**Basic steps to use angular-cli angular 2**

* npm set strict -ssl false (to remove error)
* npm install –g angular-cli
* ng new pro\_Name
* Cd pro\_Name
* npm start
* Go to localhost:4200
* If gives error during installation
  + npm config set registry <http://registry.npmjs.org/>
  + then in new cmd session try to run the module(npm start)
  + Still not working? try this
  + npm uninstall typescript -g  
    npm cache clean  
    npm install typescript@2.0.0 –g

**Component Generation**

* ng g component Compo\_Name
* Files will be generated –
* Compo\_Name.component.ts
* Compo\_Name.component.html
* Add Appropriate code in files
* Now in main app.component.ts
* Add basic structure and put selector.
* Here Goes the output

**Implementing ag-grid**

* Create a new component
* In grid-component.html,
* <**ag-grid-ng2 class="ag-fresh div-margin" style="height**: 300**px**; **width**:100%;**" [gridOptions]="gridOptions"**>  
    
  </**ag-grid-ng2**>
* In grid-component.ts file,

**gridOptions**: GridOptions = {  
 **columnDefs**: **this**.**columnDefs**,  
 **rowData**: **this**.**rowData**,  
 **enableSorting**: **true**,  
 **enableFilter**:**true**}

* Add column details in columnDefs & rowData
* {**headerName**: **"legalName"**, **field**: **"legalName"**},
* Attach json data in ts file
* Add selector in app.component.html file
* Done

**Importing from json**

* Define appropriate columns
* **rowData** = sampleCarriers;

**Implement Kendo ui-grid**

* Need to register Kendo UI npm endpoint
* <https://www.telerik.com/login/v2/telerik>
* npm login --registry=https://registry.npm.telerik.com/ --scope=@progress
* Insert credentials here in cmd
* npm install -S @progress/kendo-angular-grid
* Add this to app.module.ts to import gridmoldule from kendo,

**import** { GridModule } **from '@progress/kendo-angular-grid'**;

* Now add following selector for grid view,Define required column like this

<**kendo-grid [data]="gridData" [sortable]="{ mode: 'multiple' }"  
 [sort]="sort"**>  
 <**kendo-grid-column field="legalName" title="Legal Name" [width]="120"**>  
 </**kendo-grid-column**>  
 <**kendo-grid-column field="matchType" title="Match Type" [width]="120"**>  
 </**kendo-grid-column**>  
</**kendo-grid**>

* **[sortable]="{ mode: 'multiple' }"**this will add sorting feature to grid
* Now add this to grid.component.ts file,

**private** gridData: **any**[] = sampleCarriers;

where sampleCarriers can be any data coming from json

* Json data will be defined in .ts file as follows,
* **export const** sampleCarriers = [{….Data goes here…}];

**Implement Routing in page**

* In app.module.ts-
* **import** { RouterModule, Routes } **from '@angular/router'**;
* **imports**: [

**//other modules already here** just add this  
 RouterModule.*forRoot*(appRoutes)  
],

* **const** appRoutes: Routes = [  
   { **path**: **'content(//this is name of routing to be used in navbar)'**, **component**: ContentComponent (//this is component in which html code is there)},  
   {  
   **path**: **home**,  
   **component**: HomeComponent,  
   **data**: { **title**: **Home '** }  
   },
* {**path**: **''**,  
   **redirectTo**: **'/home'**,  
   **pathMatch**: **'full'**}
* In app.component.html-
* <**router-outlet**></**router-outlet**>
* In navbar.component.html-
* <**li**><**a routerLink="/content" routerLinkActive="active"**>Content</**a**></**li**>

**Implement Pagination in page (eg. for image gallery)**

* In app.module.ts,
* **import** {Ng2PaginationModule} **from 'ng2-pagination'**;
* Write in imports as well
* Now in component.ts file,
* **collection** = [];  
  **constructor**() {  
   **for** (**let** i = 1; i <= 100; i++) {  
   **this**.**collection**.push(**`http://lorempixel.com/output/food-q-c-320-240-2.jpg`**);  
   }  
  }

//for the time being I have chosen to display a single image no of times.

* Now in html file,on tag which you wanna repeat,

<**div class="col-md-4 col-sm-4 col-xs-6 shine\_me" \*ngFor="let item of collection | paginate: { itemsPerPage: 3, currentPage: p }"** >

<**img [src]="item" class="img-responsive"**/>  
 <**i class="shine\_effect"**></**i**>  
</**div**>

* Array of any dat type can be defined in type script as follows

{

Products:any[]=[{

Name:’djahud’

}]

}

* \*ngIf=”products && products.length”
* Eg.this condition written in table then table will appear only if there are products in list else no.
* How to give image path to img tag
* <img [src]=’item.imageUrl’ [title]=”Image-title” [style.width.px]=’imageWidth’/>(prefered)
* <img src={{item.imageUrl}} />

Where title comes sfter hovering the image.

Where imageWidth is defined in component as a property as follows,

imageWidth:number=50;

* **Event Binding**

<button (click)=’toggleImage()’></button>

**Link to standard events:**

<https://developer.mozilla.org/en-US/docs/Web/events>

* **Method signature:**
* toggleImage():void{

//code logic goes here

}

* **Conditiona statement :**

{{showImage ?’A1’ :’A2’}}

* **@Input()**
* Decorator to specify the input attributes
* Eg.

@Input() name:string;

HTML🡪

<app-star **[name]**=”items.name”></app-star>

Attached to a prop of any type

* **@Output()**
* Used for custom event binding
* @Output() **notify** :EventEmitter<**string**>=**new** EventEmitter<**string**>();
* onclick():**void**{  
   **this**.notify.emit(**`The rating** ${**this**.**rating**} **was clicked`**)  
    
  }
* HTML🡪
* <**app-star [rating]="*item*.rating" (notify)="onEventclicked(*$event*)"**></**app-star**>
* onEventclicked(message:**string**):**void**{  
  **this**.**pageTitle**=**`ProductList :`**+message;  
    
   }//in resp main parent component
* Attached to a prop declared as EventEmitter
* Use generic argument to describe payload
* **Service**
* Create a service class
* Define metadata with decorator
* Import what we need
* Eg.

@Injectable()

Export class DataService{

getdata():Itable[ ]{

return[

{data},{ data },{ data }

]

}

}

* **import** {DataService} **from "../content/data.service"**;

.

.

.

**tableData:Itable[ ];**

**constructor**(**private** \_dataService :DataService) {  
//assigning the values in service to a private variable  
 }

**Now initialize the data in life cycle hook method,**

ngOnInit() {  
 **this**.**tableData**=**this**.\_dataService.getData();  
 ***console***.log(**this**.**tableData**)  
 }

* **rxmarbles.com**
* **HTTP & Observables**
* Used to manage asynchronous data also manages using operators map ,filter
* **private \_url**=**'http://kahipremaloli.com/json/poems.json'**;
* Initialization
* **constructor**(**private** \_http :Http){  
    
  }
* **//Import {Http} from ‘@angular/http’**
* getDetails():Observable<ITable[]>{  
   **return this**.\_http.get(**this**.**\_url**)  
   .**map**((resp:Response)=> <ITable[]> resp.json());  
  }
* ngOnInit() {  
   **this**.\_dataService.getDetails()  
   .subscribe(tableData =>**this**.**tableData** = tableData,  
   errorMsg =>**this**.**errorMsg** =**'Got an Error !!'**);  
    
  }
* **Routing**
* <base href=’/’ > in main HTML
* In imports add RouterModule.forRoot( [

{path :’home’, component: HomeComponent},

{path:’about’ , component :AboutComponent},

{path :’’ ,redirectTo:’welcome’ ,pathMatch :’full’},

{path :’\*\*’ ,redirectTo:’welcome’ ,pathMatch :’full’}

] )

* Now in Html ,
* <**ul class="nav navbar-nav navbar-right"**>  
   <**li**><**a routerLink="/home" routerLinkActive="active"**>HOME</**a**></**li**>  
   <**li**><**a routerLink="/content"** >CONTENT</**a**></**li**>

</ul>

* Routing With parameters
* {path :’details/:name’, component: ’DetailsComponent’}
* In that details.component.ts file use

Import {ActivatedRoute} from ‘@angular/route’

* Inject the dependency =>

Constructor ( private \_route: ActivatedRoute){

}

* pageTitle:string=’This is :’

ngOnInit() {  
**let** name=**this**.\_route.**snapshot**.**params**[**'name'**];  
**this**.**pageTitle** + =**`**${name}**`//upending the name in url(I think)** }

* <**div class="container"**>  
   <**h1**>{{**pageTitle**}}</**h1**>

</div>

Done ☺

* Back button using Code
* <button (click)=”onBack()”>Back</button>
* Import {Router } from ‘@angular/route’
* Inject the dependency in constructor
* Write the method
* onBack():void{

this.\_router.navigate([‘/content’])

* }
* Done ☺
* Guards for URL access
* Create a service guards.service.ts
* **import** {Injectable} **from '@angular/core'  
  import** {ActivatedRouteSnapshot, CanActivate ,Router} **from '@angular/router'**@Injectable()  
  **export class** Guards **implements** CanActivate{  
   **constructor**(**private** \_router:Router){  
    
   }  
   canActivate(route:ActivatedRouteSnapshot):**boolean**{  
   **let** id =+ route.**url**[1].**path**;//Here the + sign denotes that it is converted into the string i.e in id it will get a number  
   **if**(*isNaN*(id) || id < 1 ){  
   *alert*(**'Invalid product id!'**);  
   **this**.\_router.navigate([**'/gallery'**]);  
   **return false**;  
   }  
   **return true**;  
   }  
  }
* Add as provider in app.module.ts
* In route definition add canActivate property as follows,
* { **path**: **'details/:name'**,**canActivate**:[Guards], **component**: DetailsComponent },
* Here, if we try to route with invalid link we will get alert and we will be redirected to mentioned landing url.
* ‘+’is javascript shortcut to convert a js string to numeric ID,

So that it will be easy to see tha id we got from the url.