**WEEK-1**

**Design Principles And Patterns**

**Exercise 1: Implementing the Singleton Pattern:**

**Printer.java**

public class Printer {

    private Printer() {

        System.out.println("Printer initialized.");

    }

    private static class Holder {

        private static final Printer INSTANCE = new Printer();

    }

    public static Printer getInstance() {

        return Holder.INSTANCE;

    }

    public void print(String content) {

        System.out.println("Printing: " + content);

    }

}

**Main.java**

public class Main {

    public static void main(String[] args) {

        Printer p1 = Printer.getInstance();

        Printer p2 = Printer.getInstance();

        p1.print("Document A");

        p2.print("Document B");

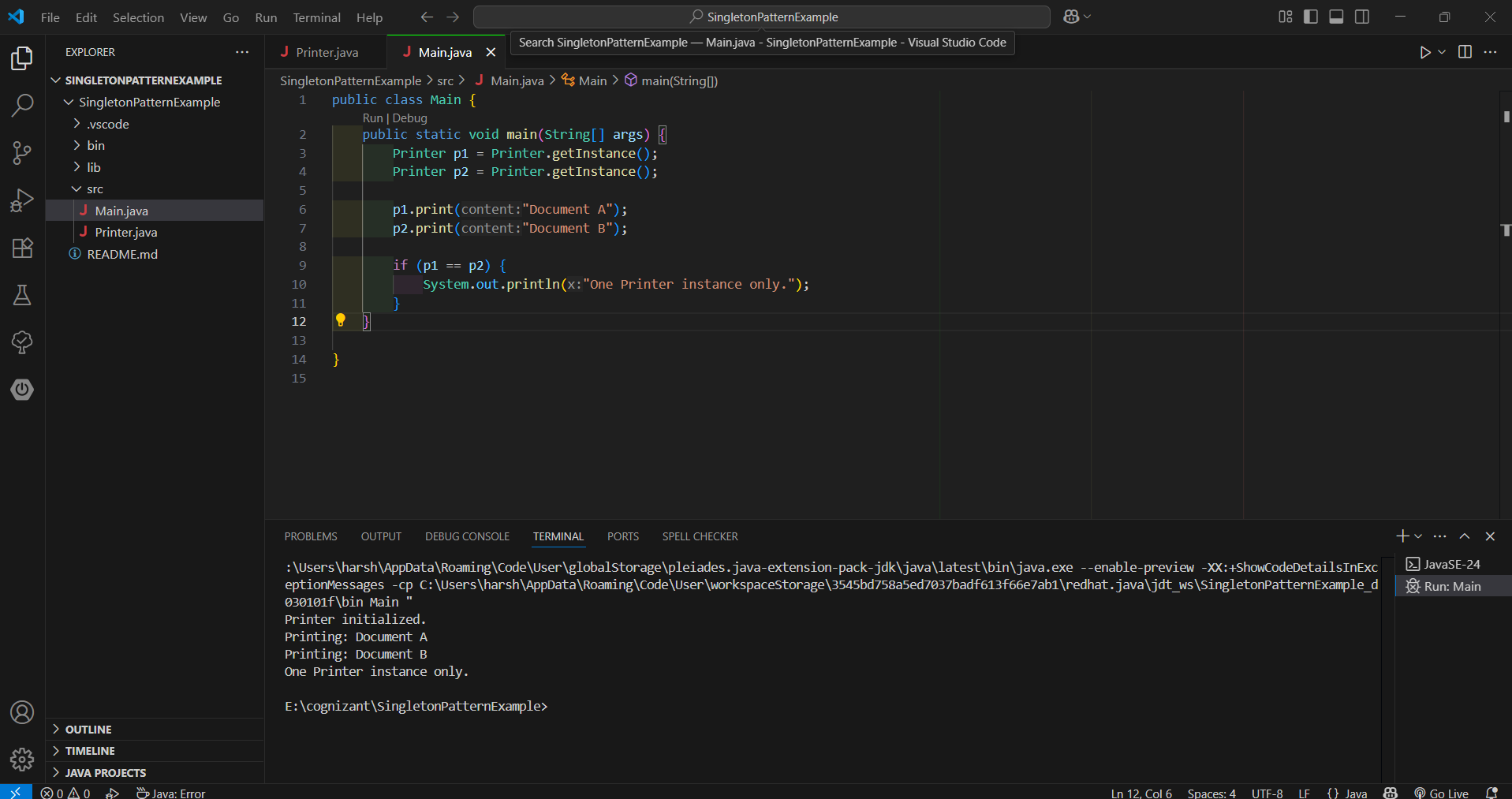
        if (p1 == p2) {

            System.out.println("One Printer instance only.");

        }

    }

}



**Exercise 2: Implementing the Factory Method Pattern**

**Shirt.java**

public class Shirt extends Product {

    public Shirt() {

        super(2001, "Polo T-Shirt", "Clothing");

    }

    @Override

    public void display() {

        System.out.println("Clothing: " + name + " [ID: " + id + ", Category: " + category + "]");

    }

}

**Laptop.java**

public class Laptop extends Product {

    public Laptop() {

        super(1002, "MacBook Air", "Electronics");

    }

    @Override

    public void display() {

        System.out.println("Laptop: " + name + " [ID: " + id + ", Category: " + category + "]");

    }

}

**Mobile.java**

public class Mobile extends Product {

    public Mobile() {

        super(1001, "OnePlus 12", "Electronics");

    }

    @Override

    public void display() {

        System.out.println("Mobile: " + name + " [ID: " + id + ", Category: " + category + "]");

    }

}

**Product.java**

public abstract class Product {

    protected int id;

    protected String name;

    protected String category;

    public Product(int id, String name, String category) {

        this.id = id;

        this.name = name;

        this.category = category;

    }

    public abstract void display();

}

**ProductFactory.java**

public class ProductFactory {

    public static Product getProduct(String type) {

        switch (type.toLowerCase()) {

            case "mobile":

                return new Mobile();

            case "laptop":

                return new Laptop();

            case "shirt":

                return new Shirt();

            default:

                throw new IllegalArgumentException("Invalid product type: " + type);

        }

    }

}

**Main.java**

import java.util.Scanner;

public class Main {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter product type (mobile/laptop/shirt): ");

        String type = scanner.nextLine();

        try {

            Product product = ProductFactory.getProduct(type);

            product.display();

        } catch (IllegalArgumentException e) {

            System.out.println("Error: " + e.getMessage());

        }

        scanner.close();

    }

}

