WEEK – 1

Implementing The Builder Pattern

Nipuna Amanapu

[namanapu@gitam.in](mailto:namanapu@gitam.in)

superset id : 6432842

**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.

1)Computer.java :

public class Computer {

private String CPU;

private String RAM;

private String storage;

private String graphicsCard;

private Computer(Builder builder) {

this.CPU = builder.CPU;

this.RAM = builder.RAM;

this.storage = builder.storage;

this.graphicsCard = builder.graphicsCard;

}

public void displayConfig() {

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

System.out.println("CPU: " + CPU);

System.out.println("RAM: " + RAM);

System.out.println("Storage: " + storage);

System.out.println("Graphics Card: " + graphicsCard);

System.out.println("-----------------------------");

}

public static class Builder {

private String CPU;

private String RAM;

private String storage;

private String graphicsCard;

public Builder setCPU(String CPU) {

this.CPU = CPU;

return this;

}

public Builder setRAM(String RAM) {

this.RAM = RAM;

return this;

}

public Builder setStorage(String storage) {

this.storage = storage;

return this;

}

public Builder setGraphicsCard(String graphicsCard) {

this.graphicsCard = graphicsCard;

return this;

}

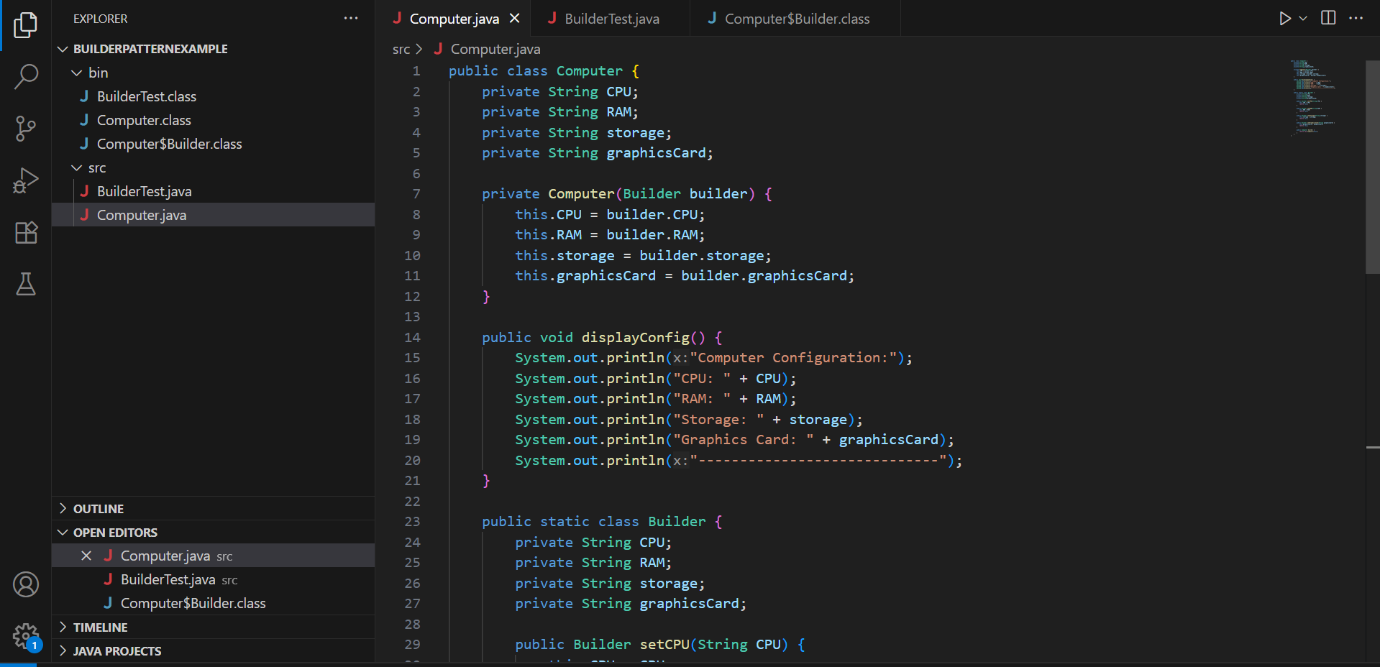
public Computer build() {

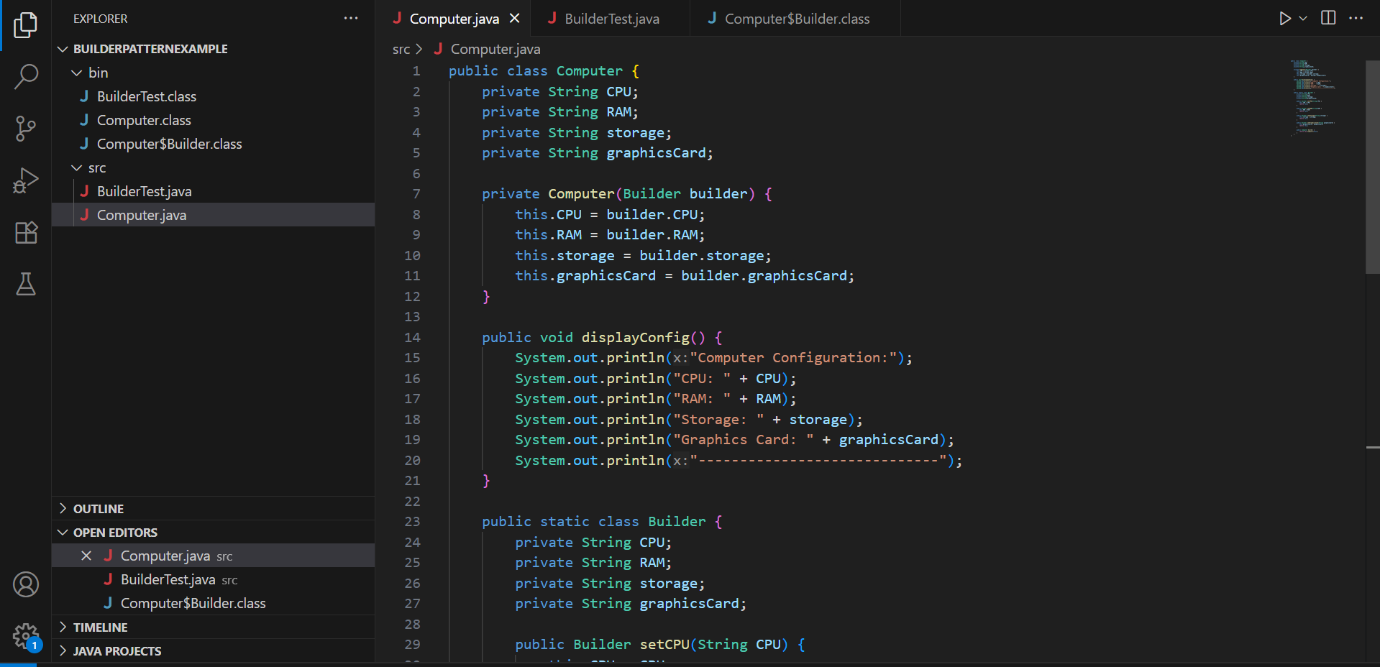
return new Computer(this);

}

}

}





2) BuilderTest.java

public class BuilderTest {

public static void main(String[] args) {

Computer basicComputer = new Computer.Builder()

.setCPU("Intel i5")

.setRAM("8GB")

.setStorage("256GB SSD")

.build();

basicComputer.displayConfig();

Computer gamingComputer = new Computer.Builder()

.setCPU("Intel i9")

.setRAM("32GB")

.setStorage("1TB SSD")

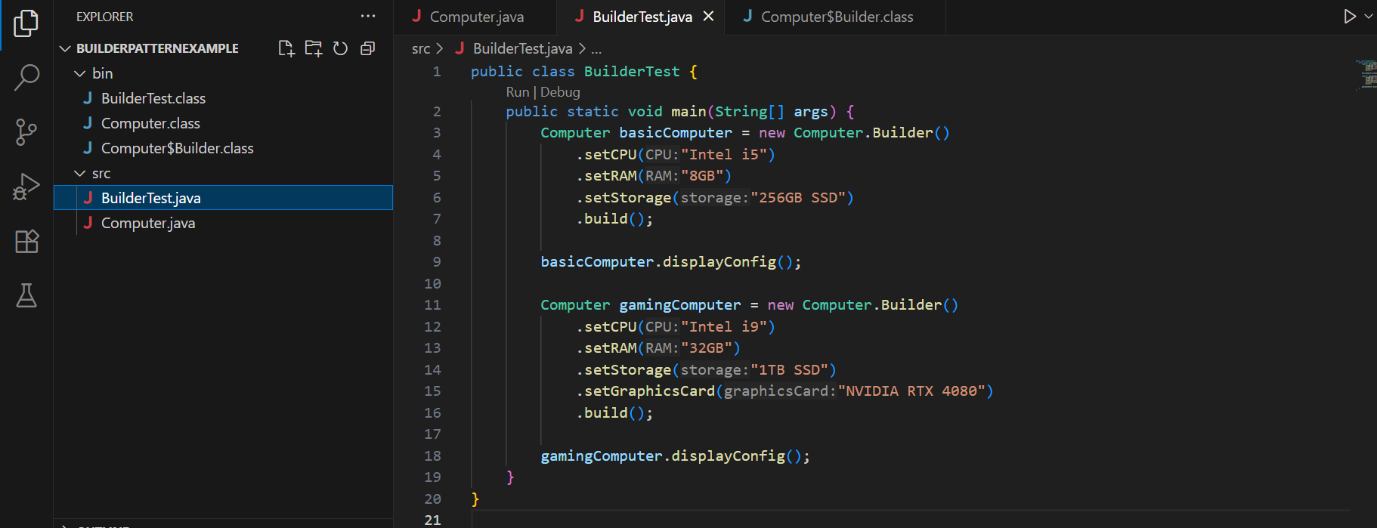
.setGraphicsCard("NVIDIA RTX 4080")

.build();

gamingComputer.displayConfig();

}

}



Output:

