**mAadhaar- Source code**

Frontend

**admin.component.css**

.image{

height:39em; background-size:cover; width:auto;

background-image:url('aadhar.jpg');

background-position:50% 50%;

}

input.ng-invalid.ng-touched {

border: 1px solid red;

}

## ****admin.component.html****

<div class="image">

<br />

<br />

<div class="container d-flex justify-content-center">

<form

[formGroup]="loginForm"

(ngSubmit)="onSubmit()"

class="w-50 p-4"

style="color: rgb(3, 11, 51)"

>

<h4 class="text-center">Admin Login</h4>

<div class="row">

<div class="form-group">

<label for="username">Admin name</label>

<input

type="text"

id="admin"

class="form-control"

formControlName="admin"

/>

<span

style="color: red"

class="small"

\*ngIf="

!loginForm.get('admin').valid && loginForm.get('admin').touched

"

>Please enter a valid admin name</span

>

</div>

</div>

<div class="row">

<div class="form-group">

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

<input

type="password"

id="password"

class="form-control"

formControlName="password"

/>

<span

style="color: red"

class="small"

\*ngIf="

!loginForm.get('password').valid &&

loginForm.get('password').touched

"

>Please enter a valid password</span

>

<br />

<span

style="color: red"

class="small"

\*ngIf="loginForm.get('password').hasError('passwordStrength')"

>

{{ loginForm.get("password").errors["passwordStrength"] }}

</span>

</div>

</div>

<br />

<button type="submit" class="btn btn-outline-danger">Login</button>

</form>

</div>

</div>

## ****admin.component.ts****

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

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

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

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

import { PasswordStrengthValidator } from "./password-strength.validators"

@Component({

selector: 'app-admin',

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

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

})

export class AdminComponent implements OnInit {

public loginForm!: FormGroup;

url="http://localhost:8080/admin";

submitted=false;

constructor(private formBuilder: FormBuilder,private http: HttpClient , private router:Router) { }

ngOnInit(): void {

this.loginForm=this.formBuilder.group({

admin : ['',Validators.required],

password : ['',Validators.compose([

Validators.required, PasswordStrengthValidator])]

})

}

onSubmit() {

this.submitted = true;

this.login();

}

login(){

this.http.get<any>(this.url).subscribe(res=>{

const log = res.find((a:any)=>{

return a.admin === this.loginForm.value.admin && a.password === this.loginForm.value.password

});

if(log){

alert("Login Success,Click Ok to continue!!");

this.loginForm.reset();

this.router.navigate(['adminhome']);

}else{

alert("Try Again!!");

}

},err=>{

alert("Something Went Wrong!!");

this.loginForm.reset();

})

}

}

## ****password-strength.validators.ts****

import { AbstractControl, ValidationErrors } from "@angular/forms"

export const PasswordStrengthValidator = function (control: AbstractControl): ValidationErrors | null {

let value: string = control.value || '';

if (!value) {

return null

}

let upperCaseCharacters = /[A-Z]+/g

if (upperCaseCharacters.test(value) === false) {

return { passwordStrength: `Upper case required` };

}

let lowerCaseCharacters = /[a-z]+/g

if (lowerCaseCharacters.test(value) === false) {

return { passwordStrength: `lower case required` };

}

let numberCharacters = /[0-9]+/g

if (numberCharacters.test(value) === false) {

return { passwordStrength: `number required` };

}

let specialCharacters = /[!@#$%^&\*()\_+\-=\[\]{};':"\\|,.<>\/?]+/

if (specialCharacters.test(value) === false) {

return { passwordStrength: `Special char required` };

}

return null;

}

## ****adminhome.component.css****

.image{

height:39em; background-size:cover; width:auto;

background-image:url('aadhar.jpg');

background-position:50% 50%;

}

## ****adminhome.component.html****

<div class="image">

<div>

<h1 class="d-flex justify-content-center" style="color: crimson">

AADHAR APPLICATION

</h1>

</div>

<div class="container">

<button class="btn btn-outline-danger mt-5" (click)="onFetchUsers()">

View All

</button>

<div class="row m-3">

<div class="col">

<p \*ngIf="fetchedUsers.length < 1">No Users Available !!</p>

<ul class="list-group" \*ngIf="fetchedUsers.length >= 1">

<li class="list-group-item" \*ngFor="let user of fetchedUsers">

<p><b>Aadhar id:</b>&nbsp;{{ user.citizenId }}</p>

<p><b>Name&nbsp;:</b>&nbsp;{{ user.name }}</p>

<p><b>Date Of Birth:</b>&nbsp;{{ user.dob }}</p>

<p><b>Email:</b>&nbsp;{{ user.emailid }}</p>

<p><b>Address:</b>&nbsp;{{ user.address }}</p>

<p><b>Mobile No:</b>&nbsp;{{ user.mobileno }}</p>

<p><b>Gender:</b>&nbsp;{{ user.gender }}</p>

<button

class="btn btn-sm btn-danger"

(click)="onDeleteUser(user.citizenId)"

>

Delete

</button>

</li>

</ul>

</div>

</div>

</div>

</div>

## ****adminhome.component.ts****

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

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

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

import { map } from 'rxjs/operators';

import { ApiService } from '../service/api.service';

import { UserData } from './userdata.model';

@Component({

selector: 'app-adminhome',

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

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

})

export class AdminhomeComponent implements OnInit {

fetchedUsers: UserData[] = [];

backendurl = 'http://localhost:8080/citizens';

constructor(private http:HttpClient,private formBuilder:FormBuilder,private api:ApiService) {}

ngOnInit(): void {

}

onFetchUsers(){

this.fetchUsers();

}

fetchUsers() {

this.http

.get(this.backendurl)

.pipe(

map((responseData) => {

const userArray: UserData[] = [];

for (const key in responseData) {

var x = { ...responseData[key] };

userArray.push(x);

}

return userArray;

})

)

.subscribe((users) => {

this.fetchedUsers = users;

});

}

onDeleteUser(id: number) {

this.http.delete(this.backendurl + '/' + id).subscribe((response) => {

console.log('User deleted: ' + response);

// this.fetchPosts();

});

}

}

## ****userdata.model.ts****

export interface UserData{

citizenId:number,

name:string,

dob:string,

mobileno:string,

address:string,

emailid:string,

gender:string

}

## ****home.component.css****

.image{

height:39em; background-size:cover; width:auto;

background-image:url('aadhar.jpg');

background-position:50% 50%;

}

## ****home.component.html****

<div class="image">

<div class="container">

<h1 class="d-flex justify-content-center" style="color: crimson">

AADHAR APPLICATION

</h1>

<a

type="button"

class="d-inline btn btn-outline-danger"

routerLink="/userlogin"

>

Citizen Login</a

>&nbsp;&nbsp;New User?<a routerLink="/register">Register Here</a>

<br />

<br />

<a

type="button"

class="d-inline btn btn-outline-danger"

routerLink="/adminlogin"

>

Admin Login

</a>

</div>

</div>

## ****home.component.ts****

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

@Component({

selector: 'app-home',

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

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

})

export class HomeComponent implements OnInit {

constructor() { }

ngOnInit(): void {

}

}

## ****register.component.css****

.image{

height:39em; background-size:cover; width:auto;

background-image:url('aadhar.jpg');

background-position:50% 50%;

}

## ****register.component.html****

<div class="image">

<br />

<h3 style="color: rgba(42, 2, 71, 0.84)" class="text-center">

Registration Page

</h3>

<div class="container mt-3 d-flex justify-content-center">

<form #f="ngForm" class="w-50 p-4" style="color: rgb(120, 55, 5)">

<div class="row mb-2">

<div class="form-group">

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

<input

type="text"

id="name"

class="form-control"

name="name"

required

ngModel

#name="ngModel"

/>

</div>

</div>

<div class="row mb-2">

<div class="form-group">

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

<input

type="text"

id="dob"

class="form-control"

name="dob"

ngModel

required

dob

#dob="ngModel"

/>

<span

id="err-dob"

style="color: red"

class="small"

\*ngIf="!dob.valid && dob.touched"

>Please enter a valid dob</span

>

</div>

</div>

<div class="row mb-2">

<div class="form-group">

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

<input

type="text"

id="emailid"

class="form-control"

name="emailid"

ngModel

required

emailid

#emailid="ngModel"

/>

<span

id="err-email"

style="color: red"

class="small"

\*ngIf="!emailid.valid && emailid.touched"

>Please enter a valid email</span

>

</div>

</div>

<div class="row mb-2">

<div class="form-group">

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

<textarea

type="text"

id="address"

class="form-control"

name="address"

ngModel

required

address

#address="ngModel"

></textarea>

<span

id="err-address"

style="color: red"

class="small"

\*ngIf="!address.valid && address.touched"

>Please enter a valid address</span

>

</div>

</div>

<div class="row mb-2">

<div class="form-group">

<label for="Phoneno">Mobile No:</label>

<input

type="text"

id="mobileno"

class="form-control"

name="mobileno"

ngModel

required

mobileno

#mobileno="ngModel"

/>

<span

id="err-mobile"

style="color: red"

class="small"

\*ngIf="!mobileno.valid && mobileno.touched"

>Please enter a valid Mobile Number</span

>

</div>

</div>

<div class="row mb-2">

<div class="form-group">

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

<input

type="text"

id="gender"

class="form-control"

name="gender"

ngModel

required

gender

#gender="ngModel"

/>

<span

id="err-gender"

style="color: red"

class="small"

\*ngIf="!gender.valid && gender.touched"

>Please enter a valid Gender</span

>

</div>

</div>

<button

class="btn btn-outline-danger m-2"

[disabled]="!f.valid"

(click)="onAddCitizen(f.value, f)"

id="register"

>

Register

</button>

</form>

</div>

</div>

## ****register.component.ts****

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

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

import { NgForm } from '@angular/forms';

@Component({

selector: 'app-register',

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

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

})

export class RegisterComponent implements OnInit {

url='http://localhost:8080/citizens';

constructor(private http:HttpClient) { }

ngOnInit(): void {

}

onAddCitizen(citizenData: {name:string,dob:string,emailid:string,gender:string,mobileno:string,address:string }, form: NgForm) {

this.http.post(this.url, citizenData).subscribe((responseData) => {

console.log(responseData);

alert("Registered Successfully!!");

form.reset();

},err=>{

alert("Something Happened!!")

});

}

}

## ****api.service.ts****

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

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

import { map } from 'rxjs/operators';

@Injectable({

providedIn: 'root'

})

export class ApiService {

constructor(private http:HttpClient) { }

UpdateUser(data:any,id:number){

return this.http.put<any>('http://localhost:8080/citizens',data)

.pipe(map((res:any)=>{

return res;

}))

}

}

## ****user.component.css****

.image{

height:39em; background-size:cover; width:auto;

background-image:url('aadhar.jpg');

background-position:50% 50%;

}

## ****user.component.html****

<div class="image">

<br />

<br />

<div class="container d-flex justify-content-center">

<form

[formGroup]="loginForm"

(ngSubmit)="onSubmit()"

class="w-50 p-4"

style="color: rgb(3, 11, 51)"

>

<h4 class="text-center">Citizen Login</h4>

<div class="row">

<div class="form-group">

<label for="citizenId">AadharId</label>

<br />

<input

type="number"

id="citizen\_id"

class="form-control"

formControlName="citizenId"

/>

</div>

</div>

<div class="row">

<div class="form-group">

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

<br />

<span class="small" style="color: darkgreen"

>Your mobile number is your password</span

>

<input

type="password"

id="password"

class="form-control"

formControlName="mobileno"

/>

<span

style="color: red"

class="small"

\*ngIf="

!loginForm.get('mobileno').valid &&

loginForm.get('mobileno').touched

"

>Please enter a valid password</span

>

</div>

</div>

<br />

<button type="submit" class="btn btn-outline-danger">Login</button>

</form>

</div>

</div>

## ****user.component.ts****

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

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

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

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

@Component({

selector: 'app-user',

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

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

})

export class UserComponent implements OnInit {

public loginForm!: FormGroup;

url="http://localhost:8080/citizens";

submitted=false;

constructor(private formBuilder: FormBuilder,private http: HttpClient , private router:Router) { }

ngOnInit(): void {

this.loginForm=this.formBuilder.group({

citizenId : ['',Validators.required],

mobileno : ['',Validators.required]

})

}

onSubmit() {

this.submitted = true;

this.login();

}

login(){

this.http.get<any>(this.url).subscribe(res=>{

const log = res.find((a:any)=>{

return a.citizenId === this.loginForm.value.citizenId && a.mobileno === this.loginForm.value.mobileno

});

if(log){

alert("Login Success,Click Ok to continue!!");

this.loginForm.reset();

this.router.navigate(['userhome']);

}else{

alert("Try Again!!");

}

},err=>{

alert("Something Went Wrong!!");

this.loginForm.reset();

})

}

}

## ****user.model.ts****

export class UserModel{

citizenId:number;

name:string;

dob:string;

mobileno:string;

address:string;

emailid:string;

gender:string;

}

## ****userhome.component.html****

<div class="image">

<div class="container">

<div class="h1 d-flex justify-content-center">AADHAR APPLICATION</div>

<div class="col card d-flex justify-content-center" style="width: 25rem">

<div class="card-body">

<h5 class="card-title" style="color: red">Your Details</h5>

<div class="card-text" \*ngFor="let user of fetchedUsers">

<p><b>Aadhar id:</b>&nbsp;{{ user.citizenId }}</p>

<p><b>Name&nbsp;:</b>&nbsp;{{ user.name }}</p>

<p><b>Date Of Birth:</b>&nbsp;{{ user.dob }}</p>

<p><b>Email:</b>&nbsp;{{ user.emailid }}</p>

<p><b>Address:</b>&nbsp;{{ user.address }}</p>

<p><b>Mobile No:</b>&nbsp;{{ user.mobileno }}</p>

<p><b>Gender:</b>&nbsp;{{ user.gender }}</p>

<button

type="button"

class="btn btn-outline-primary"

data-bs-toggle="modal"

data-bs-target="#exampleModal"

(click)="onEdit(user)"

>

Update</button

>&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;

<a

type="button"

class="btn btn-outline-primary align-right"

routerLink="/home"

>Logout</a

>

</div>

</div>

</div>

</div>

<div class="footer bg-secondary d-flex align-items-end"></div>

</div>

<!-- Modal -->

<div

class="modal fade"

id="exampleModal"

tabindex="-1"

aria-labelledby="exampleModalLabel"

aria-hidden="true"

>

<div class="modal-dialog">

<div class="modal-content">

<div class="modal-header">

<h5 class="modal-title" id="exampleModalLabel">Update User</h5>

<button

type="button"

class="btn-close"

data-bs-dismiss="modal"

aria-label="Close"

></button>

</div>

<div class="modal-body">

<form [formGroup]="loginForm" style="color: rgb(120, 55, 5)">

<div class="row mb-2">

<div class="form-group">

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

<input

type="text"

id="name"

class="form-control"

name="name"

required

formControlName="name"

/>

</div>

</div>

<div class="row mb-2">

<div class="form-group">

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

<input

type="text"

id="dob"

class="form-control"

name="dob"

formControlName="dob"

required

/>

</div>

</div>

<div class="row mb-2">

<div class="form-group">

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

<input

type="text"

id="emailid"

class="form-control"

name="emailid"

formControlName="emailid"

required

/>

</div>

</div>

<div class="row mb-2">

<div class="form-group">

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

<textarea

type="text"

id="address"

class="form-control"

name="address"

required

formControlName="address"

></textarea>

</div>

</div>

<div class="row mb-2">

<div class="form-group">

<label for="Phoneno">Mobile No:</label>

<input

type="text"

id="mobileno"

class="form-control"

name="mobileno"

required

formControlName="mobileno"

/>

</div>

</div>

<div class="row mb-2">

<div class="form-group">

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

<input

type="text"

id="gender"

class="form-control"

name="gender"

required

formControlName="gender"

/>

</div>

</div>

</form>

<div class="modal-footer">

<button

type="button"

class="btn btn-secondary"

data-bs-dismiss="modal"

>

Close

</button>

<button type="button" (click)="updateUser()" class="btn btn-primary">

Save changes

</button>

</div>

</div>

</div>

</div>

</div>

Index.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8" />

<title>AadharApp</title>

<base href="/" />

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

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

<style>

body {

background-image: url("");

background-repeat: no-repeat;

background-attachment: fixed;

background-size: cover;

}

</style>

</head>

<body>

<app-root></app-root>

</body>

</html>

[Backend](https://github.com/Archu5721/CapstoneProject/tree/main/AadharRestApi/AadharBackend)

Controller:

aadharApiexception.java

package com.example.capstone.AadharRestApi.Controller;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

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

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

import com.examples.capstone.AadharRestApi.Exception.AdminNotFoundException;

import com.examples.capstone.AadharRestApi.Exception.CitizenNotFoundException;

import com.examples.capstone.AadharRestApi.Exception.UserNotFoundException;

@ControllerAdvice

public class AadharApiException {

@ExceptionHandler(value=AdminNotFoundException.class)

public ResponseEntity<Object> handleException(AdminNotFoundException ex) {

return new ResponseEntity<Object>(ex.getMessage(), HttpStatus.NOT\_FOUND);

}

@ExceptionHandler(value=UserNotFoundException.class)

public ResponseEntity<Object> handleException(UserNotFoundException ex) {

return new ResponseEntity<Object>(ex.getMessage(), HttpStatus.NOT\_FOUND);

}

@ExceptionHandler(value=CitizenNotFoundException.class)

public ResponseEntity<Object> handleException(CitizenNotFoundException ex) {

return new ResponseEntity<Object>(ex.getMessage(), HttpStatus.NOT\_FOUND);

}

}

RestApiController.java

package com.example.capstone.AadharRestApi.Controller;

import java.util.Optional;

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

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.RestController;

import com.example.capstone.AadharRestApi.Entity.Admin;

import com.example.capstone.AadharRestApi.Entity.Citizens;

import com.example.capstone.AadharRestApi.Entity.User;

import com.example.capstone.AadharRestApi.Repository.AdminRepo;

import com.example.capstone.AadharRestApi.Repository.CitizensRepo;

import com.example.capstone.AadharRestApi.Repository.UserRepo;

import com.examples.capstone.AadharRestApi.Exception.AdminNotFoundException;

import com.examples.capstone.AadharRestApi.Exception.CitizenNotFoundException;

import com.examples.capstone.AadharRestApi.Exception.UserNotFoundException;

@RestController

public class RestApiController {

@Autowired

private AdminRepo adminrepo;

@Autowired

private UserRepo userrepo;

@Autowired

private CitizensRepo citiRepo;

// Admin

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

@GetMapping("/admin")

public Iterable<Admin> getAdmin() {

return adminrepo.findAll();

}

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

@GetMapping("/admin/{id}")

public Admin getAdmin(@PathVariable("id") Integer id) {

Optional<Admin> optAdmin = adminrepo.findById(id);

if (optAdmin.isEmpty()) {

throw new AdminNotFoundException(id);

}

return optAdmin.get();

}

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

@PostMapping("/admin")

public Admin create(@RequestBody Admin admin) {

return adminrepo.save(admin);

}

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

@PutMapping("/admin")

public Admin update(@RequestBody Admin admin) {

return adminrepo.save(admin);

}

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

@DeleteMapping("/admin/{id}")

public void deleteadmin(@PathVariable("id") Integer id) {

adminrepo.deleteById(id);

}

// User

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

@GetMapping("/user")

public Iterable<User> getUser() {

return userrepo.findAll();

}

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

@GetMapping("/user/{id}")

public User getUser(@PathVariable("id") Integer id) {

Optional<User> optUser = userrepo.findById(id);

if (optUser.isEmpty()) {

throw new UserNotFoundException(id);

}

return optUser.get();

}

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

@PostMapping("/user")

public User create(@RequestBody User user) {

return userrepo.save(user);

}

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

@PutMapping("/user")

public User update(@RequestBody User user) {

return userrepo.save(user);

}

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

@DeleteMapping("/user/{id}")

public void deleteuser(@PathVariable("id") Integer id) {

userrepo.deleteById(id);

}

// Citizens

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

@GetMapping("/citizens")

public Iterable<Citizens> getCitizens() {

return citiRepo.findAll();

}

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

@GetMapping("/citizens/{id}")

public Citizens getCiti(@PathVariable("id") Integer id) {

Optional<Citizens> optCiti = citiRepo.findById(id);

if (optCiti.isEmpty()) {

throw new CitizenNotFoundException(id);

}

return optCiti.get();

}

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

@PostMapping("/citizens")

public Citizens create(@RequestBody Citizens citi) {

return citiRepo.save(citi);

}

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

@PutMapping("/citizens")

public Citizens update(@RequestBody Citizens citi) {

return citiRepo.save(citi);

}

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

@DeleteMapping("/citizens/{id}")

public void deleteciti(@PathVariable("id") Integer id) {

citiRepo.deleteById(id);

}

}

Entity:

Admin.java

package com.example.capstone.AadharRestApi.Entity;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

@Entity

public class Admin {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id;

private String admin;

private String password;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getAdmin() {

return admin;

}

public void setAdmin(String admin) {

this.admin = admin;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

@Override

public String toString() {

return "Admin [id=" + id + ", admin=" + admin + ", password=" + password + "]";

}

}

Citizens.java

package com.example.capstone.AadharRestApi.Entity;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

@Entity

public class Citizens {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int citizenId;

private String name;

private String dob;

private String emailid;

private String mobileno;

private String gender;

private String address;

public int getCitizenId() {

return citizenId;

}

public void setCitizenId(int citizenId) {

this.citizenId = citizenId;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getDob() {

return dob;

}

public void setDob(String dob) {

this.dob = dob;

}

public String getEmailid() {

return emailid;

}

public void setEmailid(String emailid) {

this.emailid = emailid;

}

public String getMobileno() {

return mobileno;

}

public void setMobileno(String mobileno) {

this.mobileno = mobileno;

}

public String getGender() {

return gender;

}

public void setGender(String gender) {

this.gender = gender;

}

public String getAddress() {

return address;

}

public void setAddress(String address) {

this.address = address;

}

@Override

public String toString() {

return "Citizens [citizenId=" + citizenId + ", name=" + name + ", dob=" + dob + ", emailid=" + emailid

+ ",mobileno=" + mobileno + ", gender=" + gender + ", address=" + address

+ "]";

}

}

User.java

package com.example.capstone.AadharRestApi.Entity;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

@Entity

public class User {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id;

private String emailId;

private String password;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getEmail() {

return emailId;

}

public void setEmail(String email) {

this.emailId = email;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

@Override

public String toString() {

return "User [id=" + id + ", email=" + emailId + ", password=" + password + "]";

}

}

Repository:

AdminRepo.java

package com.example.capstone.AadharRestApi.Repository;

import org.springframework.data.repository.CrudRepository;

import com.example.capstone.AadharRestApi.Entity.Admin;

public interface AdminRepo extends CrudRepository<Admin, Integer>{

}

CitizensRepo.java

package com.example.capstone.AadharRestApi.Repository;

import org.springframework.data.repository.CrudRepository;

import com.example.capstone.AadharRestApi.Entity.Citizens;

public interface CitizensRepo extends CrudRepository<Citizens, Integer>{

}

UserRepo.java

package com.example.capstone.AadharRestApi.Repository;

import org.springframework.data.repository.CrudRepository;

import com.example.capstone.AadharRestApi.Entity.User;

public interface UserRepo extends CrudRepository<User, Integer>{

}

AadharBackendApplication.java

package com.example.capstone;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class AadharBackendApplication {

public static void main(String[] args) {

SpringApplication.run(AadharBackendApplication.class, args);

}

}

**EXCEPTION:**

AdminNotFoundException.java

package com.examples.capstone.AadharRestApi.Exception;

public class AdminNotFoundException extends RuntimeException{

public AdminNotFoundException(int id) {

super("Admin with id " + id + " is not Found.Pls Give another id!!");

}

}

CitizenNotFoundException.java

package com.examples.capstone.AadharRestApi.Exception;

public class CitizenNotFoundException extends RuntimeException{

public CitizenNotFoundException(int id) {

super("Citizen with id " + id + " is not Found.Pls Give another id!!");

}

}

UserNotFoundException.java

package com.examples.capstone.AadharRestApi.Exception;

public class UserNotFoundException extends RuntimeException{

public UserNotFoundException(int id) {

super("User with id " + id + " is not Found.Pls Give another id!!");

}

}

RESOURCES

Application.properties

server.port=8080

spring.datasource.url=jdbc:mysql://localhost/capstone

spring.datasource.username=root

spring.datasource.password=12345

spring.jpa.properties.hibernate.show\_sql=true

spring.jpa.properties.hibernate.format\_sql=true

spring.jpa.hibernate.ddl-auto=update

logging.level.org.hibernate=INFO

logging.level.org.hibernate.SQL=INFO

logging.level.org.hibernate.type.descriptor.sql.BasicBinder=TRACE

logging.level.org.springframework=INFO

logging.level.org.apache=INFO

**Docker file:**

FROM openjdk:17-slim

COPY target/AadharBackend-0.0.1-SNAPSHOT.jar app.jar

EXPOSE 8080

ENTRYPOINT ["java", "-jar", "app.jar"]

**Pom.xml**

<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>2.7.4</version>

<relativePath/> <!-- lookup parent from repository -->

</parent>

<groupId>com.example</groupId>

<artifactId>AadharBackend</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>AadharBackend</name>

<description>Demo project for Spring Boot</description>

<properties>

<java.version>17</java.version>

</properties>

<dependencies>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-jpa</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-devtools</artifactId>

<scope>runtime</scope>

<optional>true</optional>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

</project>

Aadhar Testng:

AadharAdmin.java

package com.capstone;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

public class AadharAdmin {

static WebDriver webdriver = null;

@Test

public void ChromeBrowser() {

System.setProperty("webdriver.chrome.driver",

"C:\\Users\\Rajini\\Documents\\chromedriver\_win32\\chromedriver.exe");

webdriver = new ChromeDriver();

webdriver.manage().window().maximize();

System.out.println("Chrome Browser is Opened!!");

}

@Test

public void TestHome() {

webdriver.get("http://localhost:4200/home");

try {

Thread.sleep(3000);

} catch (InterruptedException e) {

e.printStackTrace();

}

webdriver.findElement(By.xpath("/html[1]/body[1]/app-root[1]/app-home[1]/div[1]/div[1]/a[3]")).click();

try {

Thread.sleep(3000);

} catch (InterruptedException e) {

e.printStackTrace();

}

webdriver.get("http://localhost:4200/adminlogin");

try {

Thread.sleep(3000);

} catch (InterruptedException e) {

e.printStackTrace();

}

webdriver.findElement(By.id("admin")).sendKeys("admin2");

webdriver.findElement(By.id("password")).sendKeys("Admin002@");

webdriver.findElement(By.xpath("//button[normalize-space()='Login']")).click();

try {

Thread.sleep(3000);

} catch (InterruptedException e) {

e.printStackTrace();

}

}

}

AadharCitizen.java

package com.capstone;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

public class AadharCitizen {

static WebDriver webdriver = null;

@Test

public void ChromeBrowser() {

System.setProperty("webdriver.chrome.driver",

"C:\\Users\\Rajini\\Documents\\chromedriver\_win32\\chromedriver.exe");

webdriver = new ChromeDriver();

webdriver.manage().window().maximize();

System.out.println("Chrome Browser is Opened!!");

}

@Test

public void CitizenCheck() {

webdriver.get("http://localhost:4200/home");

try {

Thread.sleep(3000);

} catch (InterruptedException e) {

e.printStackTrace();

}

webdriver.findElement(By.xpath("/html[1]/body[1]/app-root[1]/app-home[1]/div[1]/div[1]/a[1]")).click();

try {

Thread.sleep(3000);

} catch (InterruptedException e) {

e.printStackTrace();

}

webdriver.get("http://localhost:4200/userlogin");

try {

Thread.sleep(3000);

} catch (InterruptedException e) {

e.printStackTrace();

}

webdriver.findElement(By.id("citizen\_id")).sendKeys("1");

webdriver.findElement(By.id("password")).sendKeys("7976694711");

webdriver.findElement(By.xpath("//button[normalize-space()='Login']")).click();

}

}

Aadharregister.java

package com.capstone;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

public class AadharRegister {

static WebDriver webdriver = null;

@Test

public void ChromeBrowser() {

System.setProperty("webdriver.chrome.driver",

"C:\\Users\\Rajini\\Documents\\chromedriver\_win32\\chromedriver.exe");

webdriver = new ChromeDriver();

webdriver.manage().window().maximize();

System.out.println("Chrome Browser is Opened!!");

}

@Test

public void TestHome() {

webdriver.get("http://localhost:4200/home");

try {

Thread.sleep(3000);

} catch (InterruptedException e) {

e.printStackTrace();

}

webdriver.findElement(By.xpath("/html[1]/body[1]/app-root[1]/app-home[1]/div[1]/div[1]/a[2]")).click();

webdriver.get("http://localhost:4200/register");

try {

Thread.sleep(3000);

} catch (InterruptedException e) {

e.printStackTrace();

}

webdriver.findElement(By.id("name")).sendKeys("Kalai");

webdriver.findElement(By.id("dob")).sendKeys("2012-12-12");

webdriver.findElement(By.id("emailid")).sendKeys("Kalai12@gmail.com");

webdriver.findElement(By.id("address")).sendKeys("South Street,Chennai");

webdriver.findElement(By.id("mobileno")).sendKeys("9090909090");

webdriver.findElement(By.id("gender")).sendKeys("female");

}

}

Pom.xml

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>org.mycompany</groupId>

<artifactId>AadharTesting</artifactId>

<version>0.0.1-SNAPSHOT</version>

<build>

<sourceDirectory>src</sourceDirectory>

<plugins>

<plugin>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.8.1</version>

<configuration>

<release>15</release>

</configuration>

</plugin>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-surefire-plugin</artifactId>

<version>3.0.0-M5</version>

<configuration>

<suiteXmlFiles>

<suiteXmlFile>testng.xml</suiteXmlFile>

</suiteXmlFiles>

</configuration>

</plugin>

</plugins>

</build>

<dependencies>

<dependency>

<groupId>org.seleniumhq.selenium</groupId>

<artifactId>selenium-java</artifactId>

<version>4.1.2</version>

</dependency>

<dependency>

<groupId>org.testng</groupId>

<artifactId>testng</artifactId>

<version>7.5</version>

</dependency>

</dependencies>

</project>

testngadmin.xml

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">

<suite name="Suite">

<test thread-count="5" name="Test">

<classes>

<class name="com.capstone.AadharAdmin"/>

</classes>

</test> <!-- Test -->

</suite> <!-- Suite -->

testngcitizen.xml

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">

<suite name="Suite">

<test thread-count="5" name="Test">

<classes>

<class name="com.capstone.AadharCitizen"/>

</classes>

</test> <!-- Test -->

</suite> <!-- Suite -->

testngreg.xml

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">

<suite name="Suite">

<test thread-count="5" name="Test">

<classes>

<class name="com.capstone.AadharRegister"/>

</classes>

</test> <!-- Test -->

</suite> <!-- Suite -->