Source code for mAadhar Application

Frontend: Angular, Bootstrap, and HTML/CSS

Admin-home Component

```
<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> {{ user.citizenId }}</p>
              <p><b>Name :</b> {{ user.name }}</p>
              <p><b>Date Of Birth:</b> {{ user.dob }}</p>
              <p><b>Email:</b> {{ user.emailid }}</p>

              <p><b>Address:</b> {{ user.address }}</p>
              <p><b>Mobile No:</b> {{ user.mobileno }}</p>

              <p><b>Gender:</b> {{ user.gender }}</p>
              <p><b>Status:</b> {{ user.status }}</p>

              <button
                class="btn btn-sm btn-danger"
                (click)="onDeleteUser(user.citizenId)"
              >
                Delete
              </button>

              <!-- Add the "Approve" button with an *ngIf condition -->
              <button
```

```
                class="btn btn-sm btn-success"
                *ngIf="user.status === 'pending'"
                (click)="onApprove(user)"
              >
                Approve
              </button>
            </li>
          </ul>
        </div>
      </div>
    </div>
  </div>
```

```
import { Component, OnInit } from '@angular/core';
import { UserData } from '../user-home/userdata.module';
import { HttpClient } from '@angular/common/http';
import { FormBuilder } from '@angular/forms';
import { ApiService } from '../service/api.service';

@Component({
  selector: 'app-admin-home',
  templateUrl: './admin-home.component.html',
  styleUrls: ['./admin-home.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<UserData[]>(this.backendurl)
      .subscribe((users) => {
        // Assuming your API returns the "status" field for each user
        this.fetchedUsers = users.map((user) => {
```

```
          user.status = 'pending'; // Set the status field to the appropriate
value from your API response
          return user;
        });
      });
  }

  onDeleteUser(id: number) {
    this.http.delete(this.backendurl + '/' + id).subscribe((response) => {
      console.log('User deleted: ' + response);
      // this.fetchPosts();
    });


  }
  onApprove(user: UserData) {
    // Modify the status to "approved"
    user.status = 'approved';
    // Call the API to update the status in the backend
    this.api.UpdateUser(user, user.citizenId).subscribe(
      (res) => {
        alert('User status changed to "Approved" successfully!!');
        this.fetchUsers(); // Refresh the user list after the status change
      },
      (err) => {
        alert('Something Went Wrong:/');
      }
    );
  }


}
```

Adminlogin component

```
<div class="login-container">
    <h1>Admin Login</h1>
    <form (ngSubmit)="onSubmit()">
      <div class="form-group">
        <label for="username">Username</label>
        <input type="text" id="username" [(ngModel)]="username" name="username"
required>
      </div>
```

```
      <div class="form-group">
        <label for="password">Password</label>
        <input type="password" id="password" [(ngModel)]="password"
name="password" required>
      </div>
      <button type="submit">Login</button>
    </form>
  </div>
```

```
import { Component } from '@angular/core';
import { Router } from '@angular/router'; // Import the Router class

@Component({
  selector: 'app-adminlogin',
  templateUrl: './adminlogin.component.html',
  styleUrls: ['./adminlogin.component.css']
})
export class AdminloginComponent {
  username: string = 'admin';
  password: string = 'Admin@123';

  constructor(private router: Router) {} // Inject the Router service in the
constructor

  onSubmit() {
    if (!this.username || !this.password) {
      console.log('Invalid credentials. Please try again.');
      return;
    }

    // Validate Admin password
    if (this.isAdminPasswordValid(this.password)) {
      console.log('Admin Login successful!');
      this.router.navigateByUrl('/admin-home');
    } else {
      console.log('Invalid Admin credentials. Please try again.');
    }
  }

  isAdminPasswordValid(password: string): boolean {
    // Admin password should have at least one uppercase, one lowercase, one
special character (@,#,&…), and one number
```

```
    const passwordRegex = /^(?=.*[a-z])(?=.*[A-Z])(?=.*\d)(?=.*[@#&])[A-Za-
z\d@#&]{8,}$/;
    return passwordRegex.test(password);
  }
}
```

Login component

```
<div class="login-container">
    <h1>Welcome to mAadhar Application</h1>
    <nav>
      <a routerLink="/user-login" routerLinkActive="active">User Login</a>
      <a routerLink="/adminlogin" routerLinkActive="active">Admin Login</a>
    </nav>
    <p style="margin-top: 10px; font-weight: bold;" >New User? <a
routerLink="/user-register" style="color: rgb(235, 13, 21); text-decoration:
underline;">Register Here</a></p>
  </div>

  <router-outlet></router-outlet>
```

```
import { Component } from '@angular/core';

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

}
```

Service component

```
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:8085/citizens',data)
    .pipe(map((res:any)=>{

      return res;
    }))
  }
}
```

User-home component

```
<!-- user-home.component.html -->
<div class="login-container">
    <h1>Welcome to mAadhar Application</h1>
  </div>
<div class="image">
    <div class="container">
      <!-- ... (existing code) -->
      <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> {{ user.citizenId }}</p>
            <p><b>Name :</b> {{ user.name }}</p>
            <p><b>Date Of Birth:</b> {{ user.dob }}</p>
            <p><b>Email:</b> {{ user.emailid }}</p>
            <p><b>Address:</b> {{ user.address }}</p>
            <p><b>Mobile No:</b> {{ user.mobileno }}</p>
            <p><b>Gender:</b> {{ user.gender }}</p>
            <p><b>Status:</b> {{ user.status }}</p> <!-- Add the status
column -->
            <button
            type="button"
            class="btn btn-outline-primary"
            data-bs-toggle="modal"
            data-bs-target="#exampleModal"
            (click)="onEdit(user)"
          >
            Update
```

```
          </button>
          <a
            type="button"
            class="btn btn-outline-primary align-right"
            routerLink="/login"
          >
            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>
```

```
import { HttpClient } from '@angular/common/http';
import { Component, OnInit } from '@angular/core';
```

```
import { FormBuilder, FormGroup } from '@angular/forms';
import { ApiService } from '../service/api.service';
import { UserData } from './userdata.module';
import { UserModel } from './user.module';

@Component({
  selector: 'app-user-home',
  templateUrl: './user-home.component.html',
  styleUrls: ['./user-home.component.css']
})
export class UserHomeComponent implements OnInit {
  userModelObj: UserModel = new UserModel();
  public loginForm!: FormGroup;
  fetchedUsers: UserData[] = [];
  backendurl = 'http://localhost:8080/citizens';

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

  ngOnInit(): void {
    this.loginForm = this.formBuilder.group({
      name: [''],
      dob: [''],
      emailid: [''],
      mobileno: [''],
      address: [''],
      gender: [''],
    });
    this.fetchUsers();
  }


  fetchUsers() {
    this.http
      .get<UserData[]>(this.backendurl) // Assuming UserData[] is the response
type
      .subscribe((users) => {
        // Add a "status" property to each user
        this.fetchedUsers = users.map((user) => {
          user.status = 'pending'; // Replace 'Active' with the actual status
field from your API response
          return user;
```

```
        });
      });
  }

  onEdit(user: UserData) {
    this.userModelObj.citizenId = user.citizenId;
    this.loginForm.patchValue({
      name: user.name,
      dob: user.dob,
      emailid: user.emailid,
      mobileno: user.mobileno,
      address: user.address,
      gender: user.gender,
    });
  }
  updateUser() {
    this.userModelObj.name = this.loginForm.value.name;
    this.userModelObj.emailid = this.loginForm.value.emailid;
    this.userModelObj.gender = this.loginForm.value.gender;

    this.userModelObj.address = this.loginForm.value.address;
    this.userModelObj.mobileno = this.loginForm.value.mobileno;
    this.userModelObj.dob = this.loginForm.value.dob;

    this.api.UpdateUser(this.userModelObj,
this.userModelObj.citizenId).subscribe(
      (res) => {
        alert('Updated Successfully!!');
        this.loginForm.reset();
        this.fetchUsers();
      },
      (err) => {
        alert('Something Went Wrong:/');
      }
    );
  }
}
```

User-login component

```
<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>
```

```
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-login',
  templateUrl: './user-login.component.html',
  styleUrls: ['./user-login.component.css'],
})
export class UserLoginComponent 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(['user-home']);

      }else{
        alert("Try Again!!");
      }
    },err=>{
        alert("Something Went Wrong!!");
        this.loginForm.reset();
    })


  }


}
```

User-registration component

```
<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>
```

```
import { HttpClient } from '@angular/common/http';
import { Component, OnInit } from '@angular/core';
import { FormBuilder, FormGroup, NgForm, Validators } from '@angular/forms';
import { Router } from '@angular/router';

@Component({
  selector: 'app-user-registration',
  templateUrl: './user-registration.component.html',
  styleUrls: ['./user-registration.component.css']
})
```

```
export class UserRegistrationComponent 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:stri
ng }, form: NgForm) {
    this.http.post(this.url, citizenData).subscribe((responseData) => {
      console.log(responseData);
      alert("Registered Successfully!!");
      form.reset();
    },err=>{
      alert("Something Happened!!")
    });
  }

}
```

App component

```
import { Component } from '@angular/core';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title = 'mAadhaar-Frontend';
}
```

Backend: Java Programming (Spring Boot, JPA, Hibernate)

MAadharBackendApplication.java

```
package com.aadhar;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class MAadharBackendApplication {

	public static void main(String[] args) {
		SpringApplication.run(MAadharBackendApplication.class, args);
	}

}
```

WebConfig.java

```
package com.aadhar;

import org.springframework.context.annotation.Configuration;
import org.springframework.web.servlet.config.annotation.CorsRegistry;
import org.springframework.web.servlet.config.annotation.EnableWebMvc;
import org.springframework.web.servlet.config.annotation.WebMvcConfigurer;

@Configuration
@EnableWebMvc
public class WebConfig implements WebMvcConfigurer {
```

```
	@Override
	public void addCorsMappings(CorsRegistry registry) {
		registry.addMapping("/**").allowedOrigins("http://localhost:4200") // Replace this with the URL of your Angular
                                                                                    // frontend
                                        .allowedMethods("GET", "POST", "PUT", "DELETE").allowedHeaders("*").allowCredentials(true);
	}
}
```

AadharApiException.java

```
package com.aadhar.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.aadhar.exception.CitizenNotFoundException;
import com.aadhar.exception.UserNotFoundException;

@ControllerAdvice
public class AadharApiException {

	@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.aadhar.controller;

import java.util.Optional;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.repository.CrudRepository;
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.aadhar.entity.Admin;
import com.aadhar.entity.Citizens;
import com.aadhar.entity.User;
import com.aadhar.exception.AdminNotFoundException;
import com.aadhar.exception.CitizenNotFoundException;
import com.aadhar.exception.UserNotFoundException;
import com.aadhar.repository.CitizensRepo;
import com.aadhar.repository.UserRepo;

@RestController
public class RestApiController {

	@Autowired
	private CrudRepository<Admin, Integer> adminrepo;

	@Autowired
	private UserRepo userrepo;

	@Autowired
	private CitizensRepo citiRepo;

	@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);
}
```

```
}
```

Admin.java

```
package com.aadhar.entity;


import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.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.aadhar.entity;


import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.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.aadhar.entity;


import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.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 + "]";
	}

}
```

AdminNotFoundException.java

```
package com.aadhar.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.aadhar.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.aadhar.exception;

public class UserNotFoundException extends RuntimeException {

	public UserNotFoundException(int id) {
		super("User with id " + id + " is not Found.Pls Give another
id!!");
```

```
	}
}
```

AdminRepo.java

```
package com.aadhar.repository;

import org.springframework.data.repository.CrudRepository;

import com.aadhar.entity.Admin;

public interface AdminRepo extends CrudRepository<Admin, Integer> {

}
```

CitizensRepo.java

```
package com.aadhar.repository;

import org.springframework.data.repository.CrudRepository;

import com.aadhar.entity.Citizens;

public interface CitizensRepo extends CrudRepository<Citizens, Integer> {

}
```

UserRepo.java

```
package com.aadhar.repository;

import org.springframework.data.repository.CrudRepository;

import com.aadhar.entity.User;

public interface UserRepo extends CrudRepository<User, Integer> {

}
```

Application.properties

```
spring.datasource.url=jdbc:mysql://localhost:3306/capstone_maadhar?useSSL=fal
se&serverTimezone=UTC
spring.datasource.username=root
spring.datasource.password=Rahul@2001

spring.jpa.show-sql=true
spring.jpa.hibernate.ddl-auto=update
```

```
spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQLDialect

logging.level.org.hibernate.SQL=DEBUG
```

Database

MySQL Commands

```
CREATE DATABASE capstone_maadhar;

USE capstone_maadhar;
```

Testing

AadharAdmin.java

```
package com.aadhar.test;


import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import org.testng.annotations.Test;


public class AadharAdmin {

	static WebDriver webdriver = null;


	@Test

	public void ChromeBrowser() {

		webdriver = new ChromeDriver();

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

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


	}
```

```
@Test
public void TestHome() {
	webdriver.get("http://localhost:4200/login");
	try {
		Thread.sleep(3000);
	} catch (InterruptedException e) {
		e.printStackTrace();
	}
	webdriver.findElement(By.xpath("//div[@class='login-container']")).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("username")).sendKeys("admin");
	webdriver.findElement(By.id("password")).sendKeys("Admin@123");
	webdriver.findElement(By.xpath("//button[normalize-space()='Login']")).click();
	try {
		Thread.sleep(3000);
	} catch (InterruptedException e) {
```

```
                e.printStackTrace();
            }

        }

}
```

AadharCitizen.java

```
package com.aadhar.test;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.testng.annotations.Test;

public class AadharCitizen {
        static WebDriver webdriver = null;

        @Test
        public void ChromeBrowser() {
                webdriver = new ChromeDriver();
                webdriver.manage().window().maximize();
                System.out.println("Chrome Browser is Opened!!");

        }

        @Test
```

```
public void CitizenCheck() {
	webdriver.get("http://localhost:4200/login");
	try {
		Thread.sleep(3000);
	} catch (InterruptedException e) {
		e.printStackTrace();
	}
	webdriver.findElement(By.xpath("//div[@class='login-container']")).click();
	try {
		Thread.sleep(3000);
	} catch (InterruptedException e) {
		e.printStackTrace();
	}

	webdriver.get("http://localhost:4200/user-login");
	try {
		Thread.sleep(3000);
	} catch (InterruptedException e) {
		e.printStackTrace();
	}
	webdriver.findElement(By.id("citizen_id")).sendKeys("1");
	webdriver.findElement(By.id("password")).sendKeys("9629721162");
	webdriver.findElement(By.xpath("//button[normalize-space()='Login']")).click();

}
```

```
}
```

AadharRegister.java

```
package com.aadhar.test;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.testng.annotations.Test;

public class AadharRegister {
	static WebDriver webdriver = null;

	@Test
	public void ChromeBrowser() {
		webdriver = new ChromeDriver();
		webdriver.manage().window().maximize();
		System.out.println("Chrome Browser is Opened!!");

	}

	@Test
	public void TestHome() {
		webdriver.get("http://localhost:4200/login");
		try {
			Thread.sleep(3000);
		} catch (InterruptedException e) {
```

```
            e.printStackTrace();
        }

        webdriver.findElement(By.xpath("//div[@class='login-container']")).click();
        try {
            Thread.sleep(3000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        webdriver.get("http://localhost:4200/user-register");

        webdriver.findElement(By.id("name")).sendKeys("Rahul");
        webdriver.findElement(By.id("dob")).sendKeys("11/10/2001");
        webdriver.findElement(By.id("emailid")).sendKeys("rahul@gmail.com");
        webdriver.findElement(By.id("address")).sendKeys("Chennai");
        webdriver.findElement(By.id("mobileno")).sendKeys("9876543210");
        webdriver.findElement(By.id("gender")).sendKeys("Male");
    }

}
```