CHAPTER - 1

INTRODUCTION

In today's digital age, e-commerce has become an integral part of our lives, revolutionizing the way we shop and conduct business. The convenience of online shopping platforms allows consumers to browse, select, and purchase products from the comfort of their homes. This project aims to develop a Java-based e-commerce shopping application that simulates the functionalities of a typical online shopping platform. The application is designed to provide users with a seamless shopping experience, offering a variety of products categorized under Fashion, Electronics, and Mobiles.

The primary objective of this project is to create a user-friendly console application that allows users to browse different categories and subcategories of products, add items to their shopping cart, view the contents of their cart, remove items from the cart, and generate a bill that includes the customer's name and total purchase amount. This project not only demonstrates the core functionalities of an e-commerce platform but also emphasizes best practices in software development, including modularity, maintainability, and security. By leveraging the Java programming language and adhering to a structured development process, the project aims to deliver a robust and efficient application that meets the needs of modern-day online shoppers.

Objective

The Proposed system aims to develop an interactive and dynamic Ecommerce shopping website with the help of Core java concepts

CHAPTER – 2
PLANNING

The project aims to develop a Java-based e-commerce shopping application with categories such as Fashion, Electronics, and Mobiles. The application will allow users to browse products, add them to their cart, view the cart, remove items from the cart, and generate a bill that includes the customer's name and total purchase amount.

This application must ensure high usability, providing a user-friendly console interface with intuitive navigation through categories and subcategories. Performance is critical; the system should respond quickly to user inputs and handle at least 100 products without any performance degradation. Maintainability is also essential; the code should be modular, well-documented, and easy to extend with additional features in the future. Reliability is a key consideration; the application should handle invalid inputs gracefully and ensure consistent data in the shopping cart and bill generation processes.

Functional Requirements

- Category Browsing
- Product Viewing
- Cart management
- Billing

Non-Functional Requirements

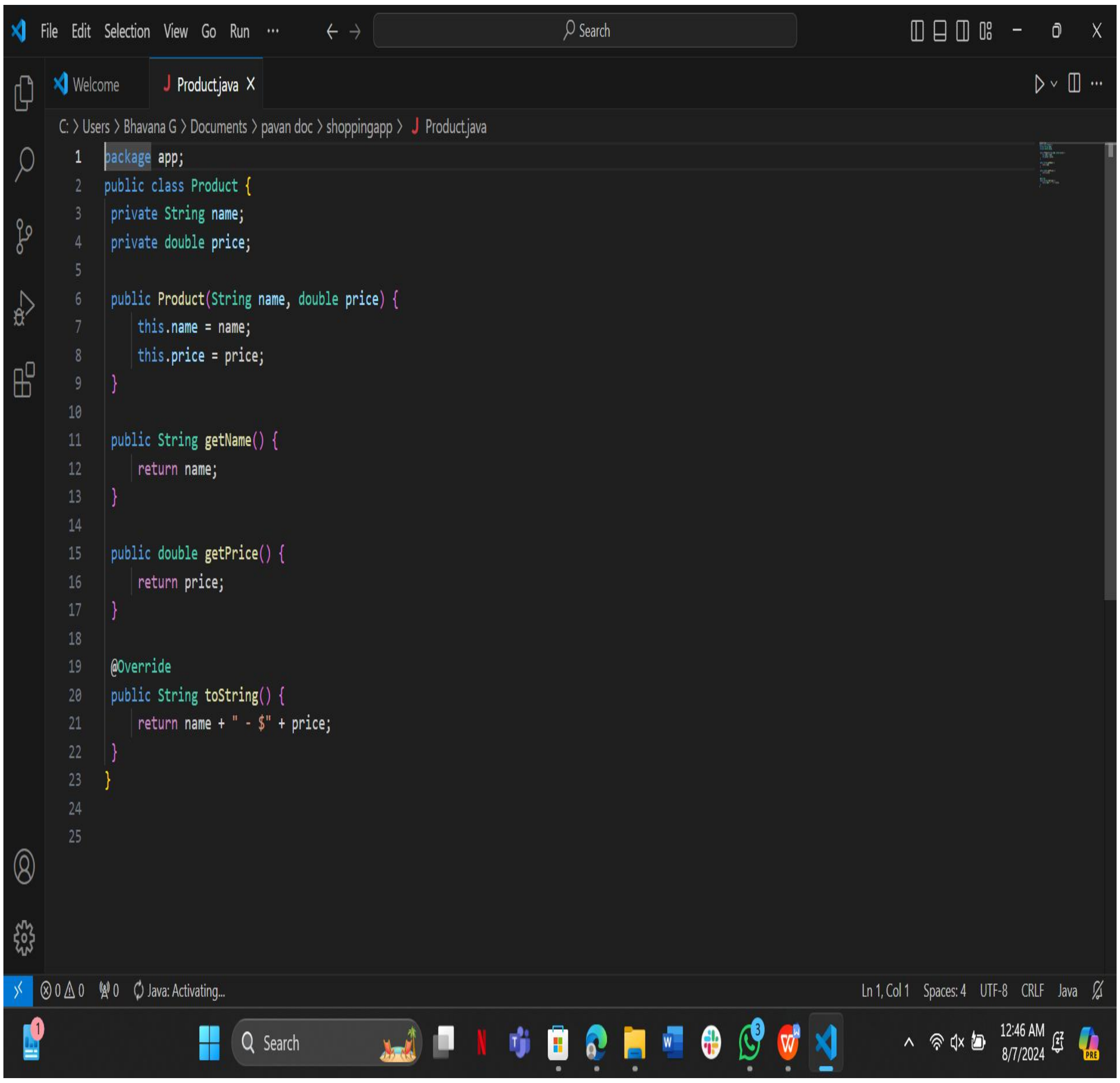
Non-functional requirements emphasize a user-friendly and intuitive interface, ensuring an engaging and dynamic experience for the players. This includes efficient error handling and a seamless user experience.

CHAPTER – 3

IMPLEMENTATION

The implementation phase involves coding the core components of the "E-Commerce Application" shopping website. Below are the detailed steps and the complete code for implementing the site, including the classes for Products, categories, sub categories and main application.

Product Class:

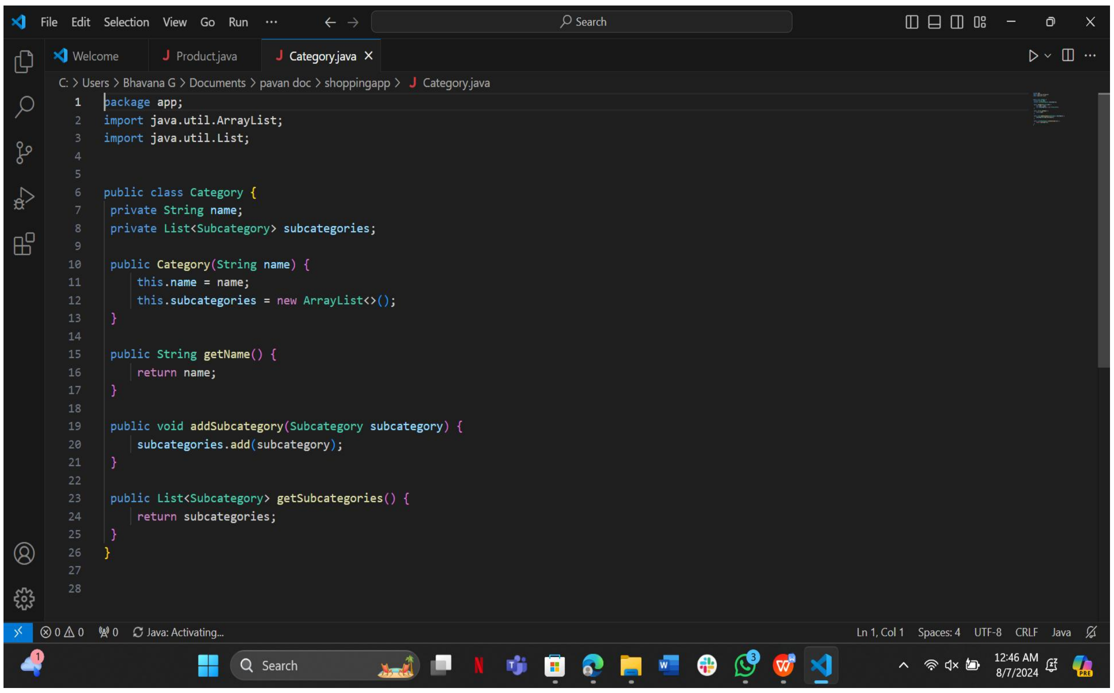


```java
package app;
public class Product {
 private String name;
 private double price;

 public Product(String name, double price) {
     this.name = name;
     this.price = price;
 }

 public String getName() {
     return name;
 }

 public double getPrice() {
     return price;
 }

 @Override
 public String toString() {
     return name + " - $" + price;
 }
}
```

File Edit Selection View Go Run ···
Search
Welcome
Product.java
C: > Users > Bhavana G > Documents > pavan doc > shoppingapp > Product.java
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Java
Java: Activating...
12:46 AM
8/7/2024

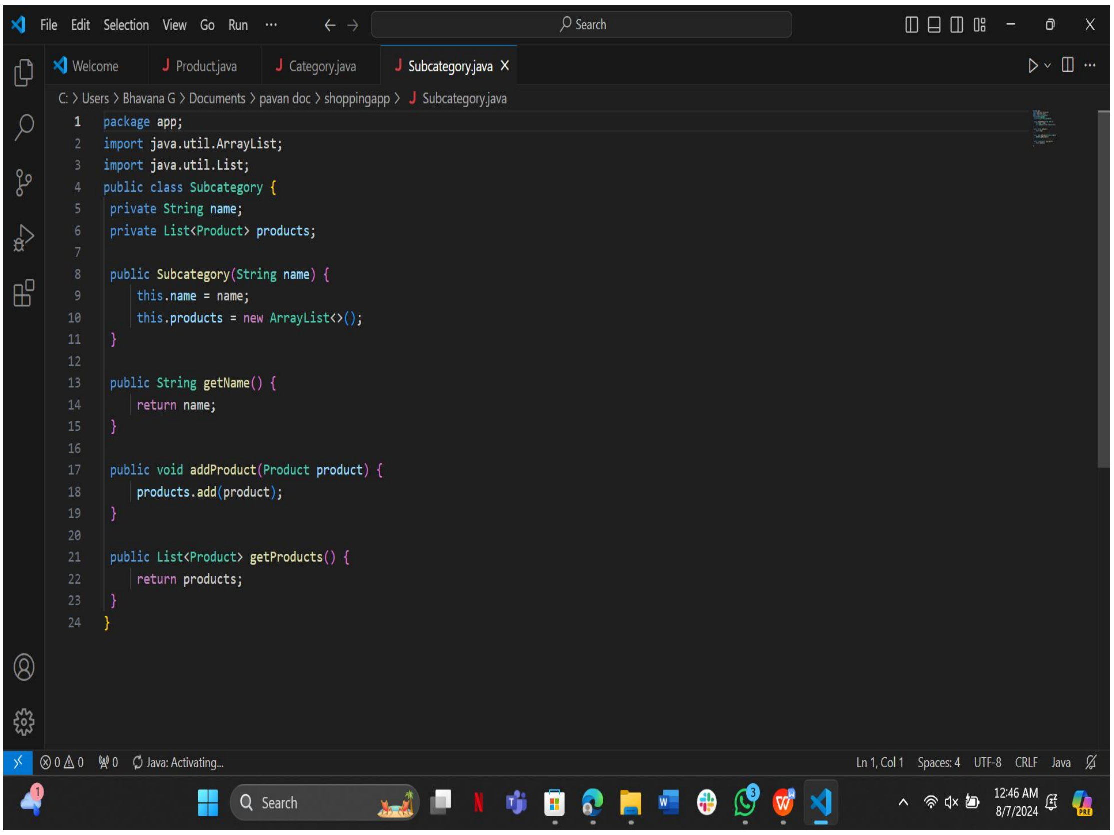
Fig 1 Product Class

Category Class



C: > Users > Bhavana G > Documents > pavan doc > shoppingapp > Category.java

```java
package app;
import java.util.ArrayList;
import java.util.List;


public class Category {
 private String name;
 private List<Subcategory> subcategories;

 public Category(String name) {
     this.name = name;
     this.subcategories = new ArrayList<>();
 }

 public String getName() {
     return name;
 }

 public void addSubcategory(Subcategory subcategory) {
     subcategories.add(subcategory);
 }

 public List<Subcategory> getSubcategories() {
     return subcategories;
 }
}
```

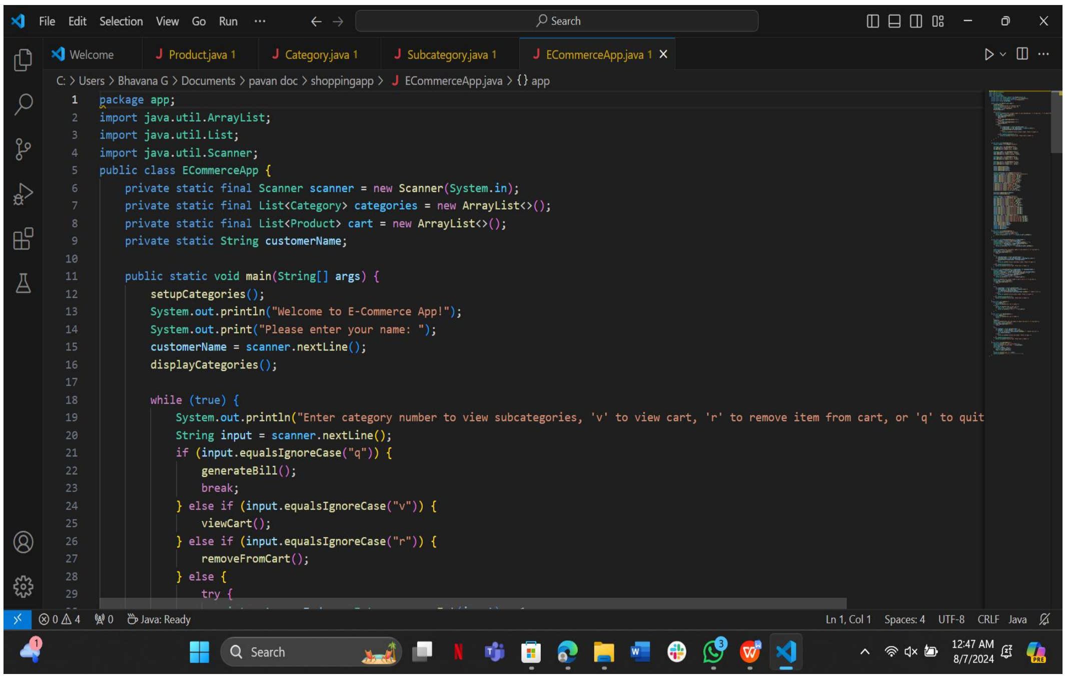


C: > Users > Bhavana G > Documents > pavan doc > shoppingapp > Subcategory.java

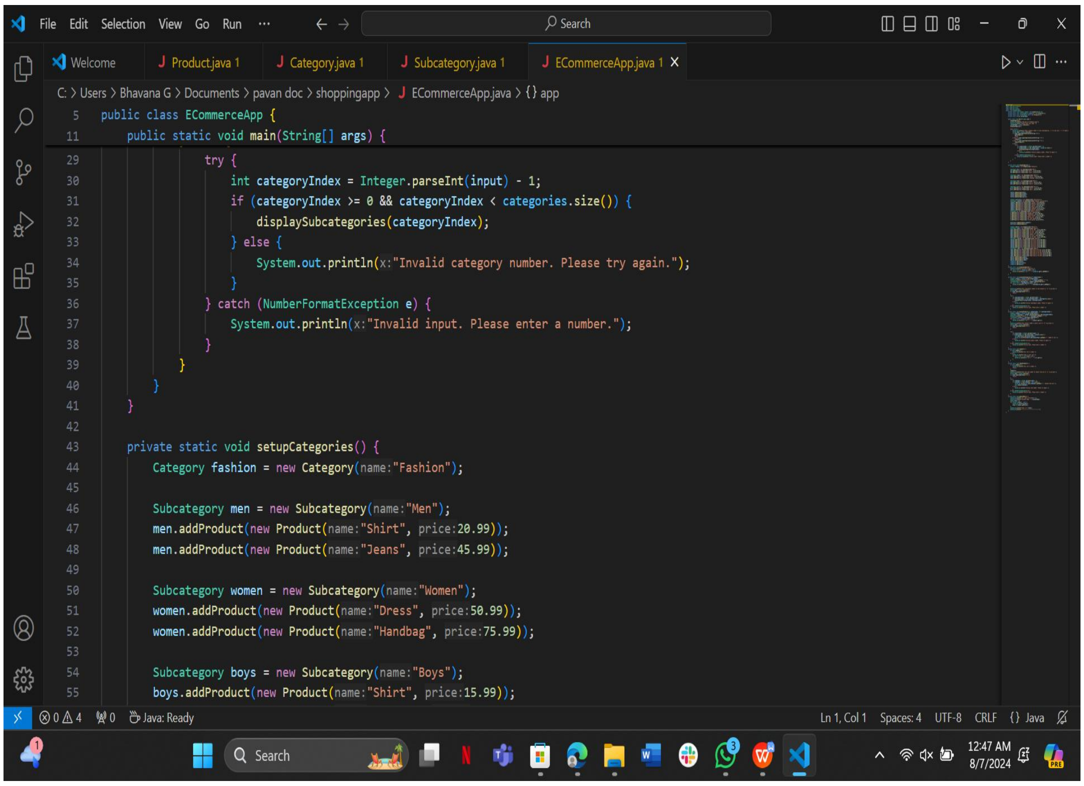
```java
package app;
import java.util.ArrayList;
import java.util.List;
public class Subcategory {
 private String name;
 private List<Product> products;

 public Subcategory(String name) {
     this.name = name;
     this.products = new ArrayList<>();
 }

 public String getName() {
     return name;
 }

 public void addProduct(Product product) {
     products.add(product);
 }

 public List<Product> getProducts() {
     return products;
 }
}
```

Sub-Category Class

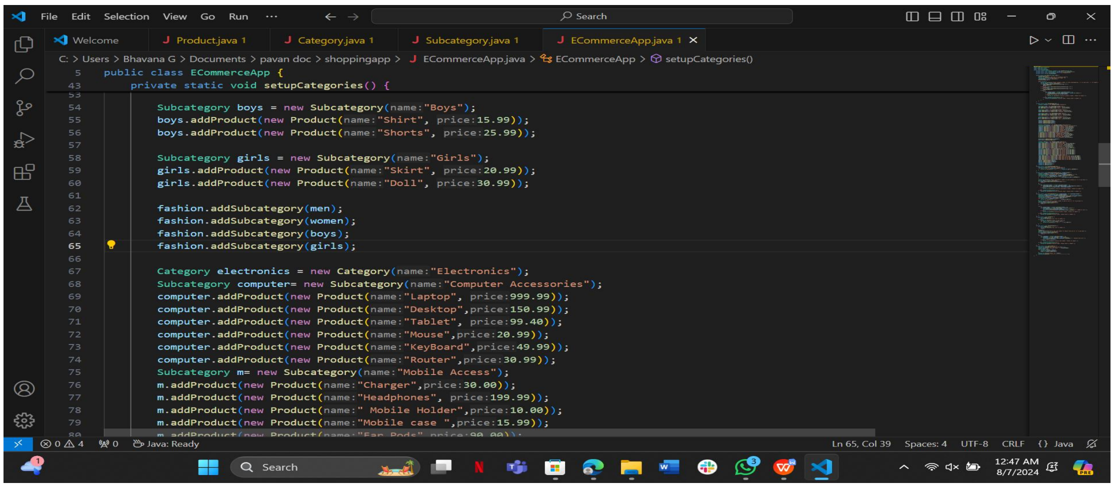
E-Commerce Class: This class starts the application run.



```
package app;
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;
public class ECommerceApp {
    private static final Scanner scanner = new Scanner(System.in);
    private static final List<Category> categories = new ArrayList<>();
    private static final List<Product> cart = new ArrayList<>();
    private static String customerName;

    public static void main(String[] args) {
        setupCategories();
        System.out.println("Welcome to E-Commerce App!");
        System.out.print("Please enter your name: ");
        customerName = scanner.nextLine();
        displayCategories();

        while (true) {
            System.out.println("Enter category number to view subcategories, 'v' to view cart, 'r' to remove item from cart, or 'q' to quit
            String input = scanner.nextLine();
            if (input.equalsIgnoreCase("q")) {
                generateBill();
                break;
            } else if (input.equalsIgnoreCase("v")) {
                viewCart();
            } else if (input.equalsIgnoreCase("r")) {
                removeFromCart();
            } else {
                try {
```

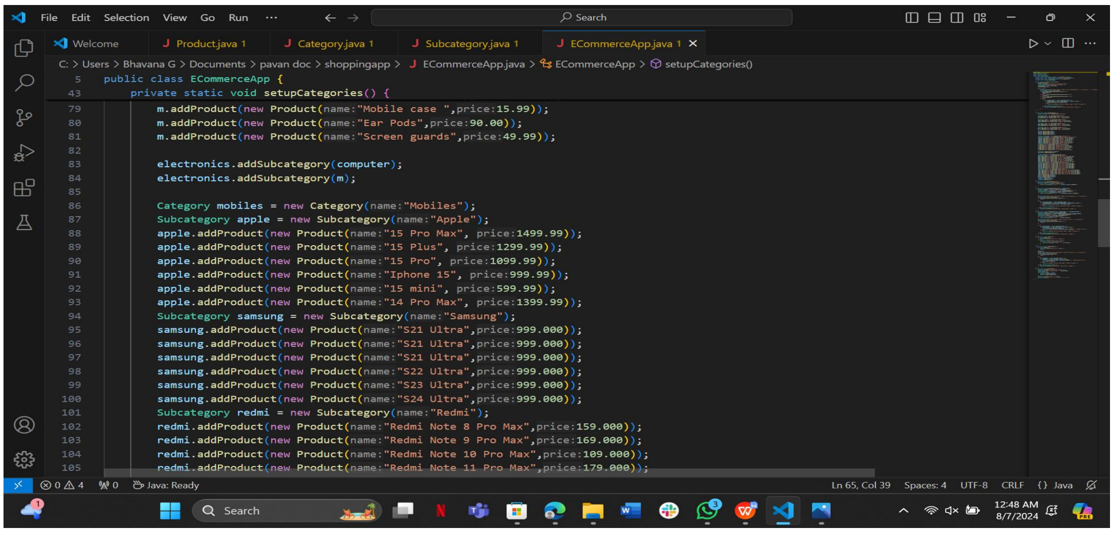


```
public class ECommerceApp {
    public static void main(String[] args) {
                try {
                    int categoryIndex = Integer.parseInt(input) - 1;
                    if (categoryIndex >= 0 && categoryIndex < categories.size()) {
                        displaySubcategories(categoryIndex);
                    } else {
                        System.out.println(x:"Invalid category number. Please try again.");
                    }
                } catch (NumberFormatException e) {
                    System.out.println(x:"Invalid input. Please enter a number.");
                }
            }
        }
    }

    private static void setupCategories() {
        Category fashion = new Category(name:"Fashion");

        Subcategory men = new Subcategory(name:"Men");
        men.addProduct(new Product(name:"Shirt", price:20.99));
        men.addProduct(new Product(name:"Jeans", price:45.99));

        Subcategory women = new Subcategory(name:"Women");
        women.addProduct(new Product(name:"Dress", price:50.99));
        women.addProduct(new Product(name:"Handbag", price:75.99));

        Subcategory boys = new Subcategory(name:"Boys");
        boys.addProduct(new Product(name:"Shirt", price:15.99));
```



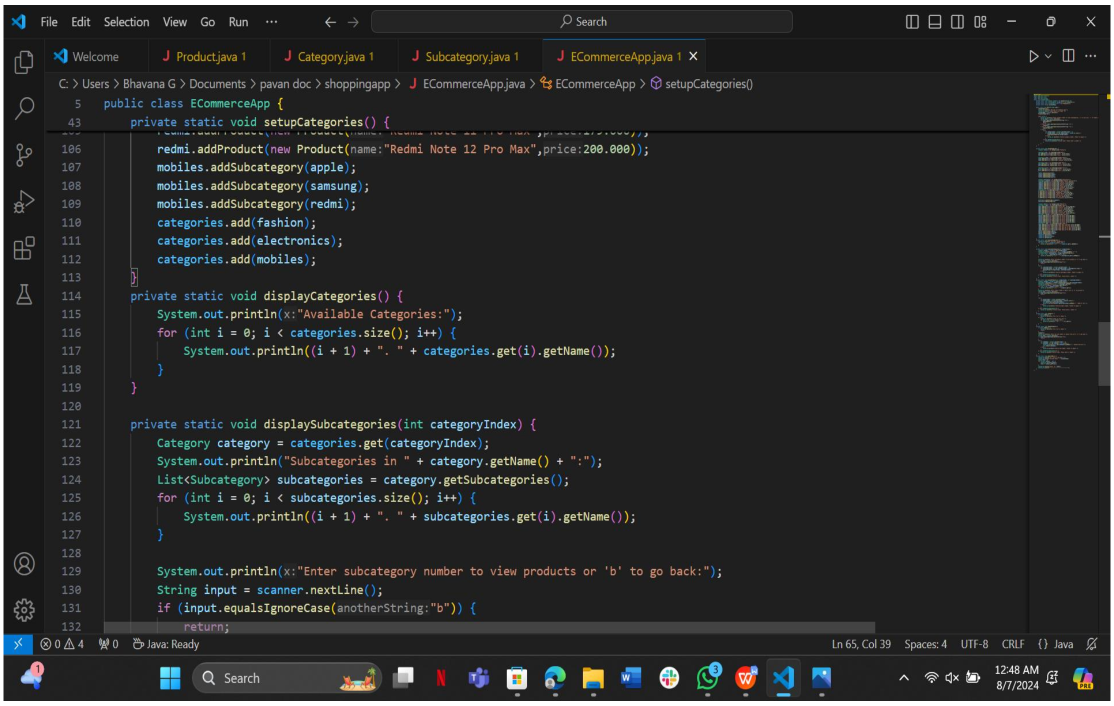
C: > Users > Bhavana G > Documents > pavan doc > shoppingapp > ECommerceApp.java > ECommerceApp > setupCategories()

```java
public class ECommerceApp {
    private static void setupCategories() {
        Subcategory boys = new Subcategory(name:"Boys");
        boys.addProduct(new Product(name:"Shirt", price:15.99));
        boys.addProduct(new Product(name:"Shorts", price:25.99));

        Subcategory girls = new Subcategory(name:"Girls");
        girls.addProduct(new Product(name:"Skirt", price:20.99));
        girls.addProduct(new Product(name:"Doll", price:30.99));

        fashion.addSubcategory(men);
        fashion.addSubcategory(women);
        fashion.addSubcategory(boys);
        fashion.addSubcategory(girls);

        Category electronics = new Category(name:"Electronics");
        Subcategory computer= new Subcategory(name:"Computer Accessories");
        computer.addProduct(new Product(name:"Laptop", price:999.99));
        computer.addProduct(new Product(name:"Desktop",price:150.99));
        computer.addProduct(new Product(name:"Tablet", price:99.40));
        computer.addProduct(new Product(name:"Mouse",price:20.99));
        computer.addProduct(new Product(name:"KeyBoard",price:49.99));
        computer.addProduct(new Product(name:"Router",price:30.99));
        Subcategory m= new Subcategory(name:"Mobile Access");
        m.addProduct(new Product(name:"Charger",price:30.00));
        m.addProduct(new Product(name:"Headphones", price:199.99));
        m.addProduct(new Product(name:" Mobile Holder",price:10.00));
        m.addProduct(new Product(name:"Mobile case ",price:15.99));
```



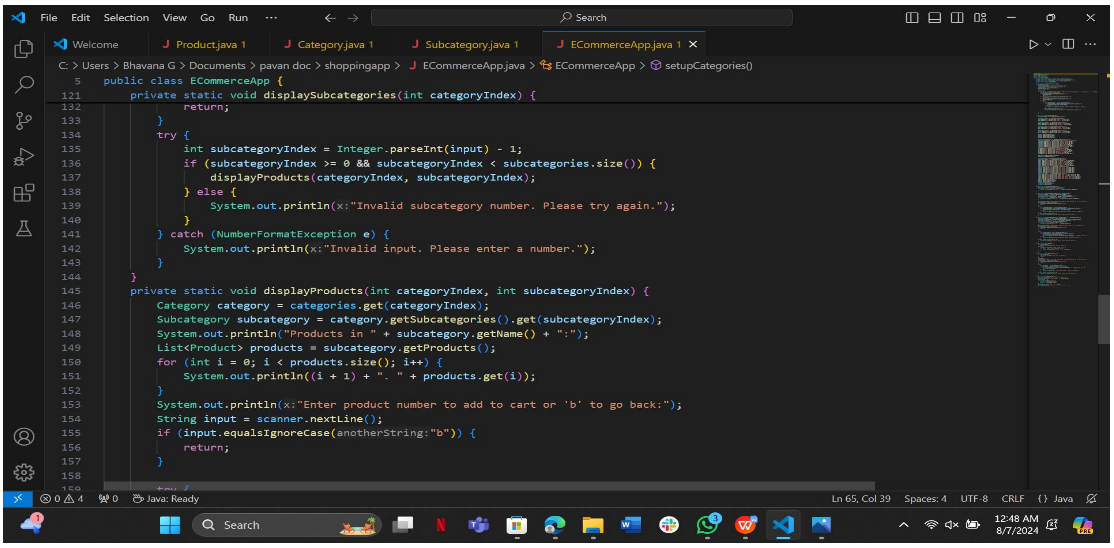
C: > Users > Bhavana G > Documents > pavan doc > shoppingapp > ECommerceApp.java > ECommerceApp > setupCategories()

```java
public class ECommerceApp {
    private static void setupCategories() {
        m.addProduct(new Product(name:"Mobile case ",price:15.99));
        m.addProduct(new Product(name:"Ear Pods",price:90.00));
        m.addProduct(new Product(name:"Screen guards",price:49.99));

        electronics.addSubcategory(computer);
        electronics.addSubcategory(m);

        Category mobiles = new Category(name:"Mobiles");
        Subcategory apple = new Subcategory(name:"Apple");
        apple.addProduct(new Product(name:"15 Pro Max", price:1499.99));
        apple.addProduct(new Product(name:"15 Plus", price:1299.99));
        apple.addProduct(new Product(name:"15 Pro", price:1099.99));
        apple.addProduct(new Product(name:"Iphone 15", price:999.99));
        apple.addProduct(new Product(name:"15 mini", price:599.99));
        apple.addProduct(new Product(name:"14 Pro Max", price:1399.99));
        Subcategory samsung = new Subcategory(name:"Samsung");
        samsung.addProduct(new Product(name:"S21 Ultra",price:999.000));
        samsung.addProduct(new Product(name:"S21 Ultra",price:999.000));
        samsung.addProduct(new Product(name:"S21 Ultra",price:999.000));
        samsung.addProduct(new Product(name:"S22 Ultra",price:999.000));
        samsung.addProduct(new Product(name:"S23 Ultra",price:999.000));
        samsung.addProduct(new Product(name:"S24 Ultra",price:999.000));
        Subcategory redmi = new Subcategory(name:"Redmi");
        redmi.addProduct(new Product(name:"Redmi Note 8 Pro Max",price:159.000));
        redmi.addProduct(new Product(name:"Redmi Note 9 Pro Max",price:169.000));
        redmi.addProduct(new Product(name:"Redmi Note 10 Pro Max",price:109.000));
        redmi.addProduct(new Product(name:"Redmi Note 11 Pro Max",price:179.000));
```

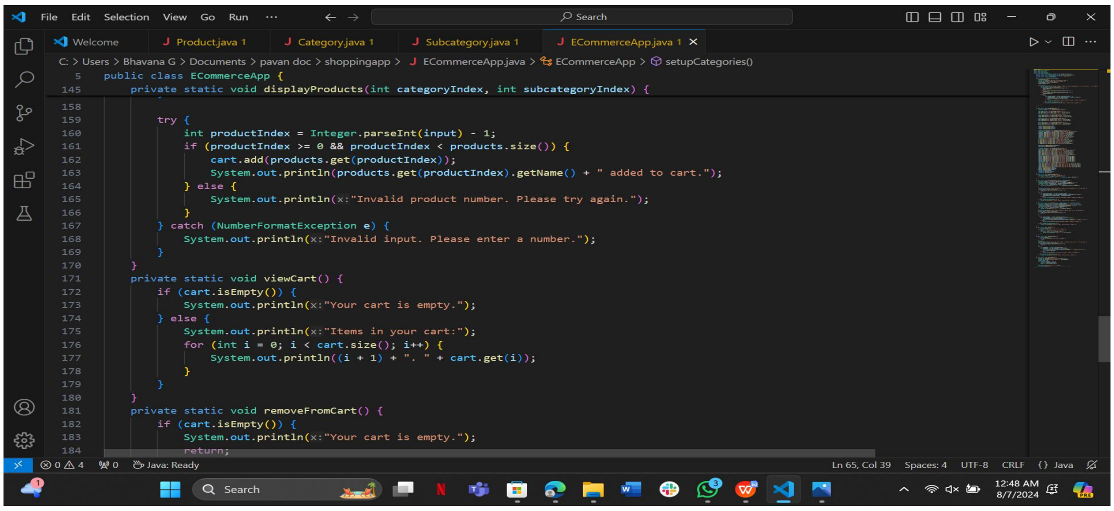


C: > Users > Bhavana G > Documents > pavan doc > shoppingapp > ECommerceApp.java > ECommerceApp > setupCategories()

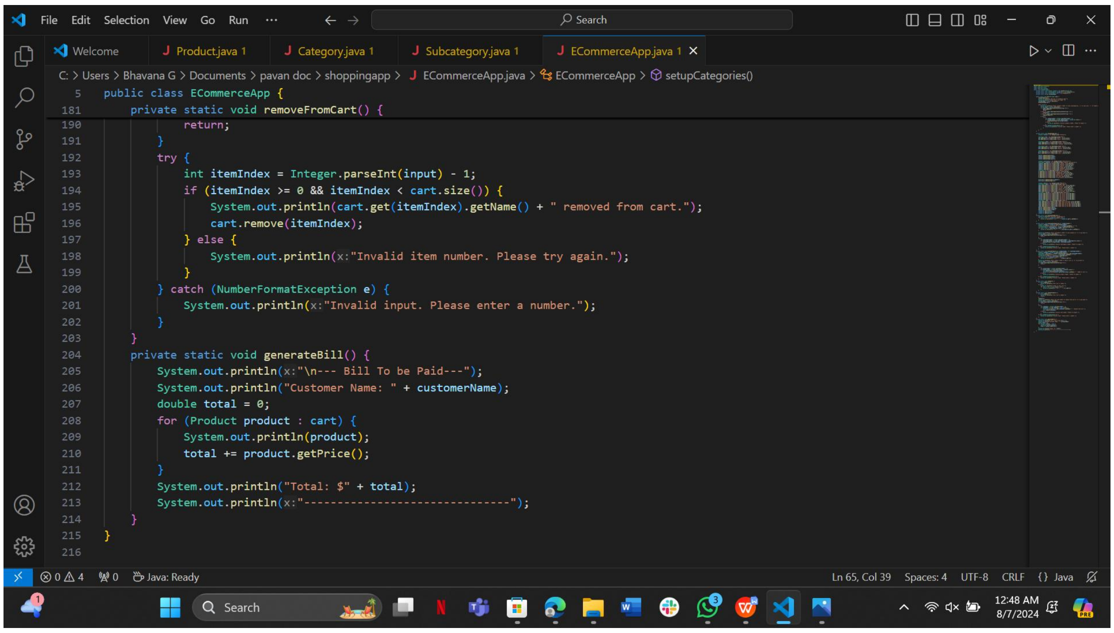
```java
public class ECommerceApp {
    private static void setupCategories() {
        redmi.addProduct(new Product(name:"Redmi Note 12 Pro Max",price:200.000));
        mobiles.addSubcategory(apple);
        mobiles.addSubcategory(samsung);
        mobiles.addSubcategory(redmi);
        categories.add(fashion);
        categories.add(electronics);
        categories.add(mobiles);
    }
    private static void displayCategories() {
        System.out.println(x:"Available Categories:");
        for (int i = 0; i < categories.size(); i++) {
            System.out.println((i + 1) + ". " + categories.get(i).getName());
        }
    }

    private static void displaySubcategories(int categoryIndex) {
        Category category = categories.get(categoryIndex);
        System.out.println("Subcategories in " + category.getName() + ":");
        List<Subcategory> subcategories = category.getSubcategories();
        for (int i = 0; i < subcategories.size(); i++) {
            System.out.println((i + 1) + ". " + subcategories.get(i).getName());
        }

        System.out.println(x:"Enter subcategory number to view products or 'b' to go back:");
        String input = scanner.nextLine();
        if (input.equalsIgnoreCase(anotherString:"b")) {
            return;
```



```java
public class ECommerceApp {
    private static void displaySubcategories(int categoryIndex) {
                return;
            }
            try {
                int subcategoryIndex = Integer.parseInt(input) - 1;
                if (subcategoryIndex >= 0 && subcategoryIndex < subcategories.size()) {
                    displayProducts(categoryIndex, subcategoryIndex);
                } else {
                    System.out.println(x:"Invalid subcategory number. Please try again.");
                }
            } catch (NumberFormatException e) {
                System.out.println(x:"Invalid input. Please enter a number.");
            }
        }
        private static void displayProducts(int categoryIndex, int subcategoryIndex) {
            Category category = categories.get(categoryIndex);
            Subcategory subcategory = category.getSubcategories().get(subcategoryIndex);
            System.out.println("Products in " + subcategory.getName() + ":");
            List<Product> products = subcategory.getProducts();
            for (int i = 0; i < products.size(); i++) {
                System.out.println((i + 1) + ". " + products.get(i));
            }
            System.out.println(x:"Enter product number to add to cart or 'b' to go back:");
            String input = scanner.nextLine();
            if (input.equalsIgnoreCase(anotherString:"b")) {
                return;
            }

            try {
```



```java
public class ECommerceApp {
    private static void displayProducts(int categoryIndex, int subcategoryIndex) {

            try {
                int productIndex = Integer.parseInt(input) - 1;
                if (productIndex >= 0 && productIndex < products.size()) {
                    cart.add(products.get(productIndex));
                    System.out.println(products.get(productIndex).getName() + " added to cart.");
                } else {
                    System.out.println(x:"Invalid product number. Please try again.");
                }
            } catch (NumberFormatException e) {
                System.out.println(x:"Invalid input. Please enter a number.");
            }
        }
        private static void viewCart() {
            if (cart.isEmpty()) {
                System.out.println(x:"Your cart is empty.");
            } else {
                System.out.println(x:"Items in your cart:");
                for (int i = 0; i < cart.size(); i++) {
                    System.out.println((i + 1) + ". " + cart.get(i));
                }
            }
        }
        private static void removeFromCart() {
            if (cart.isEmpty()) {
                System.out.println(x:"Your cart is empty.");
                return;
```



```java
public class ECommerceApp {
    private static void removeFromCart() {
                return;
            }
            try {
                int itemIndex = Integer.parseInt(input) - 1;
                if (itemIndex >= 0 && itemIndex < cart.size()) {
                    System.out.println(cart.get(itemIndex).getName() + " removed from cart.");
                    cart.remove(itemIndex);
                } else {
                    System.out.println(x:"Invalid item number. Please try again.");
                }
            } catch (NumberFormatException e) {
                System.out.println(x:"Invalid input. Please enter a number.");
            }
        }
        private static void generateBill() {
            System.out.println(x:"\n--- Bill To be Paid---");
            System.out.println("Customer Name: " + customerName);
            double total = 0;
            for (Product product : cart) {
                System.out.println(product);
                total += product.getPrice();
            }
            System.out.println("Total: $" + total);
            System.out.println(x:"------------------------------");
        }
    }
```

Fig 2 Ecommerce Class

CHAPTER – 4

TESTING

Introduction

Testing is an investigation that is carried out to offer information to stakeholders regarding the quality of the product or service being tested. Program testing also gives the business an objective, unbiased picture of the software, allowing them to grasp and comprehend the risks associated with software implementation. The process of executing a program or application with the purpose of detecting software faults is one example of a test technique.

Unit Testing

Verify that each class and method perform as expected in isolation.

Functional Testing

Validate that the application meets the functional requirements specified.

Performance Testing:

Ensure the application performs well under expected load conditions.

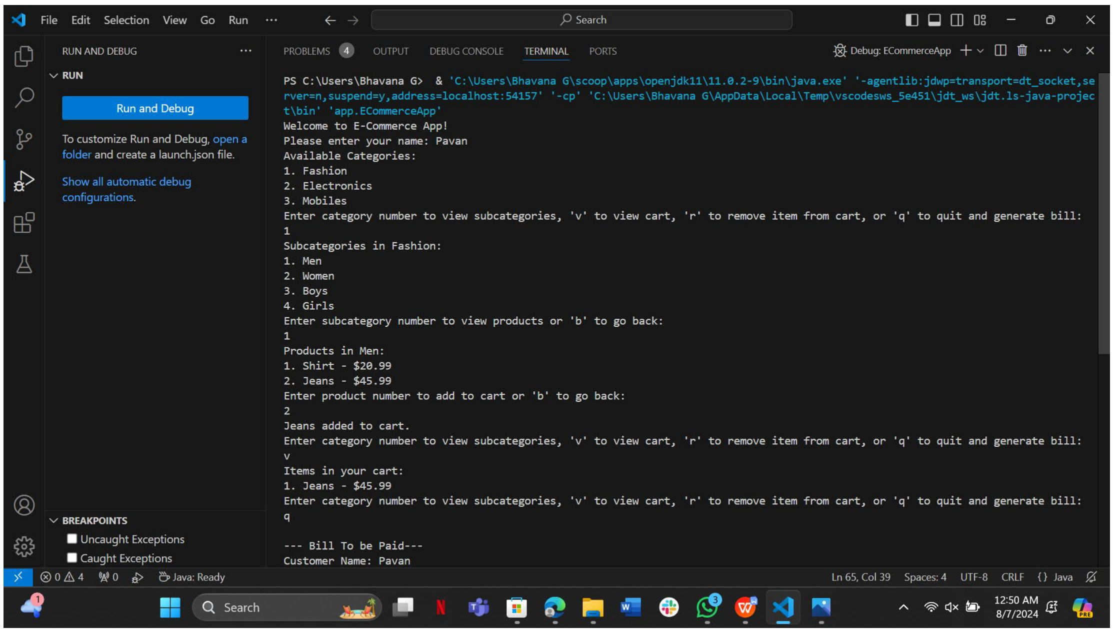
Integration Testing:

Ensure that different components of the application work together as intended. Test the interaction between the product and categories and sub category classes.

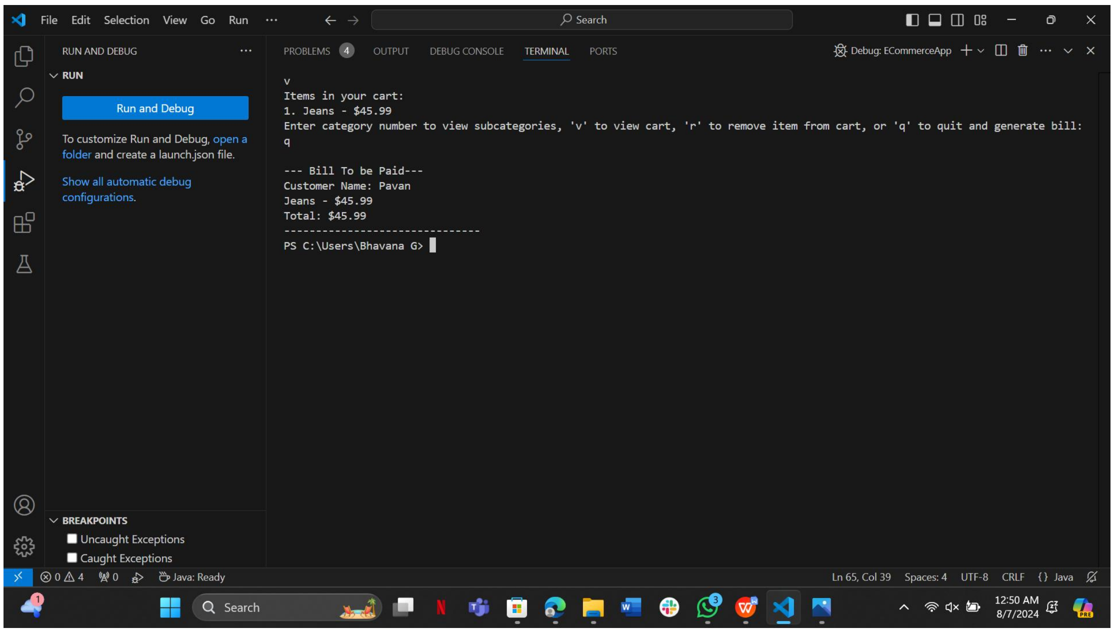
CHAPTER – 5

RESULTS

Execution of E-Commerce Application:



```
PS C:\Users\Bhavana G>  & 'C:\Users\Bhavana G\scoop\apps\openjdk11\11.0.2-9\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,se
rver=n,suspend=y,address=localhost:54157' '-cp' 'C:\Users\Bhavana G\AppData\Local\Temp\vscodesws_5e451\jdt_ws\jdt.ls-java-projec
t\bin' 'app.ECommerceApp'
Welcome to E-Commerce App!
Please enter your name: Pavan
Available Categories:
1. Fashion
2. Electronics
3. Mobiles
Enter category number to view subcategories, 'v' to view cart, 'r' to remove item from cart, or 'q' to quit and generate bill:
1
Subcategories in Fashion:
1. Men
2. Women
3. Boys
4. Girls
Enter subcategory number to view products or 'b' to go back:
1
Products in Men:
1. Shirt - $20.99
2. Jeans - $45.99
Enter product number to add to cart or 'b' to go back:
2
Jeans added to cart.
Enter category number to view subcategories, 'v' to view cart, 'r' to remove item from cart, or 'q' to quit and generate bill:
v
Items in your cart:
1. Jeans - $45.99
Enter category number to view subcategories, 'v' to view cart, 'r' to remove item from cart, or 'q' to quit and generate bill:
q

--- Bill To be Paid---
Customer Name: Pavan
```



```
v
Items in your cart:
1. Jeans - $45.99
Enter category number to view subcategories, 'v' to view cart, 'r' to remove item from cart, or 'q' to quit and generate bill:
q

--- Bill To be Paid---
Customer Name: Pavan
Jeans - $45.99
Total: $45.99
------------------------------
PS C:\Users\Bhavana G> 
```

CONCLUSION & FUTURE WORK

In conclusion, this project successfully achieves its primary objective of simulating an online shopping experience, providing valuable insights into the development of e-commerce applications. It lays the groundwork for further development and can be expanded to create a fully functional e-commerce platform, meeting the evolving needs of online shoppers and keeping pace with technological advancements.

For future Undertakings, extending the shopping site development using Web development, then it can be more visually appealing website.

REFERENCES

[1]. Myntra, Flipkart.

[2]. Core Java by Bheemesh Sir.