Backend:

User.java:

package com.demo.main.entity;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

@Entity

public class User {

@Id

@Column(name="id")

@GeneratedValue(strategy=GenerationType.IDENTITY)

private int id;

private String password;

private String name;

private String dob;

private String emailid;

private String address;

private String gender;

private String mobile;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getEmailid() {

return emailid;

}

public void setEmailid(String emailid) {

this.emailid = emailid;

}

public String getAddress() {

return address;

}

public void setAddress(String address) {

this.address = address;

}

public String getGender() {

return gender;

}

public void setGender(String gender) {

this.gender = gender;

}

public String getMobile() {

return mobile;

}

public void setMobile(String mobile) {

this.mobile = mobile;

}

public String getDob() {

return dob;

}

public void setDob(String dob) {

this.dob = dob;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

}

DBUtil.java:

package com.demo.main.dbutil;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

public class DBUtil {

private static Connection connection= null;

public static Connection getConnection() throws SQLException {

if(connection!=null) {

return connection;

}else {

String driver="com.mysql.cj.jdbc.Driver";

String url="jdbc:mysql://localhost:3306/capstone\_proj";

String user="root";

String password="15243";

try {

Class.forName(driver);

connection= DriverManager.getConnection(url,user,password);

} catch (ClassNotFoundException e) {

e.printStackTrace();

}

}

return connection;

}

}

UserRepo.java:

package com.demo.main.repo;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.stereotype.Repository;

import com.demo.main.entity.User;

@Repository

public interface UserRepo extends JpaRepository<User, Integer>{

}

LoginRepo.java:

package com.demo.main.repo;

import org.springframework.stereotype.Repository;

@Repository

public interface LoginRepo {

public int LoginValidate(String mobile , String passoword);

}

AdminRepo.java:

package com.demo.main.repo;

import org.springframework.stereotype.Repository;

@Repository

public interface AdminRepo {

public int AdminLogin(String name , String passoword);

}

UserService.java:

package com.demo.main.service;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import com.demo.main.entity.User;

import com.demo.main.repo.UserRepo;

@Service

public class UserService {

@Autowired

private UserRepo repo;

public User addUser(User u) {

return repo.save(u);

}

public List<User> getAllUser(){

return repo.findAll();

}

public User getUserById(int id) {

if(repo.findById(id).isPresent()) {

return repo.findById(id).get();

}

else {

return null;

}

}

public User updateUser(int id, User newUser) {

if(repo.findById(id).isPresent()) {

User olduser= repo.findById(id).get();

olduser.setName(newUser.getName());

olduser.setPassword(newUser.getPassword());

olduser.setEmailid(newUser.getEmailid());

olduser.setAddress(newUser.getAddress());

olduser.setGender(newUser.getGender());

olduser.setMobile(newUser.getMobile());

olduser.setDob(newUser.getDob());

return repo.save(olduser);

}

else {

return null;

}

}

public boolean deleteUser(int id) {

if(repo.findById(id).isPresent()) {

repo.deleteById(id);

return true;

}

else{

return false;

}

}

}

LoginService.java:

package com.demo.main.service;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import org.springframework.stereotype.Service;

import com.demo.main.dbutil.DBUtil;

import com.demo.main.repo.LoginRepo;

@Service

public class LoginService implements LoginRepo {

Connection connection;

int flag;

public LoginService() throws SQLException {

connection=DBUtil.getConnection();

}

@Override

public int LoginValidate(String mobile, String password) {

try {

PreparedStatement statement=connection.prepareStatement("select \* from user where mobile='"+mobile+"'");

ResultSet rs= statement.executeQuery();

while(rs.next()) {

if(rs.getString(6).equals(mobile) && rs.getString(8).equals(password)) {

flag = 1;

}else {

flag=0;

System.out.println("Invalid Username or Password");

System.out.println(flag);

}

}

} catch (SQLException e) {

e.printStackTrace();

}

return flag;

}

}

AdminService.java:

package com.demo.main.service;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import org.springframework.stereotype.Service;

import com.demo.main.dbutil.DBUtil;

import com.demo.main.repo.AdminRepo;

@Service

public class AdminService implements AdminRepo{

Connection connection;

int flag;

public AdminService() throws SQLException {

connection=DBUtil.getConnection();

}

@Override

public int AdminLogin(String name , String password) {

try {

PreparedStatement statement=connection.prepareStatement("select \* from user where name='"+name+"'");

ResultSet rs= statement.executeQuery();

while(rs.next()) {

if(rs.getString(7).equals("Admin12") && rs.getString(8).equals("admin@45")) {

flag = 1;

}else {

flag=0;

System.out.println("Invalid Admin or Password");

System.out.println(flag);

}

}

} catch (SQLException e) {

e.printStackTrace();

}

return flag;

}

}

UserController.java:

package com.demo.main;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.DeleteMapping;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.PutMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.demo.main.entity.User;

import com.demo.main.service.UserService;

@CrossOrigin(origins = "http://localhost:4200")

@RestController

@RequestMapping("/api/user")

public class UserController {

@Autowired

private UserService service;

@PostMapping("/")

public ResponseEntity<User> adduser(@RequestBody User u){

User user= service.addUser(u);

if(user!=null)

return new ResponseEntity<User>(user,HttpStatus.CREATED);

else

return new ResponseEntity<User>(user,HttpStatus.INTERNAL\_SERVER\_ERROR);

}

@GetMapping("/")

public List<User> getAllUser(){

return service.getAllUser();

}

@GetMapping("/{id}")

public ResponseEntity<User>getUserById (@PathVariable int id){

User user= service.getUserById(id);

if(user!=null)

return new ResponseEntity<User>(user, HttpStatus.FOUND);

else

return new ResponseEntity<User>(user, HttpStatus.NOT\_FOUND);

}

@PutMapping("/{id}")

public ResponseEntity<Object> updateUser(@PathVariable int id,@RequestBody User newUser){

User user= service.updateUser(id, newUser);

if (user!=null)

return new ResponseEntity<Object>(user,HttpStatus.OK);

else

return new ResponseEntity<Object>("No User Available to Update",HttpStatus.NOT\_FOUND);

}

@DeleteMapping("/{id}")

public ResponseEntity<String>deleteUser(@PathVariable int id){

boolean result = service.deleteUser(id);

if(result)

return new ResponseEntity<String>("Object Deleted",HttpStatus.OK);

else

return new ResponseEntity<String>("NO user Found", HttpStatus.NOT\_FOUND);

}

}

LoginController.java:

package com.demo.main;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.demo.main.service.LoginService;

@CrossOrigin(origins = "http://localhost:4200")

@RestController

@RequestMapping("/api/user")

public class LoginController {

@Autowired

private LoginService service;

@GetMapping("/{mobile}/{password}")

public int UserLogin(@PathVariable("mobile")String mobile1,@PathVariable("password")String password1) {

int flag= service.LoginValidate(mobile1, password1);

if(flag==0) {

return 0;

}

return flag;

}

}

AdminController.java:

package com.demo.main;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.demo.main.service.AdminService;

@CrossOrigin(origins = "http://localhost:4200")

@RestController

@RequestMapping("/api")

public class AdminController {

@Autowired

private AdminService service;

@GetMapping("/{name}/{password}")

public int AdminLogin(@PathVariable("name")String name1,@PathVariable("password")String password1) {

int flag= service.AdminLogin(name1, password1);

if(flag==0) {

return 0;

}

return flag;

}

}

CapstoneProjectApplication.java:

package com.demo.main;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.annotation.ComponentScan;

@SpringBootApplication

@ComponentScan("com.demo.main")

public class CaptoneProjectApplication {

public static void main(String[] args) {

SpringApplication.run(CaptoneProjectApplication.class, args);

}

}

Frontend:

index.html:

<!doctype html>

<html lang="en">

<head>

<meta charset="utf-8">

<title>CapstoneProject</title>

<base href="/">

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

<link rel="icon" type="image/x-icon" href="favicon.ico">

<link rel="icon" type="image/x-icon" href="favicon.ico">

<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.2/dist/css/bootstrap.min.css" integrity="sha384-xOolHFLEh07PJGoPkLv1IbcEPTNtaed2xpHsD9ESMhqIYd0nLMwNLD69Npy4HI+N" crossorigin="anonymous">

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.2.0/css/all.min.css" integrity="sha512-xh6O/CkQoPOWDdYTDqeRdPCVd1SpvCA9XXcUnZS2FmJNp1coAFzvtCN9BmamE+4aHK8yyUHUSCcJHgXloTyT2A==" crossorigin="anonymous" referrerpolicy="no-referrer" />

</head>

<body>

<app-root></app-root>

<script src="https://cdn.jsdelivr.net/npm/jquery@3.5.1/dist/jquery.slim.min.js" integrity="sha384-DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj" crossorigin="anonymous"></script>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.2/dist/js/bootstrap.bundle.min.js" integrity="sha384-Fy6S3B9q64WdZWQUiU+q4/2Lc9npb8tCaSX9FK7E8HnRr0Jz8D6OP9dO5Vg3Q9ct" crossorigin="anonymous"></script>

</body>

</html>

app.module.ts:

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

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

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

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

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

import { LoginComponent } from './login/login.component';

import { HomeComponent } from './home/home.component';

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

import { RegisterComponent } from './register/register.component';

import { AdminComponent } from './admin/admin.component';

import { AdminloginComponent } from './adminlogin/adminlogin.component';

@NgModule({

declarations: [

AppComponent,

LoginComponent,

HomeComponent,

RegisterComponent,

AdminComponent,

AdminloginComponent,

],

imports: [

BrowserModule,

AppRoutingModule,

HttpClientModule,

FormsModule,

ReactiveFormsModule

],

providers: [],

bootstrap: [AppComponent]

})

export class AppModule { }

app-routing.module.ts:

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

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

import { AdminComponent } from './admin/admin.component';

import { AdminloginComponent } from './adminlogin/adminlogin.component';

import { HomeComponent } from './home/home.component';

import { LoginComponent } from './login/login.component';

import { RegisterComponent } from './register/register.component';

const routes: Routes = [

{path:"",component:LoginComponent},

{path:"home",component:HomeComponent},

{path:"register",component:RegisterComponent},

{path:"adminlogin",component:AdminloginComponent},

{path:"admin",component:AdminComponent}

];

@NgModule({

imports: [RouterModule.forRoot(routes)],

exports: [RouterModule]

})

export class AppRoutingModule { }

userservice.service.ts:

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

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

import { map } from 'rxjs';

@Injectable({

providedIn: 'root'

})

export class UserserviceService {

constructor(private http:HttpClient) { }

getAdminLogin(name:string,password:string){

return this.http.get('http://localhost:8083/api/'+name+'/'+password);

}

getLoginUser(mobile: string,password: string){

return this.http.get('http://localhost:8083/api/user/'+mobile+'/'+password);

}

url:string="http://localhost:8083/api/user/";

url1:string="http://localhost:8083/api/user";

postData(data:any){

return this.http.post<any>(this.url,data).pipe(map((res:any)=>{

return res;

}))

}

getDataById(id:number){

return this.http.get<any>(this.url1+"/"+id).pipe(map((res:any)=>{

return res;

}))

}

updateData(data:any,id:number){

return this.http.put<any>(this.url1+"/"+id,data).pipe(map((res:any)=>{

return res;

}))

}

deleteData(id :number){

return this.http.delete<any>(this.url1+"/"+id).pipe(map((res:any)=>{

return res;

}))

}

getData(){

return this.http.get<any>(this.url).pipe(map((res:any)=>{

return res;

}))

}

}

user.ts:

export class User {

id:number=0;

password:string="";

name:string="";

dob:string="";

emailid:string="";

address:string="";

gender:string="";

mobile:string="";

}

Login.component.ts:

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

import { Router } from '@angular/router';

import { UserserviceService } from '../userservice.service';

@Component({

selector: 'app-login',

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

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

})

export class LoginComponent implements OnInit {

model:any={}

getData: boolean = false;

constructor(private userservice:UserserviceService, private router:Router) { }

ngOnInit(): void {

}

loginUser4(){

var mobile=this.model.mobile;

var password=this.model.password

this.userservice.getLoginUser(mobile,password).subscribe((res:any)=>{

this.getData=res;

if(this.getData==true){

this.router.navigateByUrl("/home");

}else{

alert("Invalid Mobile No or Password");

}

})

}

registerPage() {

this.router.navigate(['/register']);

}

AdminLogin(){

this.router.navigate(['/adminlogin']);

}

}

Login.component.html:

<form action="/action\_page.php" method="post">

<div class="imgcontainer">

<img src="../../assets/Aadhaar\_Logo.svg.webp" alt="Avatar" class="avatar">

<h2>Login Form</h2>

</div>

<div class="container">

<label for="mobile"><b>Mobile Number</b></label>

<input type="text" placeholder="Enter Mobile No" name="mobile" [(ngModel)]="model.mobile">

<label for="password"><b>Password</b></label>

<input type="password" placeholder="Enter Password" name="password" [(ngModel)]="model.password">

<button type="submit" (click)="loginUser4()">Login</button>

<label>

<input type="checkbox" checked="checked" name="remember"> Remember me

</label>

</div>

<div class="container" style="background-color:#f1f1f1">

<h2>For new users :</h2>

<button type="button" class="cancelbtn" (click)="registerPage()">Register</button>

</div>

<div class="container" style="background-color:#f1f1f1">

<h2>For Admin only :</h2>

<button type="button" class="cancelbtn" (click)="AdminLogin()">Login</button>

</div>

</form>

home.component.ts:

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

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

import { Router } from '@angular/router';

import { User } from '../login/user';

import { UserserviceService } from '../userservice.service';

@Component({

selector: 'app-home',

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

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

})

export class HomeComponent implements OnInit {

userobj:User=new User();

formValue!:FormGroup;

userData!:any;

showAdd!:boolean;

showUpdate!:boolean;

showUpdateTitle!:boolean

showAddTitle!:boolean;

constructor(private formBulider:FormBuilder, private service:UserserviceService, private router:Router) { }

ngOnInit(): void {

this.formValue=this.formBulider.group({

id:[''],

name:[''],

password:[''],

dob:[''],

emailid:[''],

address:[''],

gender:[''],

mobile:['']

})

this.getAllUser();

}

getAllUser(){

this.service.getData().subscribe(a=>{

this.userData=a;

})

}

updateUser(row:any){

this.showUpdate=true;

this.showAdd=false;

this.showUpdateTitle=true;

this.showAddTitle=false;

this.userobj.id=row.id;

this.formValue.controls['name'].setValue(row.name);

this.formValue.controls['password'].setValue(row.password);

this.formValue.controls['dob'].setValue(row.dob);

this.formValue.controls['emailid'].setValue(row.emailid);

this.formValue.controls['address'].setValue(row.address);

this.formValue.controls['gender'].setValue(row.gender);

this.formValue.controls['mobile'].setValue(row.mobile);

}

updateUserDetails(){

this.userobj.name=this.formValue.value.name;

this.userobj.password=this.formValue.value.password;

this.userobj.dob=this.formValue.value.dob;

this.userobj.mobile=this.formValue.value.mobile;

this.userobj.address=this.formValue.value.address;

this.userobj.emailid=this.formValue.value.emailid;

this.userobj.gender=this.formValue.value.gender;

this.service.updateData(this.userobj,this.userobj.id).subscribe(a=>{

alert("Record Updated Successfully");

let cancel=document.getElementById("cancel");

cancel?.click();

this.formValue.reset();

this.getAllUser();

})

}

getUserById(row:any){

this.service.getDataById(row.id).subscribe(a=>{

this.userData=a;

})

}

loginPage(){

this.router.navigateByUrl("/");

}

registerPage(){

this.router.navigateByUrl("/register");

}

}

home.component.html:

<div class="container" style="margin-top: 100px;">

<div class="image">

<img src="../../assets/Sample.png" alt="Avatar" class="avatar">

</div>

<div class="jumbotron">

<button type="button" class="btn btn-info mb-3 float-right" (click)="loginPage()">

Back To Login

</button>

<button type="button" class="btn btn-success mb-3 float-left" (click)="registerPage()">

Apply New Addhar

</button>

<div \*ngFor="let row of userData.slice(0,1)">

<button type="button" (click)="updateUser(row)" class="btn btn-primary mb-3 float-center" data-toggle="modal" data-target="#exampleModal">

Update Addhar

</button>

</div>

<button type="button" class="btn btn-danger mb-3 float-center" data-toggle="modal" data-target="#exampleModal">

Admin Portal

</button>

</div>

</div>

<div class="modal fade" id="exampleModal" tabindex="-1" role="dialog" aria-labelledby="exampleModalLabel" aria-hidden="true">

<div class="modal-dialog" role="document">

<div class="modal-content">

<div class="modal-header">

<h5 class="modal-title text-primary" \*ngIf="showUpdateTitle" id="exampleModalLabel">Update Data</h5>

<button type="button" class="close" data-dismiss="modal" aria-label="Close">

<span aria-hidden="true">&times;</span>

</button>

</div>

<div class="modal-body">

<form [formGroup]="formValue">

<div class="form-group">

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

<input formControlName="name" id="name" type="text" class="form-control" placeholder="Enter Name.." required>

</div>

<div class="form-group">

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

<input id="password" formControlName="password" type="text" class="form-control" placeholder="Enter Password.." required>

</div>

<div class="form-group">

<label for="emailid">EmailId</label>

<input id="emailid" type="email" formControlName="emailid" class="form-control" placeholder="Enter email.." required> <br>

</div>

<div class="form-group">

<label for="dob" >Date Of Birth</label>

<input id="dob" formControlName="dob" type="text" class="form-control" placeholder="Enter Dob.." required>

</div>

<div class="form-group">

<label for="address" >Address</label>

<input id="address" formControlName="address" type="text" class="form-control" placeholder="Enter address.." required>

</div>

<div class="form-group">

<label for="gender" >Gender</label>

<input id="gender" formControlName="gender" type="text" class="form-control" placeholder="Enter gender.." required>

</div>

<div class="form-group">

<label for="mobile" >Mobile No</label>

<input id="mobile" formControlName="mobile" type="text" class="form-control" placeholder="Enter mobile.." required>

</div>

<button type="button" id="cancel" class="btn btn-primary" data-dismiss="modal">Cancel</button>&nbsp;

<button type="submit" (click)= "updateUserDetails()" class="btn btn-primary">Update</button>

</form>

</div>

</div>

</div>

</div>

Adminlogin.components.ts:

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

import { Router } from '@angular/router';

import { UserserviceService } from '../userservice.service';

@Component({

selector: 'app-adminlogin',

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

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

})

export class AdminloginComponent implements OnInit {

model:any={}

getData: boolean = false;

constructor(private userservice:UserserviceService, private router:Router) { }

ngOnInit(): void {

}

adminlogin(){

var name = this.model.name;

var password = this.model.password;

this.userservice.getAdminLogin(name,password).subscribe((res:any)=>{

this.getData=res;

if(this.getData==true){

this.router.navigateByUrl("/admin")

}else{

alert("Invalid Admin")

}

})

}

}

Adminlogin.component.html:

<form action="/action\_page.php" method="post">

<div class="imgcontainer">

<img src="../../assets/Aadhaar\_Logo.svg.webp" alt="Avatar" class="avatar">

<h2>Admin Login</h2>

</div>

<div class="container">

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

<input type="text" placeholder="Enter Name" name="Name" [(ngModel)]="model.name">

<label for="password"><b>Password</b></label>

<input type="password" placeholder="Enter Password" name="password" [(ngModel)]="model.password">

<button type="submit" (click)="adminlogin()">Login</button>

<label>

<input type="checkbox" checked="checked" name="remember"> Remember me

</label>

</div>

</form>

Admin.component.ts:

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

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

import { Router } from '@angular/router';

import { User } from '../login/user';

import { UserserviceService } from '../userservice.service';

@Component({

selector: 'app-admin',

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

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

})

export class AdminComponent implements OnInit {

userobj:User=new User();

formValue!:FormGroup;

userData!:any;

showAdd!:boolean;

showUpdate!:boolean;

showUpdateTitle!:boolean

showAddTitle!:boolean;

constructor(private formBulider:FormBuilder, private service:UserserviceService, private router:Router) { }

ngOnInit(): void {

this.formValue=this.formBulider.group({

name:[''],

password:[''],

dob:[''],

emailid:[''],

address:[''],

gender:[''],

mobile:['']

})

this.getAllUser();

}

addButtonClickFunction(){

this.formValue.reset();

this.showUpdate=false;

this.showAdd=true;

this.showUpdateTitle=false;

this.showAddTitle=true;

}

postUserDetails(){

this.userobj.id=this.formValue.value.id;

this.userobj.name=this.formValue.value.name;

this.userobj.password=this.formValue.value.password;

this.userobj.dob=this.formValue.value.dob;

this.userobj.emailid=this.formValue.value.emailid;

this.userobj.address=this.formValue.value.address;

this.userobj.gender=this.formValue.value.gender;

this.userobj.mobile=this.formValue.value.mobile;

let cancel = document.getElementById("cancel");

this.service.postData(this.userobj).subscribe(a=>{

console.log(a);

alert("Record inserted Successfully")

cancel?.click();this.formValue.reset();

this.getAllUser();

})

this.router.navigateByUrl("/")

}

getAllUser(){

this.service.getData().subscribe(a=>{

this.userData=a;

})

}

deleteUser(row:any){

this.service.deleteData(row.id).subscribe(a=>{

alert("Record Deleted Successfully");

this.getAllUser();

})

}

updateUser(row:any){

this.showUpdate=true;

this.showAdd=false;

this.showUpdateTitle=true;

this.showAddTitle=false;

this.userobj.id=row.id;

this.formValue.controls['name'].setValue(row.name);

this.formValue.controls['password'].setValue(row.password);

this.formValue.controls['dob'].setValue(row.dob);

this.formValue.controls['emailid'].setValue(row.emailid);

this.formValue.controls['address'].setValue(row.address);

this.formValue.controls['gender'].setValue(row.gender);

this.formValue.controls['mobile'].setValue(row.mobile);

}

updateUserDetails(){

this.userobj.name=this.formValue.value.name;

this.userobj.password=this.formValue.value.password;

this.userobj.dob=this.formValue.value.dob;

this.userobj.mobile=this.formValue.value.mobile;

this.userobj.address=this.formValue.value.address;

this.userobj.emailid=this.formValue.value.emailid;

this.userobj.gender=this.formValue.value.gender;

this.service.updateData(this.userobj,this.userobj.id).subscribe(a=>{

alert("Record Updated Successfully");

let cancel=document.getElementById("cancel");

cancel?.click();

this.formValue.reset();

this.getAllUser();

})

}

getUserById(row:any){

this.service.getDataById(row.id).subscribe(a=>{

this.userData=a;

})

}

returnLogin(){

this.router.navigateByUrl("/");

}

}

Admin.component.html:

<div class="container" style="margin-top: 100px;">

<div class="jumbotron">

<div class="row">

<div class="col-12">

<h2 class="text-info float-left">Admin Dashboad</h2>

<button type="button" (click)="addButtonClickFunction()" class="btn btn-info mb-3 float-right" data-toggle="modal" data-target="#exampleModal">

<i class="fas fa-plus"></i> Add New User

</button>

</div>

<div class="col-12">

<table class="table table-sm table-stripped">

<thead>

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

<th>ID</th>

<th>NAME</th>

<th>PASSWORD</th>

<th>EMAIL ID</th>

<th>DOB</th>

<th>ADDRESS</th>

<th>GENDER</th>

<th>MOBILE NO</th>

<th>ACTIONS</th>

</tr>

</thead>

<tbody>

<tr \*ngFor="let row of userData" class="text-center">

<td>{{row.id}}</td>

<td>{{row.name}}</td>

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

<td>{{row.emailid}}</td>

<td>{{row.dob}}</td>

<td>{{row.address}}</td>

<td>{{row.gender}}</td>

<td>{{row.mobile}}</td>

<td class="text-right">

<button (click)="updateUser(row)" class="btn btn-outline-info pl-3 pr-3 mx-2"data-toggle="modal" data-target="#exampleModal"><i class="fas fa-pencil"></i> Edit</button>

<button (click)="deleteUser(row)" class="btn btn-outline-danger pl-3 " ><i class="fas fa-trash"></i> Delete</button>

</td>

</tr>

</tbody>

<button type="button" (click)="returnLogin()" class="btn btn-info mb-3 float-right" data-toggle="modal" data-target="#exampleModal">

Back To Login

</button>

</table>

</div>

</div>

</div>

</div>

<div class="modal fade" id="exampleModal" tabindex="-1" role="dialog" aria-labelledby="exampleModalLabel" aria-hidden="true">

<div class="modal-dialog" role="document">

<div class="modal-content">

<div class="modal-header">

<h5 class="modal-title text-primary" \*ngIf="showAddTitle" id="exampleModalLabel">Add New Data</h5>

<h5 class="modal-title text-primary" \*ngIf="showUpdateTitle" id="exampleModalLabel">Update Data</h5>

<button type="button" class="close" data-dismiss="modal" aria-label="Close">

<span aria-hidden="true">&times;</span>

</button>

</div>

<div class="modal-body">

<form [formGroup]="formValue">

<div class="form-group">

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

<input formControlName="name" id="name" type="text" class="form-control" placeholder="Enter Name.." required>

</div>

<div class="form-group">

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

<input id="password" formControlName="password" type="text" class="form-control" placeholder="Enter Password.." required>

</div>

<div class="form-group">

<label for="emailid">EmailId</label>

<input id="emailid" type="email" formControlName="emailid" class="form-control" placeholder="Enter email.." required> <br>

</div>

<div class="form-group">

<label for="dob" >Date Of Birth</label>

<input id="dob" formControlName="dob" type="text" class="form-control" placeholder="Enter Dob.." required>

</div>

<div class="form-group">

<label for="address" >Address</label>

<input id="address" formControlName="address" type="text" class="form-control" placeholder="Enter address.." required>

</div>

<div class="form-group">

<label for="gender" >Gender</label>

<input id="gender" formControlName="gender" type="text" class="form-control" placeholder="Enter gender.." required>

</div>

<div class="form-group">

<label for="mobile" >Mobile No</label>

<input id="mobile" formControlName="mobile" type="text" class="form-control" placeholder="Enter mobile.." required>

</div>

<button type="button" id="cancel" class="btn btn-primary" data-dismiss="modal">Cancel</button>&nbsp;

<button type="submit" \*ngIf="showAdd" (click)="postUserDetails()" class="btn btn-primary mx-2">Save</button>

<button type="submit" \*ngIf="showUpdate" (click)="updateUserDetails()" class="btn btn-primary">Update</button>

</form>

</div>

</div>

</div>

</div>

Testing:

CapstoneTest.java:

package com.demo.main;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

public class CapstoneTest {

WebDriver driver;

@BeforeMethod

public void beforeMethod() {

System.setProperty("webdriver.chrome.driver", "C:\\Users\\Eshwar\\Testing\\chromedriver\_win32\\chromedriver.exe");

driver= new ChromeDriver();

driver.manage().window().maximize();

driver.get("http://localhost:4200");

}

@AfterMethod

public void afterMethod() {

driver=null;

}

@Test

public void loginTest() throws Exception {

System.out.println(driver.getTitle());

Thread.sleep(3000);

driver.findElement(By.xpath("/html/body/app-root/app-login/form/div[2]/input[1]")).sendKeys("8595746785");

driver.findElement(By.cssSelector("body > app-root > app-login > form > div:nth-child(2) > input:nth-child(4)")).sendKeys("MSd\*7");

driver.findElement(By.cssSelector("body > app-root > app-login > form > div:nth-child(2) > button")).click();

}

@Test

public void registerTest() throws Exception{

Thread.sleep(3000);

driver.findElement(By.xpath("/html/body/app-root/app-login/form/div[3]/button")).click();

Thread.sleep(2000);

driver.findElement(By.xpath("//\*[@id=\"name\"]")).sendKeys("Thea");

driver.findElement(By.xpath("//\*[@id=\"password\"]")).sendKeys("T67");

driver.findElement(By.xpath("//\*[@id=\"emailid\"]")).sendKeys("thea12@gmail.com");

driver.findElement(By.xpath("//\*[@id=\"dob\"]")).sendKeys("6-8-2000");

driver.findElement(By.xpath("//\*[@id=\"address\"]")).sendKeys("No:01,New York");

driver.findElement(By.xpath("//\*[@id=\"gender\"]")).sendKeys("F");

driver.findElement(By.xpath("//\*[@id=\"mobile\"]")).sendKeys("8585969647");

driver.findElement(By.cssSelector("body > app-root > app-register > div > div > form > button")).click();

}

@Test

public void admin() throws Exception {

System.out.println(driver.getTitle());

Thread.sleep(3000);

driver.findElement(By.cssSelector("body > app-root > app-login > form > div:nth-child(4) > button")).click();

Thread.sleep(2000);

driver.findElement(By.xpath("/html/body/app-root/app-adminlogin/form/div[2]/input[1]")).sendKeys("Admin12");

driver.findElement(By.cssSelector("body > app-root > app-adminlogin > form > div.container > input:nth-child(4)")).sendKeys("admin@45");

driver.findElement(By.cssSelector("body > app-root > app-adminlogin > form > div.container > button")).click();

}

}