```
package com.restaurant.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.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.RestController;

import com.restaurant.exceptions.CartOperationException;
import com.restaurant.exceptions.ItemNotFoundException;
import com.restaurant.exceptions.UserNotFoundException;
import com.restaurant.pojo.CartItem;
import com.restaurant.pojo.FoodItem;
import com.restaurant.service.CartService;
import com.restaurant.service.FoodItemService;

@RestController
@CrossOrigin(origins = "*")
@RequestMapping("/cart")
public class CartController {
```

```
@Autowired
CartService cartService;
@Autowired
FoodItemService foodItemService;
@PostMapping("/add")
public ResponseEntity<String> addItemToCart(@RequestParam("userId") Long userId,
@RequestParam("foodItemId") Long foodItemId, @RequestParam("quantity") int quantity) {
CartItem cartItem = new CartItem();
cartItem.setQuantity(quantity);
try {
cartService.addItemToCart(userId, foodItemId, cartItem);
return ResponseEntity.ok("Item added to cart successfully");
} catch (ItemNotFoundException e) {
return ResponseEntity.status(HttpStatus.NOT_FOUND).body("Food item Not Found!");
} catch (UserNotFoundException e) {
return ResponseEntity.status(HttpStatus.NOT_FOUND)
.body("Oops, something went wrong! Please sign in again!");

} catch (CartOperationException e) {
return ResponseEntity.status(HttpStatus.BAD_REQUEST)
.body("Oops, something went wrong! Please add to cart again");
}
}

@PutMapping("/update/{cartId}/{newQuantity}")
public String updateCartItem(@PathVariable Long cartId,@PathVariable int newQuantity) {
CartItem oldCartItem=cartService.getCartItemById(cartId);
int differenceInQuantity = newQuantity-oldCartItem.getQuantity();
System.out.println(differenceInQuantity);
```

```
FoodItem food=oldCartItem.getFoodItem();
food.setAvailableQuantity(food.getAvailableQuantity()-differenceInQuantity);
foodItemService.editFoodItem(food);
System.out.println(food.getAvailableQuantity());
if(cartService.updateCartItem(cartId,newQuantity)!=null)
return "done";
return "0";
}

@GetMapping("/{userId}")
public List<CartItem> getWholeCart(@PathVariable Long userId) {
return cartService.getWholeCartByUserId(userId);
}

@DeleteMapping("/deleteFromCartOnly/{cartId}")
public String deleteFromCart(@PathVariable Long cartId) {
CartItem oldCartItem=cartService.getCartItemById(cartId);
int quantityToBeRestored = oldCartItem.getQuantity();
FoodItem food=oldCartItem.getFoodItem();
food.setAvailableQuantity(food.getAvailableQuantity()+quantityToBeRestored);
foodItemService.editFoodItem(food);
cartService.deleteFromCart(cartId);
return "done";

}

@DeleteMapping("/deleteCart/{userId}")
public String deleteCartAfterPayment(@PathVariable Long userId) {
cartService.deleteFromCartByUserId(userId);
```

```
return "done";

}
}



package com.restaurant.pojo;

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.OneToOne;

import lombok.Data;

@Entity

public class CartItem {
   @Id
   @GeneratedValue(strategy = GenerationType.IDENTITY)
   private Long cartItemId;
   @OneToOne
   @JoinColumn(name = "food_item_id")
   private FoodItem foodItem;
```

```
    private int quantity;

    private double totalFoodItemCost;


    private Long userId;


public Long getCartItemId() {

return cartItemId;

}


public void setCartItemId(Long cartItemId) {

this.cartItemId = cartItemId;

}


public FoodItem getFoodItem() {

return foodItem;

}


public void setFoodItem(FoodItem foodItem) {

this.foodItem = foodItem;

}


public int getQuantity() {

return quantity;

}


public void setQuantity(int quantity) {

this.quantity = quantity;

}
```

```
public double getTotalFoodItemCost() {

return totalFoodItemCost;

}


public void setTotalFoodItemCost(double totalFoodItemCost) {

this.totalFoodItemCost = totalFoodItemCost;

}


public Long getUserId() {

return userId;

}


public void setUserId(Long userId) {

this.userId = userId;

}


@Override

public String toString() {

return "CartItem [cartItemId=" + cartItemId + ", foodItem=" + foodItem + ", quantity=" + quantity

+ ", totalFoodItemCost=" + totalFoodItemCost + ", userId=" + userId + "]";

}


public CartItem(Long cartItemId, FoodItem foodItem, int quantity, double totalFoodItemCost, Long
userId) {

super();

this.cartItemId = cartItemId;

this.foodItem = foodItem;

this.quantity = quantity;

this.totalFoodItemCost = totalFoodItemCost;
this.userId = userId;

}


public CartItem() {

super();

// TODO Auto-generated constructor stub

}


}




package com.restaurant.repo;


import java.util.List;


import org.springframework.data.jpa.repository.JpaRepository;


import com.restaurant.pojo.CartItem;

import com.restaurant.pojo.FoodItem;


public interface CartRepo extends JpaRepository<CartItem, Long> {


public List<CartItem> findAllByUserId(Long userId);


public CartItem findByUserIdAndFoodItem(Long userId, FoodItem foodItem);


public List<CartItem> findByFoodItem(FoodItem foodItem);


public List<CartItem> findByUserId(Long userId);
```

```
}



package com.restaurant.service;

import java.util.List;

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

import com.restaurant.exceptions.CartOperationException;
import com.restaurant.exceptions.ItemNotFoundException;
import com.restaurant.exceptions.UserNotFoundException;
import com.restaurant.pojo.CartItem;
import com.restaurant.pojo.FoodItem;
import com.restaurant.pojo.User;
import com.restaurant.repo.AdminRepo;
import com.restaurant.repo.CartRepo;
import com.restaurant.repo.CategoryRepo;
import com.restaurant.repo.FoodItemRepo;
import com.restaurant.repo.UserRepo;

@Service
public class CartService {

@Autowired
CartRepo cartRepo;
```

```
@Autowired

UserRepo userRepo;

@Autowired

FoodItemRepo foodItemRepo;

@Autowired

CategoryRepo categoryRepo;

@Autowired

AdminRepo adminRepo;


public void addItemToCart(Long userId, Long foodItemId, CartItem cartItem) {

FoodItem foodItem = foodItemRepo.findById(foodItemId).orElse(null);

if (foodItem == null)

throw new ItemNotFoundException("Item Not Found");

User user = userRepo.findById(userId).orElse(null);

if (user == null)

throw new UserNotFoundException("User not found");


CartItem existingCartItem = cartRepo.findByUserIdAndFoodItem(userId, foodItem);

if (existingCartItem != null) {

existingCartItem.setQuantity(existingCartItem.getQuantity() + cartItem.getQuantity());

existingCartItem.setTotalFoodItemCost(foodItem.getDiscountedPrice() *
existingCartItem.getQuantity());

cartRepo.save(existingCartItem);

}


else {

cartItem.setFoodItem(foodItem);

cartItem.setUserId(userId);

cartItem.setTotalFoodItemCost(foodItem.getDiscountedPrice() * cartItem.getQuantity());
CartItem cart = cartRepo.save(cartItem);
if (cart == null) {
throw new CartOperationException("Failed to add Item");
}
}
}

public CartItem updateCartItem(Long cartItemId, int newQuantity) {
CartItem cartItem = cartRepo.findById(cartItemId).orElse(null);
cartItem.setQuantity(newQuantity);
if (cartItem.getQuantity() == 0) {
cartRepo.deleteById(cartItemId);
return null;
}
cartItem.setTotalFoodItemCost(cartItem.getFoodItem().getDiscountedPrice() * cartItem.getQuantity());
return cartRepo.save(cartItem);
}

public List<CartItem> getWholeCartByUserId(Long userId) {
return cartRepo.findAllByUserId(userId);
}

public CartItem getCartItemById(Long cartId) {
return cartRepo.findById(cartId).orElse(null);
}

public void deleteFromCart(Long cartId) {
cartRepo.deleteById(cartId);
}
```

```
public void deleteFromCartByFoodItems(List<FoodItem> foodItems) {
for (FoodItem foodItem : foodItems) {
List<CartItem> cartItems = cartRepo.findByFoodItem(foodItem);
for (CartItem cartItem : cartItems) {
if (cartItem != null)
cartRepo.delete(cartItem);
}
}
}

public void deleteFromCartByUserId(Long userId) {
List<CartItem> cartItems=cartRepo.findByUserId(userId);
for (CartItem cartItem : cartItems) {
if (cartItem != null)
cartRepo.delete(cartItem);
}


}

// =====================================================================//


}
```