* Create a folder ‘assets’ in the same directory we have ‘app’ folder
* Create folders ‘fonts’ and ‘css’ under folder assets
* Download bootstrap.zip and extract fonts to this directory
* Download bootstrap.css from <https://bootswatch.com/> ( Flatly ) and copy to css folder
* Create [home.component.ts](http://home.component.ts)

Home.component.ts



* Create NavbarComponent

Navbar.component.ts



* Create navbar.component.html

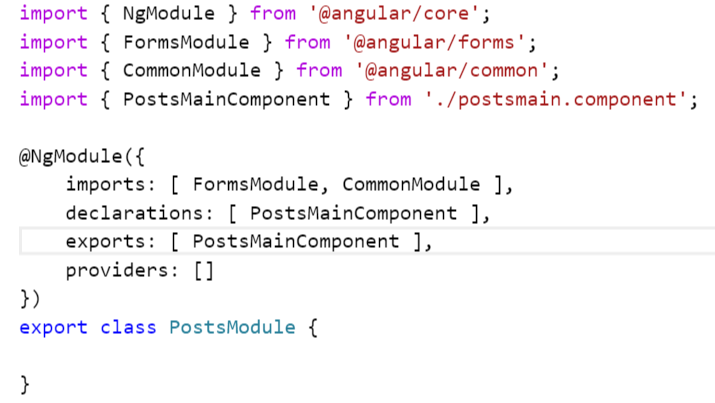


* Create posts directory and create – PostsMainComponent, posts.module

*Postsmain.component.ts*



*Posts.module.ts*

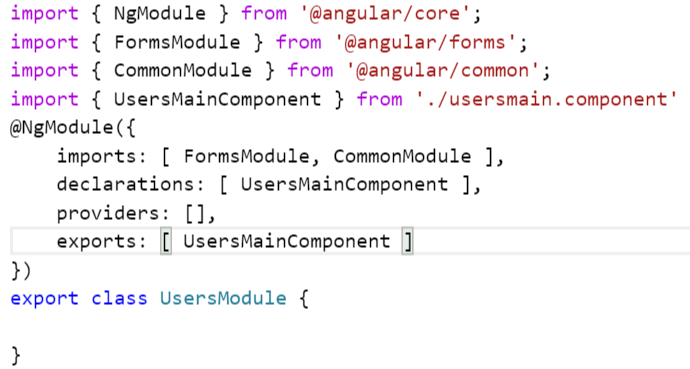


* Create users directory and create – UsersMainComponent, users.module

*Usersmain.component.ts*



*Users.module.ts*

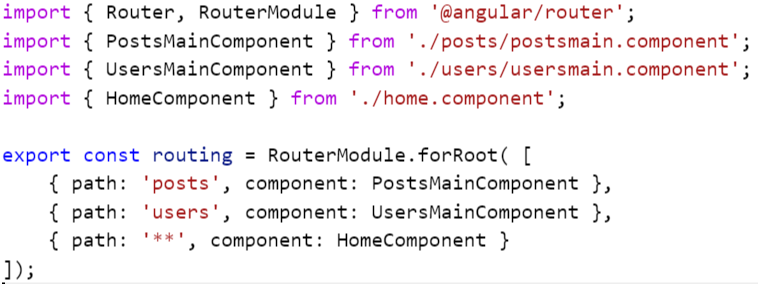


* Configure above two modules to app.module.ts

*App.module.ts*



* Create routing configuration file – app.routing.ts



* Output



* How to highlight the selected tab?
  + Use attribute – routerLinkActive

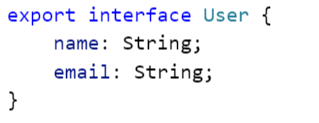


* Creating a table to display list of users from <http://jsonplaceholder.typicode.com/users>
  + Create a table structure using bootstrap class.



* Create an interface class to hold user details

*User.ts*



* Create a service class to call REST service and get details from the server

*Users.service.ts*



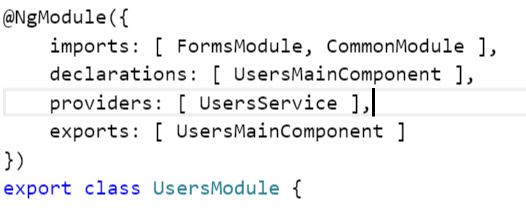
* Modify users main component to call this server and get displayed users on screen

*Usersmain.component.ts*

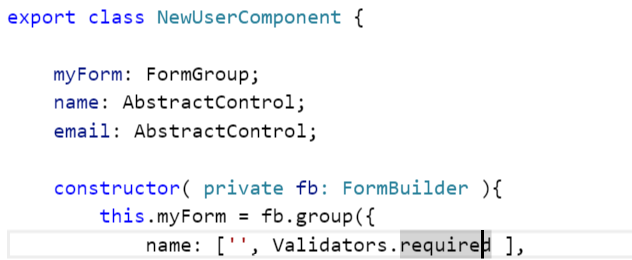


* Modify users.module.ts to include new users service.

*Users.module.ts*



* Let’s do validation for name & email attribute.
  + In build feature – required validation

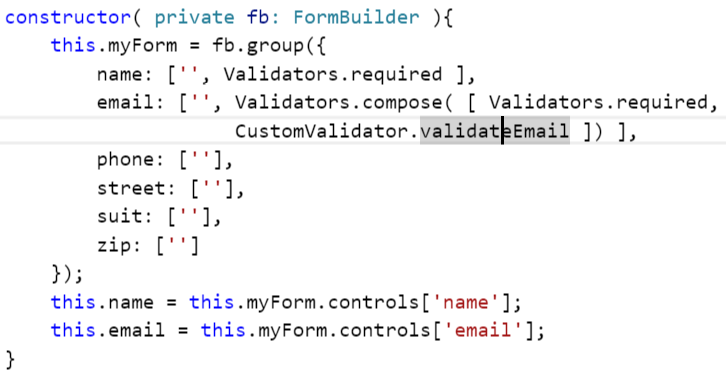


* + Create a new folder – validator and class – CustomValidator

*Customvalidator.ts*

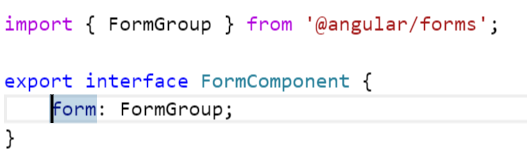


* + Custom feature – email validation



* Routing Feature: dirty checking / CanDeactivate
  + This feature will check that user provided all values and saved data. If not it will prompt user for the confirmation
  + Design an interface which can hold reference of FormGroup. ( this will become reusable component )

*Formcomponent.ts*



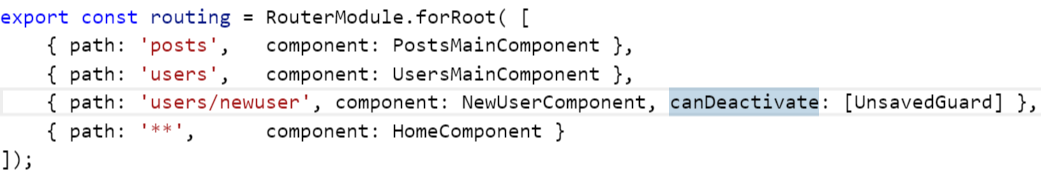
* + Design a service which will implement interface CanDeactivate

*Unsaveguard.ts*



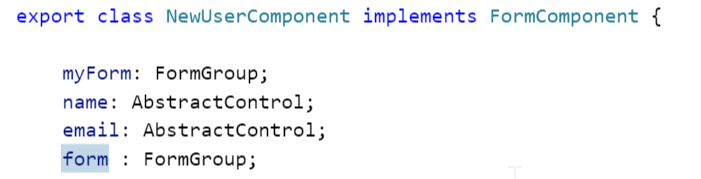
* + Update routing configuration for can-deactivate service

*App.routing.ts*

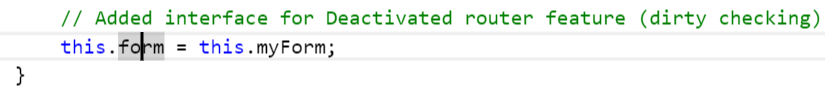


* + Update newuser component to implement FormComponent interface

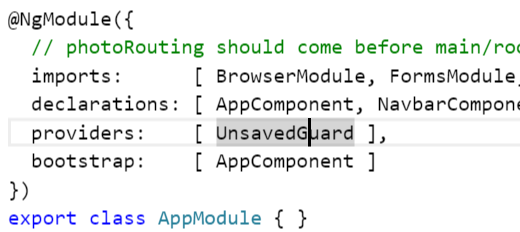
*Newuser.component.ts*

**

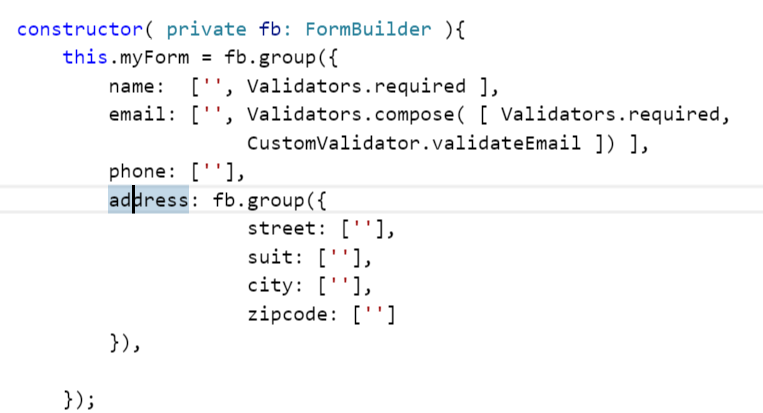
>> inside constructor defination



* + Update provider in app.module.ts

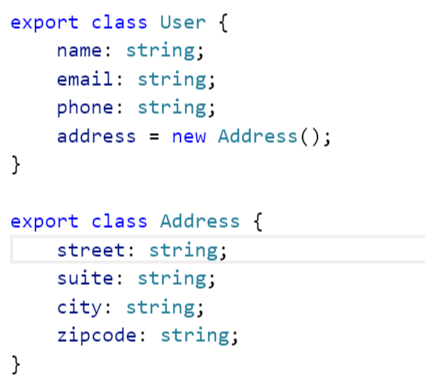


* Revised newuser.component to include address form group.



* Create domain object user to hold user details

*User.ts*

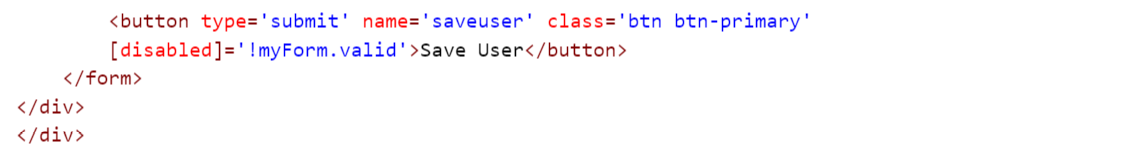


* Display values in a new user form using ngModel – two way binding

*Newuser.component.html*

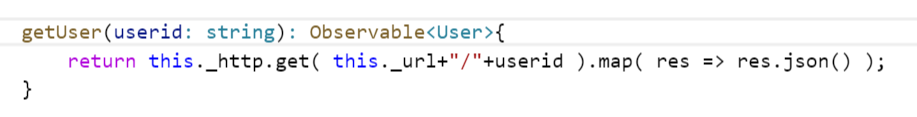






* Let’s create a functionality to Edit User details.
* Create a service method to get details for the specific user.

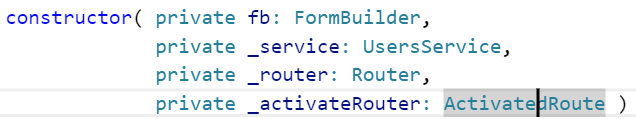
*Users.service.ts*



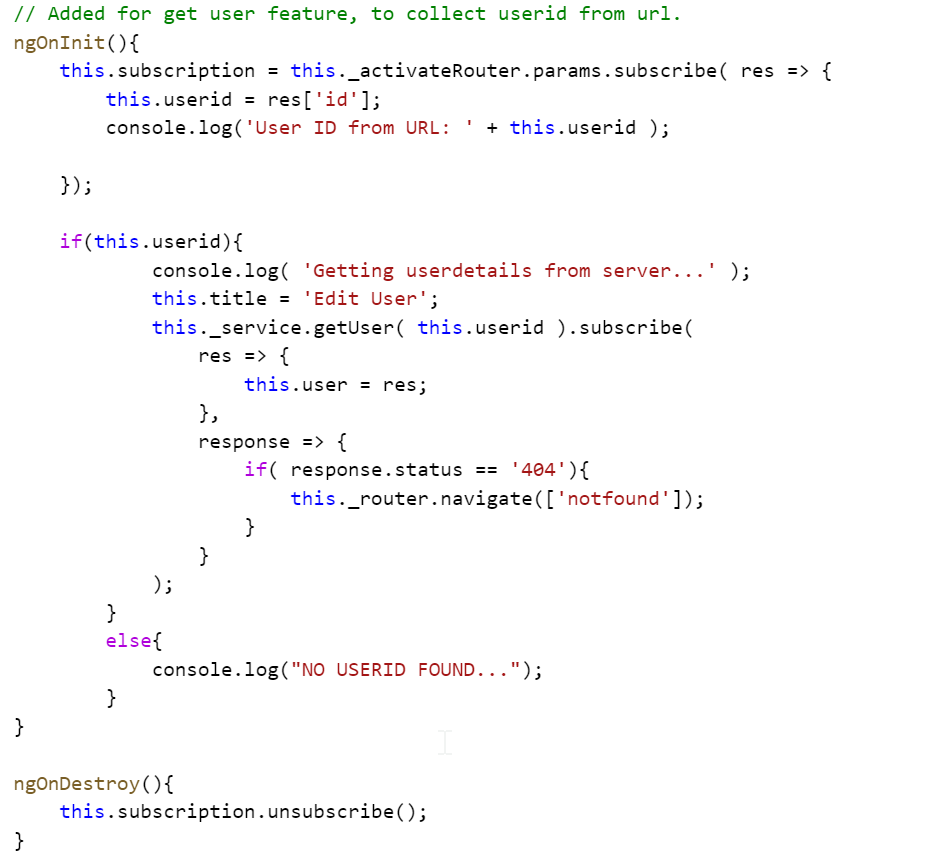
* Initialize newuser form component by calling above getUser service.

*Newuser.component.ts*

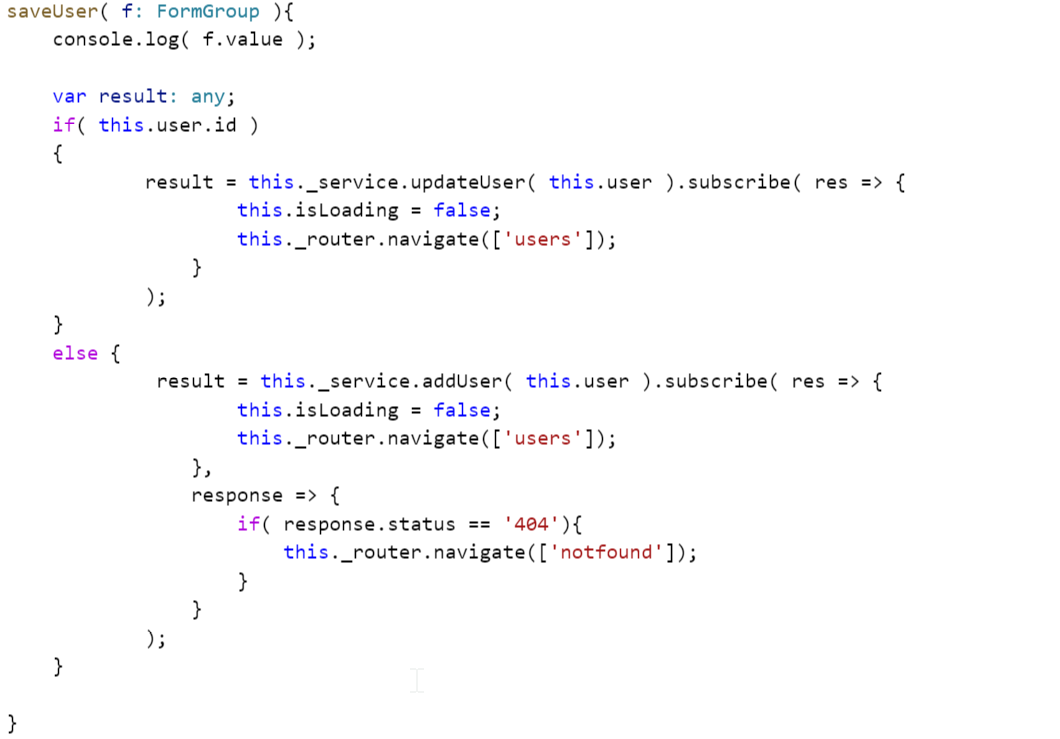
Modify constructor to include object of ActivateRouter



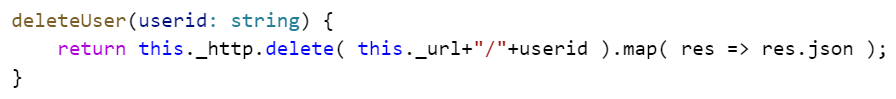
We can also use other router API like router.params for the same purpose.



* **Update user feature:**
  + We have user object where we have id attribute. If id attribute has some value that means it is going to be update otherwise it will be added.
  + Have modified below code for the same.



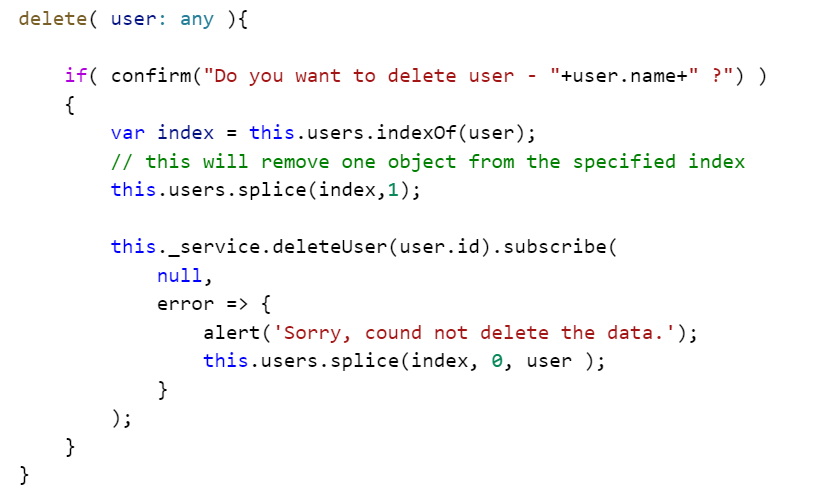
* **Delete user feature:**
  + Create a deleteService method in userservices.ts

****

* + Specify on click event – delete(user) method in users.component.html

****

* + Create a delete(user) method in usersmain.component.ts

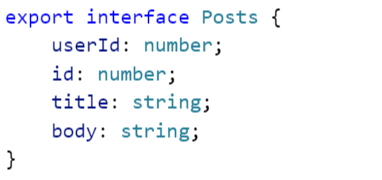
****

* Validation was just stopped working
  + Looks like in final version we cannot directly access the properties using reference variable #name.
  + We have to use below approach:
    - Name: AbstractControl; this.name = this.myForm.controls[‘name’]; OR
    - Inside the html form: myForms.controls[‘name’]

**Developing Posts tab features:**

* Create a class/interface to hold posts details

*Posts.ts*



* Create a service to collect list of posts from server

*Posts.service.ts*



* Display list of posts in posts.component.ts using below code



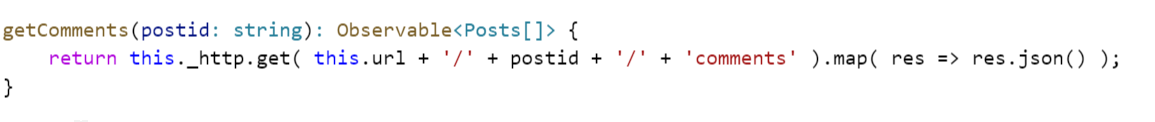
* Highlight the post on select and move-over
  + We can create a style definition called postsStyle and supply values for the hover



* + Apply class active when user selects a particular post from the list



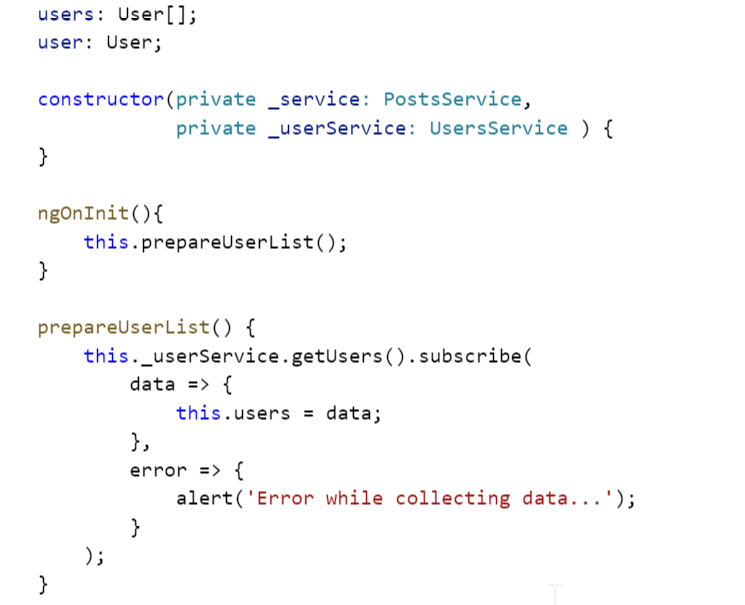
* Displaying comments with images
  + Create a service method to collect list of comments



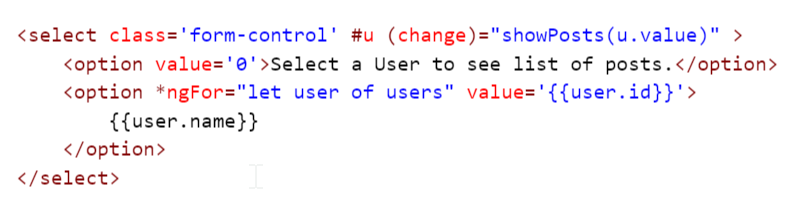
* + Create a media object a populate list of comments



* Creating dropdown box with list of users.
  + Modify postmain.component.ts for preparing list of users



* + Modify posts.component.html for the creation of dropdown box



* + Displaying list of post on selection of user name



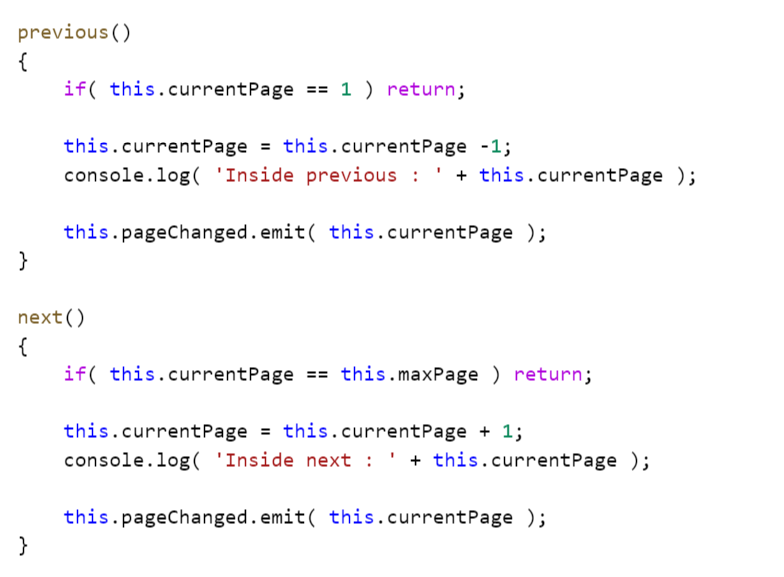
* Pagination Feature:
  + Create a bootstrap pagination component



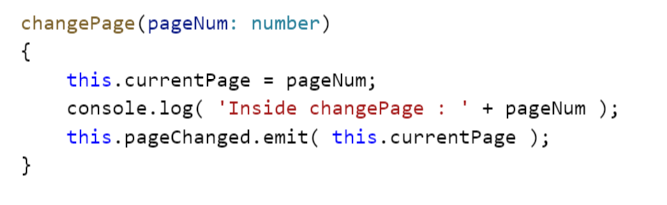
* + Implement OnChanges interface having methond ngOnChanges method which will be called each time bind will be changed



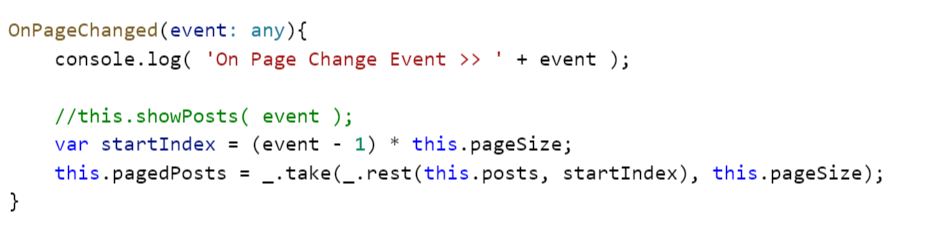
* + Configure input properties for pageSize and posts list.
  + Provide implementation of next and previous feature



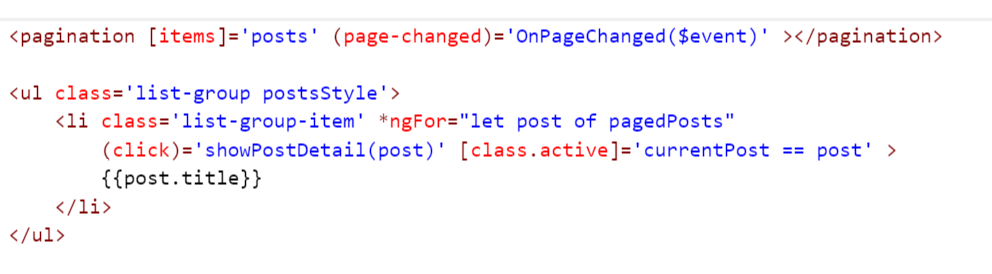
* + Provide implementation of changePage which will be called on selection of page



* + Expose value of page selection using @Output and EventEmitter
  + Provide implementation of even method on target component



* + Update post list component in post.component.ts



* Installation of javascript – underscore.js
  + Npm install dt~underscore –save
  + Install typings
  + Include script tag path in index.jsp
  + Typings install underscore –global --save

Materialize

Foundation

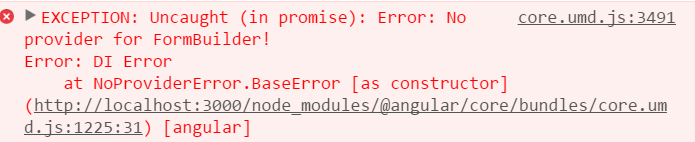
Semantic UI

**Common Errors:**

**Error-1:**

core.umd.js:3491 EXCEPTION: Uncaught (in promise): Error: No provider for FormBuilder!

Error: DI Error at NoProviderError.BaseError [as constructor]



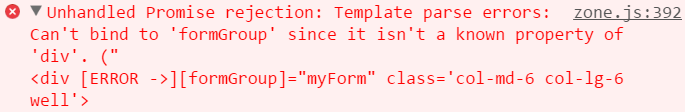
Solution: Add ReactiveFormsModule in app.module.ts

**Error-2:**

zone.js:392 Unhandled Promise rejection: Template parse errors:

Can't bind to 'formGroup' since it isn't a known property of 'div'. ("

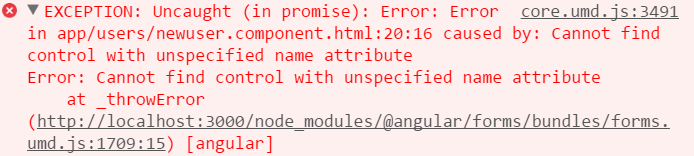
<div [ERROR ->][formGroup]="myForm" class='col-md-6 col-lg-6 well'>



Solution: include ReactiveFormModule. ( Even in sub module as well )

**Error-3:**

core.umd.js:3491 EXCEPTION: Uncaught (in promise): Error: Error in app/users/newuser.component.html:20:16 caused by: Cannot find control with unspecified name attribute Error: Cannot find control with unspecified name attribute at \_throwError

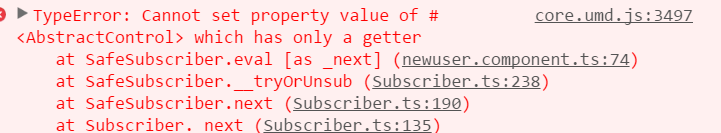


Solution: Declare & initialize all form elements in FormBuilder defination as below

Name : AbstractFormControl;

Email: AbstractFormControl;

**Error-4:**



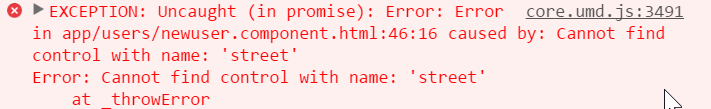
Solution: use [formGroup]=’myForm’ in <form> element.

**Error-5:**



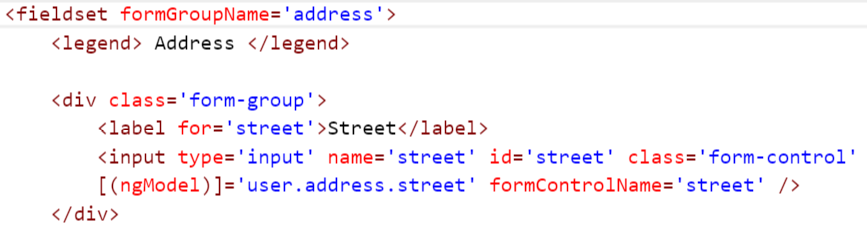
Solution: use either formControlName=’name’ or [formControl]=’name’

**Error-6:**

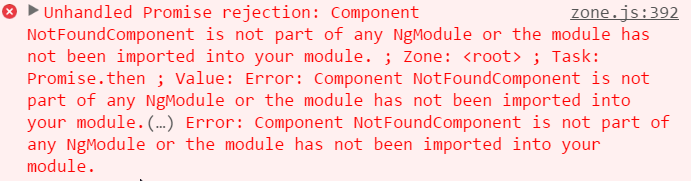


Solution: Check where street control is defined. If it is part of sub group, make sure you set

proper fromGroupName.



**Error-7:**



Solution: modify ngModel component of app.module.js and include NotFoundComponent

**Error-8:**

