**Source Code**

**Back-End (Spring-Boot):**

**File: 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>2.7.6</version>

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

</parent>

<groupId>com.kitchenstory</groupId>

<artifactId>Phase4\_Kitchen\_Story</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>Phase4\_Kitchen\_Story</name>

<description>Phase 4 Kitchen Story Backend Project</description>

<properties>

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

</properties>

<dependencies>

<dependency>

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

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

</dependency>

<dependency>

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

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

</dependency>

<dependency>

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

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

</dependency>

<dependency>

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

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

</dependency>

<dependency>

<groupId>org.thymeleaf.extras</groupId>

<artifactId>thymeleaf-extras-springsecurity5</artifactId>

</dependency>

<dependency>

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

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

<scope>runtime</scope>

<optional>true</optional>

</dependency>

<dependency>

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

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

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.springframework.security</groupId>

<artifactId>spring-security-test</artifactId>

<scope>test</scope>

</dependency>

<!-- https://mvnrepository.com/artifact/io.jsonwebtoken/jjwt -->

<dependency>

<groupId>io.jsonwebtoken</groupId>

<artifactId>jjwt</artifactId>

<version>0.9.0</version>

</dependency>

<!-- https://mvnrepository.com/artifact/javax.xml.bind/jaxb-api -->

<dependency>

<groupId>javax.xml.bind</groupId>

<artifactId>jaxb-api</artifactId>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

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

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

</plugin>

</plugins>

</build>

</project>

**Package: resource**

**File: Application.properties**

spring.data.mongodb.host=localhost

spring.data.mongodb.port=27017

spring.data.mongodb.database=kitchenstory

server.port=8080

**Package: com.kitchenstory.jwthelper**

**File: JwtUtil.java**

package com.kitchenstory.jwthelper;

import io.jsonwebtoken.Claims;

import io.jsonwebtoken.Jwts;

import io.jsonwebtoken.SignatureAlgorithm;

import org.springframework.security.core.userdetails.UserDetails;

import org.springframework.stereotype.Component;

import java.util.Date;

import java.util.HashMap;

import java.util.Map;

import java.util.function.Function;

@Component

public class JwtUtil {

private String SECRET\_KEY = "secret";

public String extractUsername(String token) {

return extractClaim(token, Claims::getSubject);

}

public Date extractExpiration(String token) {

return extractClaim(token, Claims::getExpiration);

}

public <T> T extractClaim(String token, Function<Claims, T> claimsResolver) {

final Claims claims = extractAllClaims(token);

return claimsResolver.apply(claims);

}

private Claims extractAllClaims(String token) {

return Jwts.parser().setSigningKey(SECRET\_KEY).parseClaimsJws(token).getBody();

}

private Boolean isTokenExpired(String token) {

return extractExpiration(token).before(new Date());

}

public String generateToken(UserDetails userDetails) {

Map<String, Object> claims = new HashMap<>();

return createToken(claims, userDetails.getUsername());

}

private String createToken(Map<String, Object> claims, String subject) {

return Jwts.builder().setClaims(claims).setSubject(subject).setIssuedAt(new Date(System.currentTimeMillis()))

.setExpiration(new Date(System.currentTimeMillis() + 1000 \* 60 \* 60 \* 10))

.signWith(SignatureAlgorithm.HS256, SECRET\_KEY).compact();

}

public Boolean validateToken(String token, UserDetails userDetails) {

final String username = extractUsername(token);

return (username.equals(userDetails.getUsername()) && !isTokenExpired(token));

}

}

**Package: com.kitchenstory.config**

**File: CustomUserDetails.java**

package com.kitchenstory.config;

import java.util.Collection;

import java.util.HashSet;

import org.springframework.security.core.GrantedAuthority;

import org.springframework.security.core.authority.SimpleGrantedAuthority;

import org.springframework.security.core.userdetails.UserDetails;

import com.kitchenstory.entities.AdminCred;

@SuppressWarnings("serial")

public class CustomUserDetails implements UserDetails{

private AdminCred adminCred;

public CustomUserDetails(AdminCred adminCred) {

this.adminCred = adminCred;

}

@Override

public Collection<? extends GrantedAuthority> getAuthorities() {

HashSet<SimpleGrantedAuthority> set = new HashSet<>();

set.add(new SimpleGrantedAuthority(this.adminCred.getRole()));

return set;

}

@Override

public String getPassword() {

// TODO Auto-generated method stub

return this.adminCred.getPassword();

}

@Override

public String getUsername() {

// TODO Auto-generated method stub

return this.adminCred.getUsername();

}

@Override

public boolean isAccountNonExpired() {

// TODO Auto-generated method stub

return true;

}

@Override

public boolean isAccountNonLocked() {

// TODO Auto-generated method stub

return true;

}

@Override

public boolean isCredentialsNonExpired() {

// TODO Auto-generated method stub

return true;

}

@Override

public boolean isEnabled() {

// TODO Auto-generated method stub

return true;

}

}

**Package: com.kitchenstory.config**

**File: JwtAuthEntryPoint.java**

package com.kitchenstory.config;

import java.io.IOException;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.springframework.security.core.AuthenticationException;

import org.springframework.security.web.AuthenticationEntryPoint;

import org.springframework.stereotype.Component;

@Component

public class JwtAuthEntryPoint implements AuthenticationEntryPoint{

@Override

public void commence(HttpServletRequest request, HttpServletResponse response,

AuthenticationException authException) throws IOException, ServletException {

response.sendError(401,"Unauthorized");

}

}

**Package: com.kitchenstory.config**

**File: JwtAuthFilter.java**

package com.kitchenstory.config;

import java.io.IOException;

import javax.servlet.FilterChain;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

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

import org.springframework.security.authentication.UsernamePasswordAuthenticationToken;

import org.springframework.security.core.context.SecurityContextHolder;

import org.springframework.security.core.userdetails.UserDetails;

import org.springframework.security.web.authentication.WebAuthenticationDetailsSource;

import org.springframework.stereotype.Component;

import org.springframework.web.filter.OncePerRequestFilter;

import com.kitchenstory.jwthelper.JwtUtil;

import com.kitchenstory.services.CustomUserDetailService;

@Component

public class JwtAuthFilter extends OncePerRequestFilter {

@Autowired

CustomUserDetailService customUserDetailService;

@Autowired

private JwtUtil jwtUtil;

@Override

protected void doFilterInternal(HttpServletRequest request, HttpServletResponse response, FilterChain filterChain)

throws ServletException, IOException {

String requestTokenHeader = request.getHeader("Authorization");

String username = null;

String jwtToken = null;

if(requestTokenHeader!=null && requestTokenHeader.startsWith("Bearer ")) {

jwtToken = requestTokenHeader.substring(7);

try {

username = this.jwtUtil.extractUsername(jwtToken);

}catch(Exception e) {

e.printStackTrace();

}

UserDetails userDetails = this.customUserDetailService.loadUserByUsername(username);

if(username!=null && SecurityContextHolder.getContext().getAuthentication()==null) {

UsernamePasswordAuthenticationToken usernamePasswordAuthenticationToken = new UsernamePasswordAuthenticationToken(userDetails, null, userDetails.getAuthorities());

usernamePasswordAuthenticationToken.setDetails(new WebAuthenticationDetailsSource().buildDetails(request));

SecurityContextHolder.getContext().setAuthentication(usernamePasswordAuthenticationToken);

}else {

System.out.println("Token is invalid!");

}

}

filterChain.doFilter(request, response);

}

}

**Package: com.kitchenstory.config**

**File: SecurityConfig.java**

package com.kitchenstory.config;

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

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.security.authentication.AuthenticationManager;

import org.springframework.security.config.annotation.authentication.builders.AuthenticationManagerBuilder;

import org.springframework.security.config.annotation.web.builders.HttpSecurity;

import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;

import org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurerAdapter;

import org.springframework.security.config.http.SessionCreationPolicy;

import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

import org.springframework.security.web.authentication.UsernamePasswordAuthenticationFilter;

import com.kitchenstory.services.CustomUserDetailService;

@SuppressWarnings("deprecation")

@Configuration

@EnableWebSecurity

public class SecurityConfig extends WebSecurityConfigurerAdapter{

@Autowired

private CustomUserDetailService customUserDetailService;

@Autowired

private JwtAuthEntryPoint entryPoint;

@Autowired

JwtAuthFilter authFilter;

@Override

protected void configure(HttpSecurity http) throws Exception {

http

.csrf()

.disable()

.cors().and()

.authorizeRequests()

.antMatchers("/token").permitAll()

.antMatchers("/changePassword").permitAll()

.antMatchers("/getProduct/\*\*").permitAll()

.antMatchers("/getItem/\*\*").permitAll()

.antMatchers("/order").permitAll()

.antMatchers("/getOrder/\*\*").permitAll()

.anyRequest().authenticated()

.and().sessionManagement().sessionCreationPolicy(SessionCreationPolicy.STATELESS)

.and().exceptionHandling().authenticationEntryPoint(entryPoint);

http.addFilterBefore(authFilter, UsernamePasswordAuthenticationFilter.class);

}

@Override

protected void configure(AuthenticationManagerBuilder auth) throws Exception {

auth.userDetailsService(this.customUserDetailService).passwordEncoder(passwordEncoder());

}

@Bean

public BCryptPasswordEncoder passwordEncoder() {

return new BCryptPasswordEncoder(10);

}

@Bean

public AuthenticationManager authenticationManagerBean() throws Exception {

return super.authenticationManagerBean();

}

}

**Package: com.kitchenstory.entities**

**File: AdminCred.java**

package com.kitchenstory.entities;

import org.springframework.data.annotation.Id;

import org.springframework.data.mongodb.core.mapping.Document;

@Document(collection = "AdminCred")

public class AdminCred {

@Id

private String username;

private String password;

private String role;

public AdminCred() {

super();

// TODO Auto-generated constructor stub

}

public AdminCred(String username, String password, String role) {

super();

this.username = username;

this.password = password;

this.role = role;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public String getRole() {

return role;

}

public void setRole(String role) {

this.role = role;

}

}

**Package: com.kitchenstory.entities**

**File: JwtRequest.java**

package com.kitchenstory.entities;

public class JwtRequest {

String username;

String password;

public JwtRequest() {

}

public JwtRequest(String username, String password) {

super();

this.username = username;

this.password = password;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

@Override

public String toString() {

return "JwtRequest [username=" + username + ", password=" + password + "]";

}

}

**Package: com.kitchenstory.entities**

**File: JwtResponse.java**

package com.kitchenstory.entities;

public class JwtResponse {

String token;

public JwtResponse() {

super();

}

public JwtResponse(String token) {

super();

this.token = token;

}

public String getToken() {

return token;

}

public void setToken(String token) {

this.token = token;

}

}

**Package: com.kitchenstory.entities**

**File: Order.java**

package com.kitchenstory.entities;

import org.springframework.data.annotation.Id;

import org.springframework.data.mongodb.core.mapping.DBRef;

import org.springframework.data.mongodb.core.mapping.Document;

@Document(collection = "Orders")

public class Order {

@Id

private String id;

private String name;

private String address;

private String state;

private String contact;

private String od;

private int quantity;

private String payment;

@DBRef

private Product product;

public Order() {

super();

// TODO Auto-generated constructor stub

}

public Order(String id, String name, String address, String state, String contact, String od, int quantity,

String payment, Product product) {

super();

this.id = id;

this.name = name;

this.address = address;

this.state = state;

this.contact = contact;

this.od = od;

this.quantity = quantity;

this.payment = payment;

this.product = product;

}

public String getId() {

return id;

}

public void setId(String id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getAddress() {

return address;

}

public void setAddress(String address) {

this.address = address;

}

public String getState() {

return state;

}

public void setState(String state) {

this.state = state;

}

public String getContact() {

return contact;

}

public void setContact(String contact) {

this.contact = contact;

}

public String getOd() {

return od;

}

public void setOd(String od) {

this.od = od;

}

public int getQuantity() {

return quantity;

}

public void setQuantity(int quantity) {

this.quantity = quantity;

}

public String getPayment() {

return payment;

}

public void setPayment(String payment) {

this.payment = payment;

}

public Product getProduct() {

return product;

}

public void setProduct(Product product) {

this.product = product;

}

}

**Package: com.kitchenstory.entities**

**File: Product.java**

package com.kitchenstory.entities;

import org.springframework.data.annotation.Id;

import org.springframework.data.mongodb.core.mapping.Document;

@Document(collection = "Products")

public class Product {

@Id

private String name;

private String pid;

private String category;

private String ingredient;

private int price;

private String brand;

private String quantity;

private String origin;

public Product() {

}

public Product(String name, String pid, String category, String ingredient, int price, String brand,

String quantity, String origin) {

super();

this.name = name;

this.pid = pid;

this.category = category;

this.ingredient = ingredient;

this.price = price;

this.brand = brand;

this.quantity = quantity;

this.origin = origin;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getPid() {

return pid;

}

public void setPid(String pid) {

this.pid = pid;

}

public String getCategory() {

return category;

}

public void setCategory(String category) {

this.category = category;

}

public String getIngredient() {

return ingredient;

}

public void setIngredient(String ingredient) {

this.ingredient = ingredient;

}

public int getPrice() {

return price;

}

public void setPrice(int price) {

this.price = price;

}

public String getBrand() {

return brand;

}

public void setBrand(String brand) {

this.brand = brand;

}

public String getQuantity() {

return quantity;

}

public void setQuantity(String quantity) {

this.quantity = quantity;

}

public String getOrigin() {

return origin;

}

public void setOrigin(String origin) {

this.origin = origin;

}

}

**Package: com.kitchenstory.repo**

**File: AdminCredRepo.java**

package com.kitchenstory.repo;

import org.springframework.data.mongodb.repository.MongoRepository;

import org.springframework.stereotype.Repository;

import com.kitchenstory.entities.AdminCred;

@Repository

public interface AdminCredRepo extends MongoRepository<AdminCred, String> {

public AdminCred findByUsername(String username);

}

**Package: com.kitchenstory.repo**

**File: OrderRepo.java**

package com.kitchenstory.repo;

import org.springframework.data.mongodb.repository.MongoRepository;

import org.springframework.stereotype.Repository;

import com.kitchenstory.entities.Order;

@Repository

public interface OrderRepo extends MongoRepository<Order,String>{

}

**Package: com.kitchenstory.repo**

**File: ProductRepo.java**

package com.kitchenstory.repo;

import org.springframework.data.mongodb.repository.MongoRepository;

import org.springframework.stereotype.Repository;

import com.kitchenstory.entities.Product;

@Repository

public interface ProductRepo extends MongoRepository<Product, String>{

public Product findByName(String name);

public Iterable<Product> findByNameContainingIgnoreCase(String name);

}

**Package: com.kitchenstory.services**

**File: AdminService.java**

package com.kitchenstory.services;

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

import org.springframework.stereotype.Service;

import com.kitchenstory.entities.AdminCred;

import com.kitchenstory.repo.AdminCredRepo;

@Service

public class AdminService {

@Autowired

private AdminCredRepo adminRepo;

public void saveCred(AdminCred adminCred) {

this.adminRepo.save(adminCred);

}

public AdminCred findAdmin(String username) {

AdminCred admin = this.adminRepo.findByUsername(username);

return admin;

}

}

**Package: com.kitchenstory.services**

**File: CustomUserDetailService.java**

package com.kitchenstory.services;

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

import org.springframework.security.core.userdetails.UserDetails;

import org.springframework.security.core.userdetails.UserDetailsService;

import org.springframework.security.core.userdetails.UsernameNotFoundException;

import org.springframework.stereotype.Service;

import com.kitchenstory.config.CustomUserDetails;

import com.kitchenstory.entities.AdminCred;

import com.kitchenstory.repo.AdminCredRepo;

@Service

public class CustomUserDetailService implements UserDetailsService{

@Autowired

private AdminCredRepo credRepo;

@Override

public UserDetails loadUserByUsername(String username) throws UsernameNotFoundException {

AdminCred admin = this.credRepo.findByUsername(username);

if(admin == null) {

throw new UsernameNotFoundException("NO USER FOUND!");

}

return new CustomUserDetails(admin);

}

}

**Package: com.kitchenstory.services**

**File: OrderService.java**

package com.kitchenstory.services;

import java.util.List;

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

import org.springframework.stereotype.Service;

import com.kitchenstory.entities.Order;

import com.kitchenstory.repo.OrderRepo;

@Service

public class OrderService {

@Autowired

private OrderRepo orepo;

public void saveOrder(Order order) {

this.orepo.save(order);

}

public List<Order> getOrders(){

return this.orepo.findAll();

}

public void deleteOrder(String id) {

this.orepo.deleteById(id);

}

public Order getOrder(String id) {

return this.orepo.findById(id).get();

}

}

**Package: com.kitchenstory.services**

**File: ProductService.java**

package com.kitchenstory.services;

import java.util.List;

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

import org.springframework.data.util.Streamable;

import org.springframework.stereotype.Service;

import com.kitchenstory.entities.Product;

import com.kitchenstory.repo.ProductRepo;

@Service

public class ProductService {

@Autowired

private ProductRepo prodRepo;

public void addProduct(Product product) {

this.prodRepo.save(product);

}

public List<Product> showProducts(){

return this.prodRepo.findAll();

}

public void deleteProduct(String name) {

this.prodRepo.deleteById(name);

}

public void updateProduct(String name,Product updateProduct) {

Product product = this.prodRepo.findByName(name);

product.setName(updateProduct.getName());

product.setPid(updateProduct.getPid());

product.setCategory(updateProduct.getCategory());

product.setBrand(updateProduct.getBrand());

product.setIngredient(updateProduct.getIngredient());

product.setOrigin(updateProduct.getOrigin());

product.setPrice(updateProduct.getPrice());

product.setQuantity(updateProduct.getQuantity());

this.prodRepo.save(product);

}

public Product getProduct(String name) {

return this.prodRepo.findByName(name);

}

public List<Product> getAllByName(String name){

Iterable<Product> products = this.prodRepo.findByNameContainingIgnoreCase(name);

List<Product> allProducts = Streamable.of(products).toList();

return allProducts;

}

}

**Package: com.kitchenstory.controller**

**File: AdminLoginController.java**

package com.kitchenstory.controller;

import java.util.List;

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

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

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

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

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

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

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

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

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

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

import com.kitchenstory.entities.Product;

import com.kitchenstory.services.ProductService;

@RestController

@CrossOrigin(origins = "\*")

public class AdminLoginController {

@Autowired

private ProductService prodService;

@PostMapping("/saveItem")

public ResponseEntity<?> saveItem(@RequestBody Product product){

this.prodService.addProduct(product);

return ResponseEntity.status(HttpStatus.CREATED).build();

}

@GetMapping("/getItems")

public ResponseEntity<?> getItems() {

List<Product> allItems = this.prodService.showProducts();

if(allItems.isEmpty()) {

return ResponseEntity.status(HttpStatus.NOT\_FOUND).build();

}else {

return ResponseEntity.ok(allItems);

}

}

@DeleteMapping("/deleteItem/{name}")

public ResponseEntity<?> deleteItem(@PathVariable("name") String name) {

try {

this.prodService.deleteProduct(name);

return ResponseEntity.status(HttpStatus.OK).build();

}catch(Exception e) {

System.out.println("Empty Result Data Access Exception.");

return ResponseEntity.status(HttpStatus.INTERNAL\_SERVER\_ERROR).build();

}

}

@PutMapping("/updateItem/{name}")

public ResponseEntity<?> updateItem(@PathVariable("name") String name,@RequestBody Product product) {

try {

this.prodService.updateProduct(name, product);

return ResponseEntity.status(HttpStatus.OK).build();

}catch(Exception e) {

System.out.println("Empty Result Data Access Exception.");

return ResponseEntity.status(HttpStatus.INTERNAL\_SERVER\_ERROR).build();

}

}

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

public ResponseEntity<?> getItem(@PathVariable("name") String name){

Product product = this.prodService.getProduct(name);

if(product!=null) {

return ResponseEntity.ok(product);

}else {

return ResponseEntity.status(HttpStatus.NOT\_FOUND).build();

}

}

}

**Package: com.kitchenstory.controller**

**File: JwtController.java**

package com.kitchenstory.controller;

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

import org.springframework.http.ResponseEntity;

import org.springframework.security.authentication.AuthenticationManager;

import org.springframework.security.authentication.UsernamePasswordAuthenticationToken;

import org.springframework.security.core.userdetails.UserDetails;

import org.springframework.security.core.userdetails.UsernameNotFoundException;

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

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

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

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

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

import com.kitchenstory.entities.JwtRequest;

import com.kitchenstory.entities.JwtResponse;

import com.kitchenstory.jwthelper.JwtUtil;

import com.kitchenstory.services.CustomUserDetailService;

@RestController

@CrossOrigin(origins = "\*")

public class JwtController {

@Autowired

AuthenticationManager authenticationManager;

@Autowired

CustomUserDetailService customUserDetailService;

@Autowired

JwtUtil jwtUtil;

@RequestMapping(value="/token",method = RequestMethod.POST)

public ResponseEntity<?> generateToken(@RequestBody JwtRequest jwtRequest){

System.out.println(jwtRequest);

try {

this.authenticationManager.authenticate(new UsernamePasswordAuthenticationToken(jwtRequest.getUsername(), jwtRequest.getPassword()));

}catch(UsernameNotFoundException e) {

e.printStackTrace();

}

UserDetails userDetails = this.customUserDetailService.loadUserByUsername(jwtRequest.getUsername());

String token = this.jwtUtil.generateToken(userDetails);

System.out.println("Token: "+token);

return ResponseEntity.ok(new JwtResponse(token));

}

}

**Package: com.kitchenstory.controller**

**File: SignInController.java**

package com.kitchenstory.controller;

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

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

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

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

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

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

import com.kitchenstory.entities.AdminCred;

import com.kitchenstory.services.AdminService;

@RestController

@CrossOrigin(origins = "\*")

public class SignInController {

@Autowired

private AdminService adminService;

@Autowired

private BCryptPasswordEncoder passwordEncoder;

@PutMapping("/changePassword")

public ResponseEntity<?> setPassword(@RequestBody AdminCred admin) {

AdminCred adm = this.adminService.findAdmin(admin.getUsername());

if(adm == null) {

return ResponseEntity.status(HttpStatus.NOT\_FOUND).build();

}

else { adm.setPassword(this.passwordEncoder.encode(admin.getPassword()));

this.adminService.saveCred(adm);

return ResponseEntity.ok(adm);

}

}

}

**Package: com.kitchenstory.controller**

**File: UserController.java**

package com.kitchenstory.controller;

import java.text.DateFormat;

import java.util.Calendar;

import java.util.List;

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

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

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

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

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

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

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

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

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

import com.kitchenstory.entities.Order;

import com.kitchenstory.entities.Product;

import com.kitchenstory.services.OrderService;

import com.kitchenstory.services.ProductService;

@RestController

@CrossOrigin(origins = "\*")

public class UserController {

@Autowired

private OrderService service;

@Autowired

private ProductService pservice;

@PostMapping("/order")

public ResponseEntity<?> createOrder(@RequestBody Order order){

DateFormat df = DateFormat.getDateInstance();

Calendar cl = Calendar.getInstance();

String od = df.format(cl.getTime());

order.setOd(od);

this.service.saveOrder(order);

return ResponseEntity.status(HttpStatus.CREATED).build();

}

@GetMapping("/getOrders")

public ResponseEntity<?> getAllOrders(){

List<Order> orders = this.service.getOrders();

if(orders.isEmpty()) {

return ResponseEntity.status(HttpStatus.NOT\_FOUND).build();

}

else {

return ResponseEntity.ok(orders);

}

}

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

public ResponseEntity<?> deleteOrder(@PathVariable("id") String id){

this.service.deleteOrder(id);

return ResponseEntity.status(HttpStatus.OK).build();

}

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

public ResponseEntity<?> getProducts(@PathVariable("name") String name){

List<Product> products = this.pservice.getAllByName(name);

if(products.isEmpty()) {

return ResponseEntity.status(HttpStatus.NOT\_FOUND).build();

}else {

return ResponseEntity.ok(products);

}

}

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

public ResponseEntity<?> getOrder(@PathVariable("id") String id){

Order order = this.service.getOrder(id);

if(order==null) {

return ResponseEntity.status(HttpStatus.NOT\_FOUND).build();

}else {

return ResponseEntity.ok(order);

}

}

}

**Package: com.kitchenstory.controller**

**File: UserController.java**

package com.kitchenstory;

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

import org.springframework.boot.CommandLineRunner;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

import com.kitchenstory.entities.AdminCred;

import com.kitchenstory.repo.AdminCredRepo;

@SpringBootApplication

public class Phase4KitchenStoryApplication implements CommandLineRunner {

@Autowired

private AdminCredRepo credRepo;

@Autowired

private BCryptPasswordEncoder passwordEncoder;

public static void main(String[] args) {

SpringApplication.run(Phase4KitchenStoryApplication.class, args);

}

@Override

public void run(String... args) throws Exception {

AdminCred cred = new AdminCred();

cred.setUsername("chandan@kitchenstory.com");

cred.setPassword(this.passwordEncoder.encode("Chandan@123"));

cred.setRole("ROLE\_ADMIN");

this.credRepo.save(cred);

}

}

**Front-End (Angular):**

**File: index.html**

<!doctype html>

<html lang="en">

<head>

  <meta charset="utf-8">

  <title>KitchenStory</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/bootstrap@4.0.0/dist/css/bootstrap.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.slim.min.js"

    integrity="sha384-KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"

    crossorigin="anonymous"></script>

  <script src="https://cdn.jsdelivr.net/npm/popper.js@1.12.9/dist/umd/popper.min.js"

    integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"

    crossorigin="anonymous"></script>

  <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/js/bootstrap.min.js"

    integrity="sha384-JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"

    crossorigin="anonymous"></script>

</body>

</html>

**File: styles.css**

body {

    background-image: url('https://images.unsplash.com/photo-1533109721025-d1ae7ee7c1e1?ixlib=rb-4.0.3&ixid=M3wxMjA3fDB8MHxzZWFyY2h8Mnx8Y29sb3VyfGVufDB8fDB8fHww&w=1000&q=80');

    height: fit-content;

    width: 100wh;

    background-repeat: no-repeat;

    background-size: cover;

}

**File: app.module.ts**

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

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

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

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

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

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

import { DashboardComponent } from './Components/dashboard/dashboard.component';

import { NavbarComponent } from './Components/navbar/navbar.component';

import { ChangepasswordComponent } from './Components/changepassword/changepassword.component';

import { AllitemsComponent } from './Components/allitems/allitems.component';

import { AdditemComponent } from './Components/additem/additem.component';

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

import { HttpClientModule, HTTP\_INTERCEPTORS } from '@angular/common/http';

import { AuthInterceptor } from './Services/auth.interceptor';

import { LoginService } from './Services/login.service';

import { AuthGuard } from './Services/auth.guard';

import { UpdateItemComponent } from './Components/update-item/update-item.component';

import { ProductSearchComponent } from './Components/product-search/product-search.component';

import { PlaceOrderComponent } from './Components/place-order/place-order.component';

import { OrderConfirmComponent } from './Components/order-confirm/order-confirm.component';

import { AllOrdersComponent } from './Components/all-orders/all-orders.component';

@NgModule({

  declarations: [

    AppComponent,

    HomeComponent,

    LoginComponent,

    DashboardComponent,

    NavbarComponent,

    ChangepasswordComponent,

    AllitemsComponent,

    AdditemComponent,

    UpdateItemComponent,

    ProductSearchComponent,

    PlaceOrderComponent,

    OrderConfirmComponent,

    AllOrdersComponent,

  ],

  imports: [

    BrowserModule,

    AppRoutingModule,

    FormsModule,

    HttpClientModule

  ],

  providers: [LoginService, AuthGuard, [{ provide: HTTP\_INTERCEPTORS, useClass: AuthInterceptor, multi: true }]],

  bootstrap: [AppComponent]

})

export class AppModule { }

**Service**

**File: admin.service.ts**

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

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

import { Observable } from 'rxjs';

import { Item } from '../Components/allitems/Items';

@Injectable({

  providedIn: 'root'

})

export class AdminService {

  baseUrl = 'http://localhost:8080'

  constructor(private http: HttpClient) { }

  getItems(): Observable<Item[]> {

    return this.http.get<Item[]>(`${this.baseUrl}/getItems`)

  }

  addItem(item: Item): Observable<Object> {

    return this.http.post<Object>(`${this.baseUrl}/saveItem`, item)

  }

  getItemByName(name: string): Observable<Item> {

    return this.http.get<Item>(`${this.baseUrl}/getItem/${name}`)

  }

  updateItem(name: string, item: Item): Observable<Object> {

    return this.http.put<Object>(`${this.baseUrl}/updateItem/${name}`, item)

  }

  deleteItem(name: string): Observable<Object> {

    return this.http.delete<Object>(`${this.baseUrl}/deleteItem/${name}`)

  }

}

**File: auth.guard.ts**

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

import { ActivatedRouteSnapshot, CanActivate, Router, RouterStateSnapshot, UrlTree } from '@angular/router';

import { Observable } from 'rxjs';

import { LoginService } from './login.service';

@Injectable({

  providedIn: 'root'

})

export class AuthGuard implements CanActivate {

  constructor(private loginService: LoginService, private router: Router) { }

  canActivate(

    route: ActivatedRouteSnapshot,

    state: RouterStateSnapshot): Observable<boolean | UrlTree> | Promise<boolean | UrlTree> | boolean | UrlTree {

    if (this.loginService.isLoggedIn()) {

      return true;

    }

    this.router.navigate(['login'])

    return false;

  }

}

**File: auth.interceptor.ts**

import { HttpEvent, HttpHandler, HttpInterceptor, HttpRequest } from "@angular/common/http";

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

import { Observable } from "rxjs";

import { LoginService } from "./login.service";

@Injectable()

export class AuthInterceptor implements HttpInterceptor {

    constructor(private loginService: LoginService) { }

    intercept(req: HttpRequest<any>, next: HttpHandler): Observable<HttpEvent<any>> {

        let newReq = req;

        let token = this.loginService.getToken()

        if (token != null) {

            newReq = newReq.clone({ setHeaders: { Authorization: `Bearer ${token}` } })

        }

        return next.handle(newReq)

    }

}

**File: login.service.ts**

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

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

import { Observable } from 'rxjs';

import { Admin } from '../Components/changepassword/admin';

@Injectable({

  providedIn: 'root'

})

export class LoginService {

  user="";

  url = "http://localhost:8080"

  constructor(private http: HttpClient) { }

  generateToken(credentials: any) {

    return this.http.post(`${this.url}/token`, credentials)

  }

  loginUser(token: any) {

    localStorage.setItem('token', token);

    return true;

  }

  isLoggedIn() {

    let token = localStorage.getItem('token');

    if (token == undefined || token === '' || token == null) {

      return false;

    } else {

      return true;

    }

  }

  logout() {

    localStorage.removeItem('token');

    return true;

  }

  getToken() {

    return localStorage.getItem('token');

  }

  changePassword(admin: Admin): Observable<Object> {

    return this.http.put<Object>(`${this.url}/changePassword`, admin);

  }

}

**File: user.service.ts**

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

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

import { Observable } from 'rxjs';

import { Item } from '../Components/allitems/Items';

import { Order } from '../Components/place-order/order';

@Injectable({

  providedIn: 'root'

})

export class UserService {

  baseUrl = 'http://localhost:8080'

  constructor(private http: HttpClient) { }

  searchItems(name: string): Observable<Item[]> {

    return this.http.get<Item[]>(`${this.baseUrl}/getProduct/${name}`)

  }

  orderProduct(order: Order): Observable<Object> {

    return this.http.post<Object>(`${this.baseUrl}/order`, order)

  }

  getOrder(id: number): Observable<Order> {

    return this.http.get<Order>(`${this.baseUrl}/getOrder/${id}`)

  }

  getAllOrders(): Observable<Order[]> {

    return this.http.get<Order[]>(`${this.baseUrl}/getOrders`)

  }

  deleteOrder(id: string): Observable<Object> {

    return this.http.delete<Object>(`${this.baseUrl}/delete/${id}`)

  }

}

**Component**

1. **NavBar**

**File: navbar.component.ts**

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

import { LoginService } from 'src/app/Services/login.service';

@Component({

  selector: 'app-navbar',

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

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

})

export class NavbarComponent implements OnInit {

  user="";

  public loggedIn = false;

  constructor(private loginService: LoginService) {

  }

  ngOnInit(): void {

    this.loggedIn = this.loginService.isLoggedIn()

  }

  logoutAdmin() {

    this.loginService.logout()

    location.reload()

  }

}

**File: navbar.component.html**

<header>

    <nav class="navbar navbar-expand-sm bg-dark navbar-dark">

        <a \*ngIf="!loggedIn" class="navbar-brand" routerLink="/">KitchenStory.com</a>

        <a \*ngIf="loggedIn" class="navbar-brand" routerLink="/dashboard">Admin Dashboard</a>

        <ul class="navbar-nav ml-auto">

            <li class="nav-item">

                <a \*ngIf="!loggedIn" routerLink="/login" role="button" class="btn btn-success">Admin Login</a>

                <label \*ngIf="loggedIn">{{user}}</label>

                <a \*ngIf="loggedIn" (click)="logoutAdmin()" role="button" class="btn btn-danger">Logout</a>

            </li>

        </ul>

    </nav>

</header>

1. **home**

**File:** [**home.component.ts**](http://home.component.ts)

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

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

@Component({

  selector: 'app-home',

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

  styleUrls: ['./home.component.css']

})

export class HomeComponent {

  product: any;

  constructor(private router: Router) { }

  search(name: string) {

    if (name == null) {

      alert('Please enter a product name!');

    } else {

      let url: string = "/product-search/" + name;

      this.router.navigateByUrl(url);

    }

  }

}

**File:** [**home.component.html**](http://home.component.html)

<div class="container">

    <div class="row">

        <div class="col">

            <div class="jumbotron text-center" style="background-color: transparent; color: aliceblue;">

                <h1 class="display-2" style="font-family: sans-serif;"><b>Welcome To</b></h1>

                <h2 class="display-5"><b>Kitchen Story</b></h2>

            </div>

        </div>

    </div>

    <br>

    <div class="col-8 mx-auto">

        <div class="card bg-info">

            <div class="card-header">

                <h5 class="text-center">Search essentials, grocery and more...</h5>

            </div>

            <div class="card-body">

                <div class="form-group col-8 mx-auto my-auto">

                    <input type="text" class="form-control" [(ngModel)]="product" name="product"

                        placeholder="Search for products...">

                </div>

            </div>

            <div class="card-footer text-center"><button (click)="search(product)"

                    class="btn btn-success btn-lg">Search</button></div>

        </div>

    </div>

</div>

1. **product-search**

**File:** [**product-search.component.ts**](http://home.component.ts)

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

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

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

import { UserService } from 'src/app/Services/user.service';

import { Item } from '../allitems/Items';

@Component({

  selector: 'app-product-search',

  templateUrl: './product-search.component.html',

  styleUrls: ['./product-search.component.css']

})

export class ProductSearchComponent implements OnInit {

  name!: string;

  item!: Item[];

  constructor(private userService: UserService, private route: ActivatedRoute, private router: Router) { }

  ngOnInit(): void {

    this.name = this.route.snapshot.params['name'];

    this.showItems()

  }

  showItems() {

    this.userService.searchItems(this.name).subscribe({

      next: (items: Item[]) => {

        this.item = items;

        console.log(this.item);

      }, error: (error: any) => {

        console.log(error);

        alert('No products found!');

      }

    })

  }

  selectItem(item: string) {

    let url = "/place-order/" + item;

    this.router.navigateByUrl(url);

  }

}

**File: product-search.component.html**

<div class="container">

    <div class="row mx-auto mt-4">

        <div class="col">

            <div class="card-columns">

                <div \*ngFor="let product of item" class="card bg-warning">

                    <img class=" card-img-top"

                        src="https://img.freepik.com/free-vector/shopping-supermarket-cart-with-grocery-pictogram\_1284-11697.jpg?w=740&t=st=1672168343~exp=1672168943~hmac=4018e3def7957bf35104fa91ff39940cb701fbe95e5483966941a83644f52fc0"

                        style="width: 100%; height: 250px;" alt="Card image">

                    <div class="card-header">

                        <h5>{{product.name}}</h5>

                    </div>

                    <div class="card-body">

                        <p class="card-text"><b>Quantity: </b>{{product.quantity}}</p>

                        <p class="card-text"><b>Price:</b><span> Rs </span> {{product.price}}</p>

                        <p class="card-text"><b>Ingeredient: </b>{{product.ingredient}}</p>

                        <p class="card-text"><b>Manufacturer: </b>{{product.brand}}</p>

                        <p class="card-text"><b>Country of origin: </b>{{product.origin}}</p>

                    </div>

                    <div class="card-footer">

                        <button class="btn btn-primary btn-block" (click)="selectItem(product.name)"><b>Buy

                                Now</b></button>

                    </div>

                </div>

            </div>

        </div>

    </div>

</div>

1. **place-order**

**File:** [**place-order.component.ts**](http://home.component.ts)

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

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

import { AdminService } from 'src/app/Services/admin.service';

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

import { Item } from '../allitems/Items';

import { UserService } from 'src/app/Services/user.service';

import { Order } from './order';

@Component({

  selector: 'app-place-order',

  templateUrl: './place-order.component.html',

  styleUrls: ['./place-order.component.css']

})

export class PlaceOrderComponent implements OnInit {

  item!: string;

  product: Item = new Item();

  order: Order = new Order();

  amount!: number;

  public payment = false;

  constructor(private userService: UserService, private adminService: AdminService, private route: ActivatedRoute, private router: Router) { }

  ngOnInit(): void {

    this.item = this.route.snapshot.params['name'];

    this.adminService.getItemByName(this.item).subscribe({

      next: (item) => {

        this.product = item;

        console.log(this.product);

      }, error: (error) => {

        console.log(error);

      }

    })

  }

  onPay(): boolean {

    if (this.order.payment == "COD") {

      this.payment = true;

      return this.payment;

    } else {

      return this.payment = false;

    }

  }

  onSubmit(): void {

    if (this.order.name == null || this.order.address == null || this.order.contact == null || this.order.state == null) {

      alert("Please enter the details!");

    } else {

      this.order.id = new Date().getTime().toString();

      this.order.product = this.product;

      this.userService.orderProduct(this.order).subscribe({

        next: (product) => {

          let url = "/order-confirm/" + this.order.id;

          this.router.navigateByUrl(url).then(() => { location.reload() });

        }, error: (error) => {

          console.log(error);

        }

      })

    }

  }

}

**File: place-order.component.html**

<main>

    <div class="container">

        <div class="row">

            <div class="col-md-8 mx-auto">

                <div class="card mt-2 bg-light">

                    <div class="card-header">

                        <h5 class="text-center"><b>Product:

                            </b>{{product.name}}<label>&nbsp;&nbsp;&nbsp;

                                <span><b>Quantity: </b>{{product.quantity}}</span></label>

                        </h5>

                    </div>

                    <div class="card-body">

                        <h3 class="card-title text-center">Please enter your details to create order:</h3>

                        <hr>

                        <form (ngSubmit)="onSubmit()" class="was-validated">

                            <div class="form-row">

                                <div class="form-group col-md-6">

                                    <label for="name">Enter your full name:</label> <input type="text"

                                        class="form-control" [(ngModel)]="order.name" name="name"

                                        placeholder="Enter your full name" required>

                                    <div class="valid-feedback">Valid.</div>

                                    <div class="invalid-feedback">Please fill out this field.</div>

                                </div>

                                <div class="form-group col-md-6">

                                    <label for="address">Enter shipping address:</label> <input type="text"

                                        class="form-control" [(ngModel)]="order.address" name="address"

                                        placeholder="Enter your address" required>

                                    <div class="valid-feedback">Valid.</div>

                                    <div class="invalid-feedback">Please fill out this field.</div>

                                </div>

                                <div class="form-group col-md-6">

                                    <label for="st">Enter State:</label> <input type="text" class="form-control"

                                        [(ngModel)]="order.state" name="state" placeholder="Enter your state" required>

                                    <div class="valid-feedback">Valid.</div>

                                    <div class="invalid-feedback">Please fill out this field.</div>

                                </div>

                                <div class="form-group col-md-6">

                                    <label for="cnt">Enter Contact:</label> <input type="number" class="form-control"

                                        [(ngModel)]="order.contact" name="contact" placeholder="Enter your contact"

                                        required>

                                    <div class="valid-feedback">Valid.</div>

                                    <div class="invalid-feedback">Please fill out this field.</div>

                                </div>

                                <div class="form-group col-md-6">

                                    <label for="qty">Select quantity</label> <select [(ngModel)]="order.quantity"

                                        name="quantity" class="form-control">

                                        <option value="1">1</option>

                                        <option value="2">2</option>

                                        <option value="3">3</option>

                                        <option value="4">4</option>

                                        <option value="5">5</option>

                                    </select>

                                </div>

                                <div class="form-group col-md-6">

                                    <label for="pmt">Choose payment mode:</label><select [(ngModel)]="order.payment"

                                        (click)="onPay()" name="payment" class="form-control">

                                        <option selected>Choose..</option>

                                        <option value="COD">Cash On Delivery</option>

                                        <option value="Card">Card Payment</option>

                                    </select>

                                </div>

                            </div>

                            <div>

                                <div class="text-center"><button \*ngIf="payment" type="submit"

                                        class="btn btn-success btn-lg">Place order</button>

                                    <button \*ngIf="!payment" type="button" class="btn btn-success btn-lg"

                                        data-toggle="modal" data-target="#myModal">

                                        Pay now

                                    </button>

                                </div>

                                <!-- The Modal -->

                                <div class="modal fade" id="myModal">

                                    <div class="modal-dialog modal-dialog-centered">

                                        <div class="modal-content">

                                            <!-- Modal Header -->

                                            <div class="modal-header">

                                                <h4 class="modal-title">Kindly enter your card details</h4>

                                                <button type="button" class="close"

                                                    data-dismiss="modal">&times;</button>

                                            </div>

                                            <!-- Modal body -->

                                            <div class="modal-body">

                                                <form (ngSubmit)="onSubmit()" class="was-validated">

                                                    <div class="form-group">

                                                        <label for="uname">Enter the name on the card:</label>

                                                        <input type="text" class="form-control" id="uname"

                                                            placeholder="Enter the name on card" name="cname" required>

                                                        <div class="valid-feedback">Valid.</div>

                                                        <div class="invalid-feedback">Please fill out this field.

                                                        </div>

                                                    </div>

                                                    <div class="form-group">

                                                        <label for="crd">Enter your card number:</label>

                                                        <input type="number" class="form-control" id="crd"

                                                            placeholder="Enter the card number" name="cnumber" required>

                                                        <div class="valid-feedback">Valid.</div>

                                                        <div class="invalid-feedback">Please fill out this field.

                                                        </div>

                                                    </div>

                                                    <div class="form-group">

                                                        <label for="cvvn">Enter your CVV number:</label>

                                                        <input type="password" class="form-control" id="cvvn"

                                                            placeholder="Enter your cvv" name="cvv" required>

                                                        <div class="valid-feedback">Valid.</div>

                                                        <div class="invalid-feedback">Please fill out this field.

                                                        </div>

                                                    </div>

                                                    <div class="form-group form-check">

                                                        <label class="form-check-label">

                                                            <input class="form-check-input" type="checkbox"

                                                                name="remember" required> I agree to the terms and

                                                            conditions.

                                                            <div class="valid-feedback">Valid.</div>

                                                            <div class="invalid-feedback">Check this checkbox to

                                                                continue.</div>

                                                        </label>

                                                    </div>

                                                    <div>

                                                        <h5>Payable amount =

                                                            Rs. {{(this.order.quantity) \* (this.product.price)}}

                                                        </h5>

                                                    </div>

                                                    <button type="submit" class="btn btn-success">Make Payment</button>

                                                </form>

                                            </div>

                                            <!-- Modal footer -->

                                            <div class="modal-footer">

                                                <button type="button" class="btn btn-danger"

                                                    data-dismiss="modal">Cancel</button>

                                            </div>

                                        </div>

                                    </div>

                                </div>

                            </div>

                        </form>

                    </div>

                    <div class="card-footer">

                        <h5 class="text-center">Total payable amount = Rs. {{(this.order.quantity) \*

                            (this.product.price)}}

                        </h5>

                    </div>

                </div>

            </div>

        </div>

    </div>

</main>

1. **order-confirm**

**File:** [**order-confirm.component.ts**](http://home.component.ts)

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

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

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

import { UserService } from 'src/app/Services/user.service';

import { Order } from '../place-order/order';

import { Item } from '../allitems/Items';

@Component({

  selector: 'app-order-confirm',

  templateUrl: './order-confirm.component.html',

  styleUrls: ['./order-confirm.component.css']

})

export class OrderConfirmComponent implements OnInit {

  id!: number;

  getOrder: Order = new Order();

  product: Item = new Item();

  constructor(private route: ActivatedRoute, private userService: UserService) { }

  ngOnInit(): void {

    this.id = this.route.snapshot.params['id'];

    this.userService.getOrder(this.id).subscribe({

      next: (order) => {

        this.getOrder = order

        this.product = this.getOrder.product

        console.log(this.getOrder)

      }, error: (error) => {

        console.log(error)

      }

    })

  }

}

**File: order-confirm.component.html**

<div class="container">

    <div class="row">

        <div class="col-lg-8 mx-auto mt-3">

            <div class="card  bg-light">

                <div class="card-header text-center">

                    <h3><b>Order Confirmation</b></h3>

                    <p>Dear <b>{{getOrder.name}}</b>, Thank you for shopping with us!</p>

                </div>

                <div class="card-body">

                    <h4 class="text-center"><b>Order Summary</b></h4>

                    <p class="text-center">{{getOrder.od}}</p>

                    <div class="card mx-auto">

                        <div class="card-body">

                            <p class="text-center"><b>Order Id: </b> {{getOrder.id}} <span>&nbsp;</span> <b>Product:

                                </b> {{product.name}}

                            </p>

                            <p class="text-center"><b>Weight: </b> {{product.quantity}} <span>&nbsp;</span> <b>Price:

                                </b>Rs.

                                {{product.price}}

                                <span>&nbsp;</span> <b>Ordered quantity: </b> {{getOrder.quantity}}

                            </p>

                            <hr>

                            <div class="text-center">

                                <h5><b>Order Total</b></h5>

                                <p><b>Payment mode: </b> {{getOrder.payment}}</p>

                                <p><b>Total Price: </b>Rs. {{(getOrder.quantity)\*(product.price)}}</p>

                            </div>

                            <hr>

                            <div class="text-center">

                                <h5><b>Shipping details</b></h5>

                                <p><b>Shipping address: </b>{{getOrder.address}},{{getOrder.state}}</p>

                                <p><b>Contact detail: </b>{{getOrder.contact}}</p>

                            </div>

                        </div>

                    </div>

                </div>

                <div class="card-footer  text-center">

                    <h6 class="text-center">Your order is on it's way. Thank you!</h6>

                    <button class="btn btn-success" routerLink="">Shop More</button>

                </div>

            </div>

        </div>

    </div>

</div>

1. **login**

**File:** [**login.component.ts**](http://home.component.ts)

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

import { LoginService } from 'src/app/Services/login.service';

@Component({

  selector: 'app-login',

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

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

})

export class LoginComponent {

  credentials = {

    username: '',

    password: ''

  }

  constructor(private loginService: LoginService) { }

  onSubmit() {

    if ((this.credentials.username != '' && this.credentials.password != '') && (this.credentials.username != null && this.credentials.password != null)) {

      //token generate

      this.loginService.generateToken(this.credentials).subscribe(

        (response: any) => {

          //success

          console.log(response.token);

          this.loginService.loginUser(response.token)

          window.location.href = "/dashboard"

        }, error => {

          //error

          console.log(error);

          alert("Invalid credentials!")

        }

      )

    } else {

      alert("Kindly fill details");

    }

  }

}

**File: login.component.html**

<div class="container">

    <div class="row">

        <div class="col-lg-4 col-md-8 mx-auto">

            <div class="card mt-1 bg-warning">

                <img class="card-img-top" src="https://www.w3schools.com/bootstrap4/img\_avatar1.png" alt="Card image"

                    style="width:100%; height:250px;">

                <div class="card-body">

                    <h4 class="card-title text-center">Kitchen Story Admin Login</h4>

                    <hr>

                    <form (ngSubmit)="onSubmit()">

                        <div class="form-group">

                            <label for="email"><b>Username:</b></label> <input type="text" name="username"

                                [(ngModel)]="credentials.username" class="form-control" placeholder="Enter Username"

                                id="email" required>

                        </div>

                        <div class="form-group">

                            <label for="pwd"><b>Password:</b></label> <input type="password" name="password"

                                [(ngModel)]="credentials.password" class="form-control" placeholder="Enter password"

                                id="pwd" required>

                        </div>

                        <button type="submit" class="btn btn-primary btn-block">Login</button>

                        <a routerLink="/change-password" role="button" class="btn btn-outline-danger btn-block">Change

                            Password</a>

                    </form>

                </div>

            </div>

        </div>

    </div>

</div>

1. **changepassword**

**File:** [**changepassword.component.ts**](http://home.component.ts)

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

import { LoginService } from 'src/app/Services/login.service';

import { Admin } from './admin';

@Component({

  selector: 'app-changepassword',

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

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

})

export class ChangepasswordComponent {

  admin: Admin = new Admin();

  cnfPassword: any;

  constructor(private loginService: LoginService) { }

  changePassword() {

    this.loginService.changePassword(this.admin).subscribe({

      next: (response) => {

        console.log(response);

        alert('Password changed successfully!');

      }, error: (error) => {

        console.log(error);

        alert('Username is invalid!');

      }

    })

  }

  onSubmit() {

    if (this.admin.username == null || this.admin.password == null) {

      alert("Please fill in your username and new password")

    } else if (this.admin.password != this.cnfPassword) {

      alert("Password does not match!");

    } else {

      this.changePassword()

    }

  }

}

**File: changepassword.component.html**

<div class="container">

    <div class="row">

        <div class="col-lg-5 col-md-6 mx-auto mt-5">

            <div class="card mt-5 bg-light">

                <div class="card-body">

                    <h4 class="card-title text-center">Admin Change Password</h4>

                    <hr>

                    <form (ngSubmit)="onSubmit()">

                        <div class="form-group">

                            <label for="email">Username:</label> <input type="text" name="username" class="form-control"

                                [(ngModel)]="admin.username" placeholder="Enter Username" required>

                        </div>

                        <div class="form-group">

                            <label for="pwd">New Password:</label> <input type="password" name="password"

                                [(ngModel)]="admin.password" class="form-control" placeholder="Enter new password"

                                id="pwd1" required>

                        </div>

                        <div class="form-group">

                            <label for="pwd">Confirm New Password:</label> <input type="password" name="cnfPassword"

                                [(ngModel)]="cnfPassword" class="form-control" placeholder="Confirm new password"

                                id="pwd2" required>

                        </div>

                        <button type="submit" class="btn btn-danger btn-block">Change Password</button>

                        <a routerLink="/login" role="button" class="btn btn-outline-success btn-block">Back

                            to Admin Login</a>

                    </form>

                </div>

            </div>

        </div>

    </div>

</div>

1. **dashboard**

**File: dashboard.component.html**

<br><br><br><br>

<main>

    <div class="container">

        <div class="row">

            <div class="col">

                <div class="card mx-auto bg-light">

                    <div class="card-body">

                        <h1 class="card-title text-center">Welcome To Admin Dashboard</h1><br><br>

                        <div class="card-deck">

                            <div class="card bg-secondary">

                                <div class="card-body">

                                    <h4 class="card-title">Add Grocery Items</h4>

                                    <p class="card-text">Add new items to your stock list. </p>

                                    <a routerLink="/add-item" role="button" class="btn btn-primary btn-block"><b>Add

                                            Items</b>

                                    </a>

                                </div>

                            </div>

                            <div class="card bg-warning">

                                <div class="card-body">

                                    <h4 class="card-title">Show Grocery Items</h4>

                                    <p class="card-text">Display all available grocery items.</p>

                                    <a routerLink="/all-items" role="button" class="btn btn-primary btn-block"><b>Show

                                            Items</b></a>

                                </div>

                            </div>

                            <div class="card bg-success">

                                <div class="card-body">

                                    <h4 class="card-title">Show Orders</h4>

                                    <p class="card-text">Display all orders created by users.</p>

                                    <a routerLink="/all-orders" role="button" class="btn btn-primary btn-block"><b>Show

                                            Orders</b></a>

                                </div>

                            </div>

                        </div>

                    </div>

                </div>

            </div>

        </div>

    </div>

</main>

1. **additem**

**File:** [**additem.component.ts**](http://home.component.ts)

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

import { AdminService } from 'src/app/Services/admin.service';

import { Item } from '../allitems/Items';

@Component({

  selector: 'app-additem',

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

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

})

export class AdditemComponent {

  item: Item = new Item();

  constructor(private adminService: AdminService) { }

  saveItem() {

    this.adminService.addItem(this.item).subscribe({

      next: (item) => {

        console.log("Added item");

      }, error: (errors) => {

        console.log(errors);

      }

    })

  }

  onSubmit() {

    if (this.item.name == null || this.item.brand == null || this.item.origin == null || this.item.price == null || this.item.quantity == null || this.item.pid == null || this.item.category == null || this.item.ingredient == null) {

      alert("Please Enter Complete Details!")

    } else {

      console.log(this.item);

      this.saveItem()

      alert("Item Added Successfully!")

    }

  }

}

**File: additem.component.html**

<main>

    <div class="container">

        <div class="row">

            <div class="col-md-8 mx-auto">

                <div class="card mt-5 bg-light">

                    <div class="card-body">

                        <h3 class="card-title text-center">Add Item Details:</h3>

                        <hr>

                        <form (ngSubmit)="onSubmit()">

                            <div class="form-row">

                                <div class="form-group col-md-6">

                                    <label for="pid">Product Id</label> <input type="text" class="form-control"

                                        [(ngModel)]="item.pid" name="pid" placeholder="Enter product id" required>

                                </div>

                                <div class="form-group col-md-6">

                                    <label for="pname">Product name</label> <input type="text" class="form-control"

                                        [(ngModel)]="item.name" name="name" placeholder="Enter product name" required>

                                </div>

                                <div class="form-group col-md-6">

                                    <label for="type">Category</label> <select [(ngModel)]="item.category"

                                        name="category" class="form-control">

                                        <option selected>Choose..</option>

                                        <option value="Fruits & Vegetables">Fruits & Vegetables</option>

                                        <option value="Dairy & Bakery">Dairy & Bakery</option>

                                        <option value="Staples">Staples</option>

                                        <option value="Snacks">Snacks</option>

                                        <option value="Beverages">Beverages</option>

                                        <option value="Personal Care">Personal Care</option>

                                    </select>

                                </div>

                                <div class="form-group col-md-6">

                                    <label for="ing">Ingredient</label><select [(ngModel)]="item.ingredient"

                                        name="ingredient" class="form-control">

                                        <option selected>Choose..</option>

                                        <option value="Vegetarian">Vegetarian</option>

                                        <option value="Non-Vegetarian">Non-Vegetarian</option>

                                    </select>

                                </div>

                                <div class="form-group col-md-6">

                                    <label for="price">Price(in Rs.)</label> <input type="number" class="form-control"

                                        [(ngModel)]="item.price" name="price" placeholder="Enter product M.R.P. in Rs."

                                        required>

                                </div>

                                <div class="form-group col-md-6">

                                    <label for="manft">Manufacturer</label> <input type="text" class="form-control"

                                        [(ngModel)]="item.brand" name="brand" placeholder="Enter product manufacturer"

                                        required>

                                </div>

                                <div class="form-group col-md-6">

                                    <label for="quant">Product Quantity</label> <input type="text" class="form-control"

                                        [(ngModel)]="item.quantity" name="quantity" placeholder="Enter product quantity"

                                        required>

                                </div>

                                <div class="form-group col-md-6">

                                    <label for="origin">Country Of Origin</label> <input type="text"

                                        [(ngModel)]="item.origin" class="form-control" name="origin"

                                        placeholder="Enter product country of origin" required>

                                </div>

                            </div>

                            <div class="text-center">

                                <div class="btn-group" role="group" aria-label="Basic example">

                                    <button type="submit" class="btn btn-primary btn-lg">Add Item</button>

                                    <button type="reset" class="btn btn-danger btn-lg">Reset</button>

                                </div>

                            </div>

                        </form>

                    </div>

                </div>

            </div>

        </div>

    </div>

</main>

1. **allitems**

**File: Items.ts**

export class Item {

    name!: string;

    pid!: string;

    category!: string;

    ingredient!: string;

    price!: number;

    brand!: string;

    quantity!: string;

    origin!: string;

}

**File:** [**allitems.component.ts**](http://home.component.ts)

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

import { AdminService } from 'src/app/Services/admin.service';

import { Item } from './Items';

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

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

@Component({

  selector: 'app-allitems',

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

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

})

export class AllitemsComponent implements OnInit {

  allItems!: Item[];

  constructor(private adminService: AdminService, private router: Router) {

  }

  ngOnInit(): void {

    this.getItemsList()

  }

  getItemsList() {

    this.adminService.getItems().subscribe({

      next: (items) => {

        this.allItems = items;

      }, error: (errors) => {

        console.log(errors);

        alert("No items found!");

      }

    })

  }

  updateItem(name: string) {

    let url: string = "/update-item/" + name;

    this.router.navigateByUrl(url);

  }

  deleteItem(name: string) {

    this.adminService.deleteItem(name).subscribe({

      next: (data) => {

        this.getItemsList();

      }, error: (error) => { console.log(error) }

    })

  }

}

**File: allitems.component.html**

<br>

<main>

    <div class="container">

        <div class="row">

            <div class="col">

                <h2 class="text-center text-white bg-dark">All Grocery Items</h2>

                <div class="table-responsive">

                    <table class="table table-dark table-striped table-bordered table-hover">

                        <thead class="thead-dark text-center">

                            <tr>

                                <th>S.No.</th>

                                <th>Product Id</th>

                                <th>Product Name</th>

                                <th>Category</th>

                                <th>Ingredient</th>

                                <th>Price(Rs.)</th>

                                <th>Manufacturer</th>

                                <th>Quantity</th>

                                <th>Country Origin</th>

                                <th>Action</th>

                            </tr>

                        </thead>

                        <tbody class="text-center">

                            <tr \*ngFor="let item of allItems">

                                <td></td>

                                <td><span>{{item.pid}}</span></td>

                                <td><span>{{item.name}}</span></td>

                                <td><span>{{item.category}}</span></td>

                                <td><span>{{item.ingredient}}</span></td>

                                <td><span>{{item.price}}</span></td>

                                <td><span>{{item.brand}}</span></td>

                                <td><span>{{item.quantity}}</span></td>

                                <td><span>{{item.origin}}</span></td>

                                <td>

                                    <div class="btn-group" role="group" aria-label="Basic example">

                                        <button (click)="deleteItem(item.name)" class="btn btn-danger">Delete</button>

                                        <button (click)="updateItem(item.name)" class="btn btn-info">Update</button>

                                    </div>

                                </td>

                            </tr>

                        </tbody>

                    </table>

                </div>

            </div>

        </div>

    </div>

</main>

1. **update-item**

**File:** [**update-item.component.ts**](http://home.component.ts)

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

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

import { Item } from '../allitems/Items';

import { AdminService } from 'src/app/Services/admin.service';

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

@Component({

  selector: 'app-update-item',

  templateUrl: './update-item.component.html',

  styleUrls: ['./update-item.component.css']

})

export class UpdateItemComponent implements OnInit {

  name!: string;

  item: Item = new Item();

  constructor(private adminService: AdminService, private route: ActivatedRoute) { }

  ngOnInit(): void {

    this.name = this.route.snapshot.params['name'];

    this.adminService.getItemByName(this.name).subscribe({

      next: (data) => {

        this.item = data;

      }, error: (error) => {

        console.log(error);

      }

    })

  }

  onSubmit() {

    this.adminService.updateItem(this.name, this.item).subscribe({

      next: (data) => {

        console.log(data);

        alert("Item updated successfully!");

      }, error: (error) => {

        console.log(error);

      }

    })

  }

}

**File: update-item.component.html**

<main>

    <div class="container">

        <div class="row">

            <div class="col-md-8 mx-auto">

                <div class="card mt-5 bg-light">

                    <div class="card-body">

                        <h3 class="card-title text-center">Update Grocery Item Details:</h3>

                        <hr>

                        <form (ngSubmit)="onSubmit()">

                            <div class="form-row">

                                <div class="form-group col-md-6">

                                    <label for="pid">Product Id</label> <input type="text" class="form-control"

                                        [(ngModel)]="item.pid" name="pid" placeholder="Enter product id" required>

                                </div>

                                <div class="form-group col-md-6">

                                    <label for="pname">Product name</label> <input type="text" class="form-control"

                                        [(ngModel)]="item.name" name="name" placeholder="Enter product name" required>

                                </div>

                                <div class="form-group col-md-6">

                                    <label for="type">Category</label> <select [(ngModel)]="item.category"

                                        name="category" class="form-control">

                                        <option selected>Choose..</option>

                                        <option value="Fruits & Vegetables">Fruits & Vegetables</option>

                                        <option value="Dairy & Bakery">Dairy & Bakery</option>

                                        <option value="Staples">Staples</option>

                                        <option value="Snacks">Snacks</option>

                                        <option value="Beverages">Beverages</option>

                                        <option value="Personal Care">Personal Care</option>

                                    </select>

                                </div>

                                <div class="form-group col-md-6">

                                    <label for="ing">Ingredient</label><select [(ngModel)]="item.ingredient"

                                        name="ingredient" class="form-control">

                                        <option selected>Choose..</option>

                                        <option value="Vegetarian">Vegetarian</option>

                                        <option value="Non-Vegetarian">Non-Vegetarian</option>

                                    </select>

                                </div>

                                <div class="form-group col-md-6">

                                    <label for="price">Price(in Rs.)</label> <input type="number" class="form-control"

                                        [(ngModel)]="item.price" name="price" placeholder="Enter product M.R.P. in Rs."

                                        required>

                                </div>

                                <div class="form-group col-md-6">

                                    <label for="manft">Manufacturer</label> <input type="text" class="form-control"

                                        [(ngModel)]="item.brand" name="brand" placeholder="Enter product manufacturer"

                                        required>

                                </div>

                                <div class="form-group col-md-6">

                                    <label for="quant">Product Quantity</label> <input type="text" class="form-control"

                                        [(ngModel)]="item.quantity" name="quantity" placeholder="Enter product quantity"

                                        required>

                                </div>

                                <div class="form-group col-md-6">

                                    <label for="origin">Country Of Origin</label> <input type="text"

                                        [(ngModel)]="item.origin" class="form-control" name="origin"

                                        placeholder="Enter product country of origin" required>

                                </div>

                            </div>

                            <div class="text-center">

                                <div class="btn-group" role="group" aria-label="Basic example">

                                    <button type="submit" class="btn btn-primary btn-lg">Update Item</button>

                                    <button type="reset" class="btn btn-danger btn-lg">Reset</button>

                                </div>

                            </div>

                        </form>

                    </div>

                </div>

            </div>

        </div>

    </div>

</main>

1. **all-orders**

**File:** [**all-orders.component.ts**](http://home.component.ts)

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

import { UserService } from 'src/app/Services/user.service';

import { Order } from '../place-order/order';

@Component({

  selector: 'app-all-orders',

  templateUrl: './all-orders.component.html',

  styleUrls: ['./all-orders.component.css']

})

export class AllOrdersComponent implements OnInit {

  orders!: Order[];

  constructor(private userService: UserService) { }

  ngOnInit(): void {

    this.getAllOrders()

  }

  getAllOrders() {

    this.userService.getAllOrders().subscribe({

      next: (data) => {

        this.orders = data

      }, error: (error) => {

        alert('No orders found!')

        console.log(error);

      }

    })

  }

  deleteOrder(id: string) {

    this.userService.deleteOrder(id).subscribe({

      next: (data) => {

        console.log('Successfully deleted');

        this.getAllOrders();

      }, error: (error) => {

        console.log(error);

      }

    })

  }

}

**File: all-orders.component.html**

<br>

<main>

    <div class="container">

        <div class="row">

            <div class="col">

                <h2 class="text-center text-white bg-dark">All Orders</h2>

                <div class="table-responsive">

                    <table class="table table-dark table-striped table-bordered table-hover">

                        <thead class="thead-dark text-center">

                            <tr>

                                <th>Order Id</th>

                                <th>Date</th>

                                <th>Customer Name</th>

                                <th>Address</th>

                                <th>State</th>

                                <th>Contact</th>

                                <th>Quantity</th>

                                <th>Payment</th>

                                <th>Product(Id)</th>

                                <th>Weight</th>

                                <th>Action</th>

                            </tr>

                        </thead>

                        <tbody class="text-center">

                            <tr \*ngFor="let order of orders">

                                <td><span>{{order.id}}</span></td>

                                <td><span>{{order.od}}</span></td>

                                <td><span>{{order.name}}</span></td>

                                <td><span>{{order.address}}</span></td>

                                <td><span>{{order.state}}</span></td>

                                <td><span>{{order.contact}}</span></td>

                                <td><span>{{order.quantity}}</span></td>

                                <td><span>{{order.payment}}</span></td>

                                <td><span>{{order.product.name}} ({{order.product.pid}})</span></td>

                                <td><span>{{order.product.quantity}}</span></td>

                                <td>

                                    <button (click)="deleteOrder(order.id)" class="btn btn-danger">Delete</button>

                                </td>

                            </tr>

                        </tbody>

                    </table>

                </div>

            </div>

        </div>

    </div>

</main>

**File:**  **app-routing.module.ts**

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

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

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

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

import { DashboardComponent } from './Components/dashboard/dashboard.component';

import { ChangepasswordComponent } from './Components/changepassword/changepassword.component';

import { AllitemsComponent } from './Components/allitems/allitems.component';

import { AdditemComponent } from './Components/additem/additem.component';

import { AuthGuard } from './Services/auth.guard';

import { UpdateItemComponent } from './Components/update-item/update-item.component';

import { ProductSearchComponent } from './Components/product-search/product-search.component';

import { PlaceOrderComponent } from './Components/place-order/place-order.component';

import { OrderConfirmComponent } from './Components/order-confirm/order-confirm.component';

import { AllOrdersComponent } from './Components/all-orders/all-orders.component';

const routes: Routes = [

  { path: '', component: HomeComponent, pathMatch: 'full' },

  { path: 'login', component: LoginComponent, pathMatch: 'full' },

  { path: 'dashboard', component: DashboardComponent, pathMatch: 'full', canActivate: [AuthGuard] },

  { path: 'change-password', component: ChangepasswordComponent, pathMatch: 'full' },

  { path: 'all-items', component: AllitemsComponent, pathMatch: 'full', canActivate: [AuthGuard] },

  { path: 'add-item', component: AdditemComponent, pathMatch: 'full', canActivate: [AuthGuard] },

  { path: 'update-item/:name', component: UpdateItemComponent, pathMatch: 'full', canActivate: [AuthGuard] },

  { path: 'product-search/:name', component: ProductSearchComponent, pathMatch: 'full' },

  { path: 'place-order/:name', component: PlaceOrderComponent, pathMatch: 'full' },

  { path: 'order-confirm/:id', component: OrderConfirmComponent, pathMatch: 'full' },

  { path: 'all-orders', component: AllOrdersComponent, pathMatch: 'full', canActivate: [AuthGuard] }

]

  ;

@NgModule({

  imports: [RouterModule.forRoot(routes)],

  exports: [RouterModule]

})

export class AppRoutingModule { }

**File:**  **app.component.html**

<app-navbar></app-navbar>

<router-outlet></router-outlet>

**==END==**