MEDICARE

source code

Frontend stack :

app.component.html

```
<app-navbar></app-navbar>
<router-outlet></router-outlet>
<footer>
   <div class="fixed-bottom">
      <small class="bg-light">&#169; 2012-2023- Medicare. Designed & Developed by @amanpawar. </small>  </div>
</footer>
```

app.component.spec.ts

```
import { TestBed } from '@angular/core/testing'; import {
RouterTestingModule } from '@angular/router/testing'; import {
AppComponent } from './app.component';

describe('AppComponent', () => {
beforeEach(async () => {
   await TestBed.configureTestingModule({
imports: [
      RouterTestingModule
    ],
    declarations: [
      AppComponent
    ],
  }).compileComponents();
 });

 it('should create the app', () => {
   const fixture = TestBed.createComponent(AppComponent);
const app = fixture.componentInstance;
expect(app).toBeTruthy();
 });

 it(`should have as title 'Medicare'`, () => {
   const fixture = TestBed.createComponent(AppComponent);
const app = fixture.componentInstance;
expect(app.title).toEqual('Medicare');
 });

 it('should render title', () => {
   const fixture = TestBed.createComponent(AppComponent);
fixture.detectChanges();
   const compiled = fixture.nativeElement as HTMLElement;
   expect(compiled.querySelector('.content span')?.textContent).toContain('Medicare app is running!');
 });
});
```

Cart-item.ts

```
import { Product } from "./product";

export class CartItem {
pid!: number;
name!: string;
brand!: string;
price!: number;
img!: any;    quantity!:
number;
constructor(product:
```

```
Product) {        this.pid
= product.pid;
this.name =
product.name;
this.brand =
product.brand;
this.price =
product.price;
this.img =
product.img;
      this.quantity = 1;
   }
}
```

index.html

```
<!doctype html>
<html lang="en">

<head>
  <meta charset="utf-8">
<title>Medicare</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="stylesheet" href="https://cdn.jsdelivr.net/npm/botstrap@4.0/dist/css/botstrap.min.css"     integrity="sha384-
Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm" crossorigin="anonymous"> </head>

<body>

  <app-root></app-root>


  <script src="https://code.jquery.com/jquery-3.2.1.slm.min.js"
    integrity="sha384-KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
crossorigin="anonymous"></script>
  <script src="https://cdn.jsdelivr.net/npm/poper.js@1.12.9/dist/umd/popper.min.js"
    integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
crossorigin="anonymous"></script>
  <script src="https://cdn.jsdelivr.net/npm/botstrap@4.0/dist/js/botstrap.min.js"
    integrity="sha384-JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"
crossorigin="anonymous"></script>
  <script src="https://kit.foawesome.com/a076d05399.js" crossorigin="anonymous"></script>

</body>

</html>
```

Backend :

MedicareBackendApplication.java

```
package com.medicare;

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

@SpringBootApplication
public class MedicareBackendApplication {

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

}
```

Usercontroller.java

```
package com.medicare.controller;

import java.net.URI; import
java.util.HashSet;
import java.util.Set;

import javax.annotation.PostConstruct;
import javax.validation.Valid;

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.PostMapping; import
org.springframework.web.bind.annotation.RequestBody; import
org.springframework.web.bind.annotation.RestController;
import org.springframework.web.servlet.support.ServletUriComponentsBuilder;

import com.medicare.entities.Role; import
com.medicare.entities.User; import
com.medicare.entities.UserRole;
import com.medicare.services.UserService;

@RestController @CrossOrigin(origins
= "*")
public class UserController {

	@Autowired
	private UserService userService;

	//init admin user
@PostConstruct
	public void createAdmin(){
		User admin = new User();
admin.setUsername("admin@medicare.com");
admin.setPassword("admin12345");
admin.setFirstName("Aman");
admin.setLastName("Pawar");
admin.setContactNumber("7247559948");		Role role =
new Role();			role.setRoleId(101L);
role.setRoleName("ADMIN");			UserRole ur = new
UserRole();			ur.setUser(admin);
ur.setRole(role);
			Set<UserRole> userRole = new HashSet<>();
userRole.add(ur);
			User adminCreated = this.userService.createUser(admin, userRole);
			System.out.println("Admin username: "+adminCreated.getUsername());
	}

	//create new user
	@PostMapping("/user/signup")
	public ResponseEntity<?> createNewUser(@Valid @RequestBody User user){
		Role role = new Role();
role.setRoleId(102L);
role.setRoleName("USER");			UserRole
ur = new UserRole();
ur.setUser(user);			ur.setRole(role);
		Set<UserRole> userRole = new HashSet<>();
userRole.add(ur);
		if(this.userService.getByUsername(user.getUsername())!=null) {
System.out.println("Username already exists!");
			return ResponseEntity.status(HttpStatus.INTERNAL_SERVER_ERROR).build();
		}else {
			User newUser = this.userService.createUser(user, userRole);
			URI location =
```

```
ServletUriComponentsBuilder.fromCurrentRequest().path("/{id}").buildAndExpand(newUser.getUserId()).toUri();
                                        return ResponseEntity.created(location).build();
                                }
        }
```