Angular 6 CRUD Example

By Dhiraj, 23 May,2018

👁 37K

In this article we will be building an Angular 6 application step by step from scratch with sample example. We will be generating our Angular 6 Hero application using angular CLI and then modify it to have a user management project where there will be a login screen for an admin and post login he can perform CRUD operations such as create, read, update and delete user with the sample REST API exposed using HttpClientModule. We will also be using RouterModule to have routing enabled. Also, at the end we will be integrating material designing with our Angular 6 app.

In my last examples of angular, we had covered many topics on angular such as Spring Boot Angular Example, Angular JWT Authentication and Material Design with Angular and many more and in due course we had exposed some REST APIs to be consumed by the angular client. Hence, we will be using the same set of APIs here for the CRUD operation while building our Angular 6 app. Later, in the article I will be providing the API details. But for now, here is the spring controller class for the same on github.

Table of Contents

Angular 6 Release Highlights

Angular v6 is the first release of Angular that unifies the Framework, Material and CLI. @angular/core now depends on.

- TypeScript 2.7
- RxJS 6.0.0
- tslib 1.9.0

Both the CLI and generated project have dependencies that require Node 8.9 or higher, together with NPM 5.5.1 or higher. For this project, I have npm 5.6.0 and node v8.11.2 installed on my local system. You can download the latest version of Node js from here - Node Js Official. To update NPM, you can run the following command in the terminal.

```
npm i npm@latest -g
```

If u have an @angular/cli version older then 6, then run following command to install the latest versions:

```
npm uninstall -g @angular/cli
npm cache clean
npm install -g @angular/cli
```

To install a specific version, you can use `npm install -g @angular/cli@1.4.9`

For a complete change log, you can visit here.

Recommended

- Material Sidenav Example
- Rxjs Tutorial
- Angular Data Table Example
- Spring Boot Jwt Auth
- Spring Boot Angular Spring Data Example
- Angular Material Dialog
- Angular Material App
- Typescript Tutorial
- Angular Hide Sidenav Login Page

Generating Angular 6 Project

Once, the npm and node is upgraded to the latest version, you can run following command to generate angular 6 project in any location of your choice.

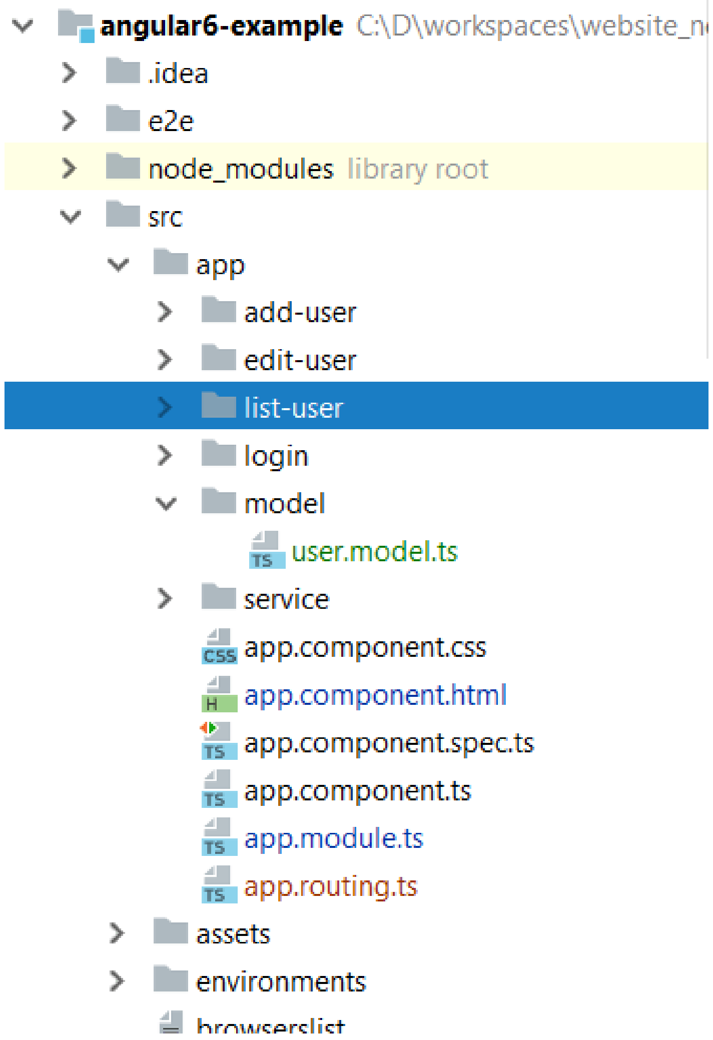
```
ng new angular6-example
```

Doing so, our angular 6 application is generated with TypeScript 2.7 & RxJS 6.RxJS 6 contains some breaking changes and hence a new package, rxjs-compat, can be installed alongside RxJS 6 to provide a compatibility layer while upgrading your code to the new syntax.

Angular 6 Project Structure

Once the project is generated, you can import it in your favourite directory and get started with it. Following will be the final structure of our project. Also, you can run following commands to see angular 6 app running at localhost:4200

```
cd angular6-example
ng serve
```



```
angular6-example C:\D\workspaces\website_n
  .idea
  e2e
  node_modules library root
  src
    app
      add-user
      edit-user
      list-user
      login
      model
        user.model.ts
      service
      app.component.css
      app.component.html
      app.component.spec.ts
      app.component.ts
      app.module.ts
      app.routing.ts
    assets
    environments
    browserslist
```

There are certain files generated with CLI command which we need to understand here. We have angular.json file generated that has all the application configuration parameters. The configuration related to welcome html file as index.html, main.ts where all the modules are bundled. You can also find the final application output directory configuration and configuration specific to environment such as dev and prod can be found here.

We have package.json file that has information about all the project dependencies. We have tsconfig.json for typescript configuration.Inside the scr/app folder we have our components defined and when the request for localhost:4200 is made, AppComponent is loaded in the browser.

As discussed above, we will have multiple components such as login and add-user, edit-user, create-user. Once an admin login sucessfully, he can perform user CRUD operation.Following are the commands to generate our components.

```
ng g component login
ng g component add-user
ng g component edit-user
ng g component list-user
```

Angular CLI Useful Commands

```
ng g component my-new-component
ng g directive my-new-directive
ng g pipe my-new-pipe
ng g service my-new-service
ng g class my-new-class
ng g guard my-new-guard
ng g interface my-new-interface
ng g enum my-new-enum
ng g module my-module
```

Angular 6 Routing

Following is our routing configurtion.We have configured to use LoginComponent as a default component.Also, do not forget to include it in the main module - `app.module.ts`

app.routing.ts

```
import { RouterModule, Routes } from '@angular/router';
import {LoginComponent} from "./login/login.component";
import {AddUserComponent} from "./add-user/add-user.component";
import {ListUserComponent} from "./list-user/list-user.component";
import {EditUserComponent} from "./edit-user/edit-user.component";

const routes: Routes = [
  { path: 'login', component: LoginComponent },
  { path: 'add-user', component: AddUserComponent },
  { path: 'list-user', component: ListUserComponent },
  { path: 'edit-user', component: EditUserComponent },
  {path : '', component : LoginComponent}
];

export const routing = RouterModule.forRoot(routes);
```

The AuthenticationService here is just a placeholder because we have actually hard-coded the username and pasword in login component. I have discussed about authentication in angular application in my another post here - JWT Authentication in Angular App

app.module.ts

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

import { AppComponent } from './app.component';
import { LoginComponent } from './login/login.component';
import {routing} from "./app.routing";
import {AuthenticationService} from "./service/auth.service";
import {ReactiveFormsModule} from "@angular/forms";
import {HttpClientModule} from "@angular/common/http";
import { AddUserComponent } from './add-user/add-user.component';
import { EditUserComponent } from './edit-user/edit-user.component';
import {ListUserComponent} from "./list-user/list-user.component";
import {UserService} from "./service/user.service";

@NgModule({
  declarations: [
    AppComponent,
    LoginComponent,
    ListUserComponent,
    AddUserComponent,
    EditUserComponent
  ],
  imports: [
    BrowserModule,
    routing,
    ReactiveFormsModule,
    HttpClientModule
  ],
  providers: [AuthenticationService, UserService],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

Service in Angular 6 Application

Following is the implementation of our UserService. It has all the API details that is required for the CRUD operation.Also, for testing purpose, you can use following fake users.

```
let fakeUsers = [{id: 1, firstName: 'Dhiraj', lastName: 'Ray', email: 'dhiraj@gmail.com'},
      {id: 1, firstName: 'Tom', lastName: 'Jac', email: 'Tom@gmail.com'},
      {id: 1, firstName: 'Hary', lastName: 'Pan', email: 'hary@gmail.com'},
      {id: 1, firstName: 'praks', lastName: 'pb', email: 'praks@gmail.com'},
    ];
    return Observable.of(fakeUsers).delay(5000);
```

user.service.ts

```
import { Injectable } from '@angular/core';
import { HttpClient } from '@angular/common/http';
import {User} from "../model/user.model";

@Injectable()
export class UserService {
  constructor(private http: HttpClient) { }
  baseUrl: string = 'http://localhost:8080/user-portal/users';

  getUsers() {
    return this.http.get<User[]>(this.baseUrl);
  }

  getUserById(id: number) {
    return this.http.get<User>(this.baseUrl + '/' + id);
  }

  createUser(user: User) {
    return this.http.post(this.baseUrl, user);
  }

  updateUser(user: User) {
    return this.http.put(this.baseUrl + '/' + user.id, user);
  }

  deleteUser(id: number) {
    return this.http.delete(this.baseUrl + '/' + id);
  }
}
```

Creating Components in Angular 6

We will have a login component. After successfull login, user will be redirected to list page and from there the user can perform crud operation. So, let us first start creating LoginComponent. To protect the website from spam and abuse, it is recommended to use captcha. In my another article, I have integrated Google Recaptcha with this LoginComponent here - Google ReCaptcha Integration with Angular and Spring Boot

login.component.html

```
<div class="row">

  <div class="col-md-6 col-md-offset-2">
    <h1>Login </h1>
    <form [formGroup]="loginForm" (ngSubmit)="onSubmit()">
      <div class="form-group">
        <label for="email">Email address:</label>
        <input type="email" class="form-control" formControlName="email" id="email">
        <div *ngIf="submitted && loginForm.controls.email.errors" class="error">
          <div *ngIf="loginForm.controls.email.errors.required">Email is required</div>
        </div>
      </div>
      <div class="form-group">
        <label for="pwd">Password:</label>
        <input type="password" class="form-control" formControlName="password" id="pwd">
        <div *ngIf="submitted && loginForm.controls.password.errors" class="error">
          <div *ngIf="loginForm.controls.password.errors.required">Password is required</div>
        </div>
      </div>
      <button class="btn btn-default">Login</button>
      <div *ngIf="invalidLogin" class="error">
        <div>Invalid credentials.</div>
      </div>
    </form>
  </div>
</div>
```

login.component.ts

```
import { Component, OnInit } from '@angular/core';
import {FormBuilder, FormGroup, Validators} from "@angular/forms";
import {Router} from "@angular/router";
import {first} from "rxjs/operators";
import {AuthenticationService} from "../service/auth.service";

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

  loginForm: FormGroup;
  submitted: boolean = false;
  invalidLogin: boolean = false;
  constructor(private formBuilder: FormBuilder, private router: Router, private authService: AuthenticationService) { }

  onSubmit() {
    this.submitted = true;
    if (this.loginForm.invalid) {
      return;
    }
    if(this.loginForm.controls.email.value == 'dhiraj@gmail.com' && this.loginForm.controls.password.value == 'password') {
        this.router.navigate(['list-user']);
    }else {
      this.invalidLogin = true;
    }
  }

  ngOnInit() {
    this.loginForm = this.formBuilder.group({
      email: ['', Validators.required],
      password: ['', Validators.required]
    });
  }



}
```

Following is list-user.component.html.

```
<div class="col-md-6">
  <h2> User Details</h2>
  <button class="btn btn-danger" (click)="addUser()"> Add User</button>
  <table class="table table-striped">
    <thead>
    <tr>
      <th class="hidden">Id</th>
      <th>FirstName</th>
      <th>LastName</th>
      <th>Email</th>
      <th>Action</th>
    </tr>
    </thead>
    <tbody>
    <tr *ngFor="let user of users">
      <td class="hidden"></td>
      <td></td>
      <td></td>
      <td></td>
      <td><button class="btn btn-danger" (click)="deleteUser(user)"> Delete</button
>
        <button class="btn btn-danger" (click)="editUser(user)" style="margin-left:
20px;"> Edit</button></td>
    </tr>
    </tbody>
  </table>
</div>
```

list-user.component.ts

```
import { Component, OnInit } from '@angular/core';
import {Router} from "@angular/router";
import {UserService} from "../service/user.service";
import {User} from "../model/user.model";

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

  users: User[];

  constructor(private router: Router, private userService: UserService) { }

  ngOnInit() {
    this.userService.getUsers()
      .subscribe( data => {
        this.users = data;
      });
  }

  deleteUser(user: User): void {
    this.userService.deleteUser(user.id)
      .subscribe( data => {
        this.users = this.users.filter(u => u !== user);
      })
  };

  editUser(user: User): void {
    localStorage.removeItem("editUserId");
    localStorage.setItem("editUserId", user.id.toString());
    this.router.navigate(['edit-user']);
  };

  addUser(): void {
    this.router.navigate(['add-user']);
  };
}
```

add-user.component.html

```
<div class="col-md-6">
  <h2 class="text-center">Add User</h2>
  <form [formGroup]="addForm" (ngSubmit)="onSubmit()">
    <div class="form-group">
      <label for="email">Email address:</label>
      <input type="email" formControlName="email" placeholder="Email" name="email"
class="form-control" id="email">
    </div>

    <div class="form-group">
      <label for="firstName">First Name:</label>
      <input formControlName="firstName" placeholder="First Name" name="firstName"
class="form-control" id="firstName">
    </div>

    <div class="form-group">
      <label for="lastName">Last Name:</label>
      <input formControlName="lastName" placeholder="Last name" name="lastName" cla
ss="form-control" id="lastName">
    </div>

    <button class="btn btn-success">Update</button>
  </form>
</div>
```

add-user.component.ts

```
import { Component, OnInit } from '@angular/core';
import {FormBuilder, FormGroup, Validators} from "@angular/forms";
import {UserService} from "../service/user.service";
import {first} from "rxjs/operators";
import {Router} from "@angular/router";

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

  constructor(private formBuilder: FormBuilder,private router: Router, private user
Service: UserService) { }

  addForm: FormGroup;

  ngOnInit() {

    this.addForm = this.formBuilder.group({
      id: [],
      email: ['', Validators.required],
      firstName: ['', Validators.required],
      lastName: ['', Validators.required]
    });

  }

  onSubmit() {
    this.userService.createUser(this.addForm.value)
      .subscribe( data => {
        this.router.navigate(['list-user']);
      });
  }

}
```

edit-user.component.html

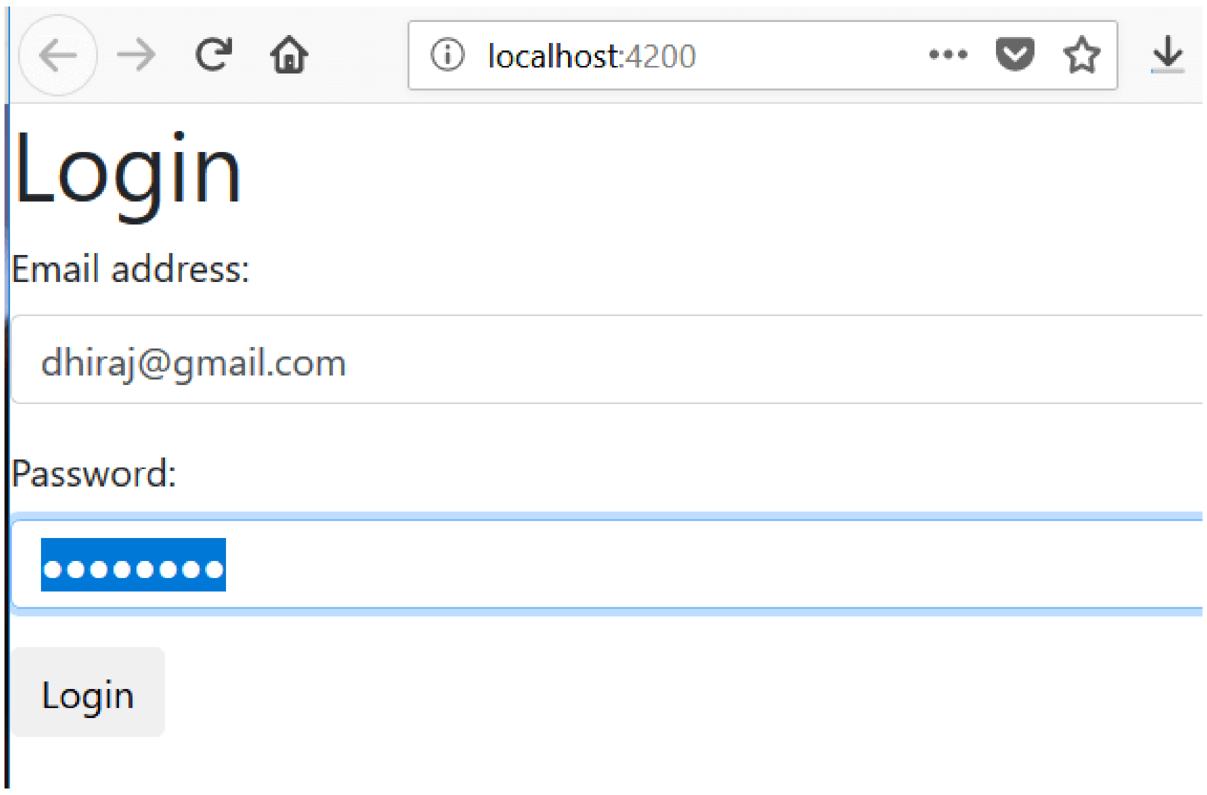
```
<div class="col-md-6">
  <h2 class="text-center">Edit User</h2>
  <form [formGroup]="editForm" (ngSubmit)="onSubmit()">
    <div class="form-group">
      <label for="email">Email address:</label>
      <input type="email" formControlName="email" placeholder="Email" name="email"
class="form-control" id="email">
    </div>

    <div class="form-group">
      <label for="firstName">First Name:</label>
      <input formControlName="firstName" placeholder="First Name" name="firstName"
class="form-control" id="firstName">
    </div>

    <div class="form-group">
      <label for="lastName">Last Name:</label>
      <input formControlName="lastName" placeholder="Last name" name="lastName" cla
ss="form-control" id="lastName">
    </div>

    <button class="btn btn-success">Update</button>
  </form>
</div>
```

edit-user.component.ts

```
import { Component, OnInit } from '@angular/core';
import {UserService} from "../service/user.service";
import {Router} from "@angular/router";
import {User} from "../model/user.model";
import {FormBuilder, FormGroup, Validators} from "@angular/forms";
import {first} from "rxjs/operators";

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

  user: User;
  editForm: FormGroup;
  constructor(private formBuilder: FormBuilder,private router: Router, private userService: UserService) { }

  ngOnInit() {
    let userId = localStorage.getItem("editUserId");
    if(!userId) {
      alert("Invalid action.")
      this.router.navigate(['list-user']);
      return;
    }
    this.editForm = this.formBuilder.group({
      id: [],
      email: ['', Validators.required],
      firstName: ['', Validators.required],
      lastName: ['', Validators.required]
    });
    this.userService.getUserById(+userId)
      .subscribe( data => {
        this.editForm.setValue(data);
      });
  }

  onSubmit() {
    this.userService.updateUser(this.editForm.value)
      .pipe(first())
      .subscribe(
        data => {
          this.router.navigate(['list-user']);
        },
        error => {
          alert(error);
        });
  }

}
```

Testing Angular 6 Application

Once our API is deployed, we can simply traverse to the client project and run the command `ng serve` and hit localhost:4200



Login

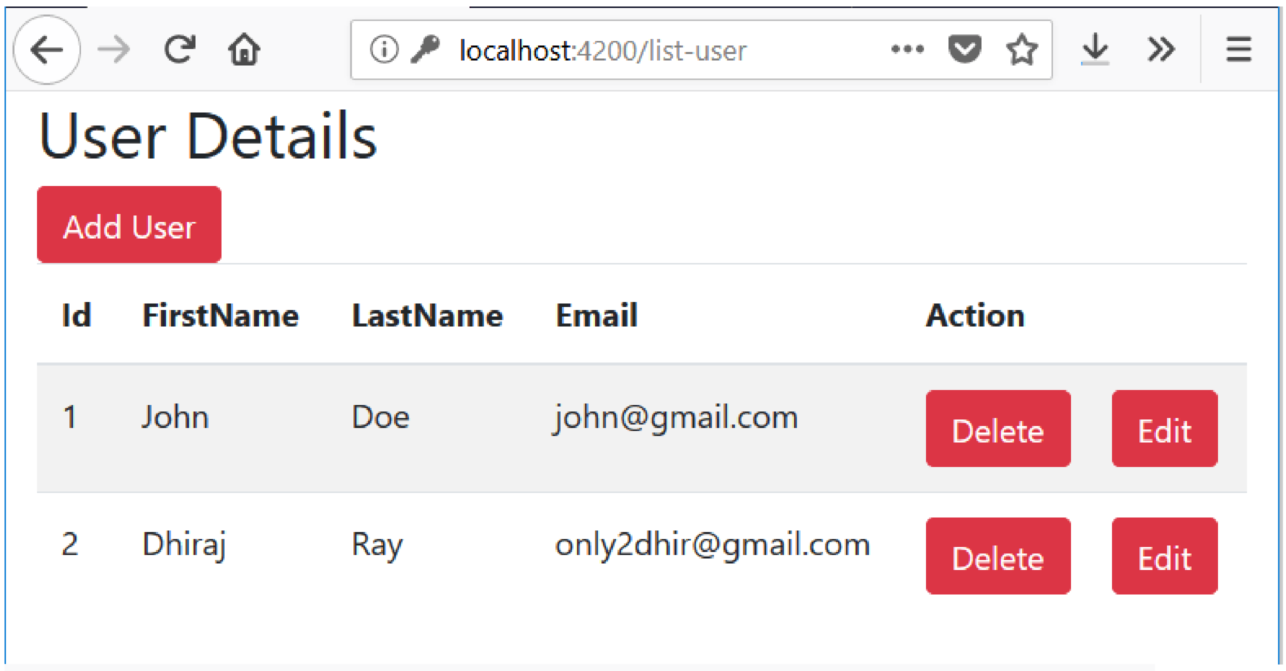
Email address:

dhiraj@gmail.com

Password:

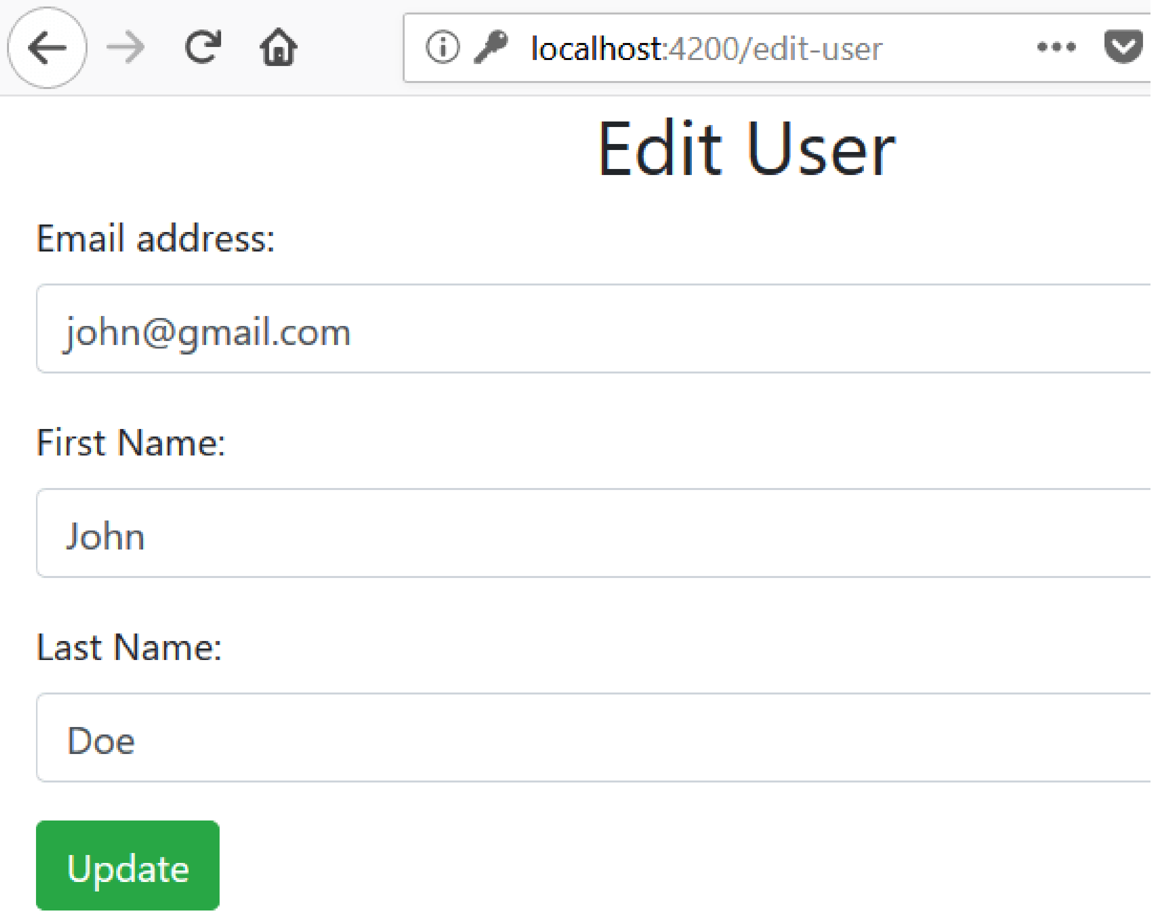
••••••••

After login, you can see following screen with list of users. On this page, you can perform actions to add, edit and delete user. Following is a sample screen for edit user.



User Details

Id	FirstName	LastName	Email	Action	
1	John	Doe	john@gmail.com	Delete	Edit
2	Dhiraj	Ray	only2dhir@gmail.com	Delete	Edit



Edit User

Email address:

john@gmail.com

First Name:

John

Last Name:

Doe

Adding Material Design in Angular 6 App

With the release of Angular 6, we can directly run `ng add @angular/material` command to add material design capabilities to an existing Angular application. By executing below command we will be installing Angular Material and the corresponding theming into the project.

Here, we will be only integrating material designing with Angular 6. Here is my another article, where we have intgegrated [SideNav with Angular 6 Material](#)

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

But in our existing project, we have included bootstrap 4 earlier to provide some styling and hence let us remove it first. To so so, open `index.html` and remove below scripts.

```
<script src="cdnjs.cloudflare.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
<script src="cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/4.0.0/js/bootstrap.mi
n.js"></script>
<link rel="stylesheet" href="cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/4.0.
0/css/bootstrap.min.css">
```

Once, these scripts are removed, run above CLI command.

Now, we will create a new module with file name `material.module.ts` and import all the material design related module here and import it into our `app.module.ts` file.

material.module.ts

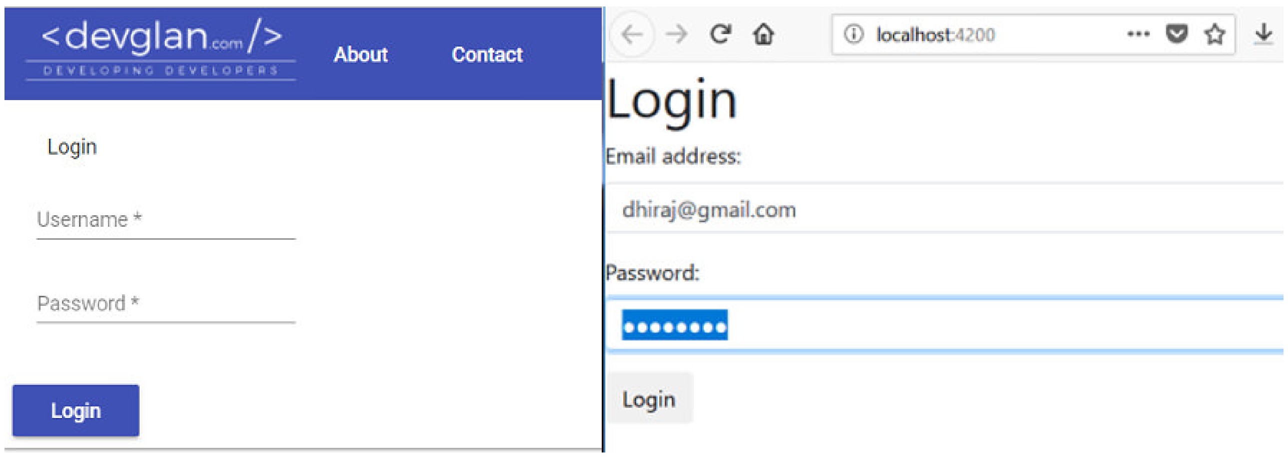
```
import {NgModule} from '@angular/core';
import { CommonModule } from '@angular/common';
import {
  MatButtonModule, MatCardModule, MatDialogModule, MatInputModule, MatTableModule,
  MatToolbarModule
} from '@angular/material';

@NgModule({
  imports: [CommonModule, MatToolbarModule, MatButtonModule, MatCardModule, MatInpu
tModule, MatDialogModule, MatTableModule],
  exports: [CommonModule, MatToolbarModule, MatButtonModule, MatCardModule, MatInpu
tModule, MatDialogModule, MatTableModule],
})
export class CustomMaterialModule { }
```

Doing so will definitely not add any material styling to our html pages. To achieve this, we require to re-write our html pages using material directives. Here, I am going to change only the `login.component.html` for demo purpose.

login.component.html

```
<mat-toolbar color="primary">
  <mat-toolbar-row>
    <span><img src="./assets/devglan.png"></span>
    <span class="example-fill-remaining-space"></span>
    <span class="example-spacer"></span>

    <button mat-button>About</button>
    <button mat-button>Contact</button>
  </mat-toolbar-row>
</mat-toolbar>

<mat-card class="example-card">
  <mat-card-header>
    <mat-card-title>Login</mat-card-title>
  </mat-card-header>
  <mat-card-content>
    <form [formGroup]="loginForm" (ngSubmit)="onSubmit()">
      <table class="example-full-width" cellspacing="0">
        <tr>
          <td>
            <mat-form-field class="example-full-width">
              <input matInput placeholder="Username" formControlName="email" name="username" required>
              <mat-error *ngIf="loginForm.controls.email.errors.required">Password is required</mat-error>
            </mat-form-field>
          </td>
        </tr>
        <tr>
          <td><mat-form-field class="example-full-width">
            <input matInput placeholder="Password" formControlName="password" type="password" name="password" required>
            <mat-error *ngIf="loginForm.controls.password.errors.required">Email is required</mat-error>
          </mat-form-field></td>
        </tr></table>
    </form>
  </mat-card-content>
  <mat-card-actions>
    <button mat-raised-button color="primary">Login</button>
  </mat-card-actions>
</mat-card>
```

Now, you can compare the difference between our previous and present(material designing) login page styling. You can download the source from github here.



Email address:

dhiraj@gmail.com

Password:

••••••••

Conclusion

In this article, we learned about Angular 6 and created a sample example project using it. The source can be downloaded from github here - Angular 6 Example

Further Reading on Angular JS

1. Material Sidenav Example

2. Rxjs Tutorial

3. Angular Data Table Example

4. Spring Boot Jwt Auth

5. Spring Boot Angular Spring Data Example

If You Appreciate What We Do Here On Devglan, You Should Consider:

1. Buy a coffee for us -

Buy Now

2. Stay Connected to: Twitter | Facebook | Google Plus

We are thankful for your never ending support.

References

Angular CLI Git

New in Angular 6

Angular CLI Git

52 Comments **Devglan** 1 **Login**

♡ **Recommend** 1 **Share** **Sort by Best**

Join the discussion…

LOG IN WITH

OR SIGN UP WITH DISQUS ?

Name

Tricksters • a month ago

Thanks for posting the code, but I could not run it on my machine as when I did ng serve on my CMD I got: Could not find module "@angular-devkit/build-angular". When I tried to look for the folder module, I could not find it in the zip file. Please help me solve this error.

1 ^ | v • Reply • Share ›

Tricksters → Tricksters • a month ago

To run the file, just do npm install after unzipping the folder, and it should work. But currently I am getting an error when I am trying to add any users into the system. It is not working properly. Please help me with this.

^ | v • Reply • Share ›

Dhiraj Ray Mod → Tricksters • a month ago

Check the server configurations. Is your server code running to persist user in the DB

^ | v • Reply • Share ›

Tricksters → Dhiraj Ray • a month ago

How to make sure that my server is running to persist user into the DB? Please help me with that.

^ | v • Reply • Share ›

Dhiraj Ray Mod → Tricksters • a month ago

https://github.com/only2dhi... or u can check user.service.ts and modify according to your implementation

^ | v • Reply • Share ›

Tricksters → Dhiraj Ray • a month ago

I am not sure how do I connect this angular 6 application with the angular 5 spring boot application.

^ | v • Reply • Share ›

Dhiraj Ray Mod → Tricksters • a month ago

Thats why the server and client are standalone project. U can plugin yor server code or u can also mock at angular service layer

^ | v • Reply • Share ›

Tricksters → Dhiraj Ray • a month ago

Can you provide an example on how to connect this project to the server side spring boot project?

^ | v • Reply • Share ›

Dhiraj Ray Mod ➦ Tricksters • a month ago

through API. Check user.service.ts

⌃ | ⌄ • Reply • Share ›

Tricksters ➦ Dhiraj Ray • a month ago

So then do I run the spring boot project alongside this project to connect to db

⌃ | ⌄ • Reply • Share ›

Dhiraj Ray Mod ➦ Tricksters • a month ago

Yes. Please follow this article to set up server. -
https://www.devglan.com/spr...

⌃ | ⌄ • Reply • Share ›

Tricksters ➦ Dhiraj Ray • a month ago

I downloaded the spring boot code from github, but now I am getting CORS error when trying to access the list of users. I have added in the proxy file from the angular 5 project and it is not working.

⌃ | ⌄ • Reply • Share ›

Dhiraj Ray ➦ Tricksters • a month ago

Best way is to handle it in server side. Annotate controller with @CrossOrigin

⌃ | ⌄ • Reply • Share ›

Tricksters ➦ Dhiraj Ray • a month ago

So just include that and the code will work then. Do I need to add anything else with it?

⌃ | ⌄ • Reply • Share ›

Dhiraj Ray ➦ Tricksters • a month ago

Above annotation requires parameter. E.g. if u use * means the API can be accessed from any browser with any domain.
Check the documentation, it's very easy

⌃ | ⌄ • Reply • Share ›

Tricksters ➦ Dhiraj Ray • a month ago

Thank You so much for helping me solve my problem. I really appreciate your help and I really liked your tutorial a lot.

⌃ | ⌄ • Reply • Share ›

omkar mhaiskar • 5 days ago

Hi,

Thanks for posting code.

With the help of article and and some other article. I have created onecomponent with crud operation with modal window for add and edit. to implement this i have used the formbuilder. except display list i am not able to do any operation. when click on edit or delete button
nothing happen. on click of add button modal form is open but after data
entry hit on submit button no data pass to component ts.

I have uploaded my project on below url.

https://github.com/Mhaiskar...

Please check and let me know what is the issue. how to resolve this issue.

Thanks in advance,
Omkar.

⌃ | ⌄ • Reply • Share ›

Sudhir • 25 days ago

First of all, All articles are Geat, Simple to understand. I have a small question. How to run the Angular project in Spring port. I mean if there are 2 projects(one for Spring boot) and other for Web, then what changes we can do in spring boot pom.xml to run the angular project in 8080 (i.e. spring boot)

⌃ | ⌄ • Reply • Share ›

Dhiraj Ray Mod ➦ Sudhir • 25 days ago

https://www.devglan.com/spr...

⌃ | ⌄ • Reply • Share ›

This comment is awaiting moderation. Show comment.

Dhiraj Ray ➦ Sudhir • 23 days ago

Sorry for the late reply. To begin with first build the angular project with ng build. This will generate the binary inside /dist folder. Now in the pom.xml of server code hardcode this location as <directory>complete-path/dist</directory>. Check this pom file for sample - https://github.com/only2dhi... and then you can package it as a war and deploy it on a standalone tomcat and hit the context - localhost:8080 to see the home page.Make sure you final war package name is ROOT to access welcome page at localhost:8080 or else localhost:8080/your-war-name

^ | ˅ • Reply • Share ›

This comment is awaiting moderation. Show comment.

Dhiraj Ray ➦ Sudhir • 23 days ago

That won't work for angular6 as it requires higher version of npm. That's why I asked u to manually build angular 6 project n run mvn clean install on server project not parent to build the war

^ | ˅ • Reply • Share ›

Junior Osho • a month ago

git hub project ?

^ | ˅ • Reply • Share ›

Dhiraj Ray ➦ Junior Osho • a month ago

https://github.com/only2dhi...

^ | ˅ • Reply • Share ›

Victor Yuji Maehira • a month ago

Hi Ray!
First of all, I really appreciate your engagement with the programmers community sharing your experience! Congratulations! Your tutorial has been very useful. I managed to run the sample properly except from the update method. Have you implemented it in the back-end?
Thanks in advance!

^ | ˅ • Reply • Share ›

Tricksters ➦ Victor Yuji Maehira • a month ago

Are you able to update the records? I am currently getting this error when I click on update: Response to preflight request doesn't pass access control check: No 'Access-Control-Allow-Origin' header is present on the requested resource. Origin 'http://localhost:4200' is therefore not allowed access. The response had HTTP status code 403.

^ | ˅ • Reply • Share ›

Victor Yuji Maehira ➦ Tricksters • a month ago

I was getting the same error, however after some adjustment I was able to create and delete a user.
I did the following:
- In the user.service.ts Angular file, I replaced the baseUrl to "http://localhost:8080/user-portal/api"
- In the UserController back-end file, I uncommented this line "@CrossOrigin(origins = "http://localhost:4200", maxAge = 3600)"

Now, I am getting http 500 at the edit operation.

Ray, sorry if I am misrepresenting the original idea. I've done some adaptations in order to make it run in my machine. Warn me if necessary, please.

^ | ˅ • Reply • Share ›

Tricksters ➦ Victor Yuji Maehira • a month ago

To even be able to list users, I had to change the url in user.service.ts file and also in comment the @CrossOrigins line in UserController.java file. After this change I was able to add and delete users but have not been able to edit users though.

^ | ˅ • Reply • Share ›

Tricksters ➦ Victor Yuji Maehira • a month ago

I am also able to add and delete but I am getting 403 error not 500 error at the edit operation.

^ | ˅ • Reply • Share ›

Dhiraj Ray ➦ Tricksters • a month ago

Take the latest change and then try. It sud be working now

⌃ | ⌄ • Reply • Share ›

Tricksters ➦ Dhiraj Ray • a month ago

Which file have you changed?

⌃ | ⌄ • Reply • Share ›

Dhiraj Ray ➦ Tricksters • a month ago

https://github.com/only2dhi... ---@RequestMapping and update() method
And in UserServiceImpl.java - update() method

⌃ | ⌄ • Reply • Share ›

Victor Yuji Maehira ➦ Dhiraj Ray • a month ago

Now it is working perfectly! Congrats! Thanks a lot!

⌃ | ⌄ • Reply • Share ›

Tricksters ➦ Victor Yuji Maehira • a month ago

@dhiraj, do you know why does it take a few seconds to navigate from edit-user page back to list-user page after clicking on Update button?

⌃ | ⌄ • Reply • Share ›

Dhiraj Ray ➦ Victor Yuji Maehira • a month ago

Yes. https://github.com/only2dhi...

⌃ | ⌄ • Reply • Share ›

Victor Yuji Maehira ➦ Dhiraj Ray • a month ago

Hi Ray!
Let me ask you more 2 questions. Is it OK to have the update method returning null in UserServiceImpl.java?
https://github.com/only2dhi...

Moreover, there should be an update method in the repository (UserRepository)?
https://github.com/only2dhi...

Thank you!

⌃ | ⌄ • Reply • Share ›

Dhiraj Ray ➦ Victor Yuji Maehira • a month ago

It's my mistake. I missed to commit these changes as I implemented update method later.

⌃ | ⌄ • Reply • Share ›

Tricksters ➦ Dhiraj Ray • a month ago

Is update method added into the UserRepository class in the previous version @DhirajRay

⌃ | ⌄ • Reply • Share ›

Dhiraj Ray ➦ Tricksters • a month ago

Yes.Please check

⌃ | ⌄ • Reply • Share ›

Nivaldo Ribeiro • a month ago

I downloaded the project but it doesn't run cus the WeService is not in the project!
Error:
Failed to load http://localhost:8080/user-portal/users: No 'Access-Control-Allow-Origin' header is present on the requested resource. Origin 'http://localhost:4200' is therefore not allowed access. The response had HTTP status code 404.

⌃ | ⌄ • Reply • Share ›

Dhiraj Ray ➦ Nivaldo Ribeiro • a month ago

The server implementation is same as of the angular 5 exmple. Hence, not included here.Check this source here - https://github.com/only2dhi...

⌃ | ⌄ • Reply • Share ›

Junior Osho • a month ago

search ? pagination ?

⌃ | ⌄ • Reply • Share ›

Yin Mon Mon Thwn • a month ago

which import can i use for fakeUsers? i have an error " return Observable.of(fakeUsers).delay(5000); "

^ | ⌄ • Reply • Share ›

Junior Osho • a month ago

Gostei muito do seu projeto inicial, mas seria bom fazer algo junto com FIREBASE, para demonstrar o poder do Angular 6 ! fica a dica para o meu amigo indiano !

^ | ⌄ • Reply • Share ›

jovel cardozo • 2 months ago

I am getting the following error while adding users:
zone.js:2969 OPTIONS http://localhost:8080/user-portal/users 0 (),
HttpErrorResponse {headers: HttpHeaders, status: 0, statusText: "Unknown Error", url: null, ok: false, …}

^ | ⌄ • Reply • Share ›

Contact us: hello@devglan.com