AMAZON CLONE

CAPSTONE 1

NAELAH ALSHIBANI

ENDPOINT 1

CONTROLLER

```java
@GetMapping("/in/stock/{productid}/{merchantid}")
public ResponseEntity isProductInStock(@PathVariable String productid, @PathVariable String merchantid) {
    if (merchantStockServices.isProductInStock(productid, merchantid)) {
        return ResponseEntity.status(200).body(new ApiResponse("Product Is In Stock"));
    }
    return ResponseEntity.status(400).body(new ApiResponse("Product Is Out Of Stock"));
}
```

SERVICE

```java
public boolean isProductInStock(String productID, String merchantID) {  2 usages
    //1.check both IDs exist
    for (int i = 0; i < productServices.getAllProducts().size(); i++)
        for (int j = 0; j < merchantServices.getMerchants().size(); j++)
            if (productServices.getAllProducts().get(i).getId().equals(productID)
                    && merchantServices.getMerchants().get(i).getId().equals(merchantID)) {
                //2.check if they are in the merchant stock
                for (MerchantStock ms : merchantStocks) {
                    if (ms.getProductID().equals(productID)
                            && ms.getMerchantID().equals(merchantID)) {
                        //3. check if product is in stock
                        if (ms.getStock() > 0) {
                            return true;
                        }
                    }
                }
            }

    return false;
}
```

ENDPOINT 2

CONTROLLER

```java
@PutMapping("/discount/{merchantid}/{productid}/{discountpercentage}")
public ResponseEntity applyDiscount(@PathVariable String merchantid, @PathVariable String productid,
                                    @PathVariable double discountpercentage) {
    int discountResult = productServices.applyDiscount(merchantid, productid, discountpercentage);
    if (discountResult == 1)
        return ResponseEntity.status(400).body(new ApiResponse("Only Merchant Can Apply Discount On Product"));
    else if (discountResult == 2)
        return ResponseEntity.status(400).body(new ApiResponse("Product ID Not Found"));

    return ResponseEntity.status(200).body(new ApiResponse("Discount Applied Successfully"));
}
```

SERVICE

```java
public int applyDiscount(String merchantID, String productID, double discountPercentage) {  1 usage
    for (Merchant merchant : merchantServices.getMerchants()) {
        if (merchant.getId().equals(merchantID)) {
            for (Product product : products) {
                if (product.getId().equals(productID)) {
                    product.setPrice(product.getPrice() - (product.getPrice() * (discountPercentage / 100)));
                    return 3; // discount applied
                }
            }
            return 2; // product id incorrect
        }
    }
    return 1; //merchant id incorrect
}
```

ENDPOINT 3

CONTROLLER

```java
@GetMapping("/same/product/category/{categoryid}")
public ResponseEntity getProductsByCategory(@PathVariable String categoryid) {
    ArrayList<Product> sameCategoryProducts = productServices.getProductsByCategory(categoryid);
    if (sameCategoryProducts.isEmpty())
        return ResponseEntity.status(400).body(new ApiResponse("Category ID Not Found"));

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

SERVICE

```java
public ArrayList<Product> getProductsByCategory(String categoryID) {  1 usage
    ArrayList<Product> sameProductCategory = new ArrayList<>();
    for (Product product : products) {
        if (product.getCategoryID().equals(categoryID)) {
            sameProductCategory.add(product);
        }
    }
    return sameProductCategory;
}
```

ENDPOINT 4

CONTROLLER

```java
@PutMapping("/update/price/{productid}/{merchantid}/{newprice}")
public ResponseEntity updateProductPrice(@PathVariable String productid, @PathVariable String merchantid,
                                         @PathVariable double newprice) {
    int updateProductPriceResult = merchantStockServices.updateProductPrice(productid, merchantid, newprice);
    if (updateProductPriceResult == 1)
        return ResponseEntity.status(400).body(new ApiResponse("Only Merchant Can Update Price"));
    else if (updateProductPriceResult == 2)
        return ResponseEntity.status(400).body(new ApiResponse("Product ID Not Found In Stock"));

    return ResponseEntity.status(200).body(new ApiResponse("Product Price Updated Successfully"));
}
```

SERVICE

```java
public int updateProductPrice(String productID, String merchantID, double newPrice) {  1 usage
    //check correct merchant id
    if (!isValidMerchantID(merchantID))
        return 1; //invalid merchant id

    for (MerchantStock ms : merchantStocks) {
        if (ms.getProductID().equals(productID) && ms.getMerchantID().equals(merchantID)) {
            for (Product product : productServices.getAllProducts()) {
                if (product.getId().equals(productID)) {
                    product.setPrice(newPrice);
                    return 0;
                }
            }
        }
    }
    return 2; //product id not in stock
}
```

ENDPOINT 5

CONTROLLER

```
@PostMapping("/add/review/{productid}/{userid}/{review}")
public ResponseEntity addReview(@PathVariable String productid, @PathVariable String userid, @PathVariable String review) {
    int reviewResult = productServices.addReview(productid, userid, review);
    if (reviewResult == 1)
        return ResponseEntity.status(400).body(new ApiResponse("Product ID Not Found"));
    else if (reviewResult == 2)
        return ResponseEntity.status(400).body(new ApiResponse("User ID Not Found"));
    else if (reviewResult == 3)
        return ResponseEntity.status(400).body(new ApiResponse("Review Must Be Longer Than 5 Letters"));

    return ResponseEntity.status(200).body(new ApiResponse("Review Added Successfully"));

}
```

SERVICE

```
public int addReview(String productID, String userID, String review) {
    //check product id
    for (Product product : products) {
        if (product.getId().equals(productID)) {
            // check user id
            for (User user : userService.getUsers()) {
                if (user.getId().equals(userID)) {
                    if (review.length() >= 5) {
                        // empty list issue
                        if (product.getReviews() == null) {
                            product.setReviews(new ArrayList<>());
                        }
                        product.getReviews().add(review);
                        return 0;
                    }
                    return 3; // review length must be longer than 5
                }

            }
            return 2; //user id not found
        }
    }
    return 1; //product id not found
}
```

ENDPOINT 5

CONTROLLER

```java
@GetMapping("/reviews/{productid}")
public ResponseEntity getProductReviews(@PathVariable String productid) {
    ArrayList<String> reviews = productServices.getProductReviews(productid);
    if (reviews == null) {
        return ResponseEntity.status(400).body(new ApiResponse("Product ID Not Found"));
    }
    return ResponseEntity.status(200).body(reviews);
}
```

SERVICE

```java
public ArrayList<String> getProductReviews(String productID) {
    for (Product product : products) {
        if (product.getId().equals(productID)) {
            return product.getReviews(); // return  list
        }
    }
    return null; //  product not found
}
```