This project demonstrates a simple Java-based e-commerce system that allows users to browse products, add items to a shopping cart, and place an order. The goal is to show how to structure and encapsulate code using Java packages and import statements, and to practice object-oriented programming principles such as encapsulation, modularity, and data abstraction.

### **Package Structure**

The project is organized into two main packages:

* com.ecommerce — contains core classes related to products and customers.
* com.ecommerce.orders — manages orders and order-related operations.

### **Class Descriptions**

#### **1. Product Class (com.ecommerce.Product)**

This class models a product in the online store. Each product has:

* itemCode – A unique identifier for the product
* itemName – The name or title of the product
* itemPrice – The cost of the product

It includes:

* A constructor to initialize the product
* Getters and setters for each attribute
* A toString() method to display product info neatly

**Purpose**: Enables product listing and easy retrieval of item details.

#### **2. Customer Class (com.ecommerce.Customer)**

This class represents a customer using the system. Each customer has:

* userId – A unique ID for the customer
* fullName – Customer’s full name
* shoppingBasket – A list (cart) holding selected products

Key methods include:

* addItemToBasket() – Adds a product to the cart
* removeItemFromBasket() – Removes a product from the cart
* calculateBasketTotal() – Computes total cost of items in the basket
* clearBasket() – Empties the shopping cart

**Purpose**: Handles shopping behaviors like adding/removing items and preparing to place orders.

#### **3. Order Class (com.ecommerce.orders.Order)**

This class models a customer’s purchase order. It contains:

* transactionId – A unique auto-generated ID for the order
* purchaser – A reference to the customer who placed the order
* purchasedItems – A list of all products in the order
* purchaseTotal – Total cost of the order
* currentStatus – Status of the order (default: “Processing”)

Key functionality:

* calculatePurchaseTotal() – Computes the total value of the order
* updateOrderStatus() – Changes the order status (e.g., Shipped, Cancelled)
* generateOrderSummary() – Prints a formatted summary of the order details

**Purpose**: Finalizes the transaction, summarizes the purchase, and tracks order status.

### **Main Program**

The Main.java class brings everything together. It demonstrates how a user would interact with the e-commerce system by:

1. Creating sample products
2. Creating a customer profile
3. Simulating product browsing and shopping
4. Adding items to a basket
5. Displaying the cart total
6. Creating an order using the selected items
7. Printing an order confirmation summary

### **Output Example**

The program displays:

* A list of available products
* Confirmation messages when items are added to the cart
* Total cost in the shopping basket
* A complete order summary, including:
  + Transaction ID
  + Customer name
  + Item list
  + Total cost
  + Order status

### **Conclusion**

This project successfully demonstrates the creation of a basic e-commerce application in Java. It applies best practices such as organized package structures, object-oriented principles, and user-friendly methods. This forms a strong foundation for more advanced systems like inventory tracking, user authentication, and payment processing.