**NAME : RACHANA J N**

**Superset ID :   6416511**

**TASK 7 : Implementing the Observer Pattern**

**Scenario:**

You are developing a stock market monitoring application where multiple clients need to be notified whenever stock prices change. Use the Observer Pattern to achieve this.

**Steps:**

1. **Create a New Java Project:**
   * Create a new Java project named **ObserverPatternExample**.
2. **Define Subject Interface:**
   * Create an interface **Stock** with methods to **register**, **deregister**, and **notify** observers.
3. **Implement Concrete Subject:**
   * Create a class **StockMarket** that implements **Stock** and maintains a list of observers.
4. **Define Observer Interface:**
   * Create an interface Observer with a method **update().**
5. **Implement Concrete Observers:**
   * Create classes **MobileApp**, **WebApp** that implement Observer.
6. **Test the Observer Implementation:**
   * Create a test class to demonstrate the registration and notification of observers.

**CODE :**

import java.util.\*;

public class TestObserver {

    public static void main(String[] args) {

        StockMarket stock = new StockMarket();

        Observer mobile = new MobileApp();

        Observer web = new WebApp();

        stock.register(mobile);

        stock.register(web);

        stock.setPrice(250.50);

        stock.setPrice(275.75);

    } }

interface Stock {

    void register(Observer o);

    void deregister(Observer o);

    void notifyObservers();

}

interface Observer {

    void update(double price);

}

class StockMarket implements Stock {

    private List<Observer> observers = new ArrayList<>();

    private double stockPrice;

    public void setPrice(double price) {

        this.stockPrice = price;

        notifyObservers();

    }

    public void register(Observer o) {

        observers.add(o);

    }

    public void deregister(Observer o) {

        observers.remove(o);

    }

    public void notifyObservers() {

        for (Observer o : observers) {

            o.update(stockPrice);

        }    } }

class MobileApp implements Observer {

    public void update(double price) {

        System.out.println("Mobile App: Stock price updated to ₹" + price);

    } }

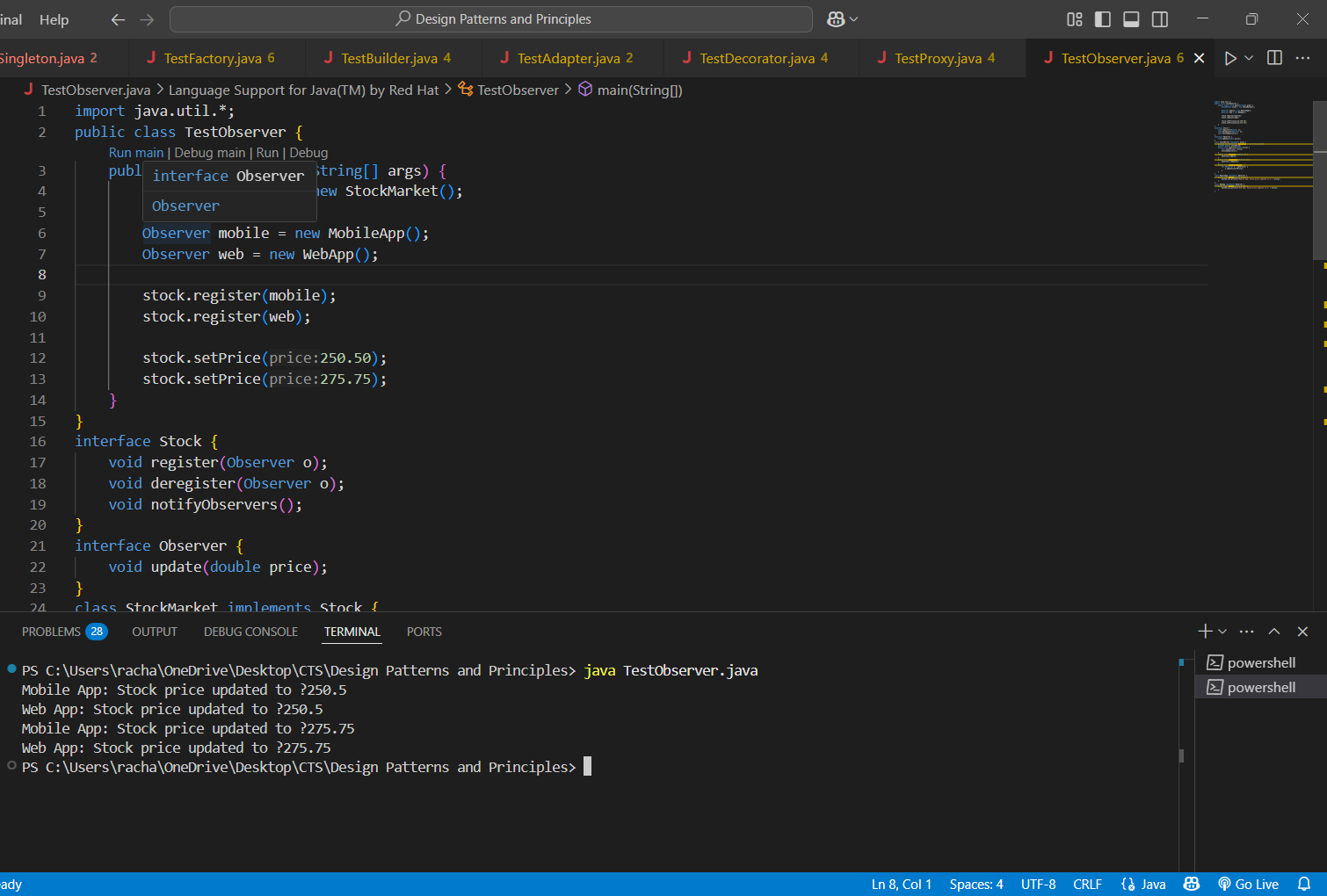
class WebApp implements Observer {

    public void update(double price) {

        System.out.println("Web App: Stock price updated to ₹" + price);

    } }

**OUTPUT :**

****