August 6

Capston 1

— Extra End Point —

Nora Alsultan

Creat aGift Card Method to User if Bought With Specific Price he get a Gift Card

Method get user id to check if he in the Data bade and a price .

Gift Card User Service

```java
// Costumer Have Gift when buy with determine price
public String giftCard(String userId, double productPrice) {  1 usage

    User user = getUserById(userId);

    if (user == null) {
        return "User not found";
    }

    double totalSpent = user.getBalance() - productPrice;

    if (totalSpent >= 1000) {
        return "Congratulations, you have a gift card with 600 Riyal!";
    } else if (totalSpent >= 800) {
        return "Congratulations, you have a gift card with 500 Riyal!";
    } else if (totalSpent >= 500) {
        return "Congratulations, you have a gift card with 200 Riyal!";
    } else {
        return "You have not spent enough to qualify for a gift card.";
    }
}
```

Gift Card User Controller

```java
@GetMapping("/card/{userId}/{productPrice}")
public ResponseEntity gift(@PathVariable String userId, @PathVariable double productPrice) {
    String giftMessage = userService.giftCard(userId, productPrice);

    if (giftMessage.equals("You have not spent enough to qualify for a gift card.")) {
        return ResponseEntity.status(400).body(new ApiResponse("Gift Card Empty"));
    }
    return ResponseEntity.status(200).body(new ApiResponse(giftMessage));
}
```

Creat Method User can add Product to Wish List

AddToWishList ProductService Class

```java
public String addProductToWishList(String userId, String productId) {  1 usage

    User user = userService.getUserById(userId); // check if user id found
    if (user == null) {
        return " User Id not found ";
    }

    Product product = getProductById(productId); // check if product id is found
    if (product == null) {
        return " Product Id not found ";
    }

    if (WishList.contains(product)) {
        return "Product is already in your wishlist"; // Product already exists in the wishlist
    }

    WishList.add(product); // Add Product to wish List
    user.setScore(user.getScore() + 1); // Increase user's score // when user added product to wish list , Score is increase
    return " Product added successfully , You have One Point Score !! ";
}
```

AddToWishList ProductController Class

```java
@PostMapping("/wishlist/{productId}/{userId}")//Add Product to WishList
public ResponseEntity wishlist( @PathVariable String productId , @PathVariable String userId ) {

    if (productService.addProductToWishList(productId,userId) != null){
        return ResponseEntity.status(200).body(new ApiResponse(" Added to Wishlist Successfully "));
    }
    return ResponseEntity.status(400).body(new ApiResponse("Sorry bad requist Check if product id  , or username is wrong"));
}
```

User can get His Wish List

GetWishList ProductService Class

```java
// Complete : first check if user found , then User want to get his WishList
public ArrayList<Product> getWishList(String userId) {  1 usage
    User user = userService.getUserById(userId);
    if (user == null) {
        return null;
    }
    return WishList;
}
```

GetWishList ProductController Class

```java
@GetMapping("/get-wishlist/{userId}") // get wishlist of the user logged in
public ResponseEntity getWishlist( @PathVariable String userId ) {
    List<Product> wishlist = productService.getWishList(userId);

    if (wishlist == null) {
        return ResponseEntity.status(400).body(new ApiResponse("User Not Found"));
    }
    return ResponseEntity.status(200).body(wishlist);
}
```

Creat Method if User have Problem and need to Connect To Support Team

Method get name and description

UserService Class

```java
// Costumer Have Problem and need to connect with support team
public String connectToSupportCenter(String Name, String issueDescription) {  2 usages

    String phoneNumberSupport = "0509875764";

    if (Name != null && !Name.isEmpty()) {
        return "Please contact our support team at phone number: " + phoneNumberSupport;
    }
    return "Error: User name is missing. Please provide a valid user name.";
}
```

UserController Class

```java
@GetMapping("/support-team/{username}/{description}")
public ResponseEntity supportTeam(@PathVariable String username, @PathVariable String description) {

   if(userService.connectToSupportCenter(username,description)== null){

       return ResponseEntity.status(400).body(new ApiResponse("Sorry, User Not Found"));
   }
   return ResponseEntity.status(200).body(userService.connectToSupportCenter(username,description));
}
```

Log In With phone number and get Verification Cod

UserService Class

```java
// user log in with phone number and get a verification Code
public String login(String phoneNumber) {  1 usage

    for (User user : users) {
        if (user.getPhoneNumber().equals(phoneNumber)) {
            int verificationCode = (int) (Math.random() * 9000) + 1000; // Generate a 4-digit random verification code
            // Send verification code logic here
            return "Verification code: " + verificationCode;
        }
    }
    return "Phone number is not correct";
}
```

UserController Class

```java
@GetMapping("/log-in/{phoneNumber}") // User login with PhoneNumber and get a verification Code
public ResponseEntity logIn(@PathVariable String phoneNumber) {
    String verificationResult = userService.login(phoneNumber);
    if (verificationResult.equals("Phone number is not correct")) {
        return ResponseEntity.status(400).body(new ApiResponse("Sorry, User Not Found"));
    }
    return ResponseEntity.status(200).body(new ApiResponse(verificationResult));
}
```

```
1   Please contact our support team at phone number: 0509875764
```

Creat Tracking ShipmentMethod get User Id to check if User Have account , and check if password Correct , write Keyword Track

Generet list of order class to get product

Creat method to generate a random number called Order Number

Track Shipment Product Service Class

```
public String trackOrderMethod(String userId, String password, String track) {  1 usage

    User user = userService.getUserById(userId);
    if (user == null) {
        return "User not found";
    }

    boolean checkPassword = password.equalsIgnoreCase(user.getPassword());
    if (!checkPassword) {
        return "Incorrect password";
    }

    if (track == null || !track.equalsIgnoreCase( anotherString: "track")) {
        return "Please write 'track' correctly to proceed";
    }

    Orders latestOrder = ordersService.getLatestOrder();
    if (latestOrder == null) {
        return "No orders available to track";
    }

    Product boughtProduct = getProductById(latestOrder.getProductId());
    if (boughtProduct == null) {
        return "Product details not found for the latest order";
    }

    int orderNumber = OrdersService.generateOrderNumber();
    LocalDate expectedDeliveryDate = LocalDate.now().plusDays( daysToAdd: 5);

    return "Track number: " + orderNumber + ", Expected Arrival Date: " + expectedDeliveryDate;
}
```

Track Shipment Product Controller Class

```
@GetMapping("/track-order/{userId}/{password}/{track}")
public ResponseEntity trackOrder(@PathVariable String userId, @PathVariable String password, @PathVariable String track) {
    String orderStatus = productService.trackOrderMethod(userId, password, track);

    if (orderStatus.equals("User not found") || orderStatus.equals("Incorrect password") ||
            orderStatus.equals("Please write 'track' correctly to proceed") ||orderStatus.equals("No orders available to track") ) {
        return ResponseEntity.status(400).body(orderStatus);
    }

    return ResponseEntity.status(200).body(orderStatus);
}
```