**Online Collaborate**

🡪Open Visual Studio Code. Create new project inside Angular\_Workspace. Set the project name as OnlineCollaborateAngular

Ng new OnlineCollaborateAngular

🡪 Install Bootstrap

Use the following command to install bootstrap in the project.

PS F:\AngularWorkSpace\OnlineCollaborateAngular> ***npm install*** [***bootstrap@3.3.7***](mailto:bootstrap@3.3.7) ***–save***

🡪 Install Angular Data Table

Use the following command to install bootstrap in the project.

PS F:\AngularWorkSpace\OnlineCollaborateAngular> ***npm install angular-datatable --save***

🡪 Generate Components  
Open the project in visual studio and then use the following command to generate Angular components:  
*ng g c login-user*  
*ng g c register-use*

*ng g c user-list*

🡪 Let's also create a service class by using the following command: -

ng g s user

🡪 Edit the **app.module.ts** file

* 1. **Import Routing** - Here, we are importing routing file (app-routing.module.ts) and include it in imports array.
  2. **Import ReactiveFormsModule** - Here, we are importing **ReactiveFormsModule** for reactive forms and specify it in imports array.
  3. **Import HttpModule** - Here, we are importing **HttpModule** for server requests and specifying it in imports array.
  4. **Register Service class** - Here, we are mentioning the service class in provider's array.

--//app.module.ts

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

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

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

import { HttpClientModule } from '@angular/common/http';

import {DataTablesModule } from 'angular-datatables';

import { AppRoutingModule } from './app-routing.module';

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

import { RegisterUserComponent } from './register-user/register-user.component';

import { UserListComponent } from './user-list/user-list.component';

import { LoginUserComponent } from './login-user/login-user.component';

@NgModule({

  declarations: [

    AppComponent,

    RegisterUserComponent,

    UserListComponent,

    LoginUserComponent

  ],

  imports: [

    BrowserModule,

    AppRoutingModule,

    FormsModule,

    ReactiveFormsModule,

    HttpClientModule,

    DataTablesModule

    ],

  providers: [],

  bootstrap: [AppComponent]

})

export class AppModule { }

🡪 Edit the **app-routing.module.ts** file

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

import { Routes, RouterModule } from '@angular/router';

import { RegisterUserComponent } from './register-user/register-user.component';

import { UserListComponent } from './user-list/user-list.component';

import { LoginUserComponent } from './login-user/login-user.component';

const routes: Routes = [

  { path: '', redirectTo: 'login-user', pathMatch: 'full' },

  { path: 'user-list', component: UserListComponent },

  { path: 'register-user', component: RegisterUserComponent },

  { path: 'login-user' , component: LoginUserComponent} ,

  ];

@NgModule({

  imports: [RouterModule.forRoot(routes)],

  exports: [RouterModule]

})

export class AppRoutingModule { }

🡪 Edit the **app.component.html** file(remove existing code and replace with following code)

<div  class="container-fluid">

  <nav class="navbar navbar-expand-sm bg-dark navbar-dark">

  <ul class="navbar-nav">

  <li class="nav-item ">

  <a routerLink="user-list" class="nav-link" class="btn-primary active" role="button" >View User</a> &nbsp;

  </li>

    <li class="nav-item">

  <a  routerLink="register-user" class="nav-link" class="btn-primary active" role="button" >Resgistration</a> &nbsp;

  </li>

  <li class="nav-item">

    <a  routerLink="login-user" class="nav-link" class="btn-primary active" role="button" >Login</a> &nbsp;

    </li>

  </ul>

  </nav>

<router-outlet></router-outlet>

</div>

🡪 Create the **user.ts** class

🡪Let's create a class by using the following command: -

*ng g class user*

🡪 Now, specify the required fields within the **User** class. (The purpose of this class is to map the specified fields with the fields of Spring entity class.)

export class User {

    user\_id:number;

    first\_name:String;

    last\_name:String;

    username:String;

    password:String;

    confirm\_password:String;

    email:String;

    role:String;

    status:String;

    isOnline:boolean;

    enabled:boolean;

}

🡪 Edit the **user.service.ts** file

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

import { HttpClient } from '@angular/common/http';

import { Observable } from 'rxjs';

@Injectable({

  providedIn: 'root'

})

export class UserService {

  private baseUrl = 'http://localhost:8000/api/';

  constructor(private http:HttpClient) { }

  getUserList(): Observable<any> {

    return this.http.get(`${this.baseUrl}`+'user-list');

        }

            createUser(user: object): Observable<object> {

        return this.http.post(`${this.baseUrl}`+'save-user', user);

          }

        deleteUser(user\_id: number): Observable<any> {

        return this.http.delete(`${this.baseUrl}/delete-user/${user\_id}`, { responseType: 'text' });

          }

        getUser(user\_id: number): Observable<Object> {

        return this.http.get(`${this.baseUrl}/user/${user\_id}`);

          }

        updateUser(user\_id: number, value: any): Observable<Object> {

        return this.http.post(`${this.baseUrl}/update-user/${user\_id}`, value);

          }

}

🡪 Edit the **register-user.component.ts** file

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

import { UserService } from '../user.service';

import {FormControl,FormGroup,Validators} from '@angular/forms';

import { User } from '../user';

@Component({

  selector: 'app-register-user',

  templateUrl: './register-user.component.html',

  styleUrls: ['./register-user.component.scss']

})

export class RegisterUserComponent implements OnInit {

  constructor(private userservice:UserService) { }

  user : User=new User();

  submitted = false

  ngOnInit(): void {

    this.submitted=false;

  }

registrationform=new FormGroup({

  first\_name:new FormControl('' , [Validators.required , Validators.minLength(10) ] ),

  last\_name:new FormControl('' , [Validators.required , Validators.minLength(10) ] ),

  username:new FormControl('' , [Validators.required , Validators.minLength(10) ] ),

  password:new FormControl('' , [Validators.required , Validators.minLength(10) ] ),

  confirm\_password:new FormControl('' , [Validators.required , Validators.minLength(10) ] ),

  email:new FormControl('',[Validators.required,Validators.email]),

  role:new FormControl('' , [Validators.required , Validators.minLength(10) ] ),

  status:new FormControl('' , [Validators.required , Validators.minLength(10) ] )

    });

    registerUser(registerUser: any)

    {

      this.user=new User();

      this.user.first\_name=this.FirstName?.value;

      this.user.last\_name=this.LastName?.value;

      this.user.username=this.Username?.value;

      this.user.password=this.Password?.value;

      this.user.confirm\_password=this.ConfirmPassword?.value;

      this.user.email=this.Email?.value;

      this.user.role=this.Role?.value;

      this.user.status=this.Status?.value;

      this.submitted = true;

      this.save();

    }

    save() {

      this.userservice.createUser(this.user)

        .subscribe(data => console.log(data), error => console.log(error));

      this.user = new User();

    }

    get FirstName(){

      return this.registrationform.get('first\_name');

    }

    get LastName(){

      return this.registrationform.get('last\_name');

    }

    get Username(){

      return this.registrationform.get('username');

    }

    get Password(){

      return this.registrationform.get('password');

    }

    get ConfirmPassword(){

      return this.registrationform.get('confirm\_password');

    }

    get Email(){

      return this.registrationform.get('email');

    }

    get Role(){

      return this.registrationform.get('role');

    }

    get Status(){

      return this.registrationform.get('status');

    }

    registrUsersForm(){

      this.submitted=false;

      this.registrationform.reset();

    }

  }

🡪 Edit the **register-user.component.html** file

<br/>

<br/>

<h3 style="font-weight: bold; font-style: italic; text-align: center">Registration Form</h3>

<div class="row">

    <div class="col-sm-4">

        <div [hidden]="submitted" style="width: 500px; ">

        <form [formGroup]="registrationform" #registerUser (ngSubmit)="registerUser(registerUser)">

            <div class="form-group">

                <label for="first\_name">First Name:-</label>

                <input type="text" class="form-control" formControlName="first\_name" data-toggle="tooltip"

                data-placement="right" title="Enter First Name" placeholder="Enter Your First Name">

                <br/>

                <label for="last\_name">Last Name:-</label>

                <input type="text" class="form-control" formControlName="last\_name" data-toggle="tooltip"

                data-placement="right" title="Enter Last Name" placeholder="Enter Your Last Name">

                <br/>

                <label for="username">User Name:-</label>

                <input type="text" class="form-control" formControlName="username" data-toggle="tooltip"

                data-placement="right" title="Enter User Name" placeholder="Enter Your User Name">

                <br/>

                <label for="password">Password:-</label>

                <input type="password" class="form-control" formControlName="password" data-toggle="tooltip"

                data-placement="right" title="Enter Password" pattern="(?=.\*\d)(?=.\*[a-z])(?=.\*[A-Z]).{8,}" placeholder="Enter Your Password">

                <br/>

                <label for="confirm\_password">Confirm Password:-</label>

                <input type="password" class="form-control" formControlName="confirm\_password" data-toggle="tooltip"

                data-placement="right" title="Enter Confirm Password" pattern="(?=.\*\d)(?=.\*[a-z])(?=.\*[A-Z]).{8,}" placeholder="Enter Your Confirm Password">

                <br/>

                <label for="email">E-mail:-</label>

                <input type="text" class="form-control" formControlName="email" data-toggle="tooltip"

                data-placement="right" title="Enter Email" pattern="[a-z0-9.\_%+-]+@[a-z0-9.-]+\.[a-z]{2,}$" placeholder="Enter Your E-Mail Address">

                <br/>

                <div class="form-group">

                    <label for="role">Select Role:-</label>

                    <select class="form-control" formControlName="role" data-toggle="tooltip"

                    data-placement="right" title="Select Role" >

                    <option value="null">--Select Role--</option>

                    <option value="Admin">Admin</option>

                    <option value="User">User</option>

                    </select>

                    </div>

                    <br/>

                    <input type="radio" id="active" name="status" value="active">

                    <label for="active">Active</label> &nbsp; &nbsp;

                    <input type="radio" id="deactive" name="status" value="deactive">

                    <label for="deactive">Deactive</label><br>

            </div>

       <div>

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

        </div>

        <div [hidden]="!submitted">

            <h4>User Added SuccessFully!</h4>

            <button (click)="registrUsersForm()" class='btn-primary'>Add More User</button>

            </div>

        </form>

        </div>

    </div>

</div>

🡪 Edit the **register-user.component.scss** file

.row {

    min-width: 50%;

    background: rgb(242, 243, 248);

    border-radius: 2px;

    border: 1px solid #000000;

    display: flex;

    flex-direction: column;

    overflow: hidden;

    align-items: center;

}

.row \* {

    box-sizing: border-box;

}

.col-sm-4 {

    display: flex;

    align-items: center;

    justify-content: center;

}

.form-control{

    width:55%;

    border-radius:5px;

    border: 2px solid rgba(76, 46, 207, 0.856);

    font-size: 13px;

    font-family:monospace;

    }

🡪 Edit the **user-list.component.ts** file

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

import {UserService} from '../user.service';

import {User} from '../user';

import {Observable, Subject} from 'rxjs';

import { Validators, FormControl, FormGroup, FormBuilder} from '@angular/forms';

import { DataTableModule } from 'angular-datatable';

@Component({

  selector: 'app-user-list',

  templateUrl: './user-list.component.html',

  styleUrls: ['./user-list.component.scss']

})

export class UserListComponent implements OnInit {

  constructor(private userservice:UserService) { }

  userArray: any[] = [];

  dtOptions: DataTables.Settings = {};

  dtTrigger: Subject<any>= new Subject();

  users!: Observable<User[]>;

  user : User=new User();

  deleteMessage=false;

  userlist:any;

  isupdated = false;

  ngOnInit() {

    this.isupdated=false;

    this.dtOptions = {

      pageLength: 6,

      stateSave:true,

      lengthMenu:[[6, 16, 20, -1], [6, 16, 20, "All"]],

      processing: true

    };

    this.userservice.getUserList().subscribe((data) =>{

    this.users =data;

    this.dtTrigger.next();

    })

  }

  deleteUser(userId: number) {

    this.userservice.deleteUser(userId)

      .subscribe(

        (        data) => {

          console.log(data);

          this.deleteMessage=true;

          this.userservice.getUserList().subscribe((data) =>{

            this.userlist =data

            })

        },

        (        error) => console.log(error));

  }

  updateUser(userId: number){

    this.userservice.getUser(userId)

      .subscribe(

        (        data) => {

          this.userlist=data

        },

        (        error) => console.log(error));

  }

  userupdateform=new FormGroup({

    user\_id:new FormControl(),

    email:new FormControl(),

    role:new FormControl()

  });

  updateUesr(updateusers: any){

    this.user=new User();

   this.user.user\_id=this.UserId!.value;

   this.user.email=this.Email!.value;

   this.user.role=this.Role!.value;

   console.log(this.Role!.value);

   this.userservice.updateUser(this.user.user\_id,this.user).subscribe(

     (    data) => {

      this.isupdated=true;

      this.userservice.getUserList().subscribe((data) =>{

        this.users =data

        })

    },

     (    error) => console.log(error));

  }

  get Email(){

    return this.userupdateform.get('email');

  }

  get Role(){

    return this.userupdateform.get('role');

  }

  get UserId(){

    return this.userupdateform.get('user\_id');

  }

  changeisUpdate(){

    this.isupdated=false;

  }

}

🡪 Edit the **user-list.component.html** file

<div class="panel panel-default">

    <div class="panel-heading">

        <h1 style="font-weight: bold; font-style: italic; text-align: center; ">Users</h1><br>

        <div class="row" [hidden]="!deleteMessage">

                  <div class="col-sm-4"></div>

                  <div class="col-sm-4">

                          <div class="alert alert-info alert-dismissible">

                                  <button type="button" class="close" data-dismiss="alert">×</button>

                                  <strong>User Data Deleted</strong>

                          </div>

                  </div>

                  <div class="col-sm-4"></div>

          </div>

      </div>

    <div class="panel-body">

        <table  class="table table-hover table-sm" datatable [dtOptions]="dtOptions"

        [dtTrigger]="dtTrigger"  >

            <thead class="thead-light">

                <tr>

                    <th>First Name</th>

                    <th>Last Name</th>

                    <th>Username</th>

                    <th>Password</th>

                    <th>Email</th>

                    <th>Role</th>

                    <th>Status</th>

                </tr>

            </thead>

            <tbody>

                 <tr \*ngFor="let user of userlist">

                    <td>{{user.first\_name}}</td>

                    <td>{{user.last\_name}}</td>

                    <td>{{user.username}}</td>

                    <td>{{user.password}}</td>

                    <td>{{user.email}}</td>

                    <td>{{user.role}}</td>

                    <td>{{user.status}}</td>

                    <td><button (click)="deleteUser(user.userid)" class='btn btn-primary'><i class="fa fa-futboll-0">Delete</i></button>

                      <button (click)="updateUser

                (user.userid)" class='btn btn-info'

                      data-toggle="modal" data-target="#myModal">Update</button>

                    </td>

                  </tr>

            </tbody><br>

        </table>

    </div>

  </div>

  <div class="modal" id="myModal">

          <div class="modal-dialog">

            <div class="modal-content">

                  <form [formGroup]="userupdateform" #updateusers (ngSubmit)="updateUesr(updateusers)">

              <!-- Modal Header -->

              <div class="modal-header">

                <h4 class="modal-title" style="text-align: center">Update User</h4>

              </div>

              <!-- Modal body -->

              <div class="modal-body" \*ngFor="let user of userlist" >

                  <div [hidden]="isupdated">

                      <input type="hidden" class="form-control"  formControlName="userid" [(ngModel)]="user.userid">

                              <div class="form-group">

                                  <label for="name">First Name</label>

                                  <input type="text" class="form-control"  formControlName="first\_name" [(ngModel)]="user.first\_name"  >

                              </div>

                              <div class="form-group">

                                <label for="name">Last Name</label>

                                <input type="text" class="form-control"  formControlName="last\_name" [(ngModel)]="user.last\_name"  >

                              </div>

                              <div class="form-group">

                                <label for="name">Username</label>

                                <input type="text" class="form-control"  formControlName="username" [(ngModel)]="user.username"  >

                              </div>

                              <div class="form-group">

                                <label for="name">Password</label>

                                <input type="password" class="form-control"  formControlName="password" [(ngModel)]="user.password"  >

                              </div>

                              <div class="form-group">

                                <label for="name">Email</label>

                                <input type="text" class="form-control" formControlName="email" [(ngModel)]="user.email">

                              </div>

                              <div class="form-group">

                                <label for="name">Role</label>

                                <input type="text" class="form-control"  formControlName="role" [(ngModel)]="user.role"  >

                              </div>

                              <div class="form-group">

                                  <label for="name">Role</label>

                                    <select class="form-control" formControlName="role" required>

                                      <option value="Admin" [selected]="'Admin' == user.role">Admin</option>

                                      <option value="User" [selected]="'User' == user.role">User</option>

                                    </select>

                              </div>

                              <div class="form-group">

                                <label for="name">Status</label>

                                <input type="text" class="form-control"  formControlName="status" [(ngModel)]="user.status"  >

                              </div>

                    </div>

                    <div [hidden]="!isupdated">

                        <h4>User Detail Updated!</h4>

                    </div>

              </div>

              <!-- Modal footer -->

              <div class="modal-footer" >

                <button type="submit" class="btn btn-success" [hidden]="isupdated">Update</button>

                <button type="button" class="btn btn-danger" data-dismiss="modal" (click)="changeisUpdate()">Close</button>

              </div>

          </form>

            </div>

          </div>

        </div>

🡪 Edit the **login-user.component.ts** file

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

@Component({

  selector: 'app-login-user',

  templateUrl: './login-user.component.html',

  styleUrls: ['./login-user.component.scss']

})

export class LoginUserComponent implements OnInit {

  constructor() { }

  ngOnInit(): void {

  }

}

🡪 Edit the **login-user.component.html** file

<div class="main">

  <p class="sign" align="center">Sign in</p>

  <form class="form1">

    <input class="un " type="text" align="center" placeholder="Username">

    <input class="pass" type="password" align="center" placeholder="Password">

    <a class="submit" align="center">Sign in</a>

     </form>

     </div>

🡪 Edit the **login-user.component.scss** file

    body {

        background-color: #f6ebee;

        font-family:monospace;

    }

    .main {

        background-color: #FFFFFF;

        width: 400px;

        height: 400px;

        margin: 7em auto;

        border-radius: 1.5em;

        box-shadow: 0px 11px 35px 2px rgba(0, 0, 0, 0.14);

    }

    .sign {

        padding-top: 40px;

        color: #8C55AA;

        font-family: 'Ubuntu', sans-serif;

        font-weight: bold;

        font-size: 23px;

    }

    .un {

    width: 76%;

    color: rgb(13, 16, 230);

    font-weight: 700;

    font-size: 14px;

    letter-spacing: 1px;

    background: rgba(250, 0, 0, 0.04);

    padding: 10px 20px;

    border-radius: 20px;

    box-sizing: border-box;

    border: 2px solid rgba(76, 46, 207, 0.856);

    margin-bottom: 50px;

    margin-left: 46px;

    text-align: center;

    margin-bottom: 27px;

    font-family:monospace;

    }

    .pass {

            width: 76%;

    color: rgb(38, 50, 56);

    font-weight: 700;

    font-size: 14px;

    letter-spacing: 1px;

    background: rgba(136, 126, 126, 0.04);

    padding: 10px 20px;

    border-radius: 20px;

    box-sizing: border-box;

    border: 2px solid rgba(76, 46, 207, 0.856);

    margin-bottom: 50px;

    margin-left: 46px;

    text-align: center;

    margin-bottom: 27px;

    font-family:monospace;

    }

    .submit {

      cursor: pointer;

        border-radius: 5em;

        color: #fff;

        background: linear-gradient(to right, #9C27B0, #E040FB);

        border: 0;

        padding-left: 40px;

        padding-right: 40px;

        padding-bottom: 10px;

        padding-top: 10px;

        font-family:monospace;

        margin-left: 35%;

        font-size: 13px;

        box-shadow: 0 0 20px 1px rgba(0, 0, 0, 0.04);

    }

🡪Install following:

npm install jquery --save

npm install datatables.net --save

npm install datatables.net-dt --save

npm install angular-datatables@6.0.0 --save

npm install @types/jquery --save-dev

npm install @types/datatables.net --save-dev

    🡪 Edit angular.json file at line 30. styles and scripts:

"styles": [

"src/styles.scss",

"node\_modules/datatables.net-dt/css/jquery.dataTables.css",

"node\_modules/bootstrap/dist/css/bootstrap.css"

],

"scripts": [

"node\_modules/jquery/dist/jquery.js",

"node\_modules/datatables.net/js/jquery.dataTables.js",

"node\_modules/bootstrap/dist/js/bootstrap.js"

]

🡪Save All.

🡪Run the SpringBoot Application

🡪Run the Angular project