**REPRESENTATIVE COMPONENT**

1. **REPRESENTATIVE COMPONENT.TS:**
2. import { Component, OnInit,Input } from '@angular/core';
3. import { getTestBed } from '@angular/core/testing';
4. import { UserService } from '../user.service';
5. import { Router } from '@angular/router';
7. @Component({
8. selector: 'app-representative',
9. templateUrl: './representative.component.html',
10. styleUrls: ['./representative.component.css']
11. })
12. export class RepresentativeComponent implements OnInit {
13. address:any;
14. addressId:any;
15. address1:any;
16. userId:any;
17. message:String;
18. users:any;
19. userFarmsId:any;
20. farms:any;
21. userId1:any;
22. constructor(private service:UserService, private router :Router) {
24. }
26. ngOnInit(){
27. this.service.currentMessage.subscribe(message => this.message=message);
28. this.userId=this.message;
29. }
31. ViewProfile(){
32. this.service.currentMessage.subscribe(message => this.message=message);
33. this.userId=this.message;
34. console.log(this.userId);
35. this.service.getUserById(this.userId).subscribe((data:any) => {console.log(data); this.address1=data;});
37. }
38. ShowFarmers(){
40. this.service.currentMessage.subscribe(message => this.message=message);
41. this.userId=this.message;
42. console.log(this.userId);
43. this.service.getUserById(this.userId).subscribe((data:any) => {console.log(data); this.address=data;});
44. this.service.getFarmersById(this.userId).subscribe((data:any) => {console.log(data); this.users=data;});
46. }
47. showFarms(){
48. this.service.currentMessage.subscribe(message => this.message=message);
49. this.userId=this.message;
50. console.log(this.userId);
51. this.service.getFarmsById(this.addressId).subscribe((data:any) => {console.log(data); this.farms=data;});
53. }
54. }

**2.REPRESENTATIVE COMPONENT.HTML:**

<!DOCTYPE html>

<html lang="en">

<head>

  <title>Bootstrap Example</title>

  <meta charset="utf-8">

  <meta charset="utf-8">

  <meta name="viewport" content="width=device-width, initial-scale=1">

  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">

  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>

  <meta name="viewport" content="width=device-width, initial-scale=1">

  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css">

  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

  <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>

  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.0/js/bootstrap.min.js"></script>

</head>

<style>

.button {

  background-color: #4CAF50; /\* Green \*/

  border: none;

  color: white;

  padding: 15px 32px;

  text-align: center;

  text-decoration: none;

  display: inline-block;

  font-size: 16px;

  margin: 4px 2px;

  cursor: pointer;

}

.button2 {background-color: #008CBA;} /\* Blue \*/

.button3 {background-color: #f44336;} /\* Red \*/

</style>

<script type="text/javascript">

    function googleTranslateElementInit() {

      new google.translate.TranslateElement({pageLanguage: 'en'}, 'google\_translate\_element');

    }

    </script>

    <script type="text/javascript" src="//translate.google.com/translate\_a/element.js?cb=googleTranslateElementInit"></script>

    <div class = "container">

<h1><p>                     Welcome Representative {{address.user.userName}}.</p></h1>

<div class = "row">

<div class = "col-7"></div>

<div class = "col-5">

    <button class="button" (click) = "ShowFarmers()">Show Farmers</button>

    <table class="table table-bordered table-striped table-hover table-sm text-center">

        <thead>

            <tr class="bg-info text-white">

                <TH>FARMER ID</TH>

                <th>USERNAME</th>

                <th>NAME</th>

                <th>MOBILE NUMBER</th>

                <th>AGE</th>

                <th>ROLE</th>

        </thead>

        <tbody>

            <tr \*ngFor='let user of users'>

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

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

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

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

                <td>{{user.age}}</td>

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

    </tr>

    </tbody>

    </table>

    </div>

    <button class="button button2" routerLink="/viewprofile">ViewProfile</button>

    <br/><br/><br/><br/><br/><br/><br/><br/>

<button class="button button2" routerLink="/addfarm">Add Farm</button>

<br/><br/><br/><br/><br/><br/><br/><br/>

<button class="button button3"  routerLink="/addfarmer">Add Farmer</button>

<br/><br/><br/><br/><br/><br/><br/><br/>

<div class = "col-6"></div>

</div>

</div>

<div class = "col-5">

    ENTER THE USERID TO DISPLAY FARMS:<input type="text" [(ngModel)]="addressId" ><br/>

    <button class="button" (click) = "showFarms()">Show Farms</button>

    <table class="table table-bordered table-striped table-hover table-sm text-center">

        <thead>

            <tr class="bg-info text-white">

                <TH>FARM Id</TH>

                <th>Crop</th>

                <th>Crop Quantity</th>

                <th>Field Area</th>

                <th>Kind Of Crop</th>

        </thead>

        <tbody>

            <tr \*ngFor='let farm of farms'>

                <td>{{farm.farmId}}</td>

                <td>{{farm.crop}}</td>

                <td>{{farm.cropQuantity}}</td>

                <td>{{farm.fieldArea}}</td>

                <td>{{farm.kindOfCrop}}</td>

    </tr>

    </tbody>

    </table>

**3.USERSERVICE.TS (REPRESENTATIVE COMPONENT):**

 getUserById(userId:any){

    return this.httpClient.get('AGRIAPI/webapi/myresource/viewProfile/' + userId);

   }

   getFarmersById(userId: any) {

    return this.httpClient.get('AGRIAPI/webapi/myresource/getFarmers/' + userId);

  }

  getFarmsById(addressId: any) {

    return this.httpClient.get('AGRIAPI/webapi/myresource/getFarms/' + addressId);

  }

**4.METHODS RELATED TO GETUSERBYID(),GETFARMSBYID(),GETFARMERSBYID():**

**MYRESOURCE:**

@Path("viewProfile/{userId}")

@GET

@Produces(MediaType.***APPLICATION\_JSON***)

**public** Address viewProfile(@PathParam("userId") **int** userId){

AddressDAO userDao = **new** AddressDAO();

Address users= userDao.viewProfile(userId);

**return** users;

}

@Path("getFarmers/{userId}")

@GET

@Produces(MediaType.***APPLICATION\_JSON***)

**public** List<User> getFarmers(@PathParam("userId") String qualification){

UserDAO userDao = **new** UserDAO();

List<User> users = userDao.getFarmers(qualification);

**return** users;

}

@Path("getFarms/{addressId}")

@GET

@Produces(MediaType.***APPLICATION\_JSON***)

**public** List<FarmDetails> getFarms(@PathParam("addressId") **int** addressId){

FarmDetailsDAO farmdetailsDao = **new** FarmDetailsDAO();

List<FarmDetails> farmdetails = farmdetailsDao.getFarmDetailsById(addressId);

**return** farmdetails;

}

5.**METHODS IN HIBERNATE.TEMPLATE:**

HIBERNATE TEMPLATE:

**public** **static** User getObjectByUserId(**int** userId) {

String queryString = "from User where userId = userId";

Query query = *sessionFactory*.openSession().createQuery(queryString);

query.setInteger("userId", userId);

Object queryResult = query.uniqueResult();

User user = (User)queryResult;

**return** user;

}

**public** **static** Object getObjectById(**int** userId){

String queryString ="from Address where userId = :userId";

Query query = *sessionFactory*.openSession().createQuery(queryString);

query.setLong("userId", userId);

Object queryResult = query.uniqueResult();

Address user = (Address)queryResult;

**return** user;

}

**public** **static** List<User> getObjectByquali(String qualification) {

// **TODO** Auto-generated method stub

**int** userId = Integer.*parseInt*(qualification);

String queryString ="from User where qualification = :userId";

Query query = *sessionFactory*.openSession().createQuery(queryString);

query.setString("userId", qualification);

List queryResult = query.list();

List<User> userList = (List<User>)queryResult;

**return** userList;

}

**public** **static** List<FarmDetails> getObjectByAddressId(**int** addressId) {

// **TODO** Auto-generated method stub

String queryString ="from FarmDetails where addressId = :addressId";

Query query = *sessionFactory*.openSession().createQuery(queryString);

query.setLong("addressId", addressId);

Object queryResult = query.list();

List<FarmDetails> farm = (List<FarmDetails>)queryResult;

**return** farm;

}

**6.VIEWPROFILE METHOD IN ADDRESSDAO:**

**IN AddressDAO:**

**public** Address viewProfile(**int** userId) {

// **TODO** Auto-generated method stub

**return** (Address)HibernateTemplate.*getObjectById*(userId);

}

**7.GET FARMERS() IN USERDAO:**

**IN UserDAO():**

**public** List<User> getFarmers(String qualification) {

// **TODO** Auto-generated method stub

**return** (List<User>)HibernateTemplate.*getObjectByquali*(qualification);

**8.GETFARMS() IN FARMSETAILSDAO():**

**IN FarmDetailsDAO():**

**public** List<FarmDetails> getFarmDetailsById(**int** addressId) {

**return** (List<FarmDetails>)HibernateTemplate.*getObjectByAddressId*(addressId);

}

**ADDFARMER COMPONENT:**

1. **ADD FARMER COMPONENT.TS:**
2. import { Component, OnInit } from '@angular/core';
3. import { UserService } from '../user.service';
4. import { Router } from '@angular/router';
5. @Component({
6. selector: 'app-add-farmer',
7. templateUrl: './add-farmer.component.html',
8. styleUrls: ['./add-farmer.component.css']
9. })
10. export class AddFarmerComponent implements OnInit {
11. user : any;
12. address :any;
13. userId:any;
14. message:String;
15. message1:any;
17. constructor(private router: Router,private service: UserService) {
18. this.address={pincode:'',city:'',state:'',
19. user:{ userName:'', password:'',role:'farmer',fullName:'',mobileNumber:'',age:'',qualification:this.message}};
20. }

23. ngOnInit(): void {
24. this.service.currentMessage.subscribe(message => this.message=message);
25. console.log(this.message);

28. }
29. register(registerForm : any): void{
30. console.log(this.address);
31. this.service.currentMessage.subscribe(message => this.message=message);
32. this.userId=this.message;
33. this.address.user.qualification=this.userId;
34. console.log(this.userId);
35. console.log(this.service.registerUser(this.address).subscribe((result:any) => {console.log(result);}));
36. console.log(this.address);

39. alert("registration of Farmer is done!." );
40. this.router.navigate(['representative']);

43. }
45. }

**2.ADDFARMER COMPONENT.HTML:**

<!DOCTYPE html>

<html lang="en">

<head>

  <title>Bootstrap Example</title>

  <meta charset="utf-8">

  <meta name="viewport" content="width=device-width, initial-scale=1">

  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">

  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>

  <style>

    body {

      background-repeat: no-repeat;

      background-size: cover;

      background-color: wheat;

    }

    .header {

      text-align: center;

      color: white;

    }

    .di {

      background-color:green;

      padding-left: 32;

      margin-left: 400;

      margin-right: 400;

      color: white;

      margin-bottom: 45;

      border-radius: 5px;

      padding-bottom: 30;

      width: 350px;

    }

    .name {

      padding-top: 20px;

      border-radius: 5px;

      padding-left: 25px;

    }

    .name1{

      padding-top: 15px;

      border-radius: 5px;

      padding-left: 105px;

      padding-bottom: 15px;

    }

    input {

      margin-top: 8;

      padding-right: 300;

      border-radius: 5px;

      padding-top: 5;

      width: 250px;

      color: black;

    }

    select {

      padding-right: 338;

      margin-top: 10;

      border-radius: 5px;

      padding-top: 5;

    }

    .id {

      padding-right: 350;

      margin-top: 10;

      border-radius: 5px;

      padding-top: 5;

    }

    button {

      color: white;

      padding-top: 12px;

      padding-bottom: 12px;

      padding-right: 208;

      padding-left: 225;

      margin-right: 83px;

      margin-top: 8px;

      text-align: center;

      border: 0;

      border-radius: 5px;

    }

    .heading{

      font-size: 45px;

      color: green;

      text-align: center;

    }

  </style>

</head>

<body background="assets/images/farm.jpg">

  <div class="container">

<div class="container" align=center>

    <fieldset>

        <legend>ADD FARMER</legend>

        <form #registerForm="ngForm" (ngSubmit)="register(registerForm.value)">

            <div class="name">

                UserName:<br />

                <input type="text" name="userName" [(ngModel)]='address.user.userName'><br />

              </div>

              <div class="name">

                Password:<br />

                <input type="password" name="password"><br />

              </div>

              <div class="name">

                Confirm Password:<br />

                <input type="password" name="password" [(ngModel)]='address.user.password'><br />

              </div>

              <div class="name">

                FullName:<br />

                <input type="text" name="fullName" [(ngModel)]='address.user.fullName'><br />

              </div>

              <div class="name">

                Age:<br />

                <input type="number" name="age" [(ngModel)]='address.user.age'><br />

              </div>

              <div class="name">

                Mobile Number:<br />

                <input type="text" name="mobileNumber" [(ngModel)]='address.user.mobileNumber'><br />

              </div>

              <div class="name">

                Pin-Code:<br />

                <input type="number" name="pincode" [(ngModel)]='address.pincode'><br />

              </div>

              <div class="name">

                City:<br />

                <input type="text" name="city" [(ngModel)]='address.city'><br />

              </div>

              <div class="name">

                State:<br />

                <input type="text" name="state" [(ngModel)]='address.state'><br />

              </div>

              <div class="name1">

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

            </div>

        </form>

    </fieldset>

    </div>

</div>

</body>

</html>

**ADDFARMER Doesn’t have userservice, methods in dao because im using the RegUser one which is used for registration of user.**

**1.ADD FARM:**

**ADDFARM COMPONENT.TS:**

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

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

@Component({

  selector: 'app-add-farm',

  templateUrl: './add-farm.component.html',

  styleUrls: ['./add-farm.component.css']

})

export class AddFarmComponent implements OnInit {

  farmdetails:any;

  addresses:any;

  constructor(private service:UserService) {

    this.farmdetails={crop:'',kindOfCrop:'',fieldArea:'',cropQuantity:'',date:'',

  address:{addressId:'',city:'',state:'',pincode:''}};

   }

  ngOnInit(): void {

    this.service.getAllAddress().subscribe((data: any) => {console.log(data); this.addresses = data; });

  }

addfarm(farmForm : any): void{

  this.service.addfarm(this.farmdetails).subscribe((result:any) => {console.log(result);});

  console.log(this.farmdetails);

  alert("details are recorded!.");

}

}

**2.ADD FARM COMPONENT.HTML:**

<!DOCTYPE html>

<html>

<head>

<meta name="viewport" content="width=device-width, initial-scale=1">

<meta name="viewport" content="width=device-width, initial-scale=1">

    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css" integrity="sha384-HSMxcRTRxnN+Bdg0JdbxYKrThecOKuH5zCYotlSAcp1+c8xmyTe9GYg1l9a69psu" crossorigin="anonymous">

    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">

    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"></script>

    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"></script>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>

    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js"></script>

    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">

<style>

 body{

      background-image: url(./assets/images/farm1.jpg);

      background-repeat: no-repeat;

      background-size: cover;

      background-color: wheat;

    }

\* {

  box-sizing: border-box;

}

/\* Add styles to the form container \*/

.container {

  position: absolute;

  right: 0;

  margin: 20px;

  max-width: 500px;

min-height: 520px;

  padding: 16px;

  background-color: white;

}

/\* Full-width input fields \*/

input[type=text], input[type=Crop] {

  width: 100%;

  padding: 15px;

  margin: 5px 0 22px 0;

  border: none;

  background: #f1f1f1;

}

input[type=number], input[type=CropQuantity] {

  width: 100%;

  padding: 15px;

  margin: 5px 0 22px 0;

  border: none;

  background: #f1f1f1;

}

input[type=number], input[type=FieldArea] {

  width: 100%;

  padding: 15px;

  margin: 5px 0 22px 0;

  border: none;

  background: #f1f1f1;

}

input[type=number], input[type=Date] {

  width: 100%;

  padding: 15px;

  margin: 5px 0 22px 0;

  border: none;

  background: #f1f1f1;

}

input[type=text]:focus, input[type=password]:focus {

  background-color: #ddd;

  outline: none;

}

/\* Set a style for the submit button \*/

.btn {

  background-color: #4CAF50;

  color: white;

  padding: 16px 20px;

  border: none;

  cursor: pointer;

  width: 100%;

  opacity: 0.9;

}

.btn:hover {

  opacity: 1;

}

</style>

</head>

<body background="./assets/images/farm1.jpg">

<div class="container" align=center>

    <fieldset>

        <legend>ADD FARM DETAILS</legend>

        <form #farmform="ngForm" (ngSubmit)="addfarm(farmform.value)">

            <label for="Crop"><b>Crop</b></label>

    <input type="text" placeholder="Enter Crop Name" name="crop" required [(ngModel)]='farmdetails.crop'>

    <label for="CropQuantity"><b>CropQuantity</b></label>

    <input type="number" placeholder="Enter Crop Quantity" name="cropQuantity" required   [(ngModel)]='farmdetails.cropQuantity'>

    <label for="FieldArea"><b>FieldArea</b></label>

    <input type="number" placeholder="Enter Field Area" name="fieldArea" required [(ngModel)]='farmdetails.fieldArea'>

 <label for="KindOfCrop"><b>KindOfCrop</b></label>

    <input type="text" placeholder="Enter Kind Of Crop" name="kindOfCrop" required [(ngModel)]='farmdetails.kindOfCrop'>

    <label for="KindOfCrop"><b>Select Address</b></label>

      <select name="address" [(ngModel)]='farmdetails.address.addressId'><br/>

            <option value={{address.addressId}} \*ngFor="let address of addresses">{{address.addressId}} </option>

        </select><br/>

         <button class="btn btn-primary" >ADD FARM</button>

        </form>

    </fieldset>

    </div>

</body>

</html>

**USERSERVICE.TS:**

addfarm(farmdetails : any){

    console.log('inside service: ');

    return this.httpClient.post('AGRIAPI/webapi/myresource/addfarm', farmdetails);

  }

getAllAddress(): any {

    return this.httpClient.get('AGRIAPI/webapi/myresource/getAllAddress');

   }

**METHODS IN MYRESOURCE:**

@Path("addfarm")

@POST

@Consumes(MediaType.***APPLICATION\_JSON***)

**public** FarmDetails addFarm(FarmDetails farmdetails){

System.***out***.println("Data received in addfarm: "+farmdetails);

FarmDetailsDAO farmdetailsDao= **new** FarmDetailsDAO();

farmdetailsDao.register(farmdetails);

**return** farmdetails;}

@Path("getAllAddress")

@GET

@Produces(MediaType.***APPLICATION\_JSON***)

**public** List<Address> getAllAddress(){

System.***out***.println("Recieved in getAllAddress : " );

AddressDAO addressDao = **new** AddressDAO();

List<Address> addressList = addressDao.getAllAddress() ;

**return** addressList;

}

**3. FARMDETAILSDAO:**

**public** **int** register(FarmDetails farmdetails) {

**return** HibernateTemplate.*addObject*(farmdetails);

}

**4.ADDRESSDAO:**

**public** List<Address> getAllAddress() {

List<Address> addressList=(List)HibernateTemplate.*getObjectListByQuery*("From Address");

**return** addressList;

}

**DONE:**

**IMPORTANT: ADD (STREET) IN ADDRESS , I FORGOT TO ADD THAT.**