Project Overview and Components

1. Project Overview

Your system will allow customers to:

- View product catalog
- Add products to the cart
- Remove products from the cart
- View the cart
- Checkout and generate a receipt
- Store product and customer data using files

2. Project Components

a. Product Structure

```
typedef struct {
    int id;
    char name[50];
    float price;
    int stock;
} Product;
```

b. Cart Structure

```
typedef struct {
    int productId;
    int quantity;
} CartItem;
```

```
typedef struct {
```

```
    CartItem items[50];
    int itemCount;
} ShoppingCart;
```

3. Functionalities and Code Outline

a. Display Product Catalog

You can store products in a file (e.g., products.txt) and read them for display.

```
void displayProducts() {
    FILE *file = fopen("products.txt", "r");
    if (!file) {
        printf("Error: Could not open product file.\n");
        return;
    }

    Product product;
    printf("ID\tName\t\tPrice\tStock\n");
    printf("-----------------------------------\n");
     while (fscanf(file, "%d %s %f %d", &product.id, product.name, &product.price, &product.stock) !=
EOF) {
        printf("%d\t%s\t\t%.2f\t%d\n", product.id, product.name, product.price, product.stock);
    }
    fclose(file);
}
```

b. Add Product to Cart

```
void addToCart(ShoppingCart *cart, int productId, int quantity) {
    for (int i = 0; i < cart->itemCount; i++) {
```

```
        if (cart->items[i].productId == productId) {
            cart->items[i].quantity += quantity;
            printf("Quantity updated in cart.\n");
            return;
        }
    }
    cart->items[cart->itemCount].productId = productId;
    cart->items[cart->itemCount].quantity = quantity;
    cart->itemCount++;
    printf("Product added to cart.\n");
}
```

c. Remove Product from Cart

```
void removeFromCart(ShoppingCart *cart, int productId) {
    for (int i = 0; i < cart->itemCount; i++) {
        if (cart->items[i].productId == productId) {
            for (int j = i; j < cart->itemCount - 1; j++) {
                cart->items[j] = cart->items[j + 1];
            }
            cart->itemCount--;
            printf("Product removed from cart.\n");
            return;
        }
    }
    printf("Product not found in cart.\n");
}
```

d. View Cart

```
void viewCart(ShoppingCart *cart) {
    if (cart->itemCount == 0) {
        printf("Your cart is empty.\n");
        return;
    }

    printf("Product ID\tQuantity\n");
    printf("-------------------------\n");
    for (int i = 0; i < cart->itemCount; i++) {
        printf("%d\t\t%d\n", cart->items[i].productId, cart->items[i].quantity);
    }
}
```

e. Checkout and Generate Receipt

```
void checkout(ShoppingCart *cart) {
    FILE *file = fopen("products.txt", "r");
    if (!file) {
        printf("Error: Could not open product file.\n");
        return;
    }

    Product product;
    float total = 0;
    printf("\n--- Receipt ---\n");
    printf("Product\tQuantity\tPrice\tTotal\n");
```

```
    for (int i = 0; i < cart->itemCount; i++) {
        rewind(file);
        while (fscanf(file, "%d %s %f %d", &product.id, product.name, &product.price, &product.stock)
!= EOF) {
            if (product.id == cart->items[i].productId) {
                float itemTotal = cart->items[i].quantity * product.price;
                total += itemTotal;
                    printf("%s\t%d\t\t%.2f\t%.2f\n", product.name, cart->items[i].quantity, product.price,
itemTotal);
                break;
            }
        }
    }
    fclose(file);

    printf("\nTotal Amount: %.2f\n", total);
    printf("Thank you for shopping!\n");
}
```

4. Main Menu

```
int main() {
    ShoppingCart cart = {.itemCount = 0};
    int choice, productId, quantity;

    while (1) {
        printf("\n--- E-Commerce Shopping Cart ---\n");
        printf("1. View Products\n");
```

```
printf("2. Add to Cart\n");
printf("3. Remove from Cart\n");
printf("4. View Cart\n");
printf("5. Checkout\n");
printf("6. Exit\n");
printf("Enter your choice: ");
scanf("%d", &choice);

switch (choice) {
    case 1:
        displayProducts();
        break;
    case 2:
        printf("Enter Product ID: ");
        scanf("%d", &productId);
        printf("Enter Quantity: ");
        scanf("%d", &quantity);
        addToCart(&cart, productId, quantity);
        break;
    case 3:
        printf("Enter Product ID to remove: ");
        scanf("%d", &productId);
        removeFromCart(&cart, productId);
        break;
    case 4:
        viewCart(&cart);
        break;
```

```
            case 5:
                checkout(&cart);
                return 0;
            case 6:
                printf("Exiting... Thank you!\n");
                return 0;
            default:
                printf("Invalid choice. Try again.\n");
        }
    }
}
```

5. Products File (products.txt Format Example)

1 iPhone 999.99 10

2 Laptop 799.99 5

3 Headphones 199.99 15

4 Keyboard 49.99 20

5 Mouse 29.99 25

6. How to Compile and Run the Program

Save your code in a file named ecommerce.c.

Compile the program using:

gcc ecommerce.c -o ecommerce

Run the program using:

./ecommerce

7. Future Improvements (Optional)

- Add user authentication (login system).

- Allow updating product stock after each purchase.

- Implement error handling for file operations and inputs.

- Add sorting and searching options for products.

- Save cart state to a file for persistence.