**DIGITAL NURTURE 4.0 DEEP SKILLING JAVA FSE-WEEK1**

**NAME: SIVITHA GUNASEKARAN**

**SUPERSET ID: 6413354**

**WEEK 1: DESIGN PATTERNS AND PRINCIPLES**

**Exercise 3: Implementing the Builder Pattern**

**Scenario:**

You are developing a system to create complex objects such as a Computer with multiple optional parts. Use the Builder Pattern to manage the construction process.

**Steps:**

1. **Create a New Java Project:**
   * Create a new Java project named **BuilderPatternExample**.
2. **Define a Product Class:**
   * Create a class **Computer** with attributes like **CPU**, **RAM**, **Storage**, etc.
3. **Implement the Builder Class:**
   * Create a static nested Builder class inside Computer with methods to set each attribute.
   * Provide a **build()** method in the Builder class that returns an instance of Computer.
4. **Implement the Builder Pattern:**
   * Ensure that the **Computer** class has a private constructor that takes the **Builder** as a parameter.
5. **Test the Builder Implementation:**
   * Create a test class to demonstrate the creation of different configurations of Computer using the Builder pattern.

**CODE SAMPLES  
BuilderPatternExample.java**

class Computer {

private String CPU;

private String RAM;

private String storage;

private String graphicsCard;

private String operatingSystem;

private Computer(Builder builder) {

this.CPU = builder.CPU;

this.RAM = builder.RAM;

this.storage = builder.storage;

this.graphicsCard = builder.graphicsCard;

this.operatingSystem = builder.operatingSystem;

}

public static class Builder {

private String CPU;

private String RAM;

private String storage;

private String graphicsCard;

private String operatingSystem;

public Builder(String CPU, String RAM) {

this.CPU = CPU;

this.RAM = RAM;

}

public Builder setStorage(String storage) {

this.storage = storage;

return this;

}

public Builder setGraphicsCard(String graphicsCard) {

this.graphicsCard = graphicsCard;

return this;

}

public Builder setOperatingSystem(String os) {

this.operatingSystem = os;

return this;

}

public Computer build() {

return new Computer(this);

}

}

public String toString() {

return "Computer Configuration:\n"

+ "CPU: " + CPU + "\n"

+ "RAM: " + RAM + "\n"

+ "Storage: " + (storage != null ? storage : "Not specified") + "\n"

+ "Graphics Card: " + (graphicsCard != null ? graphicsCard : "Not specified") + "\n"

+ "Operating System: " + (operatingSystem != null ? operatingSystem : "Not specified");

}

}

public class BuilderPatternExample {

public static void main(String[] args) {

Computer basicComputer = new Computer.Builder("Intel i5", "8GB").build();

Computer gamingComputer = new Computer.Builder("Intel i9", "32GB")

.setStorage("1TB SSD")

.setGraphicsCard("NVIDIA RTX 4080")

.setOperatingSystem("Windows 11")

.build();

System.out.println("Basic Computer:");

System.out.println(basicComputer);

System.out.println("\nGaming Computer:");

System.out.println(gamingComputer);

}

}

**OUTPUT:**

Basic Computer:

Computer Configuration:

CPU: Intel i5

RAM: 8GB

Storage: Not specified

Graphics Card: Not specified

Operating System: Not specified

Gaming Computer:

Computer Configuration:

CPU: Intel i9

RAM: 32GB

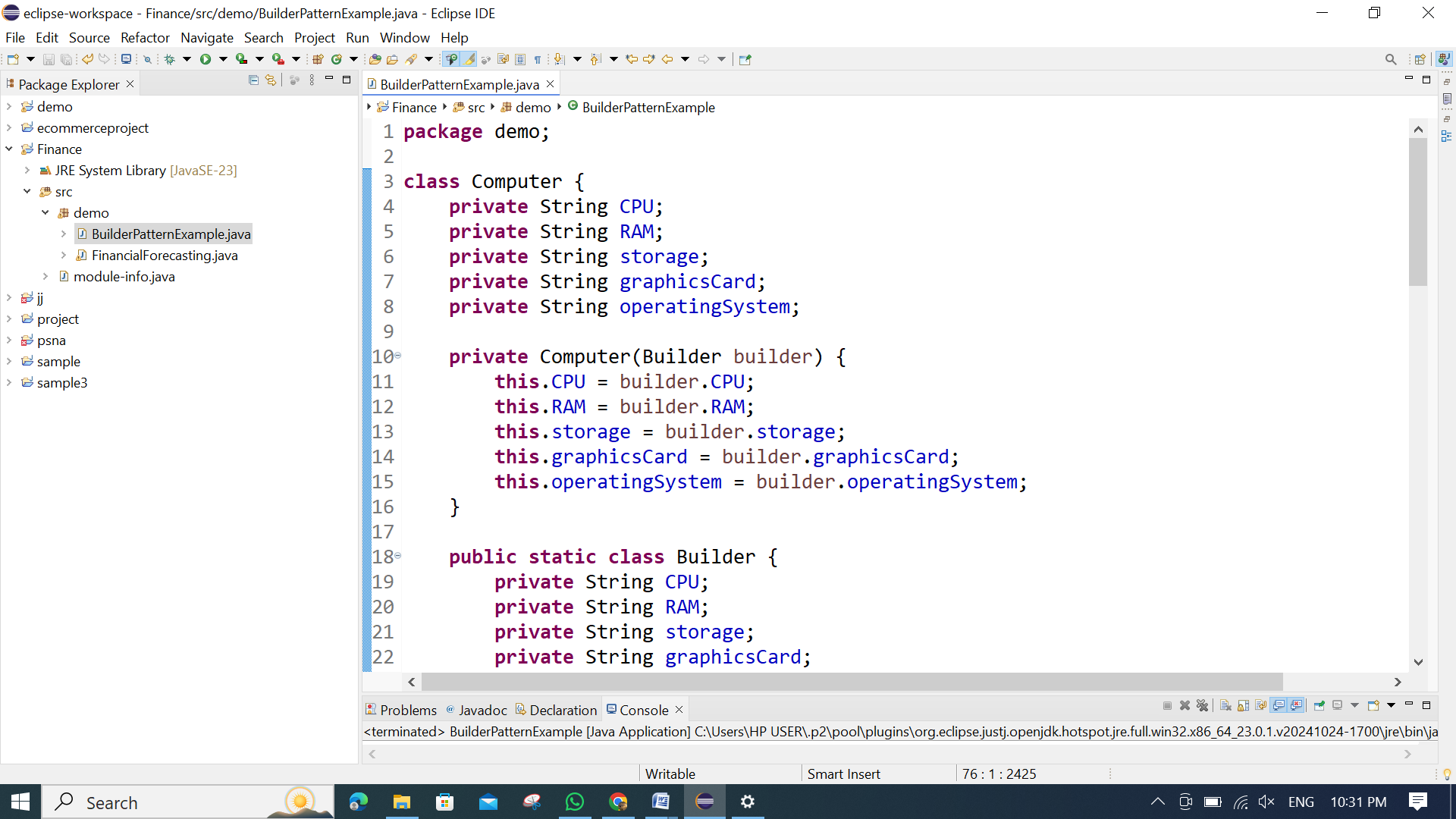
Storage: 1TB SSD

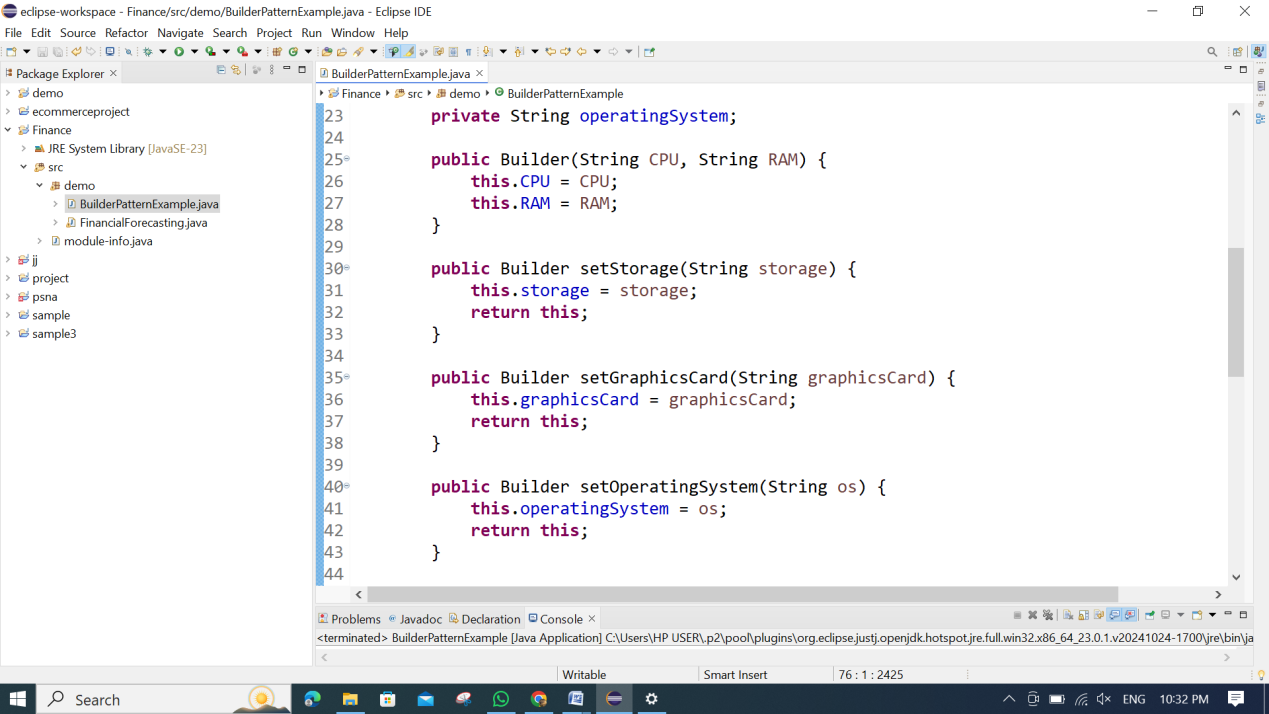
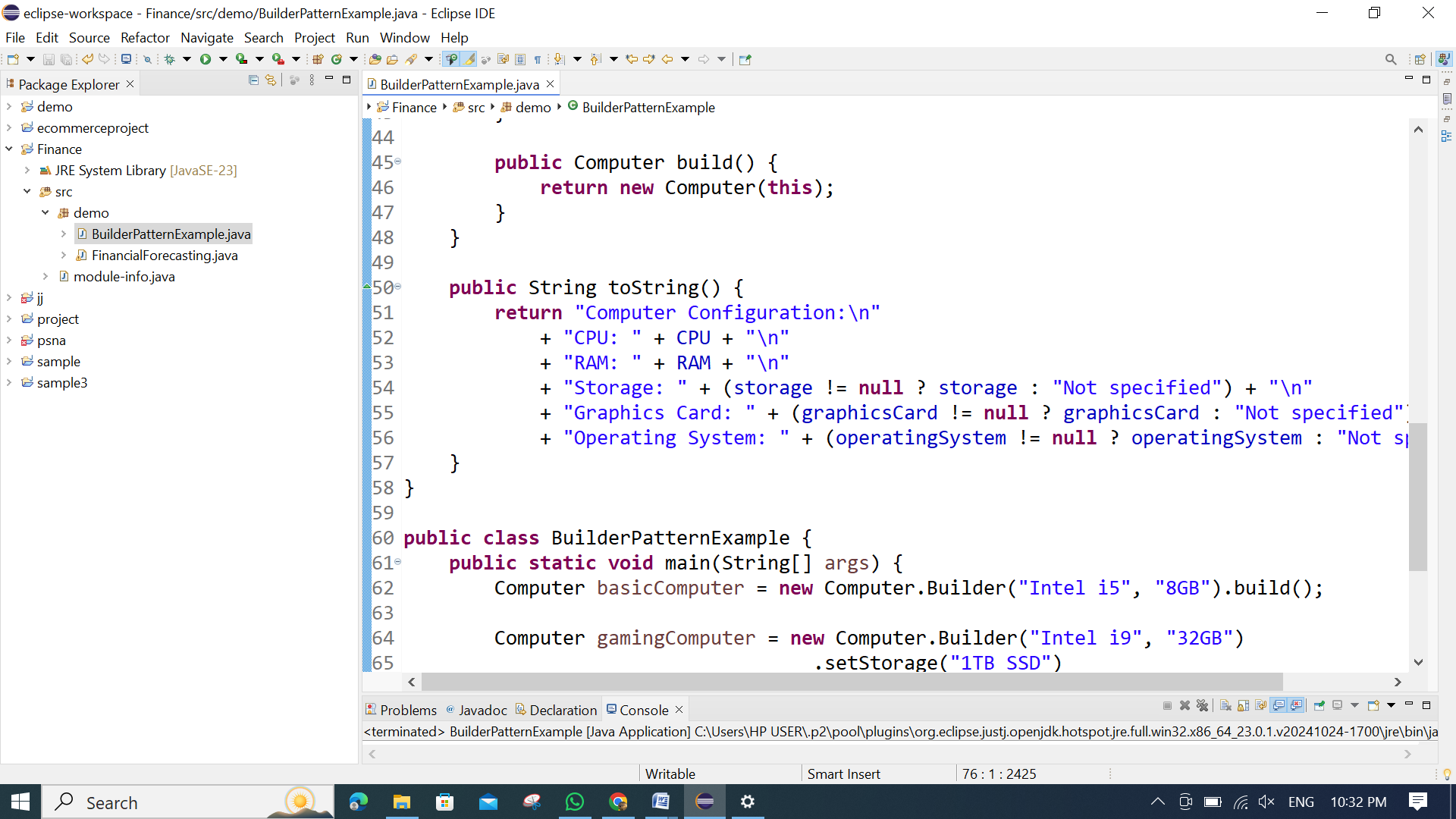
Graphics Card: NVIDIA RTX 4080

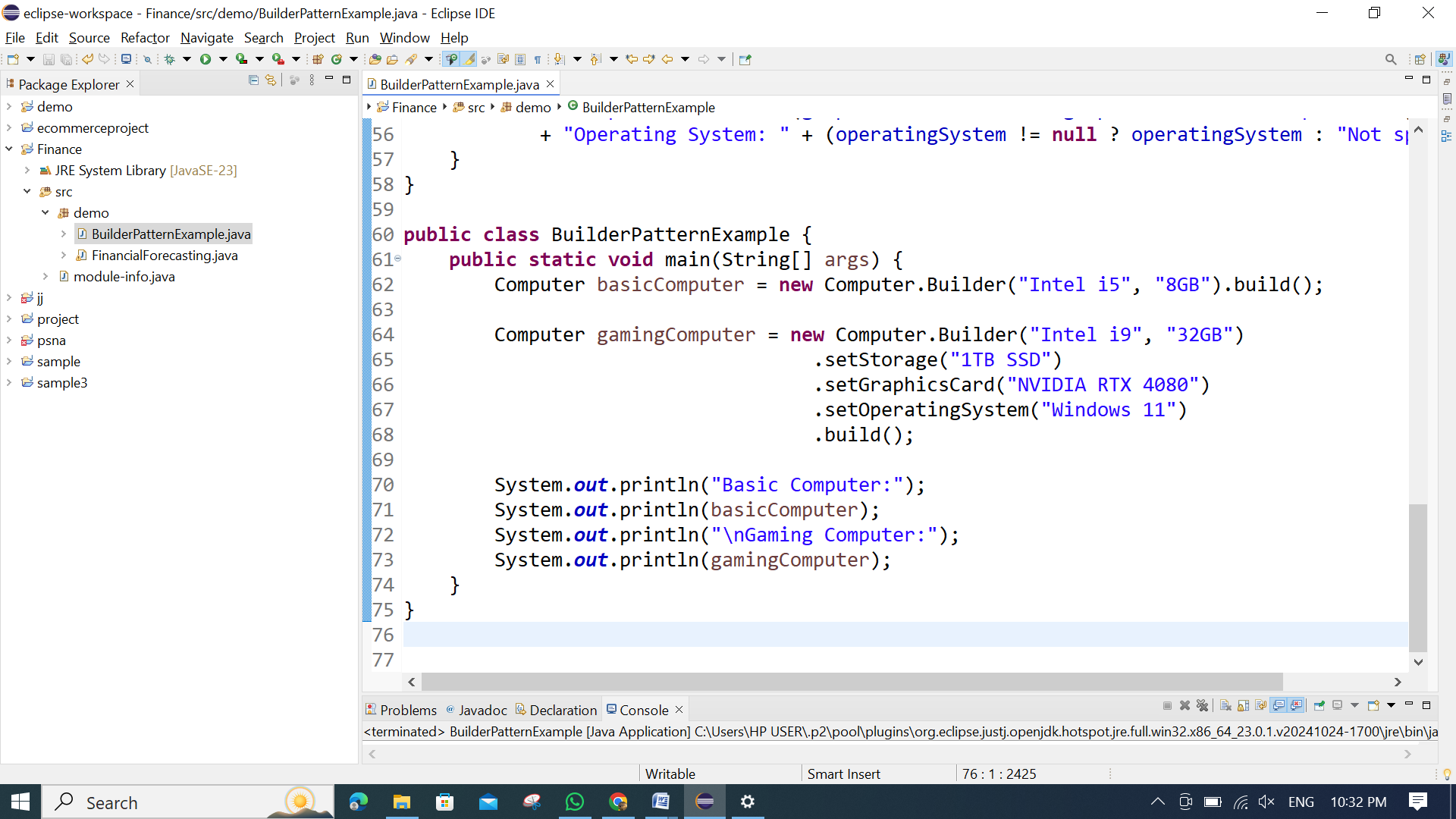
Operating System: Windows 11

**MY SCREENSHOT PROOFS:**

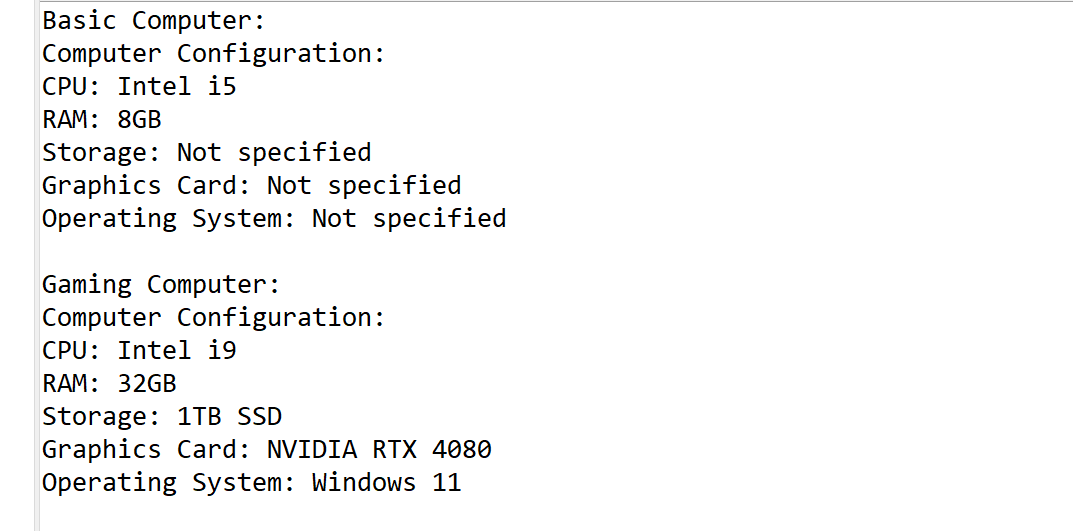
**BuilderPatternExample.java**

****

** **

****

**OUTPUT:**

****