myForm=new FormGroup({
name:new FormControl(''),
age:new FormControl(''),
address:new FormControl(''),
jop:new FormControl('')
});
```

فى نهاية كل سطر نضغط حرف و , لاننا سوف نكتب سطر اخر وفى السطر الاخير لا نضع هذه العلامة (,)

☜ عملنا اربع متغيرات لكل متغير منهم كنترول

☜ لاحظ ان المكون Contactme هو مكون قد انشاءته فانت لست ملزم باختيار نفس اسم ال component يمكنك اختيار اى component تريده

لكن يجب ان تضع كود ال ts فى و ال html فى المكون الواحد بمعنى ان المكون يتكون من اربع اجزاء منهم ال ال html وال ts

هناك مكونات اخرى لاتهمنا لاننا نشتغل على المكون الواحد من المفترض انك قد علمت لكن تكرار القول يمنعنا من الوقوع فى الخطا او النسيان .

الان نذهب الى المكون الخاص بال view

contactme.component.html

نقوم بعمل فورم للعرض

شكل الفورم النهائى

```
<form [formGroup] ="myForm" (ngSubmit)="onSubmit()">
    <div>
        <label >Name:</label>
        <input type="text" formControlName="name">
    </div>
    <div>
        <label >Age:</label>
        <input type="text" formControlName="age">
    </div>
    <div>
        <label >Address:</label>
        <input type="text" formControlName="address">
    </div>
    <div>
        <label >jop:</label>
        <input type="text" formControlName="jop">
    </div>
    <input type="submit" value="send from Data">
</form>
```

[formGroup] هنا حددنا نوع الفورم لان الفورم تم بناءه فى المديول بطريقة ال Reactive forms model

myForm : هو اسم الفورم فى ال ts

```
export class BooksComponent implements OnInit {
myForm=new FormGroup({
```

ngSubmit : هو حدث زرار الارسال وهو يساوى الفاكشن اعطيناها اسم افتراضى onSubmit

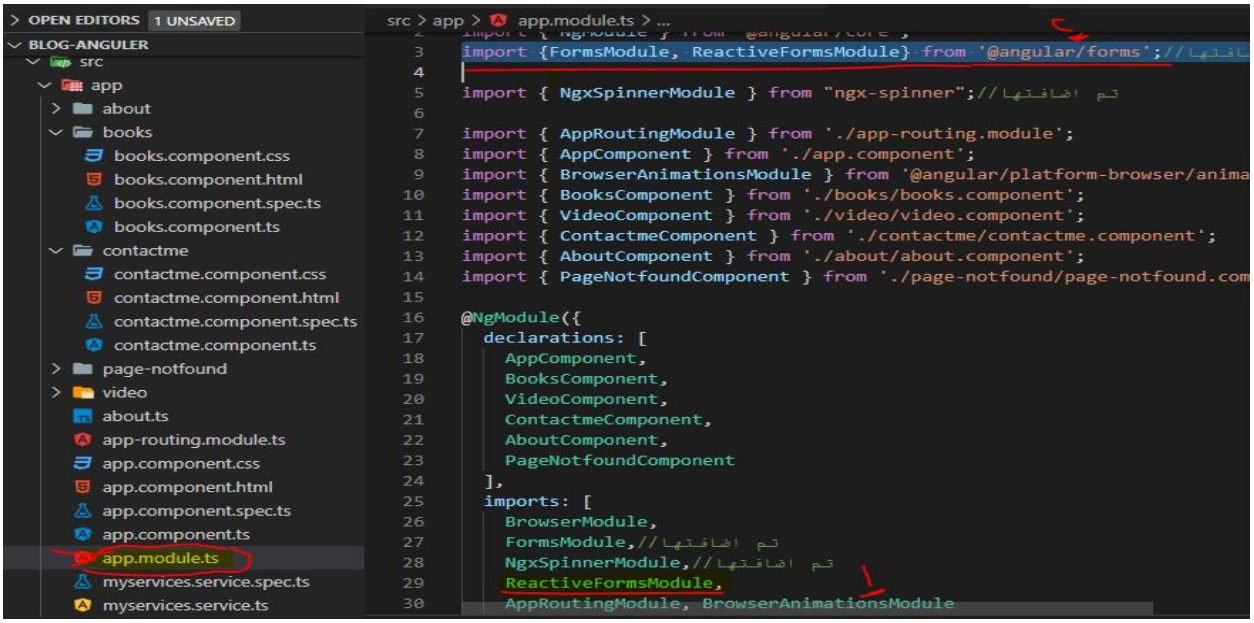
ثم ذهبنا الى مكون ال

books.component.ts

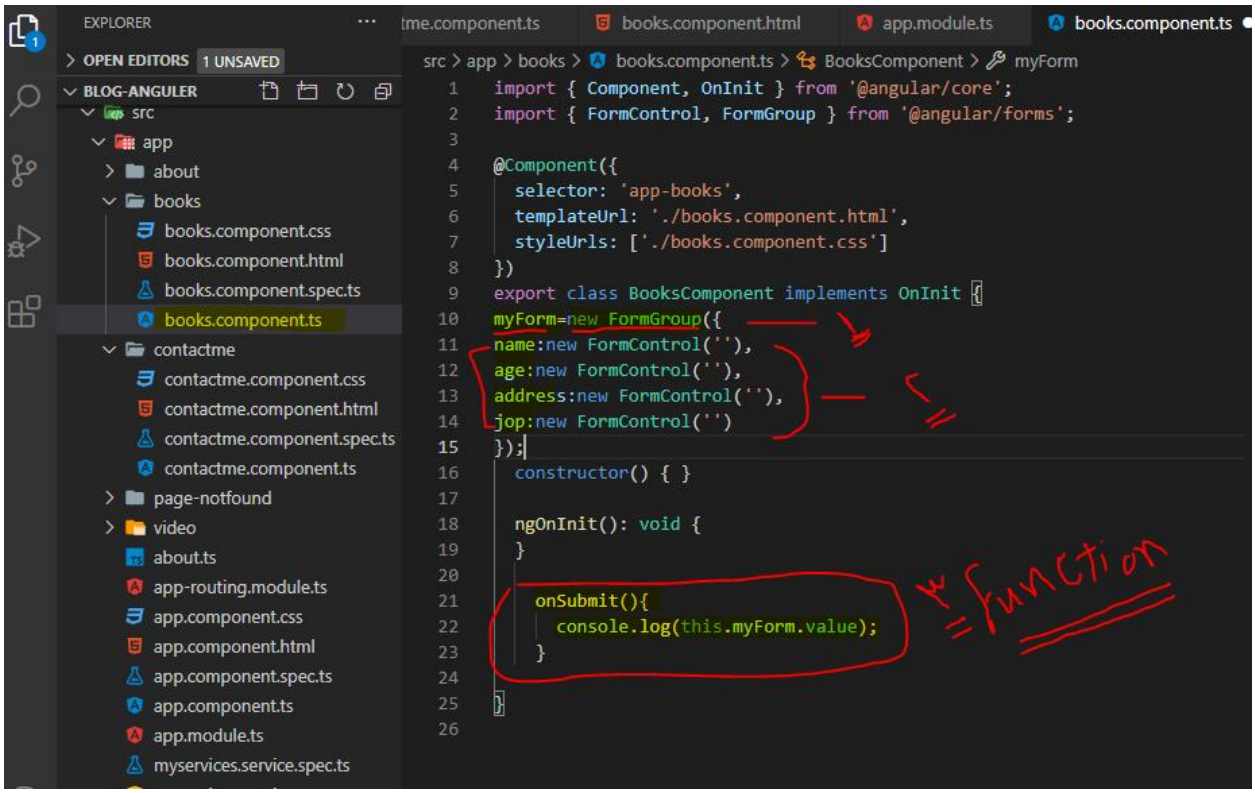
وعملنا الفاكشن على سبيل المثال اخترنا اسم onSubmit

```
export class BooksComponent implements OnInit {
myForm=new FormGroup({
name:new FormControl(''),
age:new FormControl(''),
address:new FormControl(''),
jop:new FormControl('')
});
  constructor() { }

  ngOnInit(): void {
  }

    onSubmit(){
      console.log(this.myForm.value);
    }

}
```

هذه الفاكشن تطبع القيم النصية التى سيتم كتابتها فى حقول الادخال مراجعة سريعة



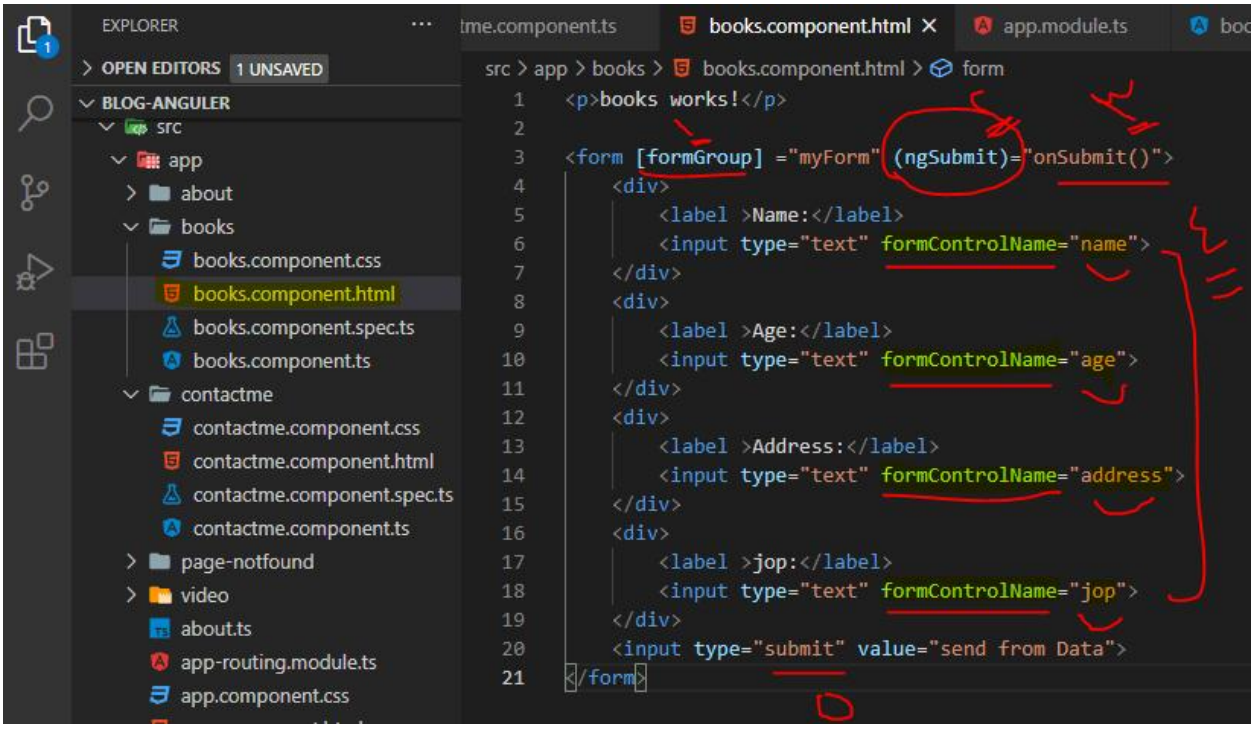
قمنا بتضمين مفهوم ReactiveFormsModule



1- اسم الفورم يساوى FormGroup

2- اسم المتغيرات كل منهم يساوى FormControl

3- هى الفاكشن المربوطة بزرار الارسال فى ال html فى الفورم



1- هو الطريقة التى تعمل بها الفورم

2- هو حدث الارسال فى افورم

3- هى الفاكشن التى تطبع لنا البارمترات المدخلة فى الكونسول

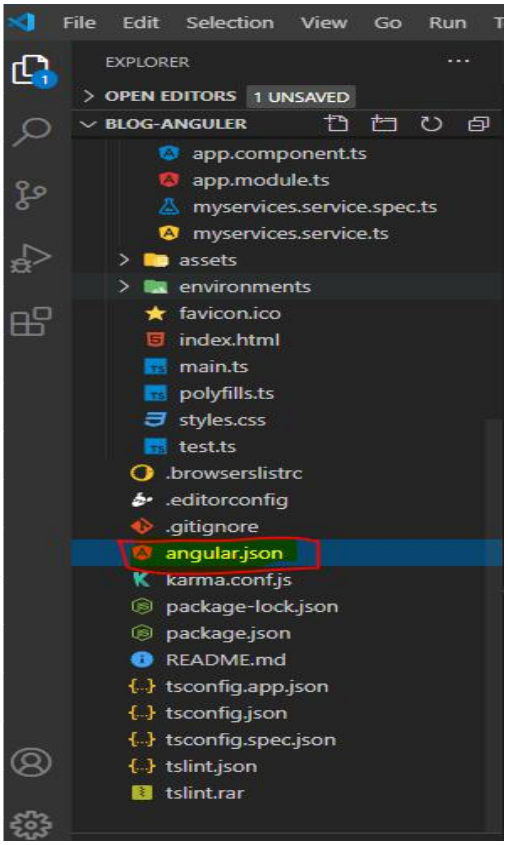
4- اسم المتغير فى الفورم كنترول

5- هو زرار الارسال

Angular Project Deployment on Live Server

نكتب هذا السطر فى ال terminal

```
ng build --prod
```



سيقوم تلقائيا بانشاء ملف يدعى

dist

اذا ذهبنا الى ملف ال

angular.json

سنجد مسار الملف بداخله

ندخل الى ملف ال dist

ونضغط الملف الذى بداخله بامتداد

Zip

نذهب الى موقع الاستضافة ونضيف الملف المضغوط ثم نفك الضغط ونعمل move للملفات المضغوطة الى المسار المناسب ويمكنك معرفة ذلك من موقع الاستضافة المجانى

هناك العديد من المواقع لنشر الموقع الخاص بك كل ما ستحتاج اليه اميل فقط لتفعيل حساب مجانى

ولكن الويب سايت المجانى لايمكنك فيه ان تختار اسم الدومين

و يمكنك رفع ملفات او داتا بيز ولكن بحجم محدد اذا ارد ان تزيد من مساحة الاستضافة على السيرفر تستطيع شراء مساحة وتستطيع معرفة الاسعار وتختار ما هو مناسب لك.

للاطلاع على المزيد عن anguler يمكنكم زيارة الموقع الخاص

https://angular.io/docs

نصائح قبل نشر الموقع
تاكد من ان الموقع responsive بمعنى انه متجاوب على جميع احجام الشاشات ويمكنكم استعمال مكتبة البوت ستراب لفعل ذلك ولا شك ان استخدام مكتبة ال fontawesome واستعمال الايقونات سيجعل الموقع افضل

واذا اردت ان نقوم بعمل تنسيقات خاصة بال css يالتاكيد ستحصل على مظهر افضل ويممكنك استخدام animation عند الضغط على لينك او صورة

هذه هى البداية لمن اراد البدء فى anguler يمكن بعد ذلك ان تربط ال anguler مــع asp عــن طريــق asp.netcor ونقــوم بعمليــات الاتصال بالسرفر وتقوما بانشاء database من نوع sql server ومن ثم نقوم بعمل جمل الاضافة والحذف والتعديل يمكن ان تبحثوا وتتقدموا للامام

اترككم في رعاية الله وحفظه.

الخاتمة

الحمد لله تعالى الذي قدرنا الى فعل ذلك ، وكتب لنا التوفيق والسداد، ونسأل الله أن ينال تقديركم وإعجابكم

بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ

. "مَنْ عَمِلَ صَالِحًا مِّن ذَكَرٍ أَوْ أُنثَىٰ وَهُوَ مُؤْمِنٌ فَلَنُحْيِيَنَّهُ حَيَاةً طَيِّبَةً، وَلَنَجْزِيَنَّهُمْ أَجْرَهُم بِأَحْسَنِ مَا كَانُوا يَعْمَلُونَ"، صدق الله العظيم.

سورة النحل - الآية 97

. رَبَّنَا اغْفِرْ لِي وَلِوَالِدَيَّ وَلِلْمُؤْمِنِينَ يَوْمَ يَقُومُ الْحِسَابُ

سورة ابراهيم - الآية 41

ولا تنسوا الاخلاص فى الدعاء الى جدى وعمتى رحمه الله عليهم

واسالكم الدعاء لى بالتوفيق والسداد