Module 9: Validating Angular Forms

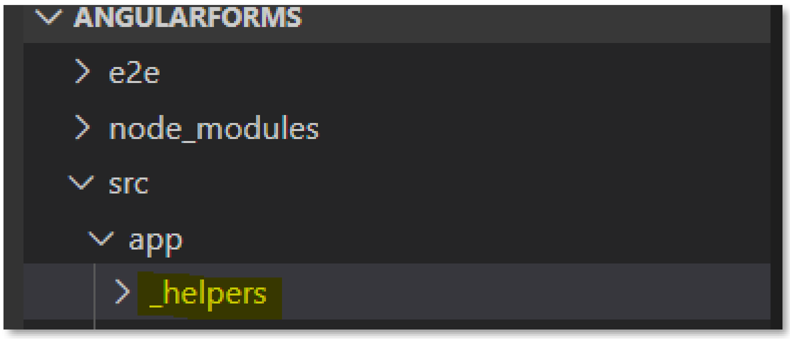
Demo Document 1 – Template driven user registration form with validation

Creating Template driven user registration form and add validation to form

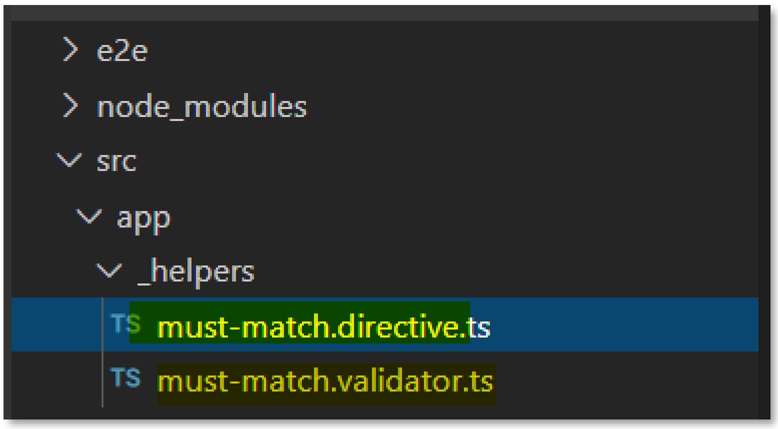
In this demo, we will see how to add validation for different fields for template driven form created in module 8 demo 1

Step 1: Open project Module8-Demo1, created in module 8 demo1

Step 2: Create **"_helpers"** folder under the app folder as shown below.



Step 3- Create must-match directive and must-match validator files as shown below.



Purpose of this is to validate fields in forms. It verifies if the password entered in the confirm password field matches the password entered in the password field. If you enter different password in password and confirm password fields then mustmatch directive will trigger error message.

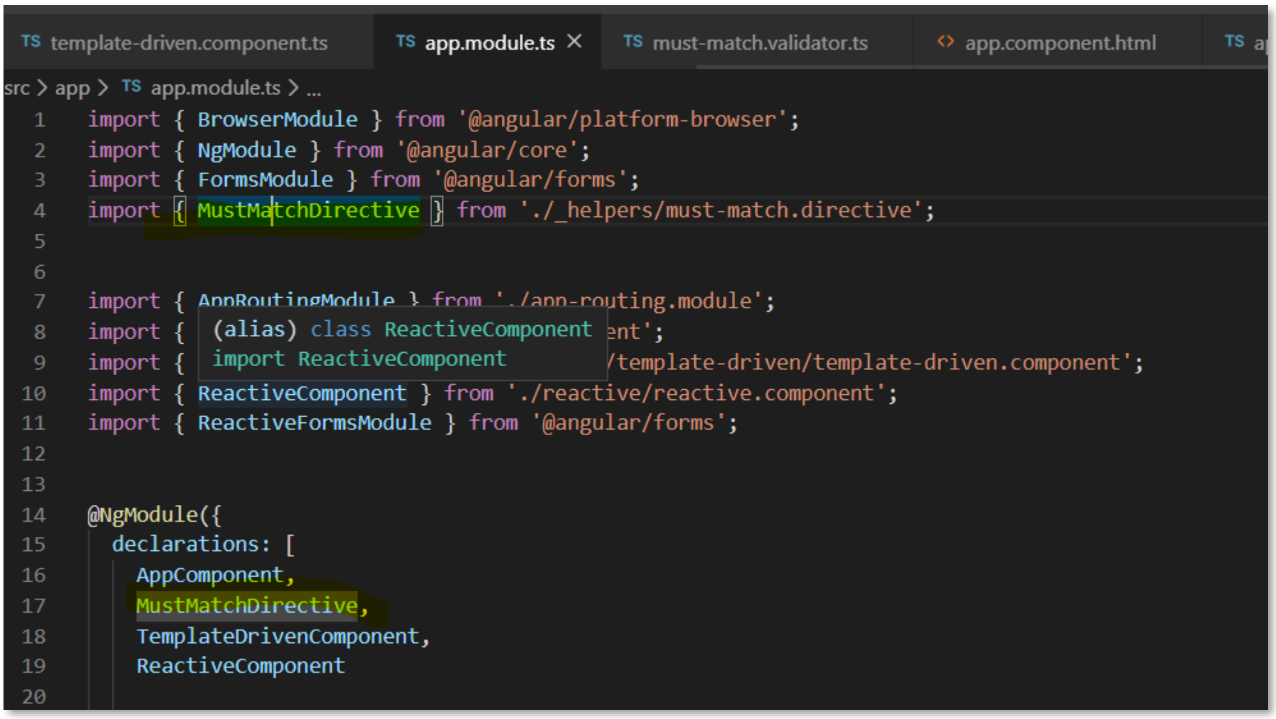
Step 4 – The must-match.directive.ts contains the below code

```
import { Directive, Input } from '@angular/core';
import { NG_VALIDATORS, Validator, ValidationErrors, FormGroup } from '@angular/fo
rms';

import { MustMatch } from './must-match.validator';

@Directive({
    selector: '[mustMatch]',
    providers: [{ provide: NG_VALIDATORS, useExisting: MustMatchDirective, multi:
true }]
})
export class MustMatchDirective implements Validator {
    @Input('mustMatch') mustMatch: string[] = [];

    validate(formGroup: FormGroup): ValidationErrors {
        return MustMatch(this.mustMatch[0], this.mustMatch[1])(formGroup);
    }
}
```

Step 5 – The must-match.validator.ts contains the below code

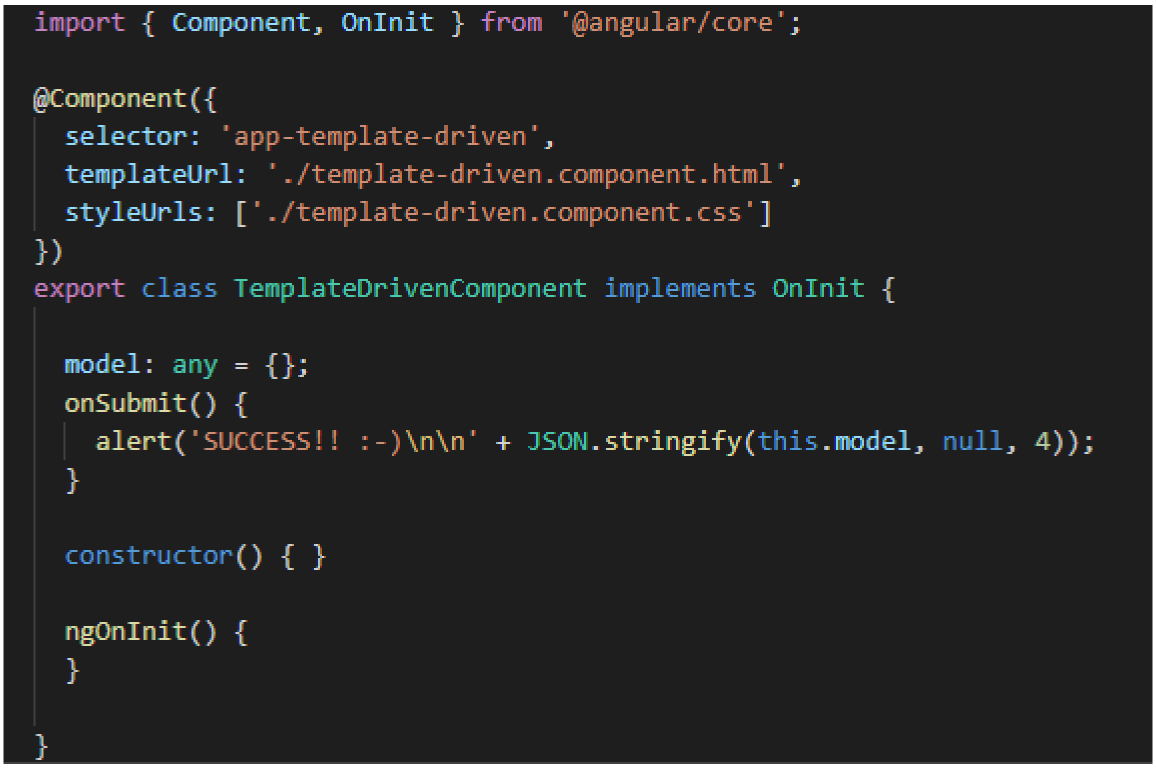
```
import { FormGroup } from '@angular/forms';

// custom validator to check that two fields match
export function MustMatch(controlName: string, matchingControlName: string) {
    return (formGroup: FormGroup) => {
        const control = formGroup.controls[controlName];
        const matchingControl = formGroup.controls[matchingControlName];

        // return null if controls haven't initialised yet
        if (!control || !matchingControl) {
          return null;
        }

        // return null if another validator has already found an error on the matc
hingControl
        if (matchingControl.errors && !matchingControl.errors.mustMatch) {
            return null;
        }

        // set error on matchingControl if validation fails
        if (control.value !== matchingControl.value) {
            matchingControl.setErrors({ mustMatch: true });
        } else {
            matchingControl.setErrors(null);
        }
    }
}
```

Step 6- Open app.module.ts file and import mustmatch directive as below.


TS template-driven.component.ts
TS app.module.ts ×
TS must-match.validator.ts
<> app.component.html
src > app > TS app.module.ts > ...
(alias) class ReactiveComponent
import ReactiveComponent

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { FormsModule } from '@angular/forms';
import { MustMatchDirective } from './_helpers/must-match.directive';


import { AppRoutingModule } from './app-routing.module';
import {                                   ent';
import {                                   /template-driven/template-driven.component';
import { ReactiveComponent } from './reactive/reactive.component';
import { ReactiveFormsModule } from '@angular/forms';


@NgModule({
  declarations: [
    AppComponent,
    MustMatchDirective,
    TemplateDrivenComponent,
    ReactiveComponent

```

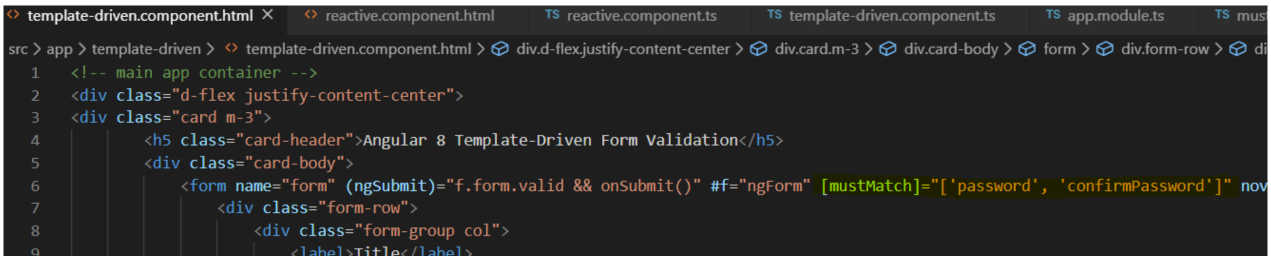
Step 7- Open template-driven.component.ts file and create a **“onSubmit()”** method



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

@Component({
  selector: 'app-template-driven',
  templateUrl: './template-driven.component.html',
  styleUrls: ['./template-driven.component.css']
})
export class TemplateDrivenComponent implements OnInit {

  model: any = {};
  onSubmit() {
    alert('SUCCESS!! :-)\n\n' + JSON.stringify(this.model, null, 4));
  }

  constructor() { }

  ngOnInit() {
  }

}
```

Step 8- Open template-driven.component.html and add built-in validators to validate the form fields.

Add mustmatch directive for password validation



template-driven.component.html × | reactive.component.html | reactive.component.ts | template-driven.component.ts | app.module.ts | mus

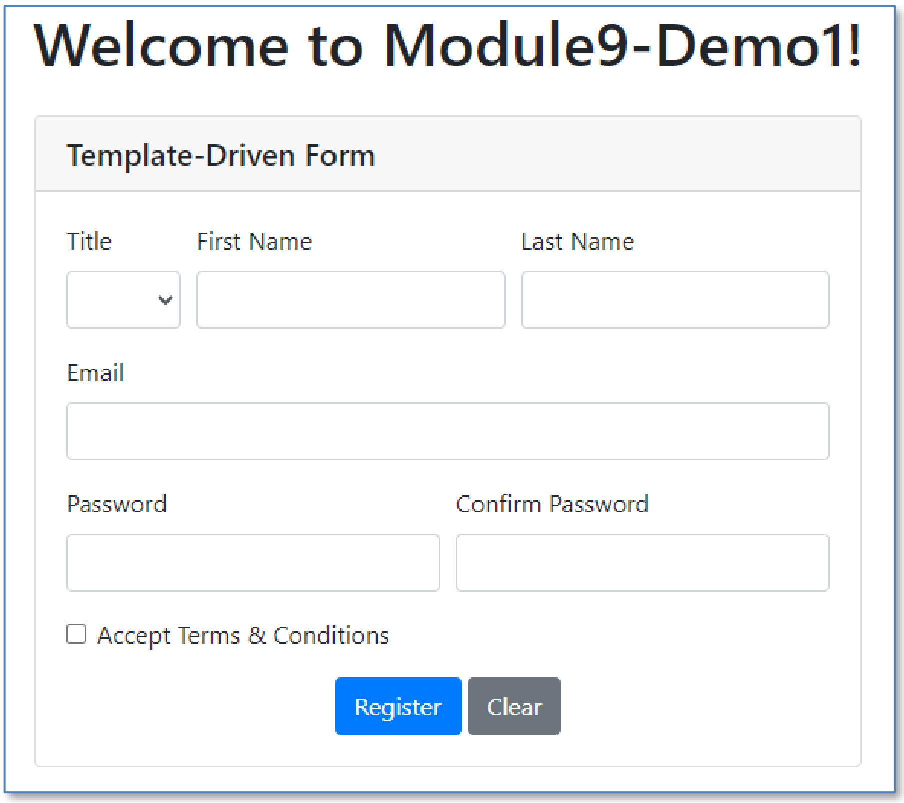
src > app > template-driven > template-driven.component.html > div.d-flex.justify-content-center > div.card.m-3 > div.card-body > form > div.form-row > di

```
<!-- main app container -->
<div class="d-flex justify-content-center">
<div class="card m-3">
        <h5 class="card-header">Angular 8 Template-Driven Form Validation</h5>
        <div class="card-body">
            <form name="form" (ngSubmit)="f.form.valid && onSubmit()" #f="ngForm" [mustMatch]="['password', 'confirmPassword']" nov
                <div class="form-row">
                    <div class="form-group col">
                        <label>Title</label>
```

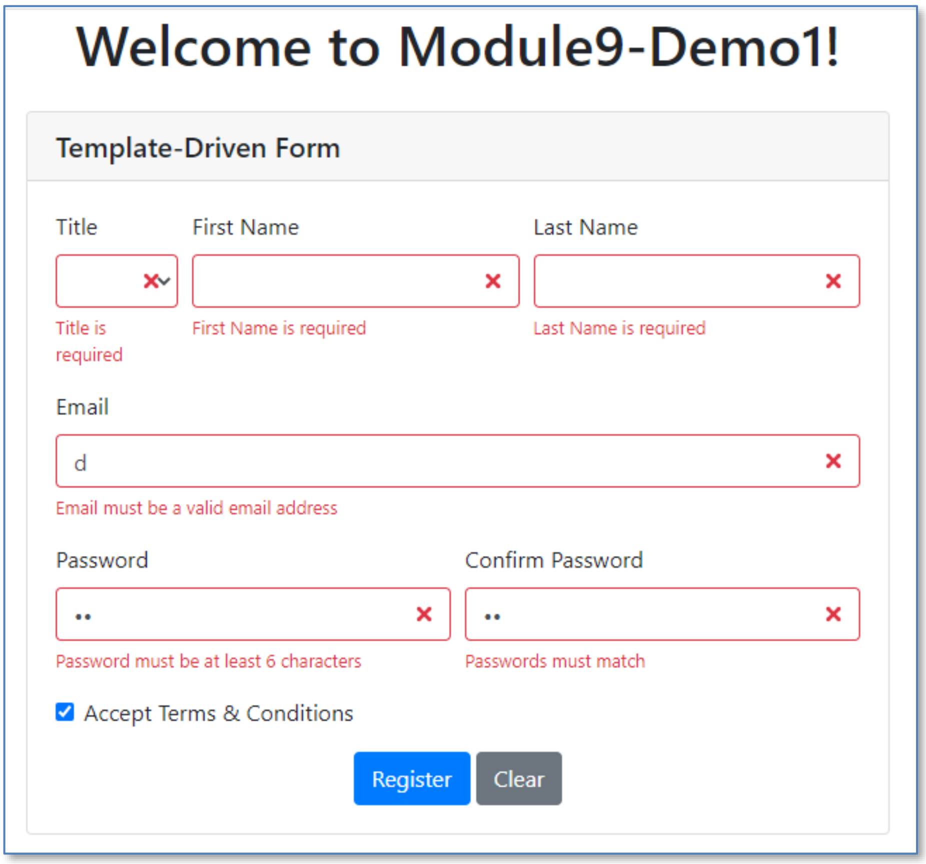
Note that we are adding all validation in .html file for template driven form

Step 9- Run app using ng serve

- This will open below window



- Now click on register button, you will see that all validation messages will be triggered as below


Welcome to Module9-Demo1!

Template-Driven Form

Title

Title is required

First Name

First Name is required

Last Name

Last Name is required

Email

d

Email must be a valid email address

Password

••

Password must be at least 6 characters

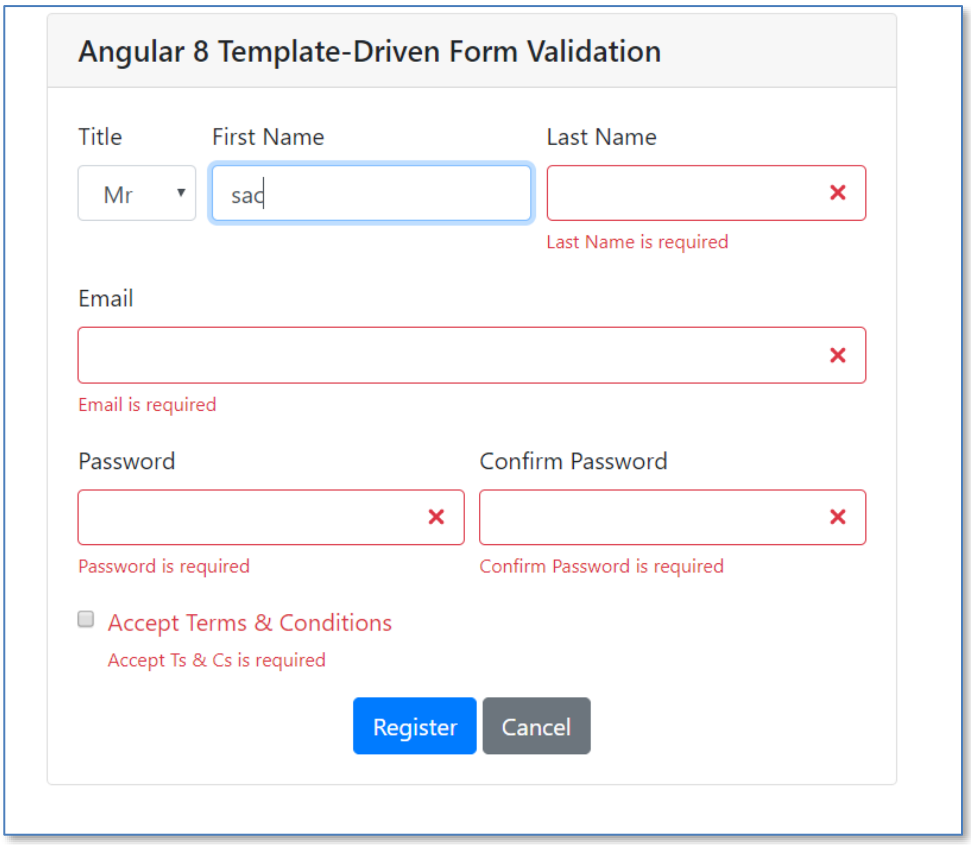
Confirm Password

••

Passwords must match

☑ Accept Terms & Conditions

- Once you start adding values to fields validation errors will be removed


Angular 8 Template-Driven Form Validation

Title	First Name	Last Name
Mr	sac	
		Last Name is required

Email

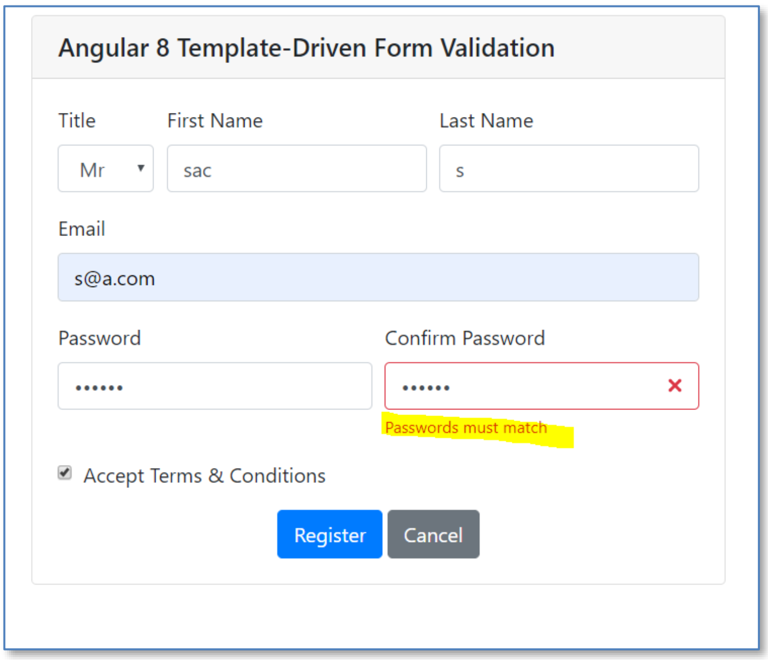
Email is required

Password	Confirm Password
Password is required	Confirm Password is required

☐ Accept Terms & Conditions

Accept Ts & Cs is required

- If you enter different password in password and confirm password fields then must-match directive will trigger error message


Angular 8 Template-Driven Form Validation

Title	First Name	Last Name
Mr	sac	s

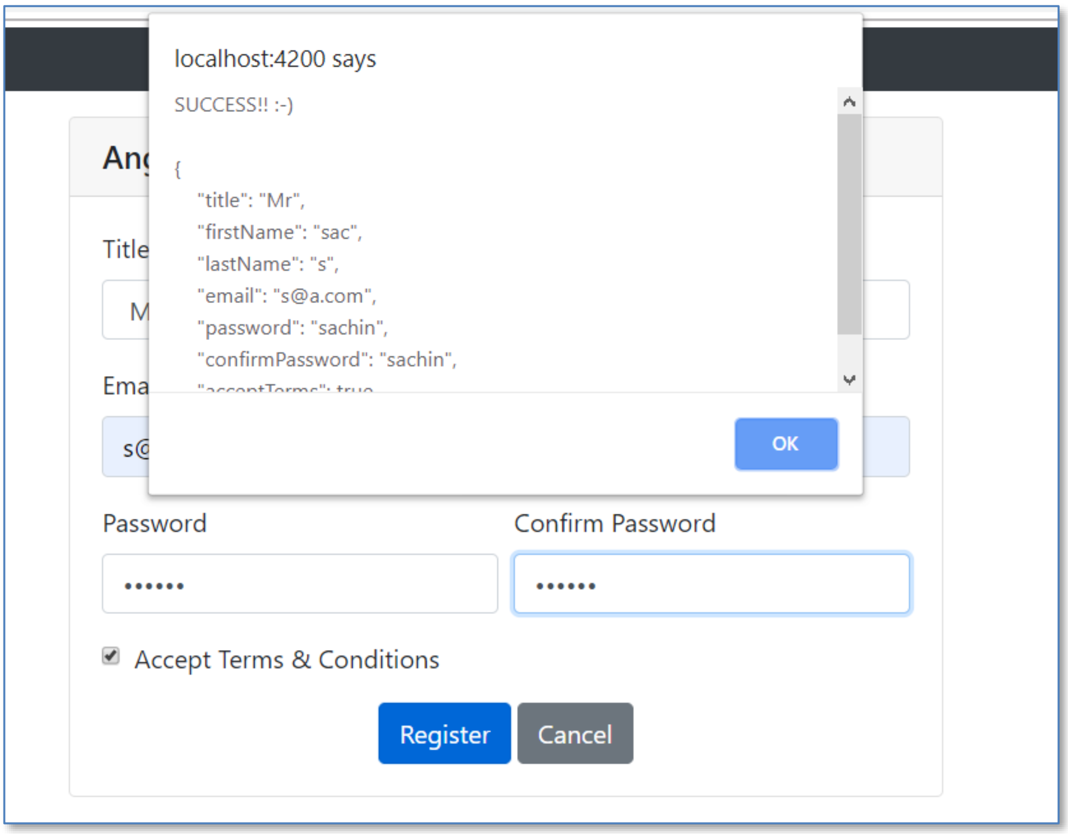
Email

s@a.com

Password	Confirm Password
••••••	••••••
	Passwords must match

☑ Accept Terms & Conditions

Step 10 – Once all errors are corrected and form is submitted it will show success popup as shown below


localhost:4200 says

SUCCESS!! :-)

```
{
    "title": "Mr",
    "firstName": "sac",
    "lastName": "s",
    "email": "s@a.com",
    "password": "sachin",
    "confirmPassword": "sachin",
```