Complaint Redressal System
Project Source Code

Version History:

Author:	Shanmugam D
Purpose:	Project Source Code and GITHUB
Date:	27th March 2022.
Version:	1.0

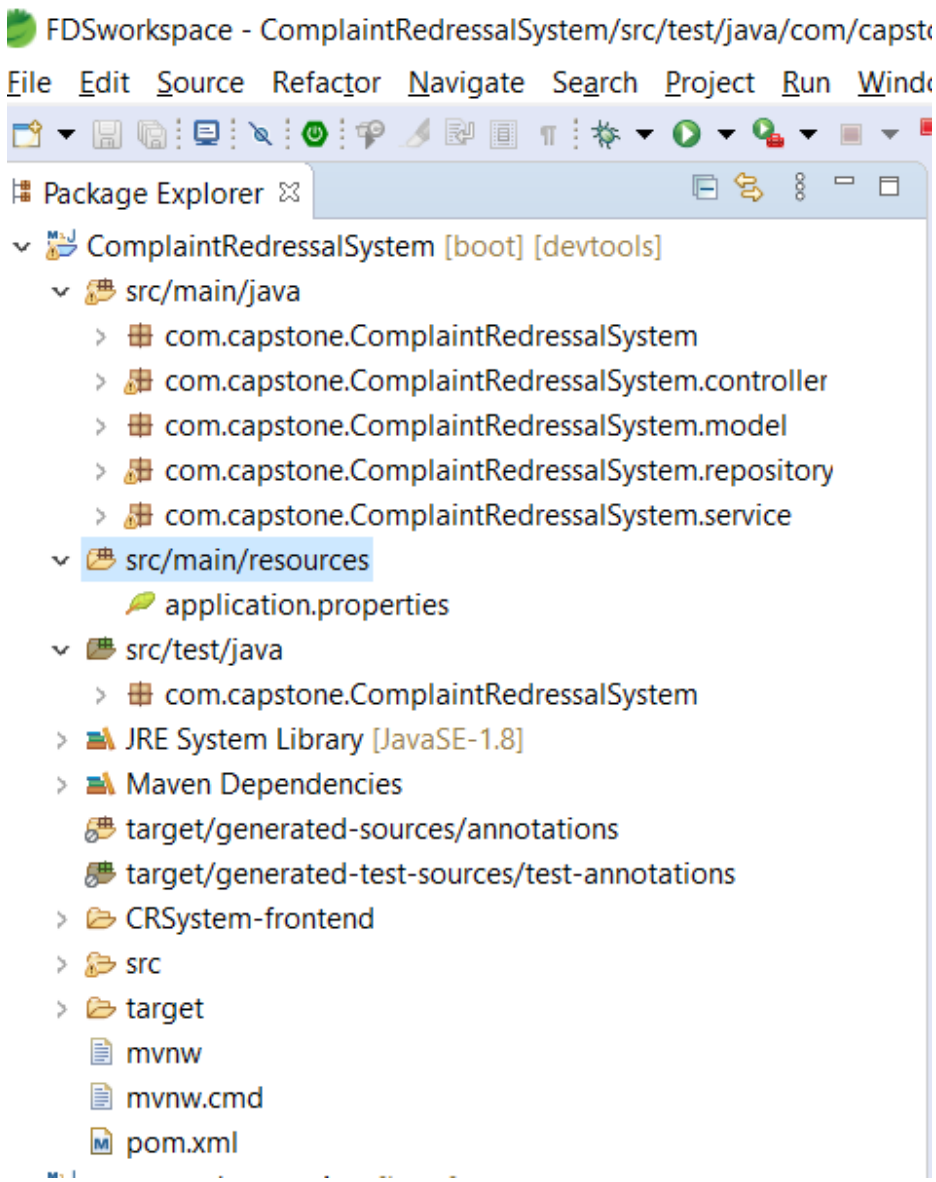
Table of Content:

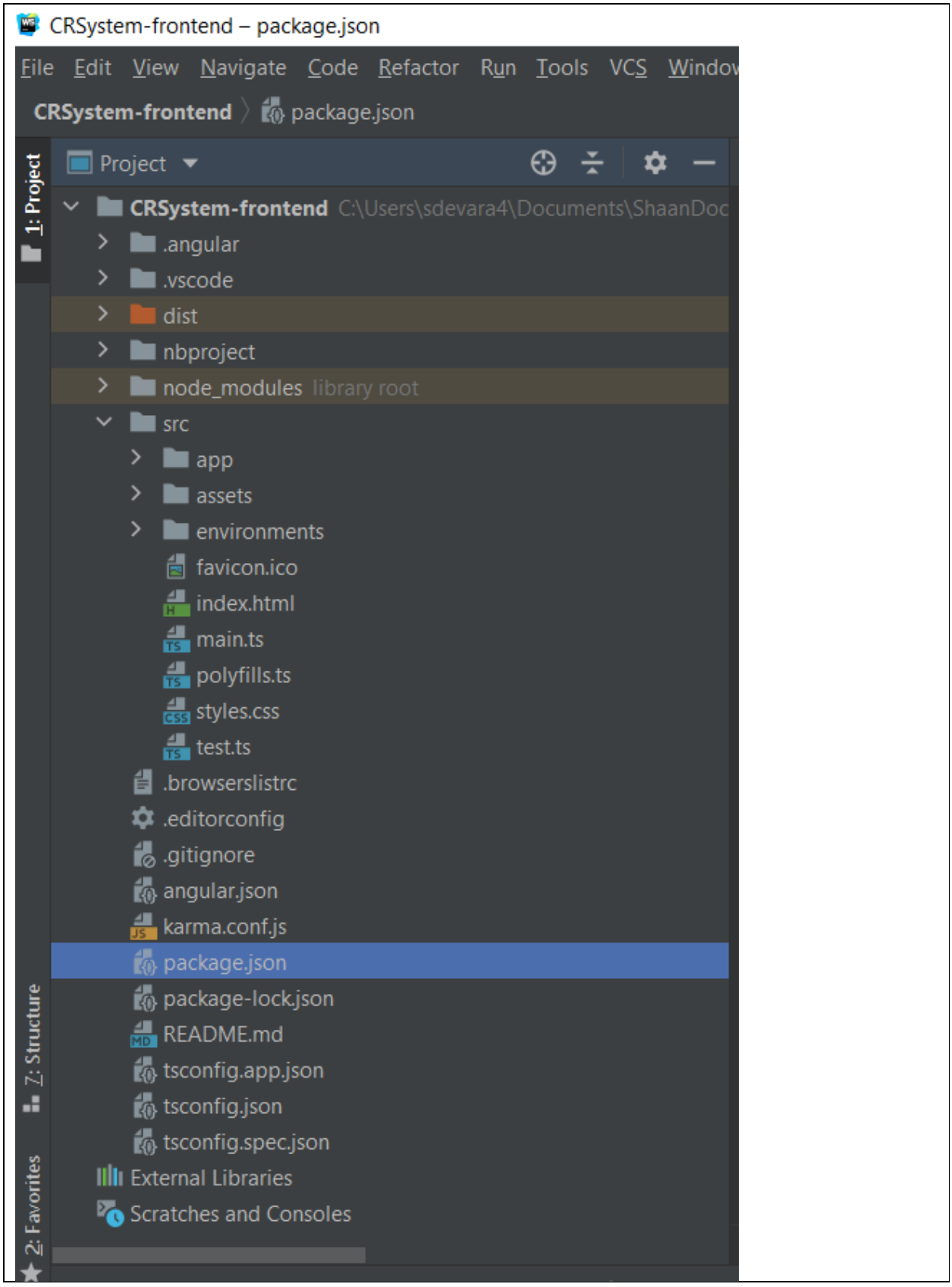
Project GITHUB Link:

Repository Name:	CapStone-ComplaintRedressalSystem
Github Link:	https://github.com/dshaan05/CapStone-ComplaintRedressalSystem

Project Code:

Folder Structure




CRSystem-frontend – package.json
File Edit View Navigate Code Refactor Run Tools VCS Windov
CRSystem-frontend package.json
Project
CRSystem-frontend C:\Users\sdevara4\Documents\ShaanDoc
.angular
.vscode
dist
nbproject
node_modules library root
src
app
assets
environments
favicon.ico
index.html
main.ts
polyfills.ts
styles.css
test.ts
.browserslistrc
.editorconfig
.gitignore
angular.json
karma.conf.js
package.json
package-lock.json
README.md
tsconfig.app.json
tsconfig.json
tsconfig.spec.json
External Libraries
Scratches and Consoles
1: Project
7: Structure
2: Favorites

AdminController.java

```java
package com.capstone.ComplaintRedressalSystem.controller;

import java.util.List;
import java.util.Optional;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.validation.annotation.Validated;
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.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.capstone.ComplaintRedressalSystem.model.Admin;
import com.capstone.ComplaintRedressalSystem.repository.AdminRepository;
import com.capstone.ComplaintRedressalSystem.service.AdminService;

@CrossOrigin(origins = "http://localhost:4200")
@RestController
@RequestMapping("/api")
public class AdminController {

	@Autowired
	private AdminService adminService;
	private AdminRepository adminRepo;

	@GetMapping("/get-admin")
	public ResponseEntity<List<Admin>> getAdminDetails()
	{
		List<Admin> admin = adminService.findAll();
		return new ResponseEntity<List<Admin>>(admin, HttpStatus.OK);
	}


	@GetMapping("/get-adminlist")
	public ResponseEntity<List<Admin>> findAllBySoftDelete()
	{
		List<Admin> admin = adminRepo.findAllBySoftDelete("0");
		return new ResponseEntity<List<Admin>>(admin, HttpStatus.OK);
	}



	@GetMapping("/get-admin/{id}")
	public ResponseEntity<Admin> getAdminById(@PathVariable("id") Long id)
	{
```

```
		Admin adminDetails = adminService.findById(id);
		return new ResponseEntity<Admin>(adminDetails, HttpStatus.OK);
	}

	@DeleteMapping("/delete-admin/{id}")
	public ResponseEntity<String> deleteAdmin(@PathVariable("id") Long id)
	{
		//adminService.deleteAdminById(id);
		adminService.deleteAdminById(id);
		return new ResponseEntity<String>("Admin is deleted Successfully",
HttpStatus.OK);
	}

	@PostMapping("/save-admin")
	public ResponseEntity<Admin> saveAdmin(@Validated @RequestBody Admin admin)
	{
		Admin adminSave = adminService.saveAdmin(admin);
		return new ResponseEntity<Admin>(adminSave, HttpStatus.OK);
	}


}
```

Admin.java

```
package com.capstone.ComplaintRedressalSystem.model;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table(name = "admin")
public class Admin
{
	@Id
	@GeneratedValue(strategy = GenerationType.IDENTITY)
	@Column(name = "aid")
	private long id;

	@Column(name = "user_name")
	private String username;

	public String getUsername() {
		return username;
	}

	public void setUsername(String username) {
		this.username = username;
	}

	@Column(name = "email_id")
```

```
	private String emailId;

	@Column(name = "password")
	private String password;

	@Column(name = "soft_delete")
	private int softDelete;

	public Admin() {
		super();
	}

	public Admin(long id, String username, String emailId, String password, int
softDelete) {
		super();
		this.id = id;
		this.username = username;
		this.emailId = emailId;
		this.password = password;
		this.softDelete = softDelete;
	}

	public long getId() {
		return id;
	}

	public void setId(long id) {
		this.id = id;
	}

	public String getEmailId() {
		return emailId;
	}

	public void setEmailId(String emailId) {
		this.emailId = emailId;
	}

	public String getPassword() {
		return password;
	}

	public void setPassword(String password) {
		this.password = password;
	}

	public int getSoftDelete() {
		return softDelete;
	}

	public void setSoftDelete(int softDelete) {
		this.softDelete = softDelete;
	}

	@Override
```

```
	public String toString() {
		return "Admin [id=" + id + ", Username=" + username + ", emailId=" +
emailId + ", password=" + password
				+ ", softDelete=" + softDelete + "]";
	}

}
```

AdminRepository.java

```
package com.capstone.ComplaintRedressalSystem.repository;

import java.util.List;

import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.query.Param;
import org.springframework.web.bind.annotation.CrossOrigin;
import org.springframework.transaction.annotation.Transactional;
import org.springframework.data.jpa.repository.Modifying;

import com.capstone.ComplaintRedressalSystem.model.Admin;

//@CrossOrigin(origins = "http://localhost:4200")
public interface AdminRepository extends JpaRepository<Admin, Long>{

	@Transactional
	@Modifying(clearAutomatically = true)
	@Query("SELECT a FROM Admin a where a.softDelete like %?1%")
	List<Admin> findAllBySoftDelete(String id);

}
```

AdminService.java

```
package com.capstone.ComplaintRedressalSystem.service;

import java.util.List;
import java.util.Optional;

import com.capstone.ComplaintRedressalSystem.model.Admin;

public interface AdminService {

	List<Admin> findAll();

	Admin findById(Long id);

	void deleteAdminById(Long id);

	Admin saveAdmin(Admin admin);

}
```

AdminServiceimpl.java

```java
package com.capstone.ComplaintRedressalSystem.service;

import java.util.List;
import java.util.Optional;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.jpa.repository.Query;
import org.springframework.http.ResponseEntity;
import org.springframework.stereotype.Service;

import com.capstone.ComplaintRedressalSystem.model.Admin;
import com.capstone.ComplaintRedressalSystem.repository.AdminRepository;

@Service
public class AdminServiceImpl implements AdminService{

	@Autowired
	private AdminRepository adminRepo;

	@Override
	public List<Admin> findAll()
	{
		return adminRepo.findAll();
	}

	@Override
	public Admin findById(Long id) {
		if(adminRepo.findById(id).isPresent()) {
			return adminRepo.findById(id).get();
		}
		return null;

	}

	@Override
	public void deleteAdminById(Long id) {
		Admin admin = findById(id);
		//adminRepo.deleteById(id);
		adminRepo.delete(admin);
	}

	@Override
	public Admin saveAdmin(Admin admin) {
		adminRepo.save(admin);
		return admin;
	}



}
```

application.properties

```
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.datasource.url=jdbc:mysql://localhost:3306/crsystem
spring.datasource.username=root
spring.datasource.password=Shaan@1015

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5Dialect
spring.jpa.show-sql = true
logging.level.org.springframework = info
spring.jpa.hibernate.ddl-auto=update
```

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>2.6.3</version>
		<relativePath/> <!-- lookup parent from repository -->
	</parent>
	<groupId>com.capstone</groupId>
	<artifactId>ComplaintRedressalSystem</artifactId>
	<version>0.0.1-SNAPSHOT</version>
	<name>ComplaintRedressalSystem</name>
	<description>Complaint Redressal System</description>
	<properties>
		<java.version>1.8</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-starter-data-rest</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.projectlombok</groupId>
                        <artifactId>lombok</artifactId>
                        <optional>true</optional>
                </dependency>
                <dependency>
                        <groupId>org.springframework.boot</groupId>
                        <artifactId>spring-boot-starter-test</artifactId>
                        <scope>test</scope>
                </dependency>
                <dependency>
                        <groupId>org.springframework.restdocs</groupId>
                        <artifactId>spring-restdocs-mockmvc</artifactId>
                        <scope>test</scope>
                </dependency>
        </dependencies>

        <build>
                <plugins>
                        <plugin>
                                <groupId>org.asciidoctor</groupId>
                                <artifactId>asciidoctor-maven-plugin</artifactId>
                                <version>1.5.8</version>
                                <executions>
                                        <execution>
                                                <id>generate-docs</id>
                                                <phase>prepare-package</phase>
                                                <goals>
                                                        <goal>process-asciidoc</goal>
                                                </goals>
                                                <configuration>
                                                        <backend>html</backend>
                                                        <doctype>book</doctype>
                                                </configuration>
                                        </execution>
                                </executions>
                                <dependencies>
                                        <dependency>

        <groupId>org.springframework.restdocs</groupId>
                                                <artifactId>spring-restdocs-
asciidoctor</artifactId>
                                                <version>${spring-restdocs.version}</version>
                                        </dependency>
                                </dependencies>
                        </plugin>
                        <plugin>
                                <groupId>org.springframework.boot</groupId>
                                <artifactId>spring-boot-maven-plugin</artifactId>
                                <configuration>
                                        <excludes>
                                                <exclude>
```

```
                                        <groupId>org.projectlombok</groupId>
                                        <artifactId>lombok</artifactId>
                                    </exclude>
                                </excludes>
                            </configuration>
                        </plugin>
                    </plugins>
                </build>

</project>
```

Agular UI Source Code:

Add-admin-component.html

```
<form (ngSubmit)= "saveAdmin()">

    <input type="text"
           name="username" [(ngModel)] = "admin.username"
           placeholder="Enter Username: "/>

    <input type="text"
           name="password" [(ngModel)] = "admin.password"
           placeholder="Enter Password: "/>

    <input type="text"
           name="emailId" [(ngModel)] = "admin.emailId"
           placeholder="Enter EmailId: "/>

    <button type="submit">Add Admin</button>
    <!-- <button *ngIf ="admin.id"  (click) ="deleteAdmin(admin.id)">Delete</button>
-->



</form>
```

Add-admin-component.spec.ts

```
import { ComponentFixture, TestBed } from '@angular/core/testing';

import { AddAdminComponent } from './add-admin.component';

describe('AddAdminComponent', () => {
  let component: AddAdminComponent;
  let fixture: ComponentFixture<AddAdminComponent>;

  beforeEach(async () => {
    await TestBed.configureTestingModule({
      declarations: [ AddAdminComponent ]
    })
    .compileComponents();
  });

  beforeEach(() => {
    fixture = TestBed.createComponent(AddAdminComponent);
```

```
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

  it('should create', () => {
    expect(component).toBeTruthy();
  });
});
```

Add-admin-component.ts

```
import { Component, OnInit } from '@angular/core';
import { ActivatedRoute, Router } from '@angular/router';
import { Admin } from 'src/app/models/admin';
import { AdminService } from 'src/app/services/admin.service';

@Component({
  selector: 'app-add-admin',
  templateUrl: './add-admin.component.html',
  styleUrls: ['./add-admin.component.css']
})
export class AddAdminComponent implements OnInit {

  admin: Admin = new Admin();
  constructor(private _adminService: AdminService,
              private _router: Router,
              private _activatedRoute: ActivatedRoute) { }

  ngOnInit(): void {
    const isIdPresent = this._activatedRoute.snapshot.paramMap.has('id');
    if(isIdPresent){
      const id = Number(this._activatedRoute.snapshot.paramMap.get('id'));
      this._adminService.getAdminbyId(id).subscribe(
        data => this.admin = data
      )
      // this._adminService.deleteAdmin(id).subscribe(
      //   data => this.admin = data
      // )

    }
  }

  saveAdmin() {
    this._adminService.saveAdmin(this.admin).subscribe(
      data => {
        console.log('response', data);
        this._router.navigateByUrl("/admin");
      }
    )

  }

  deleteAdmin(id: number) {
    this._adminService.deleteAdmin(id).subscribe(
      data => {
```

```
        console.log('deleted admin', data);
        this._router.navigateByUrl("/admin");
      }
    )
  }
}
```

Admin.service.spec.ts

```
import { TestBed } from '@angular/core/testing';

import { AdminService } from './admin.service';

describe('AdminService', () => {
  let service: AdminService;

  beforeEach(() => {
    TestBed.configureTestingModule({});
    service = TestBed.inject(AdminService);
  });

  it('should be created', () => {
    expect(service).toBeTruthy();
  });
});
```

Admin.service.ts

```
import { HttpClient } from '@angular/common/http';
import { Injectable } from '@angular/core';
import { map, Observable } from 'rxjs';
import { Admin } from '../models/admin';

@Injectable({
  providedIn: 'root'
})
export class AdminService {

  private getUrl: string = "http://localhost:4200/api/get-admin";
  private getSaveUrl: string = "http://localhost:4200/api/save-admin";
  private deleteUrl: String = "http://localhost:4200/api/delete-admin";

  constructor(private _httpClient: HttpClient) { }

  getAdmin() : Observable<Admin[]> {
    return this._httpClient.get<Admin[]>(this.getUrl).pipe(
      map(responce => responce)
    )
  }

  saveAdmin(admin: Admin): Observable<Admin> {
    return this._httpClient.post<Admin>(this.getSaveUrl, admin);
  }

  getAdminbyId(id: number): Observable<Admin> {
    return this._httpClient.get<Admin>(`${this.getUrl}/${id}`).pipe(
```

```
      map(responce => responce)
    )

  }

  deleteAdmin(id: number): Observable<any> {
    return this._httpClient.delete(`${this.deleteUrl}/${id}`, {responseType:
'text'});
  }
}
```

MYSQL DB Query:

```
CREATE DB crsystem;
CREATE TABLE admin (
   aid int NOT NULL AUTO_INCREMENT,
   user_name varchar(255),
   email_id varchar(255),
   password varchar(255),
   soft_delete int,
         PRIMARY KEY (aid)
);
```