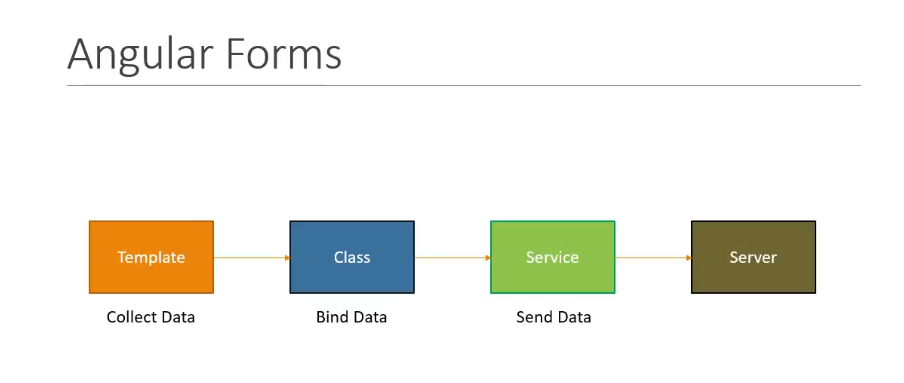
**Angular Form Fields :**

****

The Component template collects the html to collect the user data. The component class handles data binding.The collected data is sent to the server through a service. And to achiee this Angular provides the two approaches**.**

**1.Template driven form :** Most of the code is written in template

**2.Reactive forms or Model Driven form : Most** of the code is written in Component class

**1.Template driven form :** Using this approach most of the code is written under html template**.**

**Some keypoints we will use related to TDF :**

* It is easy to use and is very similar to Angular JS (Angular 1) form
* Two way data binding using ngModel
* Bulky html code and minimal component code
* In this approach Angular provides the ng-form directive also ng-model directive , Automatically tracks the form and form element state and validity

**Drawbacks of TDF :**

* Unit testing is a challenge. the form validation logic can not be unit tested. the only way to test the logic is to run and end to end test with a browser.
* Readibility of the code decreases with complex forms and validation.

**Q-> When should we go for TDF approach?**

Suitable for simple scenario where unit testing can be handled by the browser for more complex form with complex validation and where unit testing is absolutely necessasarygo with Reactive form.

**Steps to start working on TDF approach?**

1. Generate a new CLI Project.
2. Add the html form on which we have to work with
3. Data Binding
4. Tracking state and validity
5. Providing visual feedback
6. Displaying error message
7. Posting Data to server

**Command to install latest version of Angular CLI in your local machine :**

**npm install –g @angular/cli@latest**

Create a new Angular project : **ng new tdf**

After creating project goto tdf folder : **cd .\tdf**

Under this folder : **ng serve –o**

**Final step is to bootstrap styling to our project.**

**Note :** BootStrap is a css framework that gives us a variety of classes that we can apply to make our html look better.To add bootstrap css goto getBootstrap.com -> Documenttation-> Getting Started-> scrolldown and copy the css.

**Now in our Visul studion code : open index.html paste the css at the end of <head> tag**

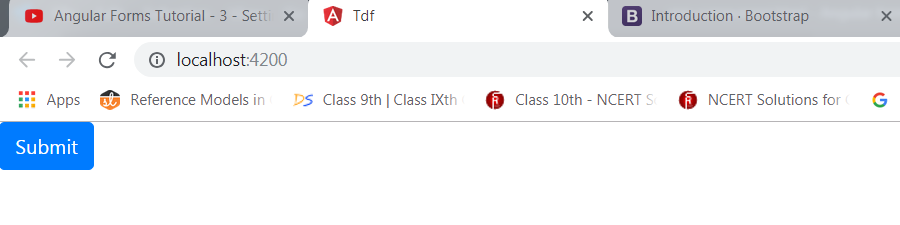
<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css" integrity="sha384-MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO" crossorigin="anonymous">

**Now open app.component.html :**

**Remove all the previous code and write**

<button class="btn btn-primary">Submit</button>

**And o/p :**

****

**So we can say that bootstrap is woring.**

**Adding Form html :**

We will be adding a bootcamp enrollment form so in app.component.html we begin with a div tag of class **container-fluid**

<div class="container-fluid">

<h1>BootCamp Enrollment form</h1>

<form>

<div class="form-group">

<label>Name</label>

<input type="text" class="form-control">

</div>

</form>

</div>

Here **form-group and form-control** are bootstrap classes.

App.componetnt.ts :

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

@Component({

selector: 'app-root',

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

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

})

export class AppComponent {

topics= ['Angular','React','Vue'];

}

App.component.html :

<div class="container-fluid">

<h1>BootCamp Enrollment form</h1>

<form>

<div class="form-group">

<label>Name</label>

<input type="text" class="form-control">

</div>

<div class="form-group">

<label>Email</label>

<input type="email" class="form-control">

</div>

<div class="form-group">

<label>Phone</label>

<input type="tel" class="form-control">

</div>

<div class="form-group">

<select class="custom-select">

<option selected>I am interested in </option>

<option \*ngFor="let topic of topics"> {{ topic }}</option>

</select>

</div>

<div>

<label>Time Preferance</label>

<div class="form-check">

<input class="form-check-input" type="radio" name="timePreference" value="morningBAtch">

<label class="form-check-label">Morning (9AM to 12PM)</label>

</div>

<div class="form-check">

<input class="form-check-input" type="radio" name="timePreference" value="eveningBAtch">

<label class="form-check-label">Evening (6PM to 9PM)</label>

</div>

</div>

<div class="form-check mb-3">

<input class="form-check-input" type="checkbox">

<label class="formcheck-label">

Send me Promotional Offer

</label>

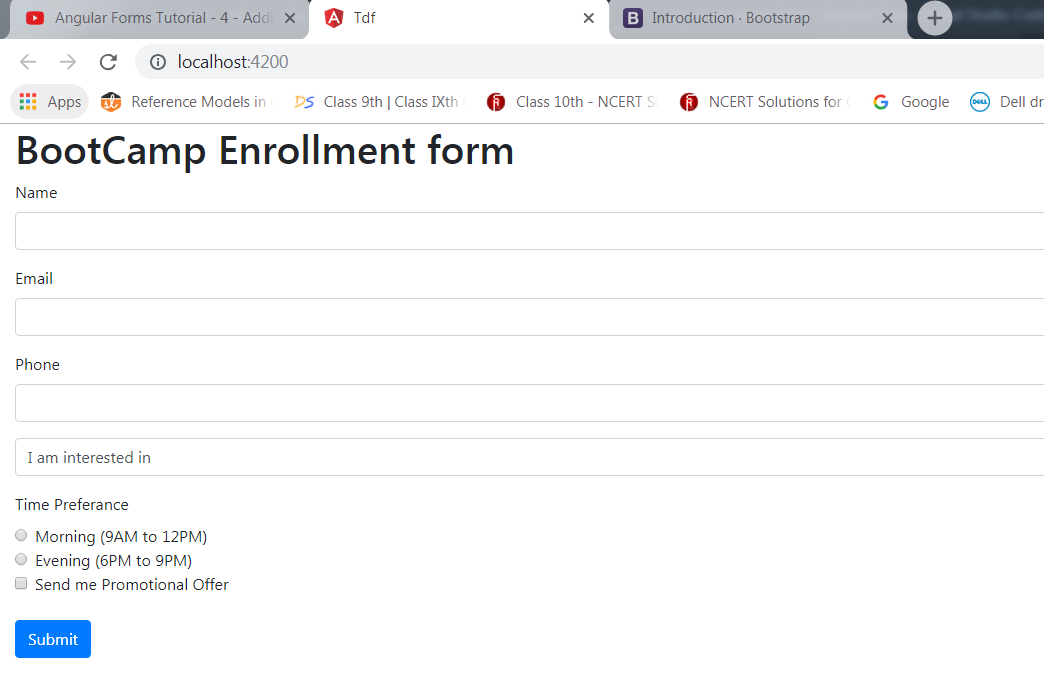
</div>

<button class="btn btn-primary" type="submit">Submit</button>

</form>

</div>

O/P:



**Data Binding with form fields :** Till now we just created an html form nothing related to Angular. First step we start to working with angular form is to import the forms module.so in **app.module.ts** we just import **FormsModule** and we add FormsModule in the import array

import { BrowserModule } from '@angular/platform-browser';

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

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

import { AppComponent } from './app.component';

@NgModule({

declarations: [

AppComponent

],

imports: [

BrowserModule,

FormsModule

],

providers: [],

bootstrap: [AppComponent]

})

export class AppModule { }

Now if we go to app.component.html we see its still a basic html form but behind the scene Angular’s magic is happening. Anytime we use a form tag Angular attaches a ng-form directive to the form tag which gives us the valuable information about that particular form. It tells us what the value to different controls are and whether the value is valid or invalid.

**How do we get a hold of refrence to ng-form directive?**

**Ans🡪** Using a template refrence variable

**So to the form tag we are going to add a template reference variable named userform and to this we assign a string “ngForm”**

<form #useForm="ngForm">

**So the ngForm directive exports itself as a string ngForm and by assigning it to a reference variable we have a reference to the directive itself. now we just mention the directive gives an access to the value of the form control. it does so using the value properties. So using interpolation we are going to pick the form control value.**

<h1>BootCamp Enrollment form</h1>

<form #userForm="ngForm">

{{ userForm.value | json }}

**But this is not sufficient to map all the form control value to bind so we need to add ngModel directive with each form fields and also need to give name directive for each form controller**

<div class="container-fluid">

<h1>BootCamp Enrollment form</h1>

<form #userForm="ngForm">

{{ userForm.value | json }}

<div class="form-group">

<label>Name</label>

<input type="text" class="form-control" name="useName" ngModel >

</div>

<div class="form-group">

<label>Email</label>

<input type="email" class="form-control" name="email"ngModel>

</div>

<div class="form-group">

<label>Phone</label>

<input type="tel" class="form-control" name="Phone"ngModel>

</div>

<div class="form-group">

<select class="custom-select" name="Topic" ngModel>

<option selected>I am interested in </option>

<option \*ngFor="let topic of topics"> {{ topic }}</option>

</select>

</div>

<div>

<label>Time Preferance</label>

<div class="form-check" >

<input class="form-check-input" ngModel type="radio" name="timePreference" value="morningBAtch">

<label class="form-check-label">Morning (9AM to 12PM)</label>

</div>

<div class="form-check">

<input class="form-check-input" ngModel type="radio" name="timePreference" value="eveningBAtch">

<label class="form-check-label">Evening (6PM to 9PM)</label>

</div>

</div>

<div class="form-check mb-3">

<input class="form-check-input" name="subscription" ngModel type="checkbox">

<label class="formcheck-label">

Send me Promotional Offer

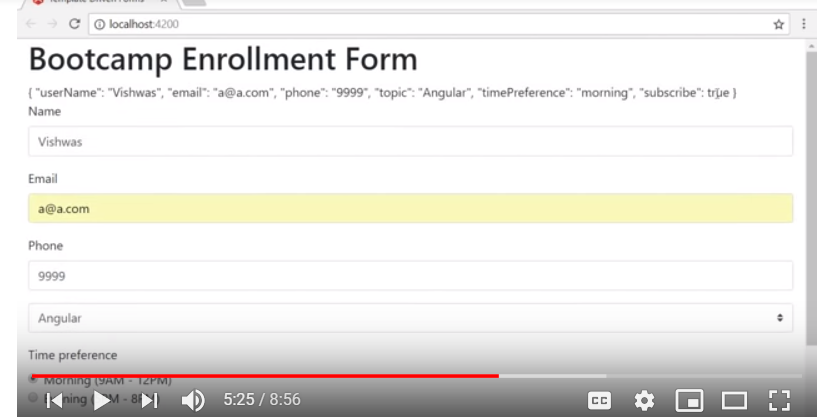
</label>

</div>

<button class="btn btn-primary" type="submit">Submit</button>

</form>

</div>

**O/P: **