COMSATS UNIVERSITY ISLAMABAD, ABBOTTABAD

Design patterns lab Assignment # 02

Submitted by:

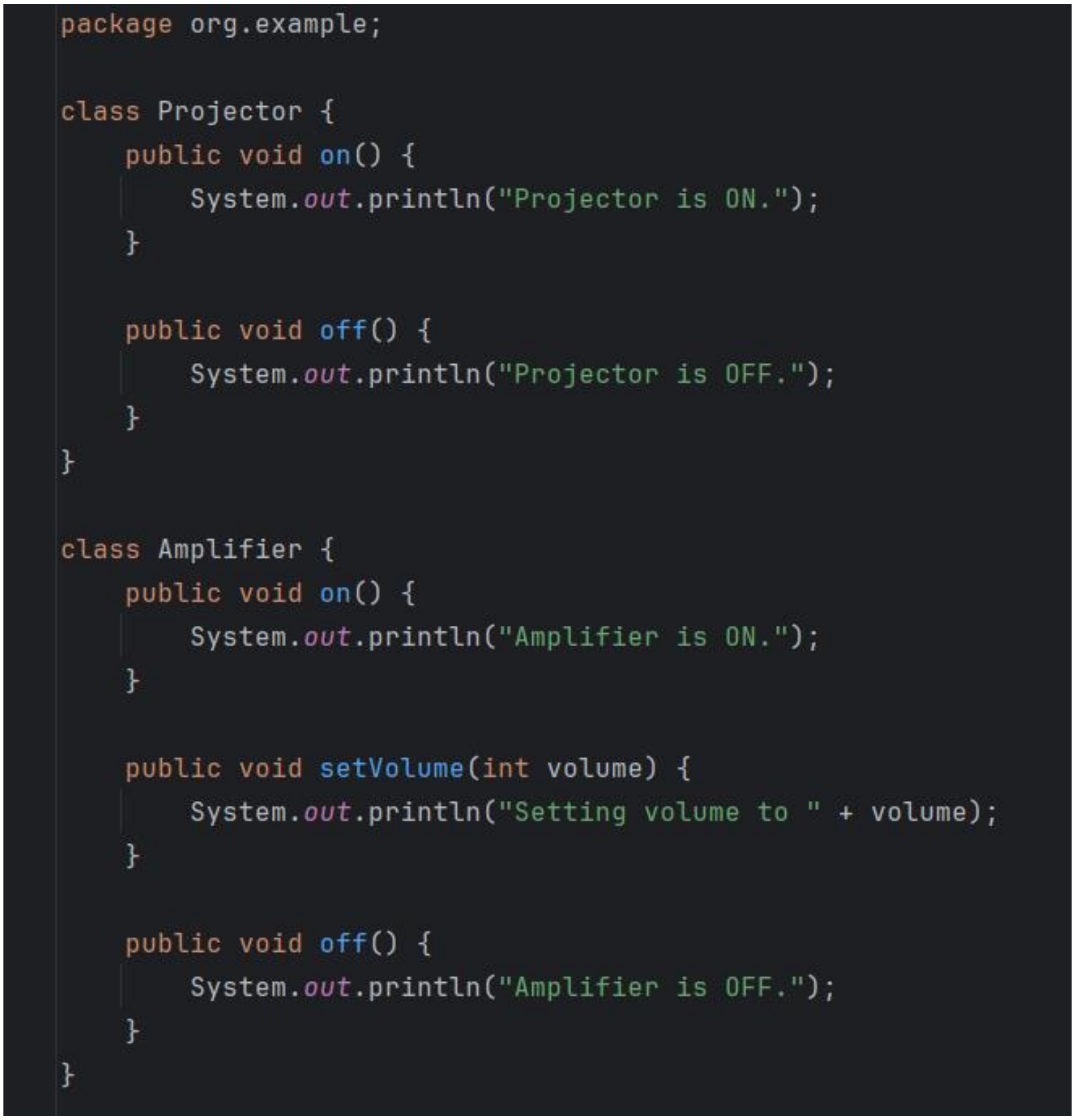
Laiba binte tahir FA21-BSE-019

Contents

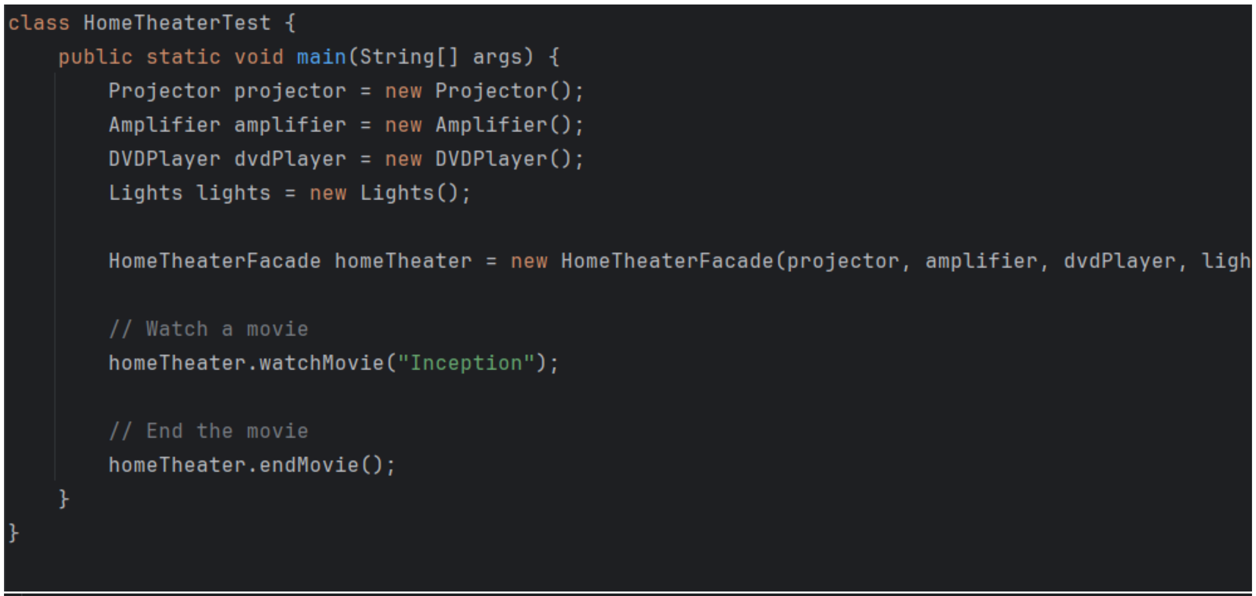
Implementation of Facade pattern

Task -01 Home Theater System

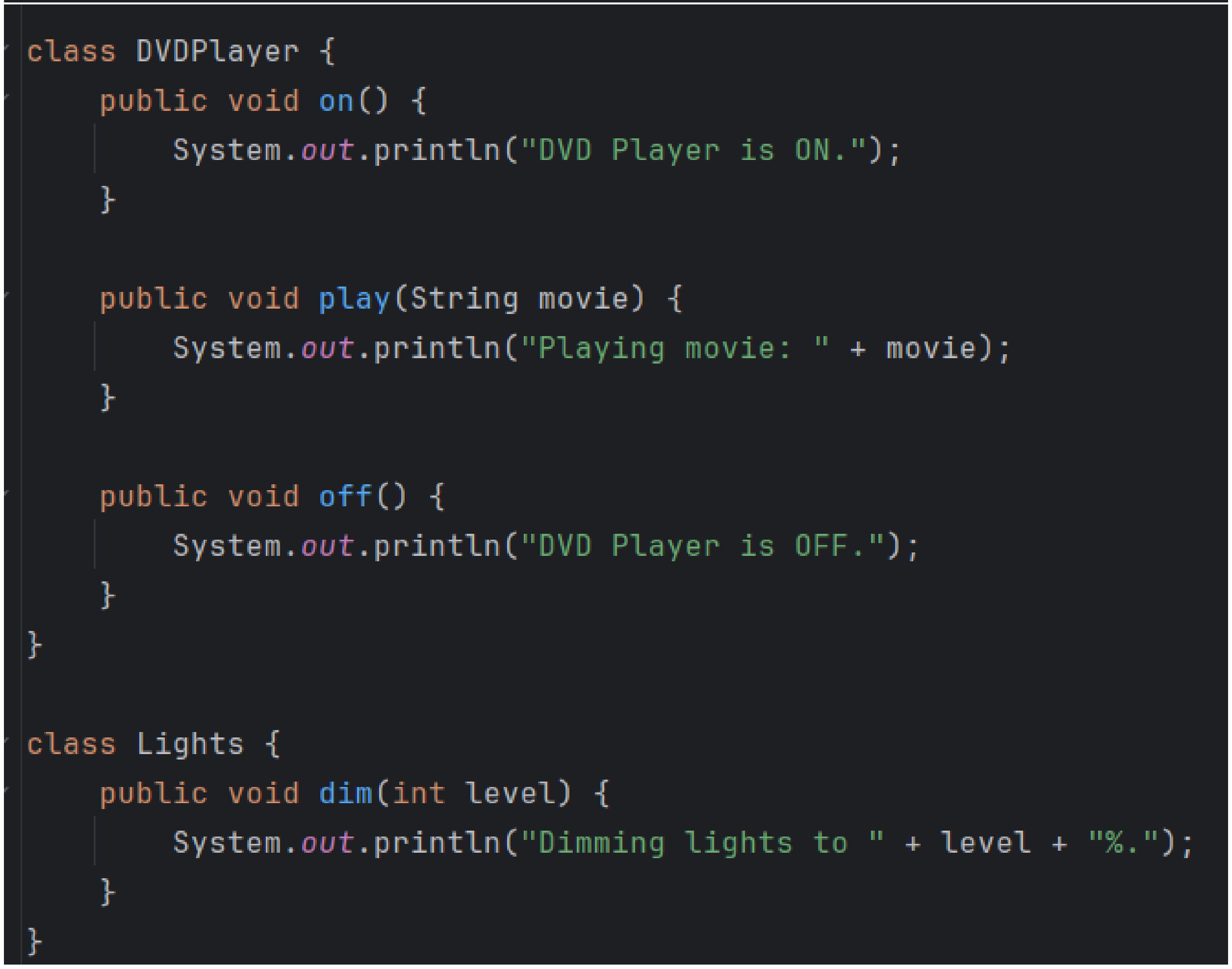
Code



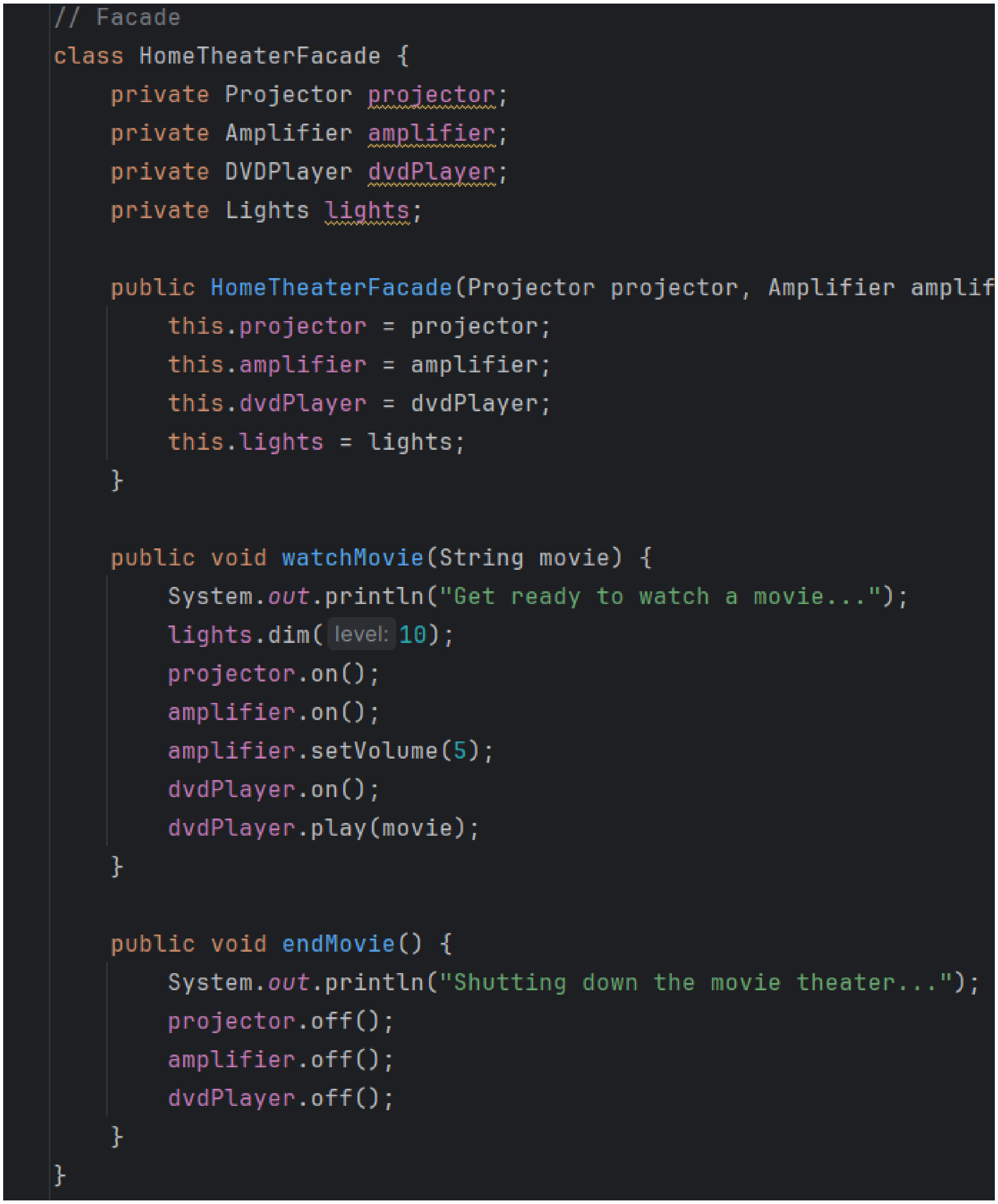
```java
package org.example;

class Projector {
    public void on() {
        System.out.println("Projector is ON.");
    }

    public void off() {
        System.out.println("Projector is OFF.");
    }
}

class Amplifier {
    public void on() {
        System.out.println("Amplifier is ON.");
    }

    public void setVolume(int volume) {
        System.out.println("Setting volume to " + volume);
    }

    public void off() {
        System.out.println("Amplifier is OFF.");
    }
}
```



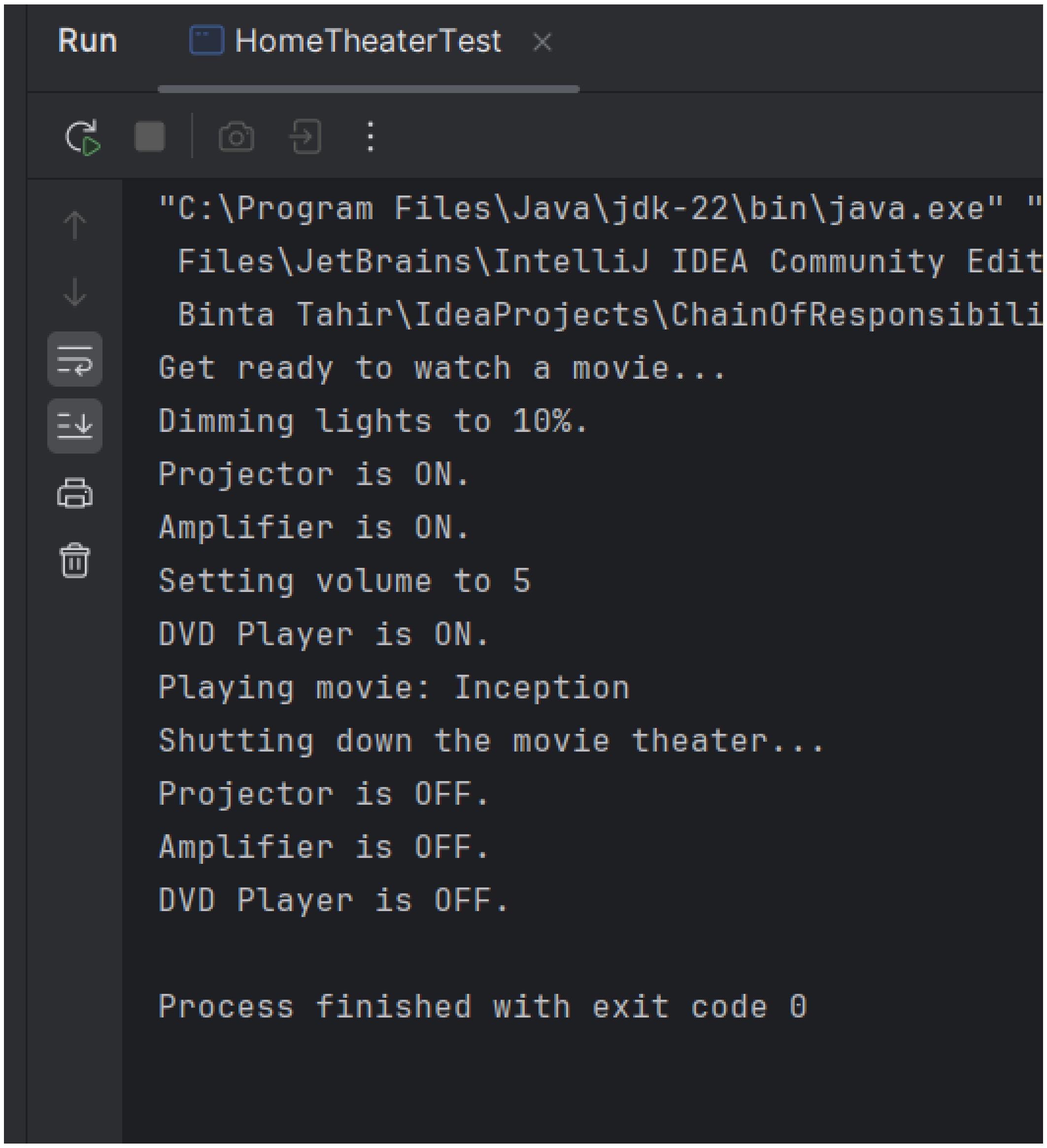
```java
class HomeTheaterTest {
    public static void main(String[] args) {
        Projector projector = new Projector();
        Amplifier amplifier = new Amplifier();
        DVDPlayer dvdPlayer = new DVDPlayer();
        Lights lights = new Lights();

        HomeTheaterFacade homeTheater = new HomeTheaterFacade(projector, amplifier, dvdPlayer, ligh

        // Watch a movie
        homeTheater.watchMovie("Inception");

        // End the movie
        homeTheater.endMovie();
    }
}
```



```java
class DVDPlayer {
    public void on() {
        System.out.println("DVD Player is ON.");
    }

    public void play(String movie) {
        System.out.println("Playing movie: " + movie);
    }

    public void off() {
        System.out.println("DVD Player is OFF.");
    }
}

class Lights {
    public void dim(int level) {
        System.out.println("Dimming lights to " + level + "%.");
    }
}
```

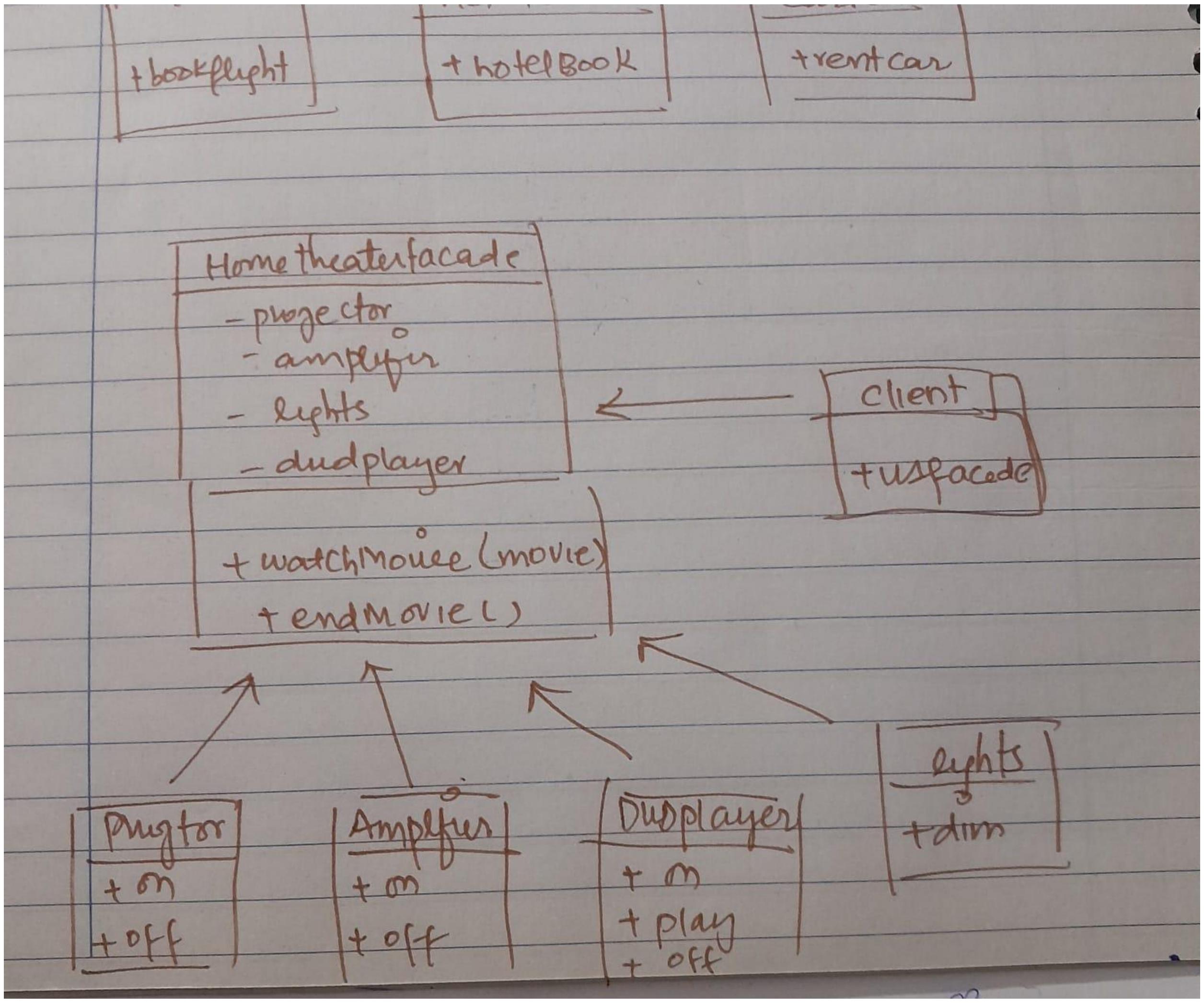


```java
// Facade
class HomeTheaterFacade {
    private Projector projector;
    private Amplifier amplifier;
    private DVDPlayer dvdPlayer;
    private Lights lights;

    public HomeTheaterFacade(Projector projector, Amplifier amplif
        this.projector = projector;
        this.amplifier = amplifier;
        this.dvdPlayer = dvdPlayer;
        this.lights = lights;
    }

    public void watchMovie(String movie) {
        System.out.println("Get ready to watch a movie...");
        lights.dim( level: 10);
        projector.on();
        amplifier.on();
        amplifier.setVolume(5);
        dvdPlayer.on();
        dvdPlayer.play(movie);
    }

    public void endMovie() {
        System.out.println("Shutting down the movie theater...");
        projector.off();
        amplifier.off();
        dvdPlayer.off();
    }
}
```



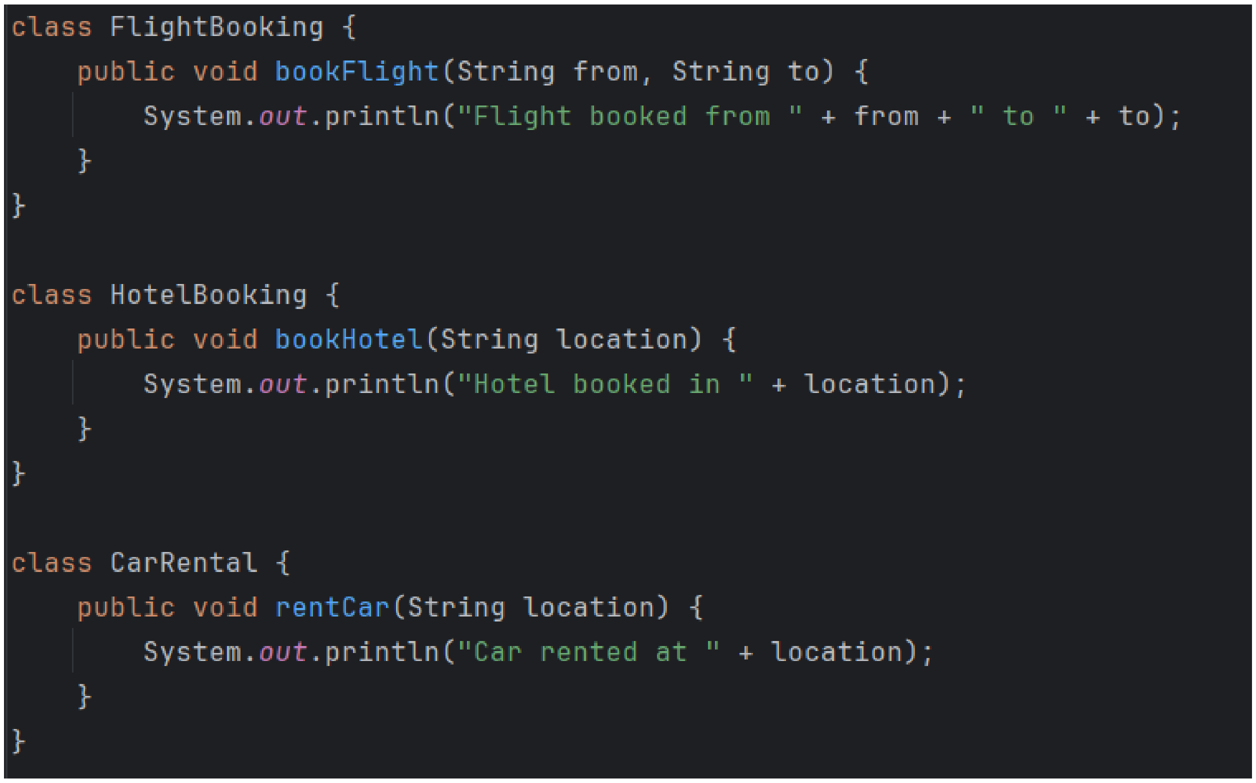
```
"C:\Program Files\Java\jdk-22\bin\java.exe" "
 Files\JetBrains\IntelliJ IDEA Community Edit
 Binta Tahir\IdeaProjects\ChainOfResponsibili
Get ready to watch a movie...
Dimming lights to 10%.
Projector is ON.
Amplifier is ON.
Setting volume to 5
DVD Player is ON.
Playing movie: Inception
Shutting down the movie theater...
Projector is OFF.
Amplifier is OFF.
DVD Player is OFF.

Process finished with exit code 0
```

Class diagram

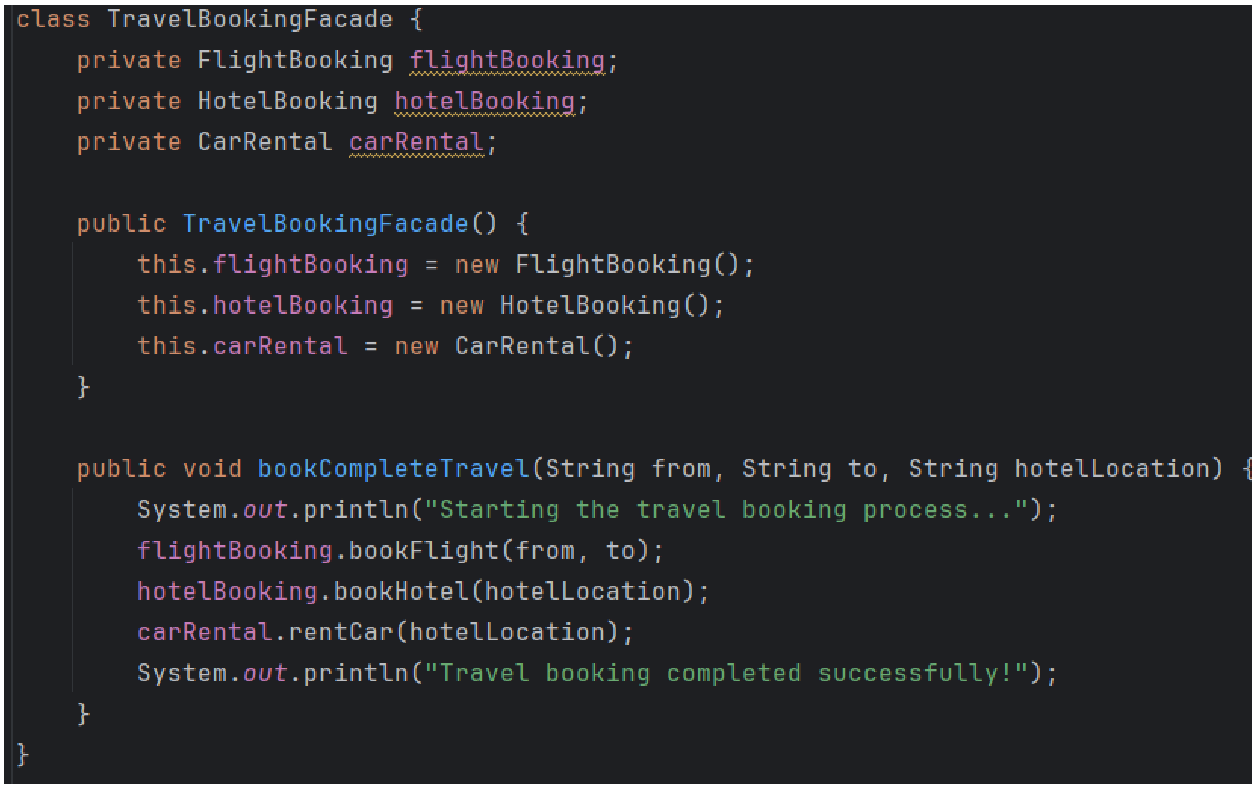


Task-02 Flight booking

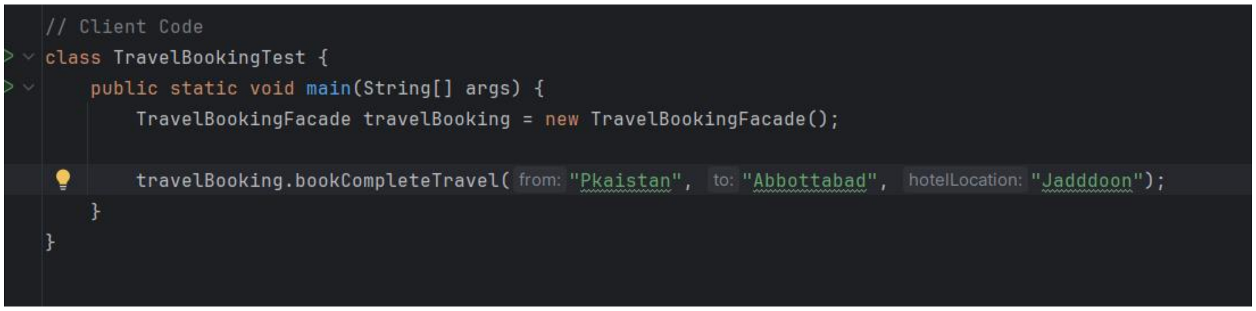
Code



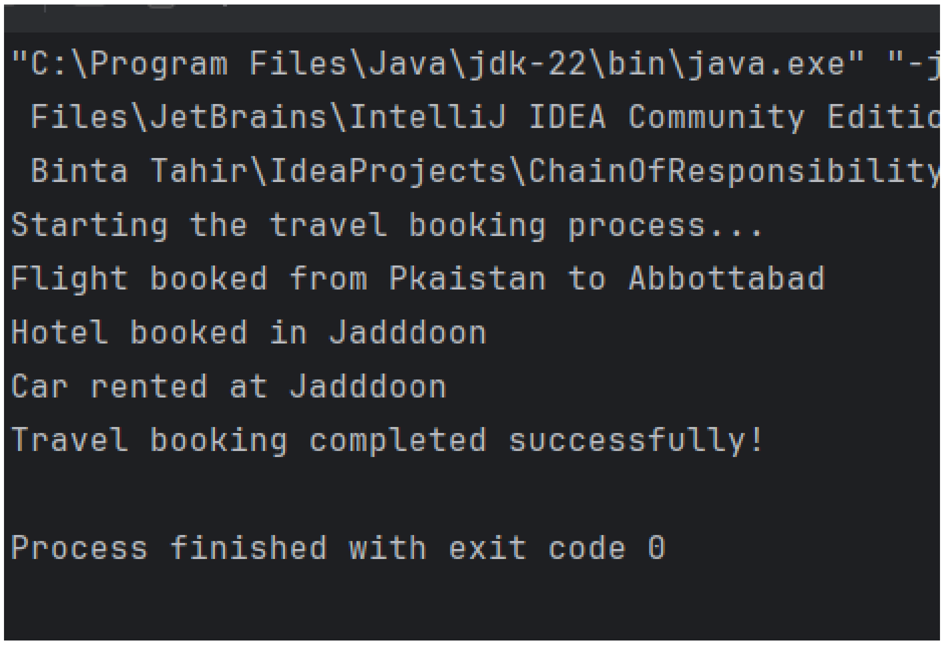
```java
class FlightBooking {
    public void bookFlight(String from, String to) {
        System.out.println("Flight booked from " + from + " to " + to);
    }
}

class HotelBooking {
    public void bookHotel(String location) {
        System.out.println("Hotel booked in " + location);
    }
}

class CarRental {
    public void rentCar(String location) {
        System.out.println("Car rented at " + location);
    }
}
```



```java
class TravelBookingFacade {
    private FlightBooking flightBooking;
    private HotelBooking hotelBooking;
    private CarRental carRental;

    public TravelBookingFacade() {
        this.flightBooking = new FlightBooking();
        this.hotelBooking = new HotelBooking();
        this.carRental = new CarRental();
    }

    public void bookCompleteTravel(String from, String to, String hotelLocation) {
        System.out.println("Starting the travel booking process...");
        flightBooking.bookFlight(from, to);
        hotelBooking.bookHotel(hotelLocation);
        carRental.rentCar(hotelLocation);
        System.out.println("Travel booking completed successfully!");
    }
}
```

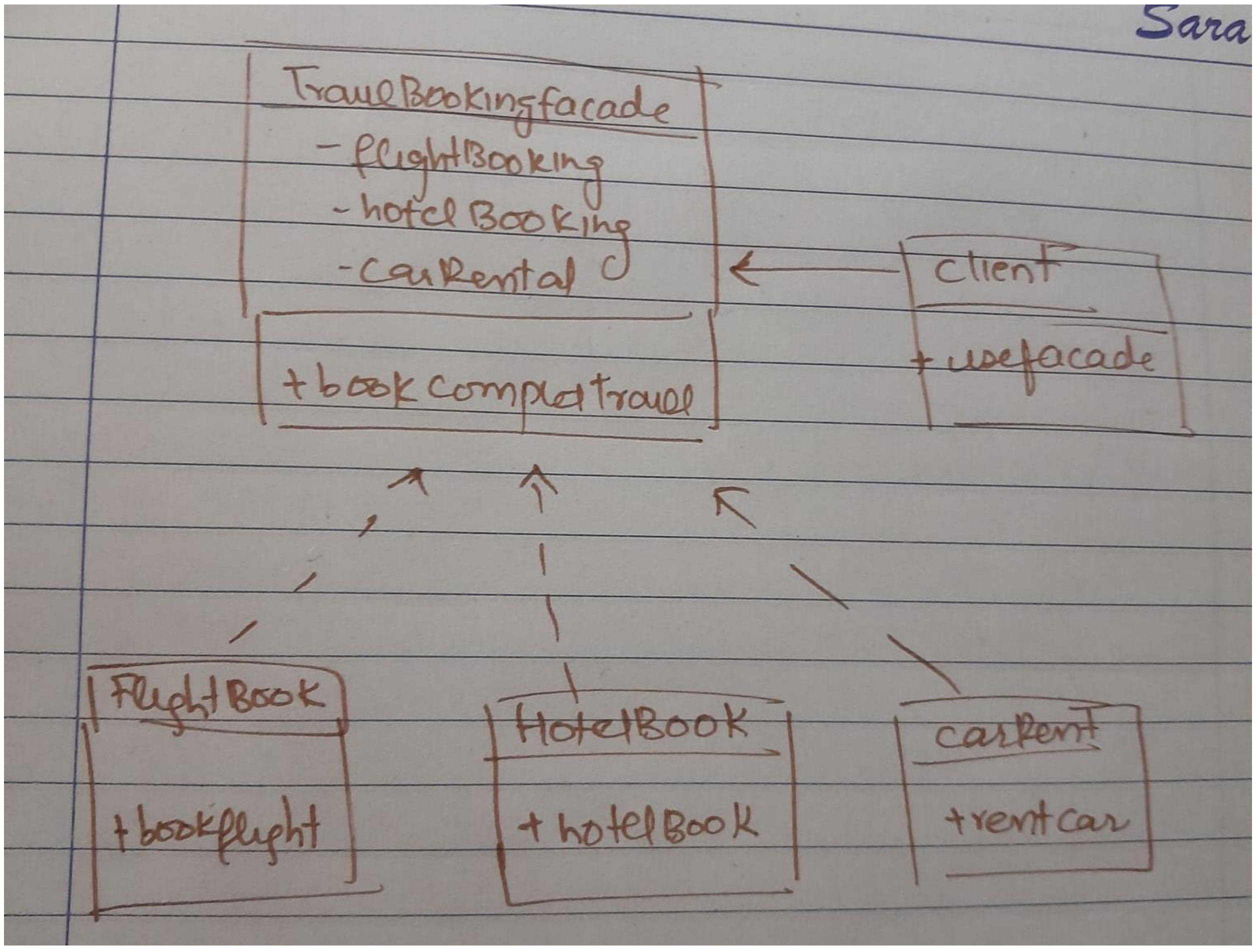


```java
// Client Code
class TravelBookingTest {
    public static void main(String[] args) {
        TravelBookingFacade travelBooking = new TravelBookingFacade();

        travelBooking.bookCompleteTravel( from: "Pkaistan",  to: "Abbottabad",  hotelLocation: "Jadddoon");
    }
}
```



```
"C:\Program Files\Java\jdk-22\bin\java.exe" "-j
 Files\JetBrains\IntelliJ IDEA Community Editic
 Binta Tahir\IdeaProjects\ChainOfResponsibility
Starting the travel booking process...
Flight booked from Pkaistan to Abbottabad
Hotel booked in Jadddoon
Car rented at Jadddoon
Travel booking completed successfully!

Process finished with exit code 0
```

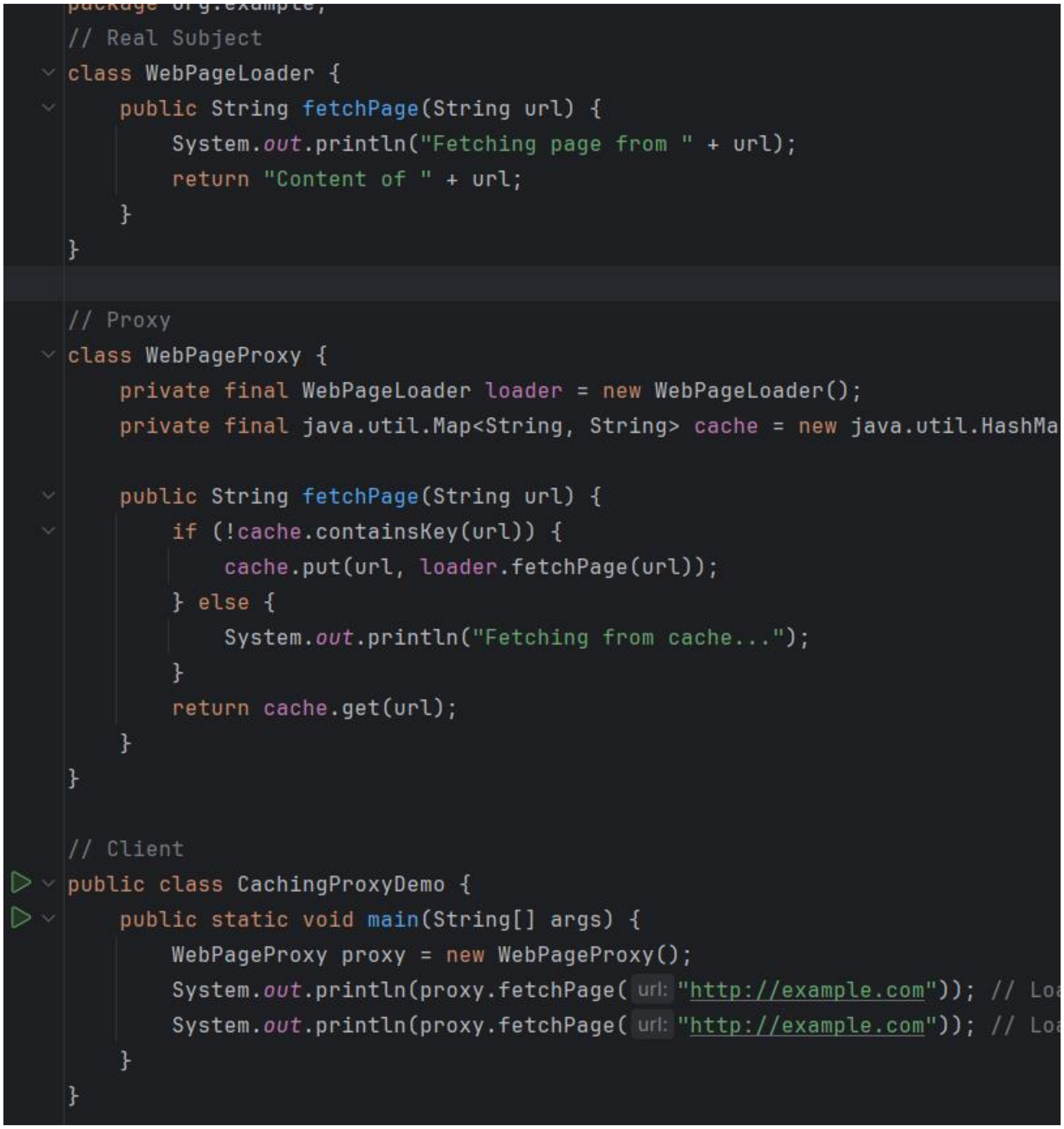
Class diagram



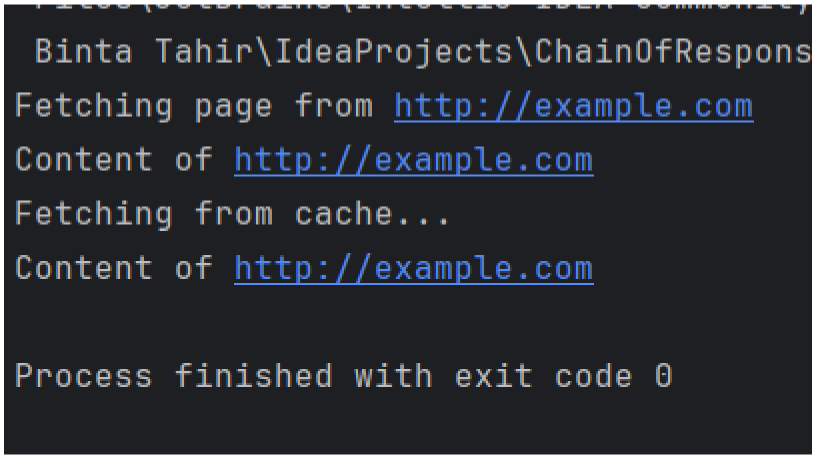
Implementation of proxy design pattern:

Task 1: Caching Proxy in Web Applications

Code

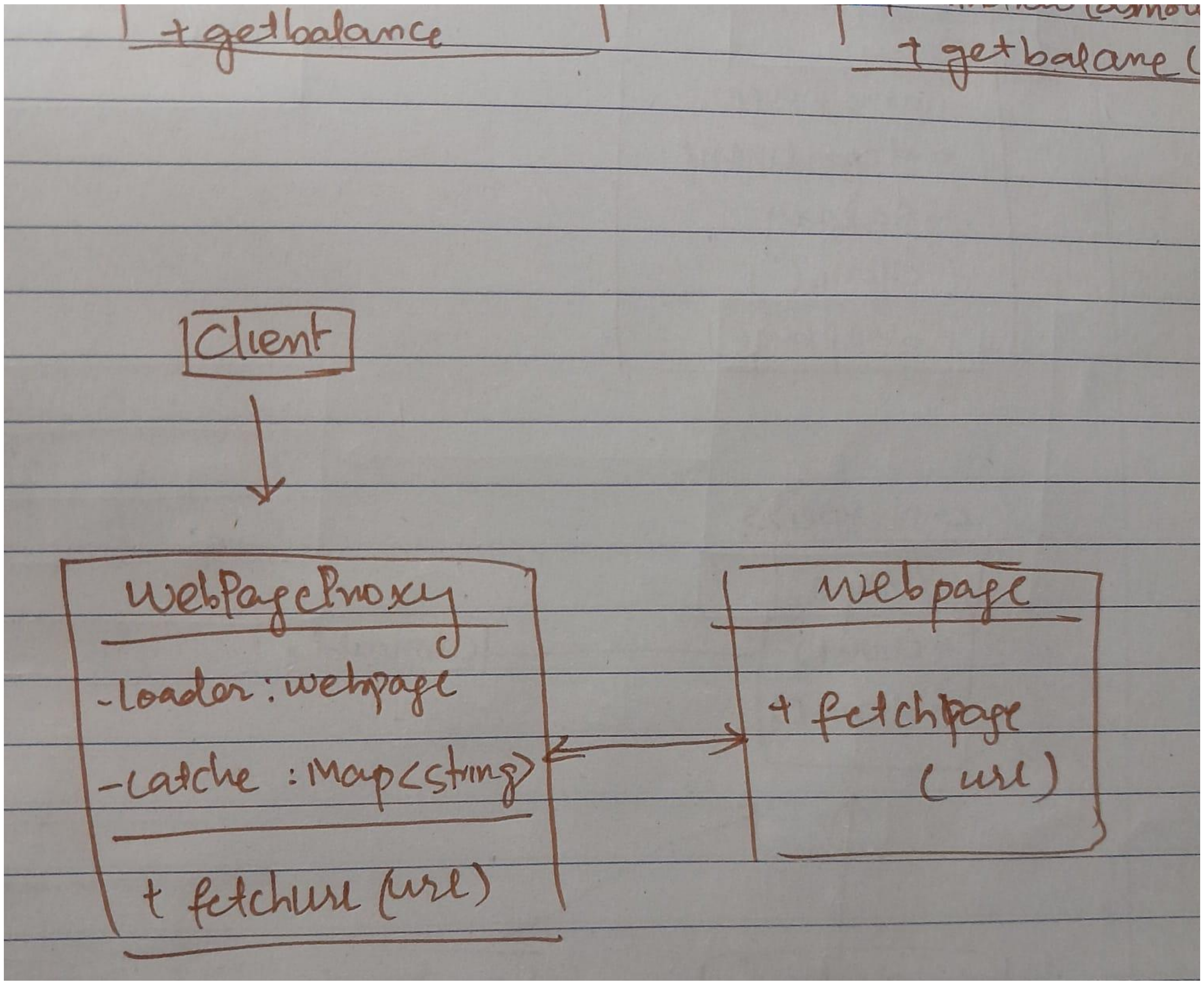


```
package org.example;
// Real Subject
class WebPageLoader {
    public String fetchPage(String url) {
        System.out.println("Fetching page from " + url);
        return "Content of " + url;
    }
}

// Proxy
class WebPageProxy {
    private final WebPageLoader loader = new WebPageLoader();
    private final java.util.Map<String, String> cache = new java.util.HashMa

    public String fetchPage(String url) {
        if (!cache.containsKey(url)) {
            cache.put(url, loader.fetchPage(url));
        } else {
            System.out.println("Fetching from cache...");
        }
        return cache.get(url);
    }
}

// Client
public class CachingProxyDemo {
    public static void main(String[] args) {
        WebPageProxy proxy = new WebPageProxy();
        System.out.println(proxy.fetchPage( url: "http://example.com")); // Loa
        System.out.println(proxy.fetchPage( url: "http://example.com")); // Loa
    }
}
```



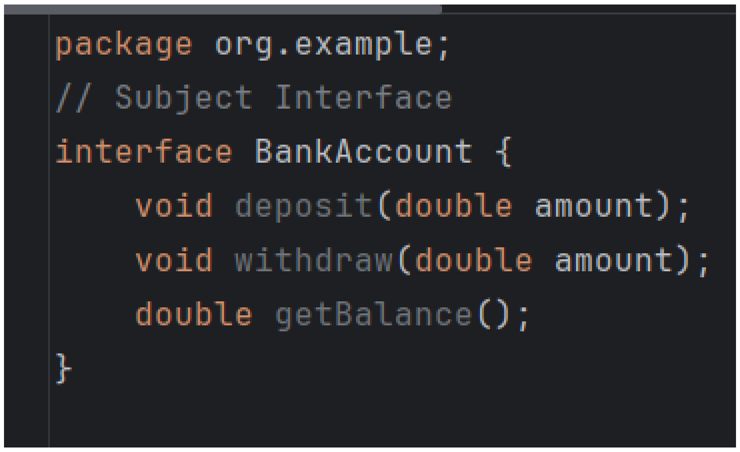
```
 Binta Tahir\IdeaProjects\ChainOfRespons
Fetching page from http://example.com
Content of http://example.com
Fetching from cache...
Content of http://example.com

Process finished with exit code 0
```

Class diagram

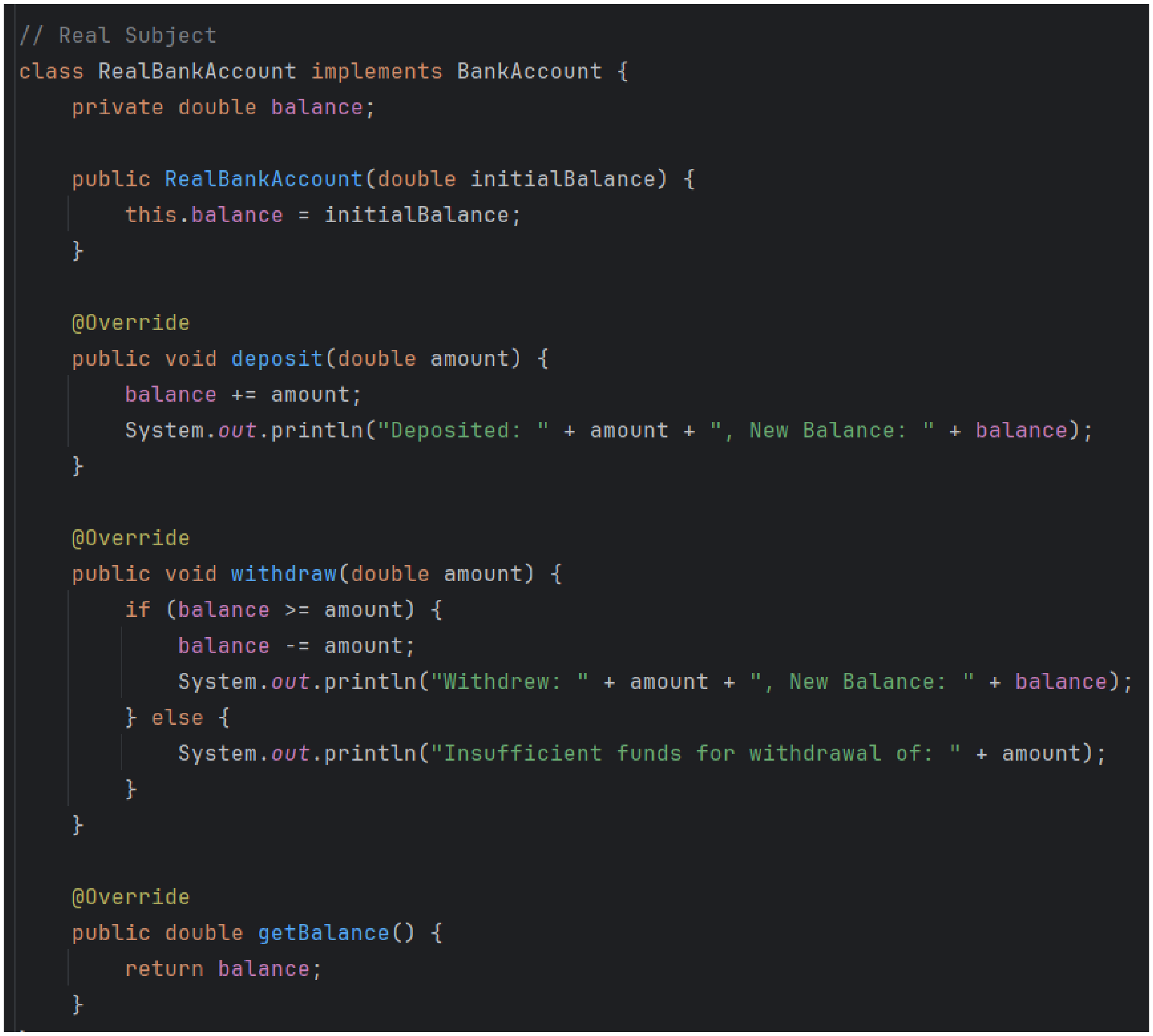


Task 2: Protection Proxy in Online Banking Systems

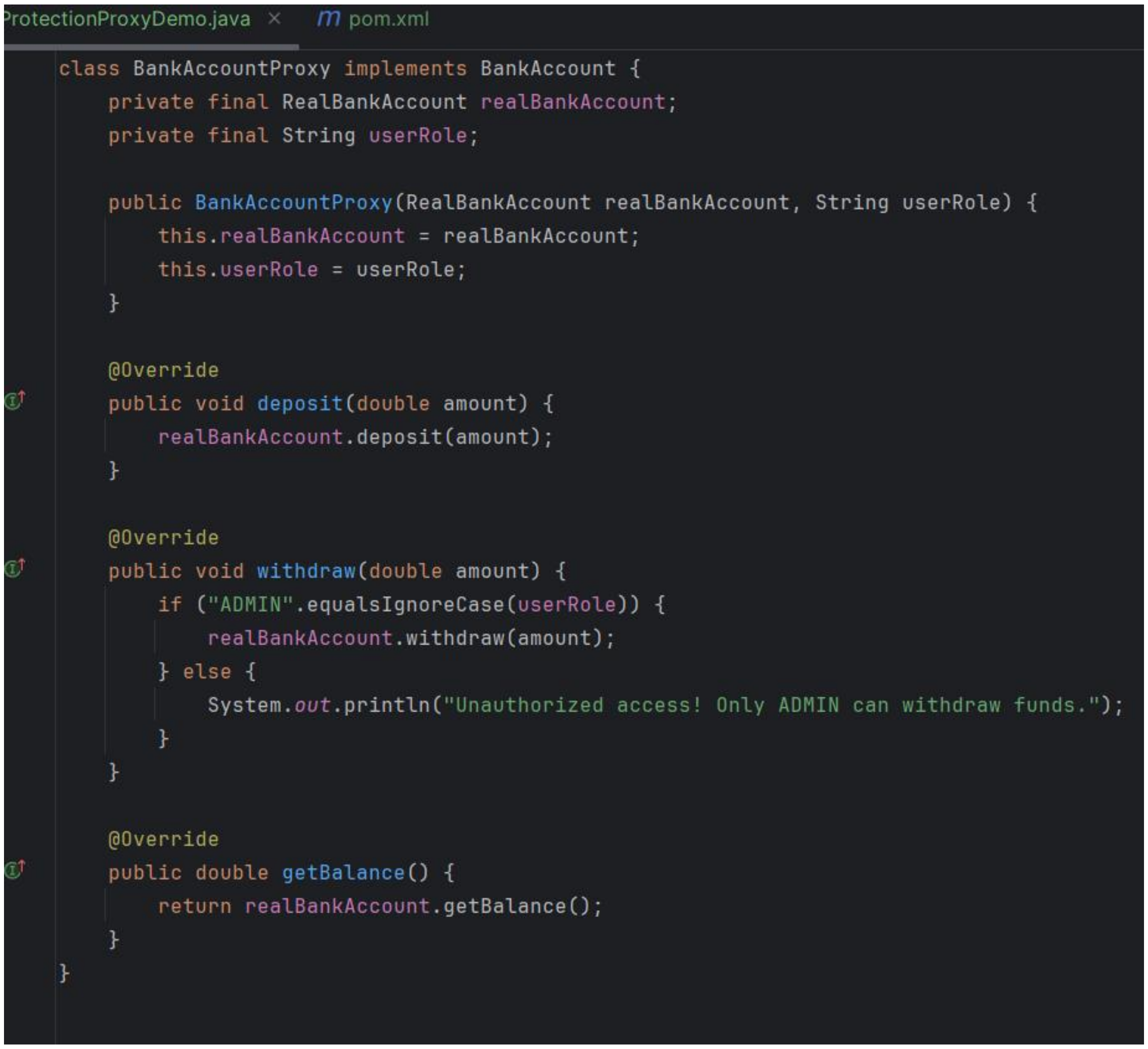
Code



```java
package org.example;
// Subject Interface
interface BankAccount {
    void deposit(double amount);
    void withdraw(double amount);
    double getBalance();
}
```

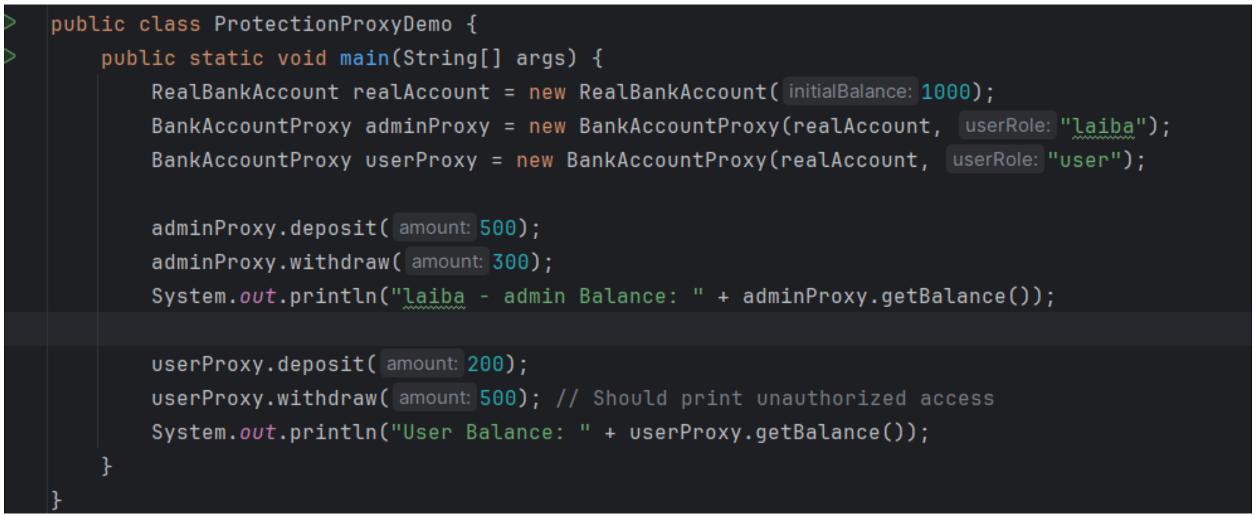


```java
// Real Subject
class RealBankAccount implements BankAccount {
    private double balance;

    public RealBankAccount(double initialBalance) {
        this.balance = initialBalance;
    }

    @Override
    public void deposit(double amount) {
        balance += amount;
        System.out.println("Deposited: " + amount + ", New Balance: " + balance);
    }

    @Override
    public void withdraw(double amount) {
        if (balance >= amount) {
            balance -= amount;
            System.out.println("Withdrew: " + amount + ", New Balance: " + balance);
        } else {
            System.out.println("Insufficient funds for withdrawal of: " + amount);
        }
    }

    @Override
    public double getBalance() {
        return balance;
    }
```

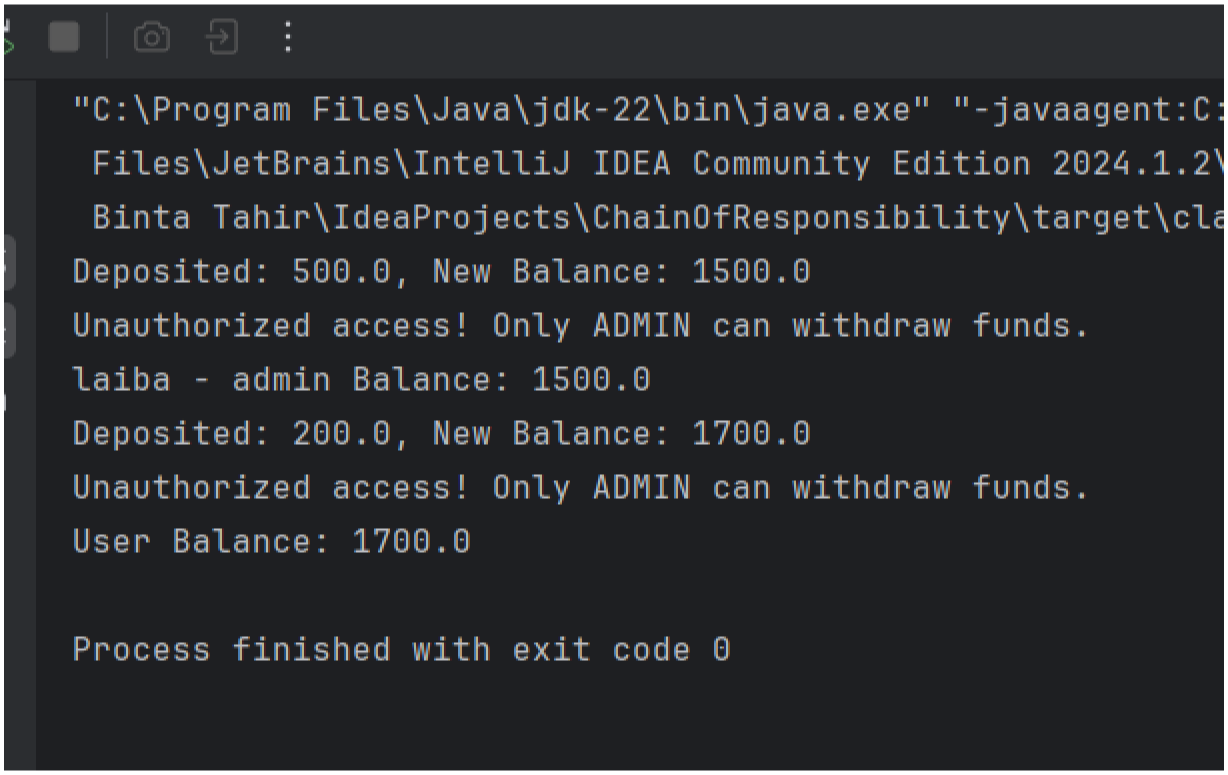


ProtectionProxyDemo.java × m pom.xml

```java
class BankAccountProxy implements BankAccount {
    private final RealBankAccount realBankAccount;
    private final String userRole;

    public BankAccountProxy(RealBankAccount realBankAccount, String userRole) {
        this.realBankAccount = realBankAccount;
        this.userRole = userRole;
    }

    @Override
    public void deposit(double amount) {
        realBankAccount.deposit(amount);
    }

    @Override
    public void withdraw(double amount) {
        if ("ADMIN".equalsIgnoreCase(userRole)) {
            realBankAccount.withdraw(amount);
        } else {
            System.out.println("Unauthorized access! Only ADMIN can withdraw funds.");
        }
    }

    @Override
    public double getBalance() {
        return realBankAccount.getBalance();
    }
}
```

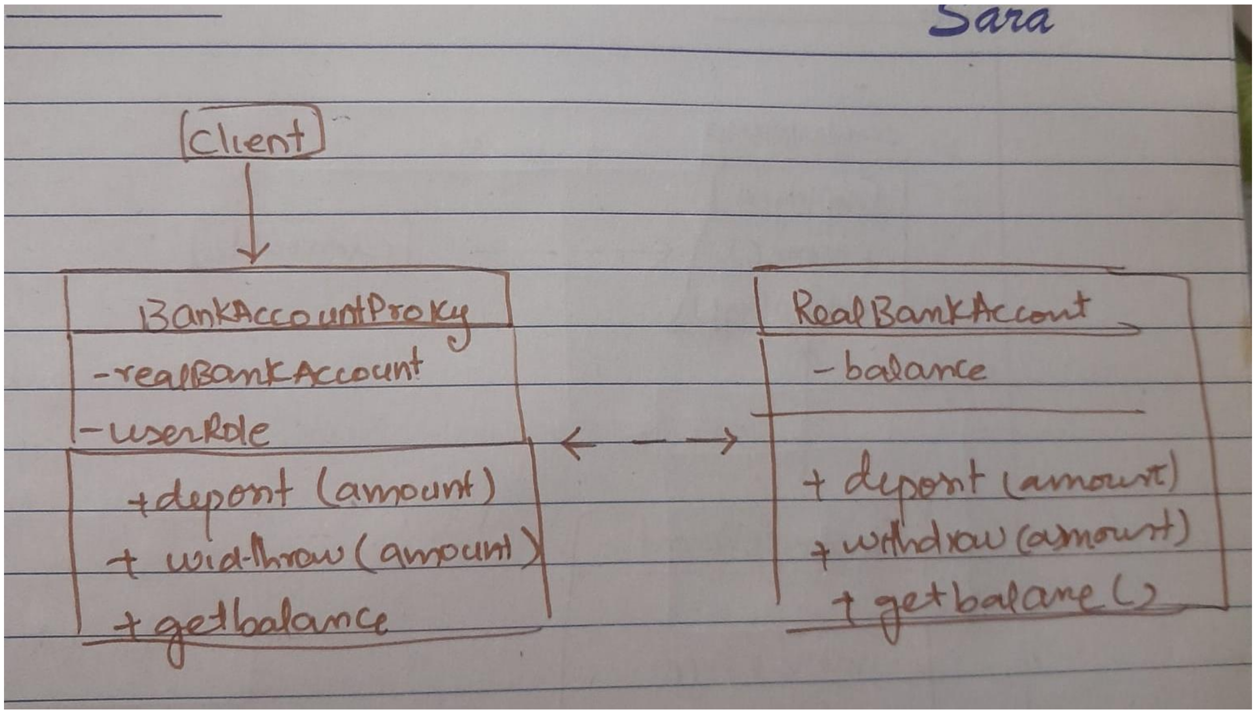


```java
public class ProtectionProxyDemo {
    public static void main(String[] args) {
        RealBankAccount realAccount = new RealBankAccount( initialBalance: 1000);
        BankAccountProxy adminProxy = new BankAccountProxy(realAccount,  userRole: "laiba");
        BankAccountProxy userProxy = new BankAccountProxy(realAccount,  userRole: "user");

        adminProxy.deposit( amount: 500);
        adminProxy.withdraw( amount: 300);
        System.out.println("laiba - admin Balance: " + adminProxy.getBalance());

        userProxy.deposit( amount: 200);
        userProxy.withdraw( amount: 500); // Should print unauthorized access
        System.out.println("User Balance: " + userProxy.getBalance());
    }
}
```



```
"C:\Program Files\Java\jdk-22\bin\java.exe" "-javaagent:C:
 Files\JetBrains\IntelliJ IDEA Community Edition 2024.1.2\
 Binta Tahir\IdeaProjects\ChainOfResponsibility\target\cla
Deposited: 500.0, New Balance: 1500.0
Unauthorized access! Only ADMIN can withdraw funds.
laiba - admin Balance: 1500.0
Deposited: 200.0, New Balance: 1700.0
Unauthorized access! Only ADMIN can withdraw funds.
User Balance: 1700.0

Process finished with exit code 0
```

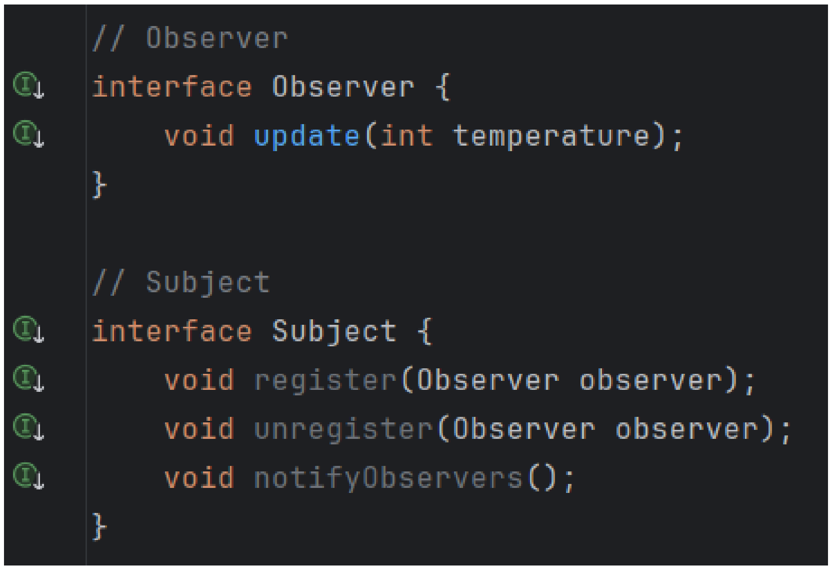
Class Diagram



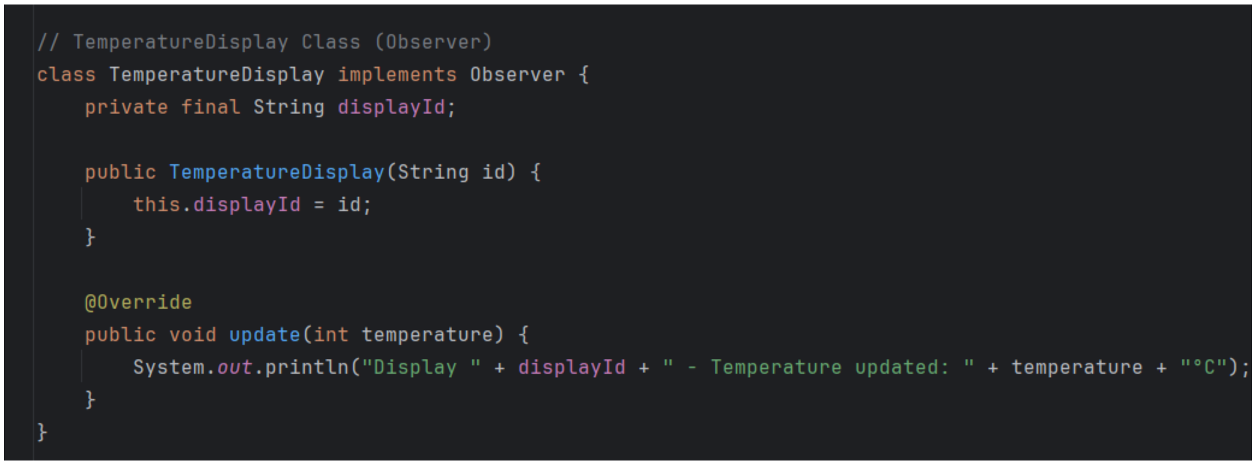
Implementation of observer pattern

Task -01 Weather application

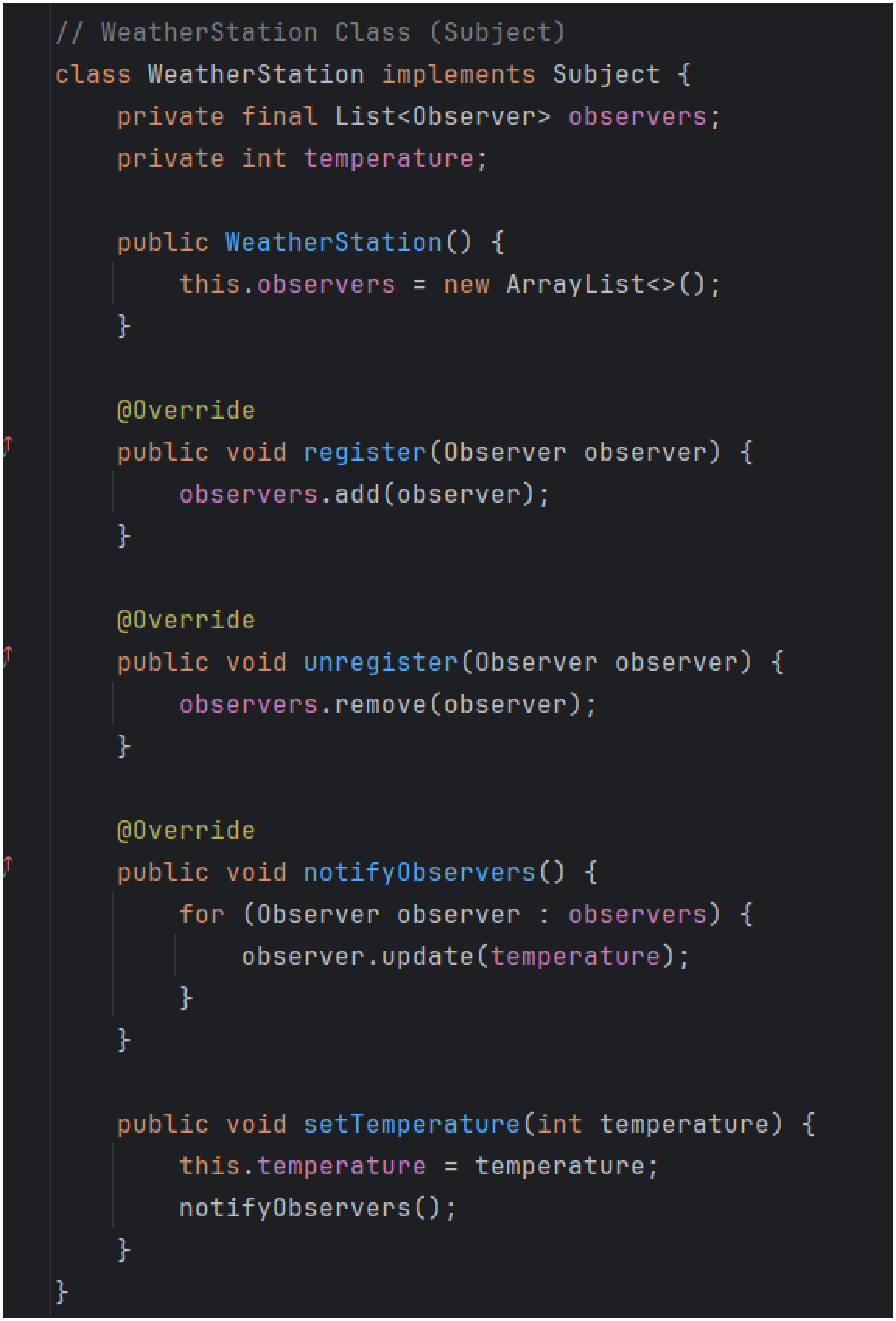
Code



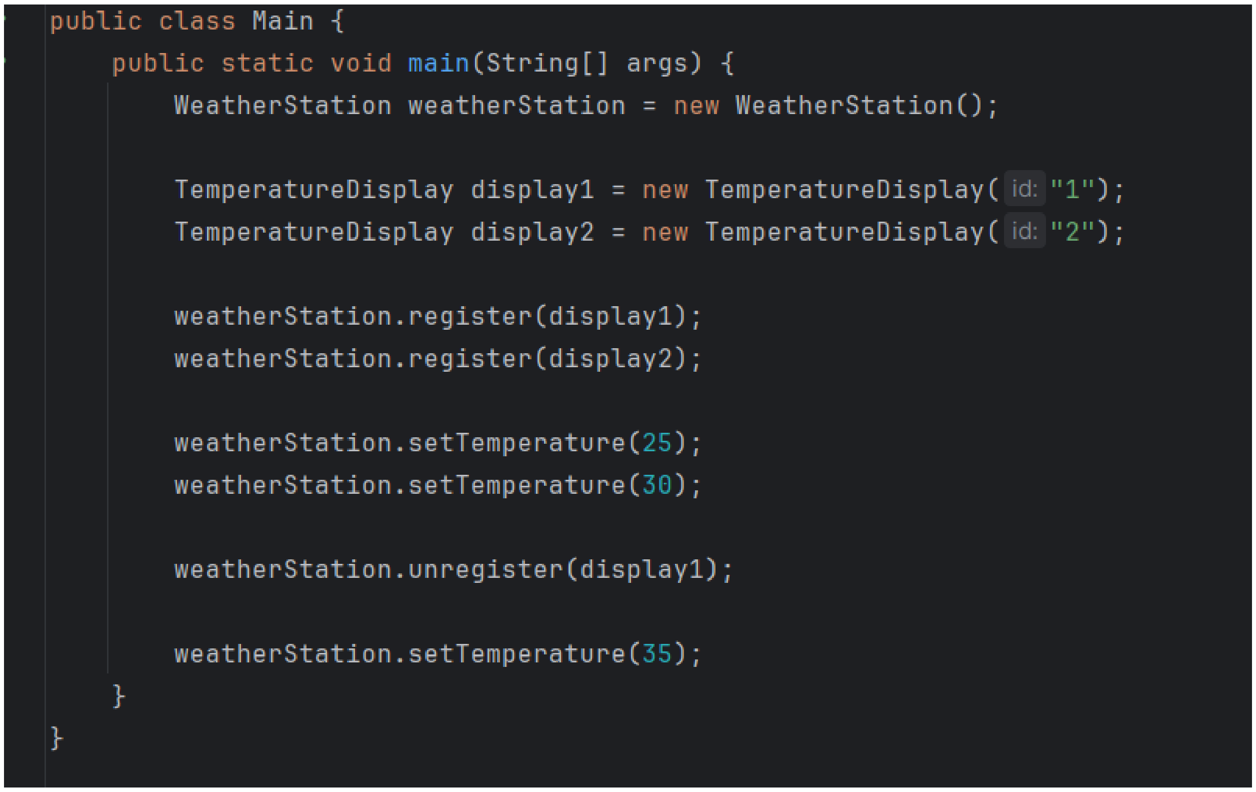
```java
// Observer
interface Observer {
    void update(int temperature);
}

// Subject
interface Subject {
    void register(Observer observer);
    void unregister(Observer observer);
    void notifyObservers();
}
```



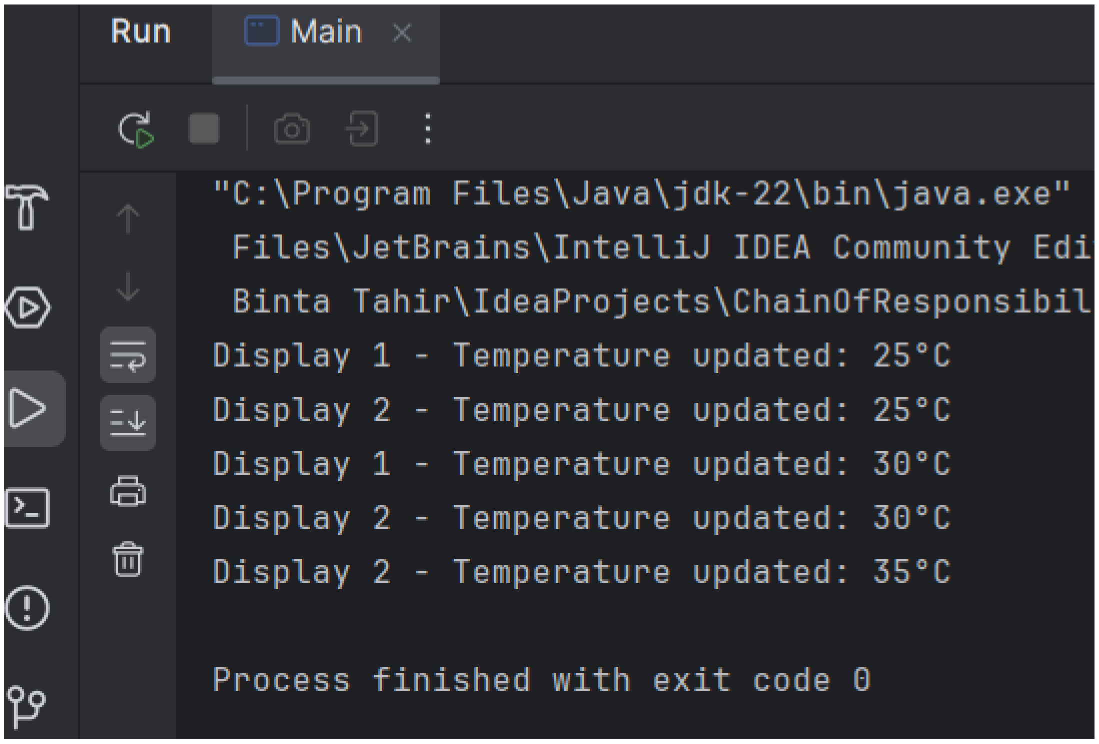
```java
// TemperatureDisplay Class (Observer)
class TemperatureDisplay implements Observer {
    private final String displayId;

    public TemperatureDisplay(String id) {
        this.displayId = id;
    }

    @Override
    public void update(int temperature) {
        System.out.println("Display " + displayId + " - Temperature updated: " + temperature + "°C");
    }
}
```



```java
// WeatherStation Class (Subject)
class WeatherStation implements Subject {
    private final List<Observer> observers;
    private int temperature;

    public WeatherStation() {
        this.observers = new ArrayList<>();
    }

    @Override
    public void register(Observer observer) {
        observers.add(observer);
    }

    @Override
    public void unregister(Observer observer) {
        observers.remove(observer);
    }

    @Override
    public void notifyObservers() {
        for (Observer observer : observers) {
            observer.update(temperature);
        }
    }

    public void setTemperature(int temperature) {
        this.temperature = temperature;
        notifyObservers();
    }
}
```

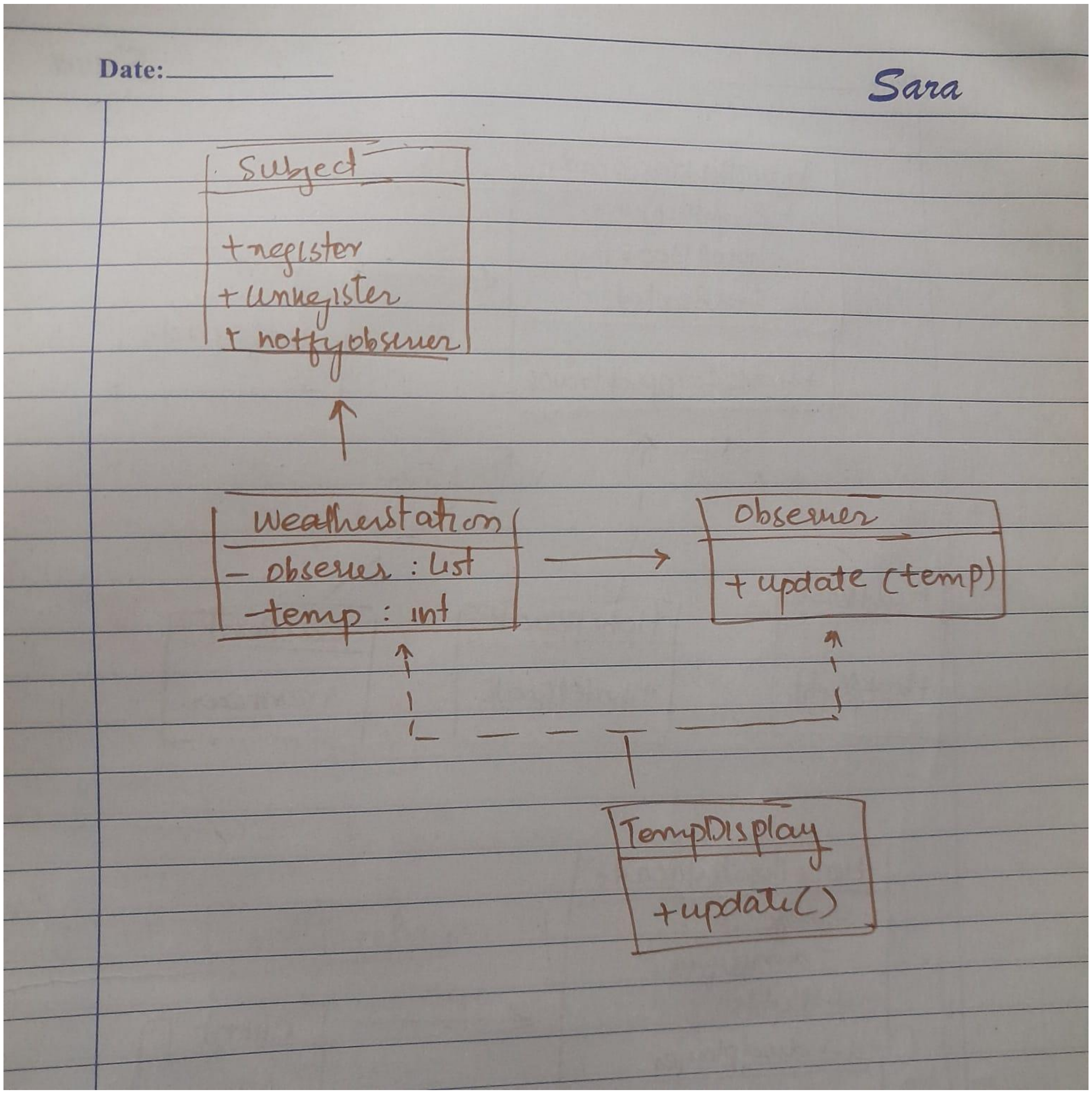


```java
public class Main {
    public static void main(String[] args) {
        WeatherStation weatherStation = new WeatherStation();

        TemperatureDisplay display1 = new TemperatureDisplay( id: "1");
        TemperatureDisplay display2 = new TemperatureDisplay( id: "2");

        weatherStation.register(display1);
        weatherStation.register(display2);

        weatherStation.setTemperature(25);
        weatherStation.setTemperature(30);

        weatherStation.unregister(display1);

        weatherStation.setTemperature(35);
    }
}
```



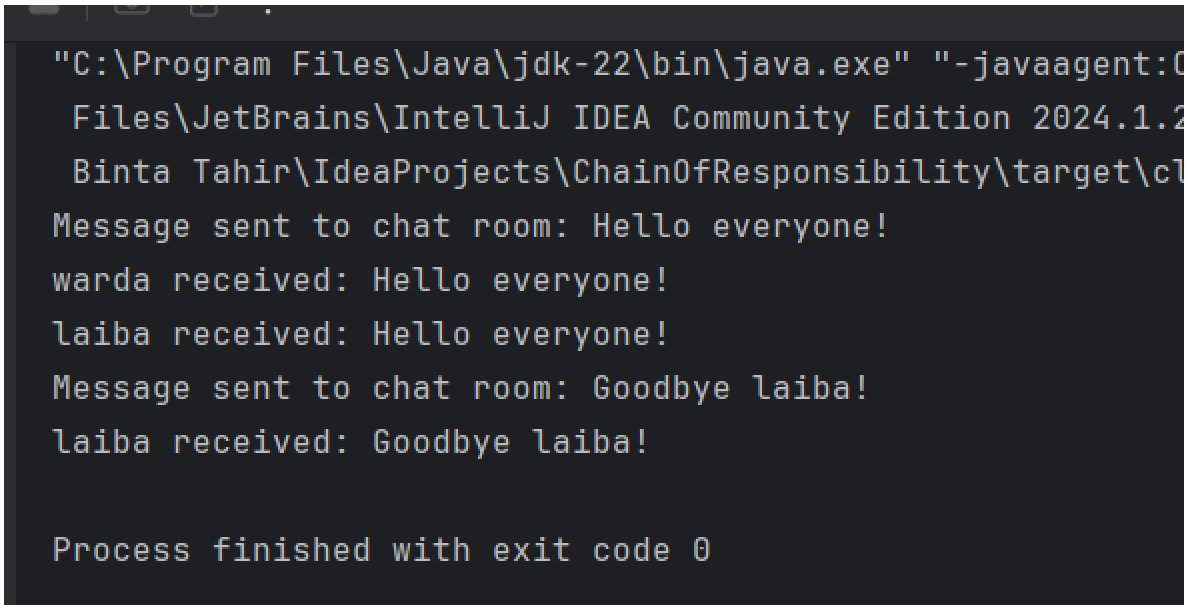
```
"C:\Program Files\Java\jdk-22\bin\java.exe"
 Files\JetBrains\IntelliJ IDEA Community Edi
 Binta Tahir\IdeaProjects\ChainOfResponsibil
Display 1 - Temperature updated: 25°C
Display 2 - Temperature updated: 25°C
Display 1 - Temperature updated: 30°C
Display 2 - Temperature updated: 30°C
Display 2 - Temperature updated: 35°C

Process finished with exit code 0
```

Class diagram

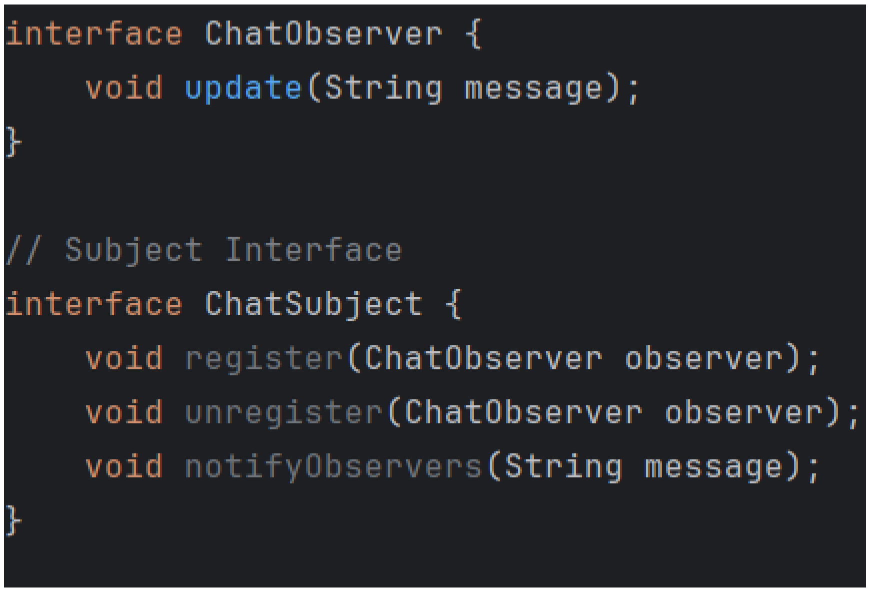


Task -02 Chat application

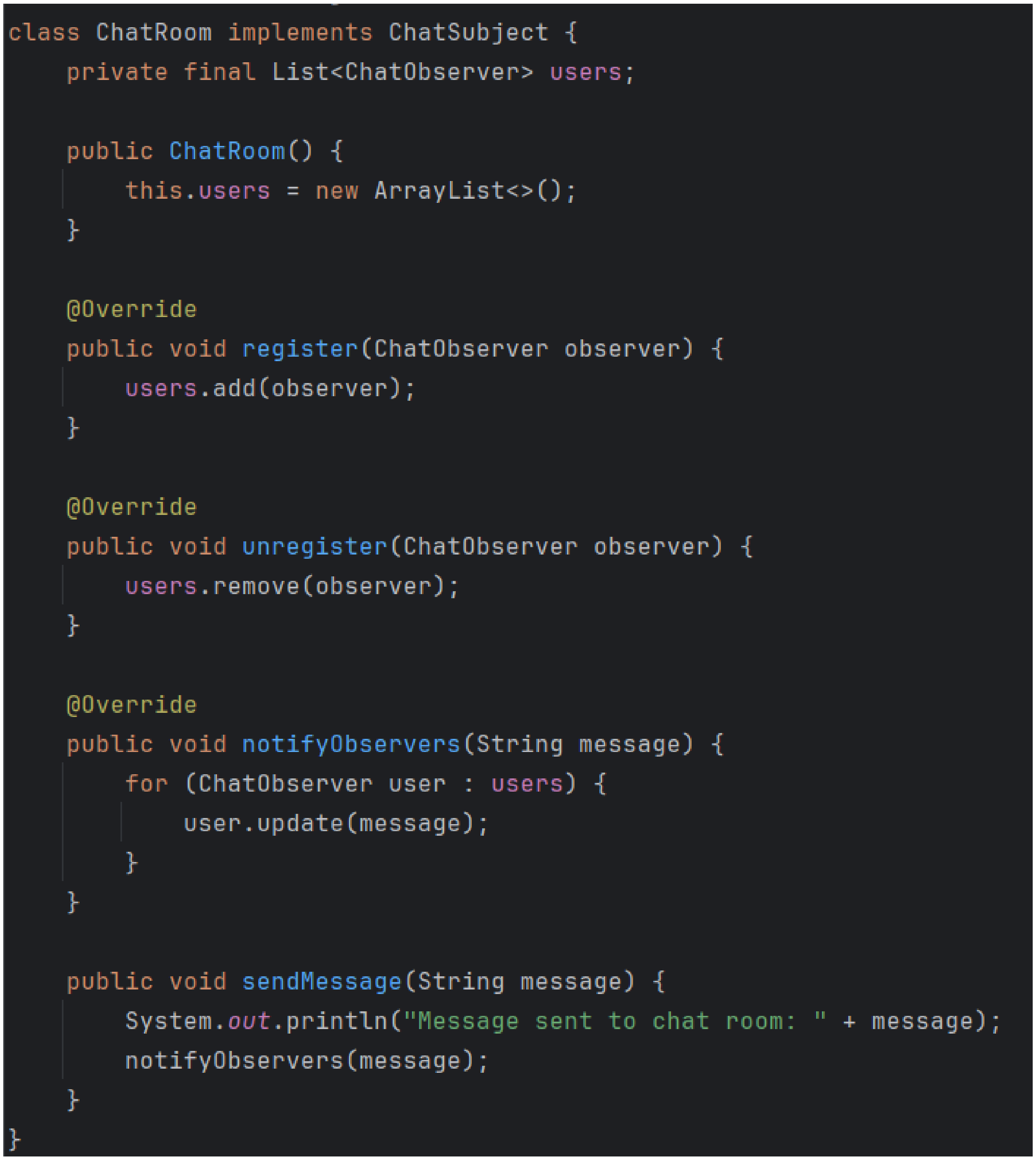
Code



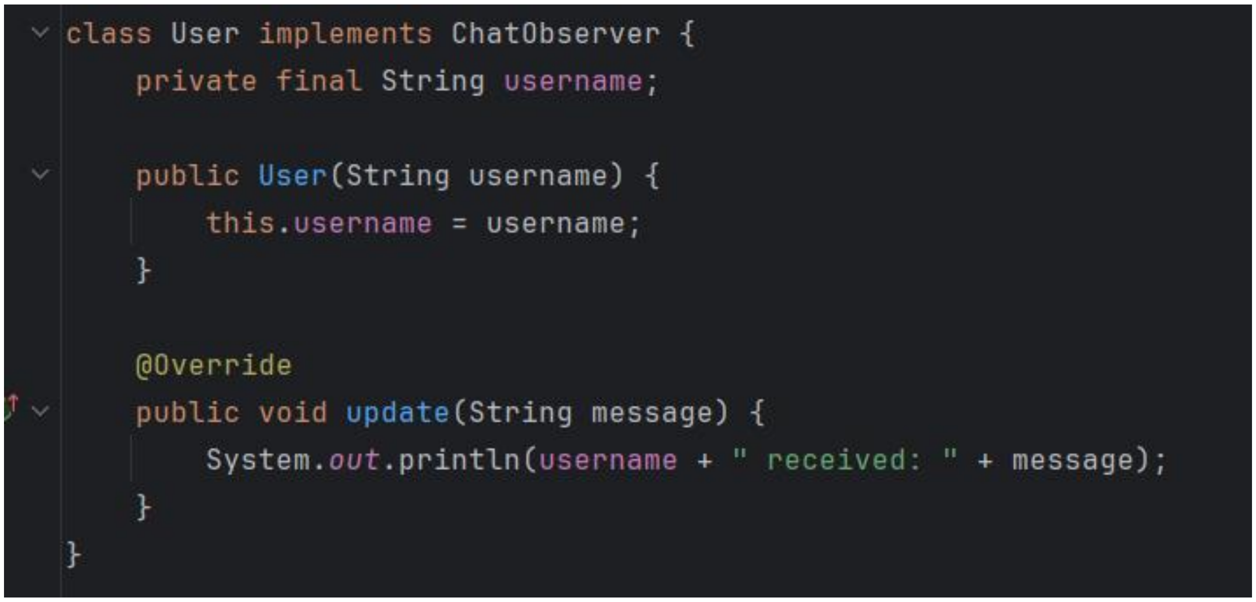
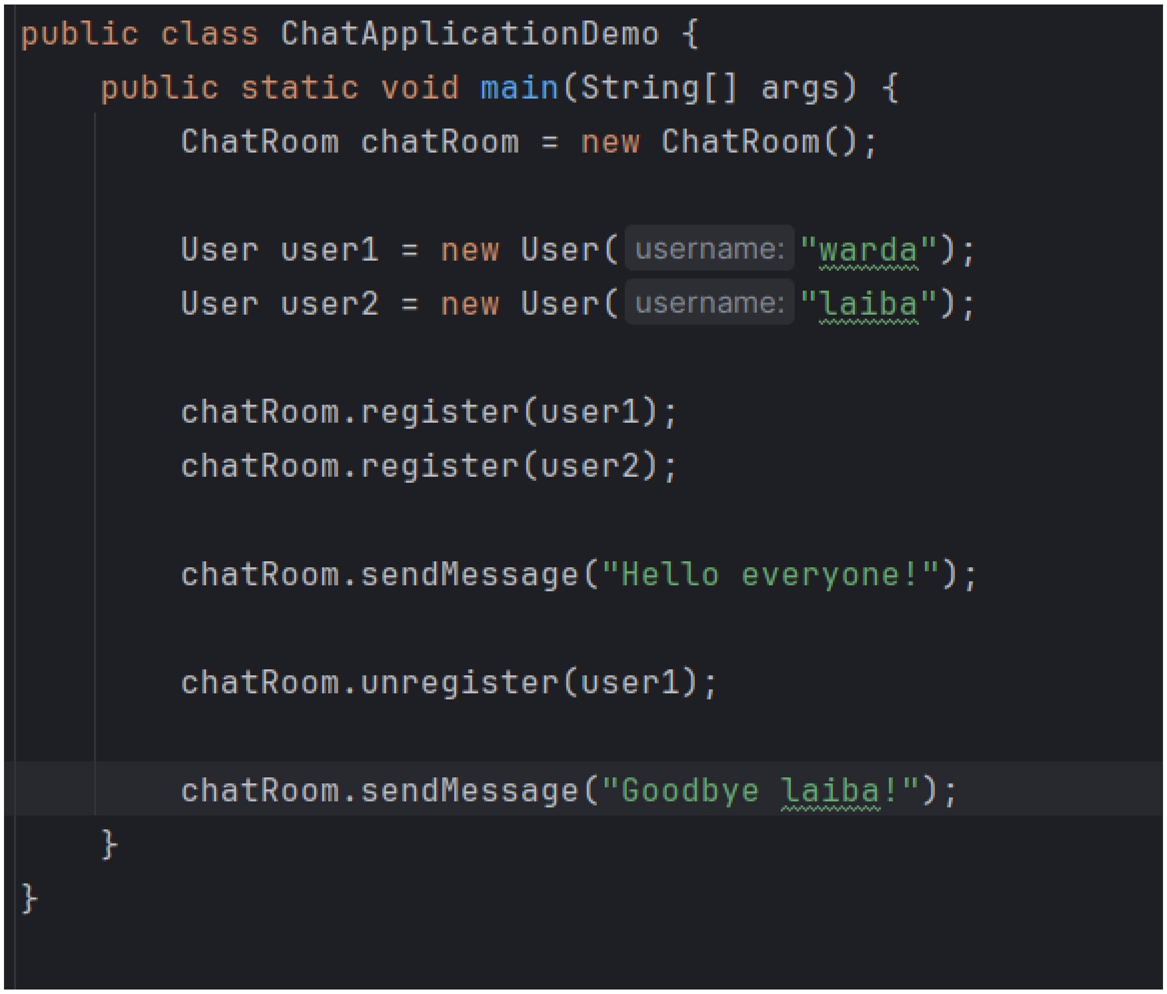
```
"C:\Program Files\Java\jdk-22\bin\java.exe" "-javaagent:C
 Files\JetBrains\IntelliJ IDEA Community Edition 2024.1.2
 Binta Tahir\IdeaProjects\ChainOfResponsibility\target\cl
Message sent to chat room: Hello everyone!
warda received: Hello everyone!
laiba received: Hello everyone!
Message sent to chat room: Goodbye laiba!
laiba received: Goodbye laiba!

Process finished with exit code 0
```



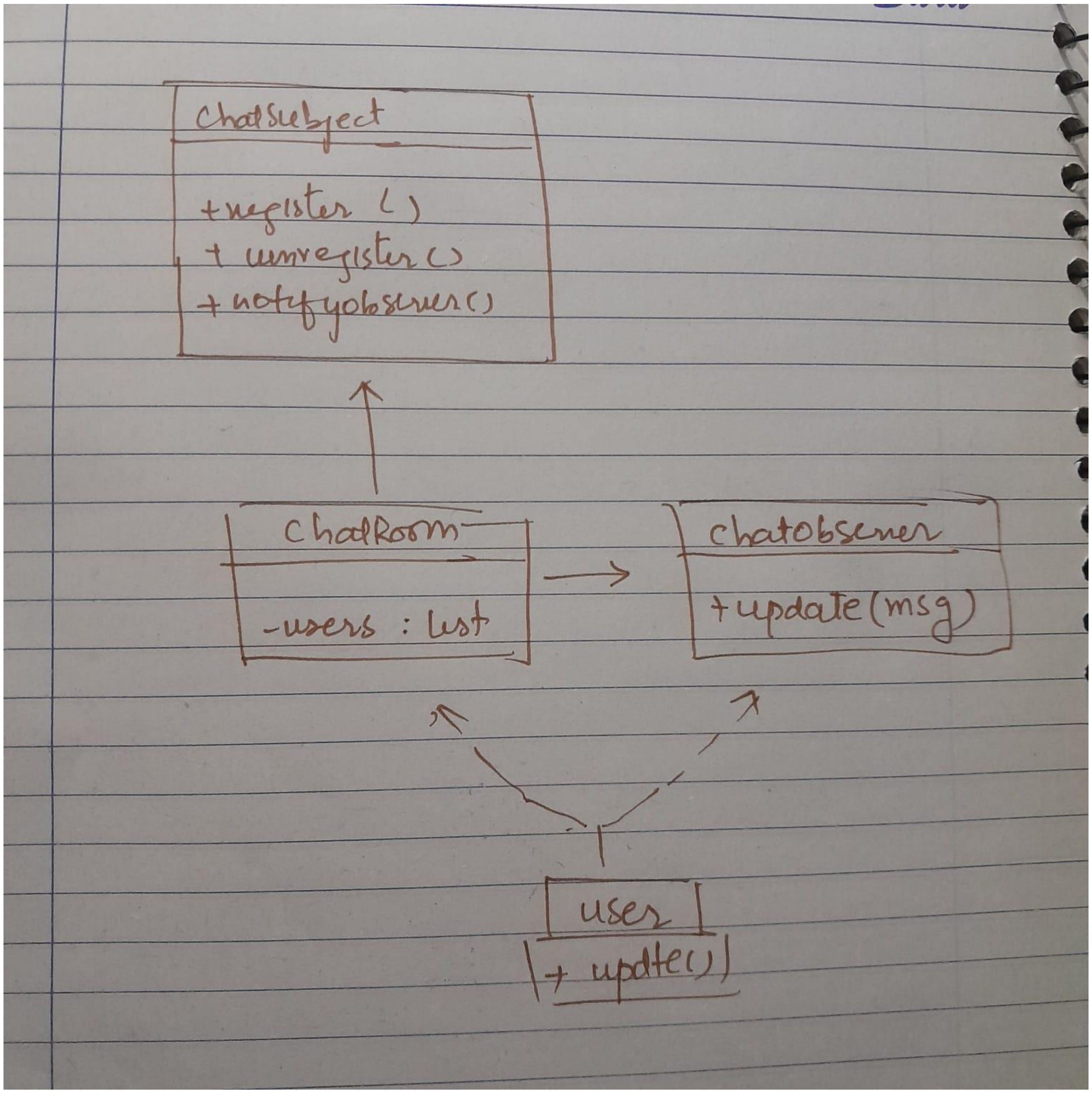
```java
interface ChatObserver {
    void update(String message);
}

// Subject Interface
interface ChatSubject {
    void register(ChatObserver observer);
    void unregister(ChatObserver observer);
    void notifyObservers(String message);
}
```



```java
class ChatRoom implements ChatSubject {
    private final List<ChatObserver> users;

    public ChatRoom() {
        this.users = new ArrayList<>();
    }

    @Override
    public void register(ChatObserver observer) {
        users.add(observer);
    }

    @Override
    public void unregister(ChatObserver observer) {
        users.remove(observer);
    }

    @Override
    public void notifyObservers(String message) {
        for (ChatObserver user : users) {
            user.update(message);
        }
    }

    public void sendMessage(String message) {
        System.out.println("Message sent to chat room: " + message);
        notifyObservers(message);
    }
}
```


```java
class User implements ChatObserver {
    private final String username;

    public User(String username) {
        this.username = username;
    }

    @Override
    public void update(String message) {
        System.out.println(username + " received: " + message);
    }
}
```

```java
public class ChatApplicationDemo {
    public static void main(String[] args) {
        ChatRoom chatRoom = new ChatRoom();

        User user1 = new User( username: "warda");
        User user2 = new User( username: "laiba");

        chatRoom.register(user1);
        chatRoom.register(user2);

        chatRoom.sendMessage("Hello everyone!");

        chatRoom.unregister(user1);

        chatRoom.sendMessage("Goodbye laiba!");
    }
}
```

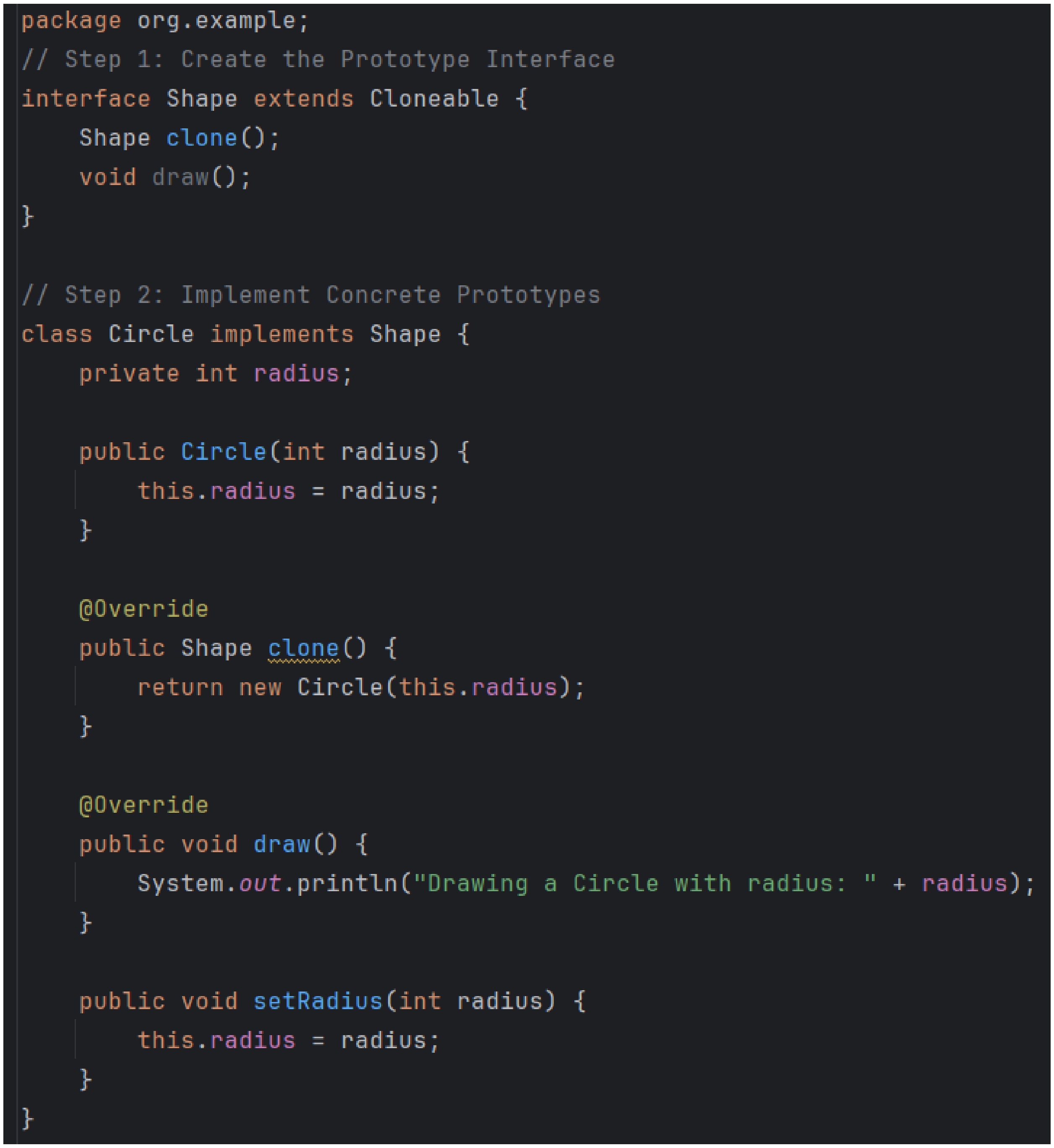
Class diagram:



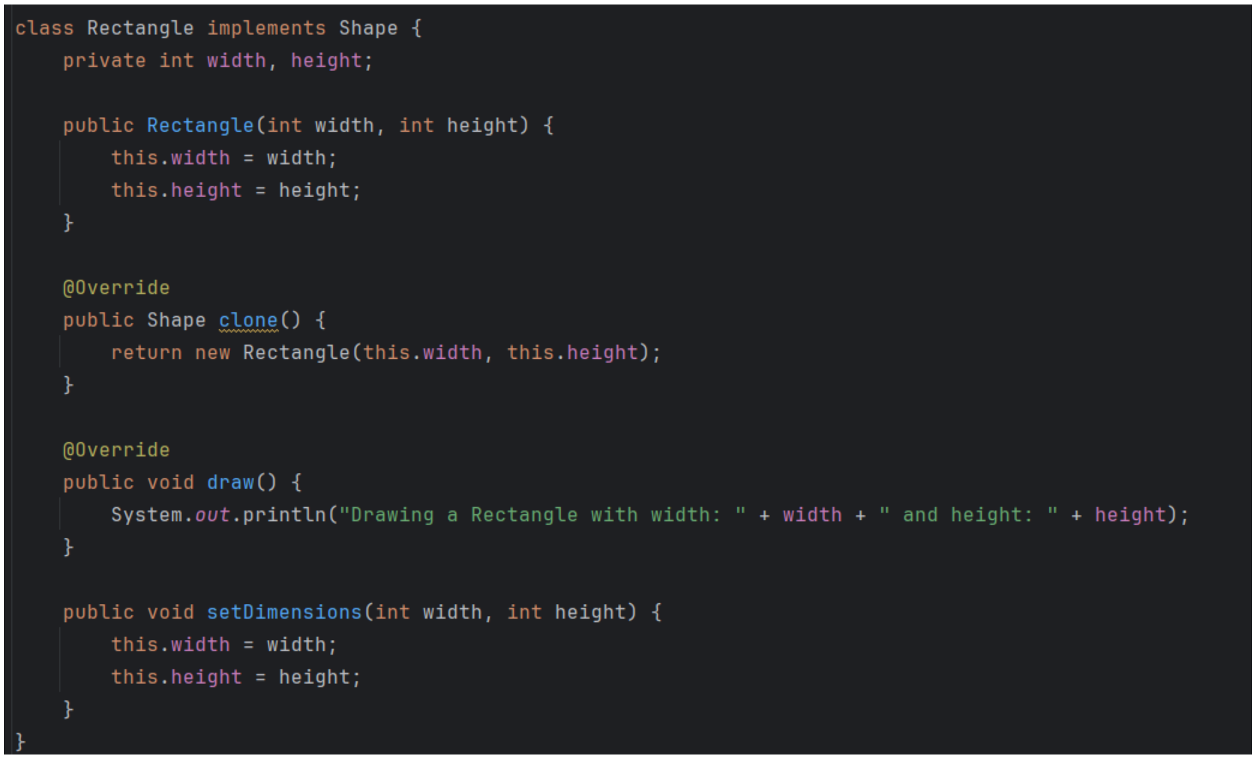
Implementation of prototype design pattern:

Task 1: Cloning Shapes

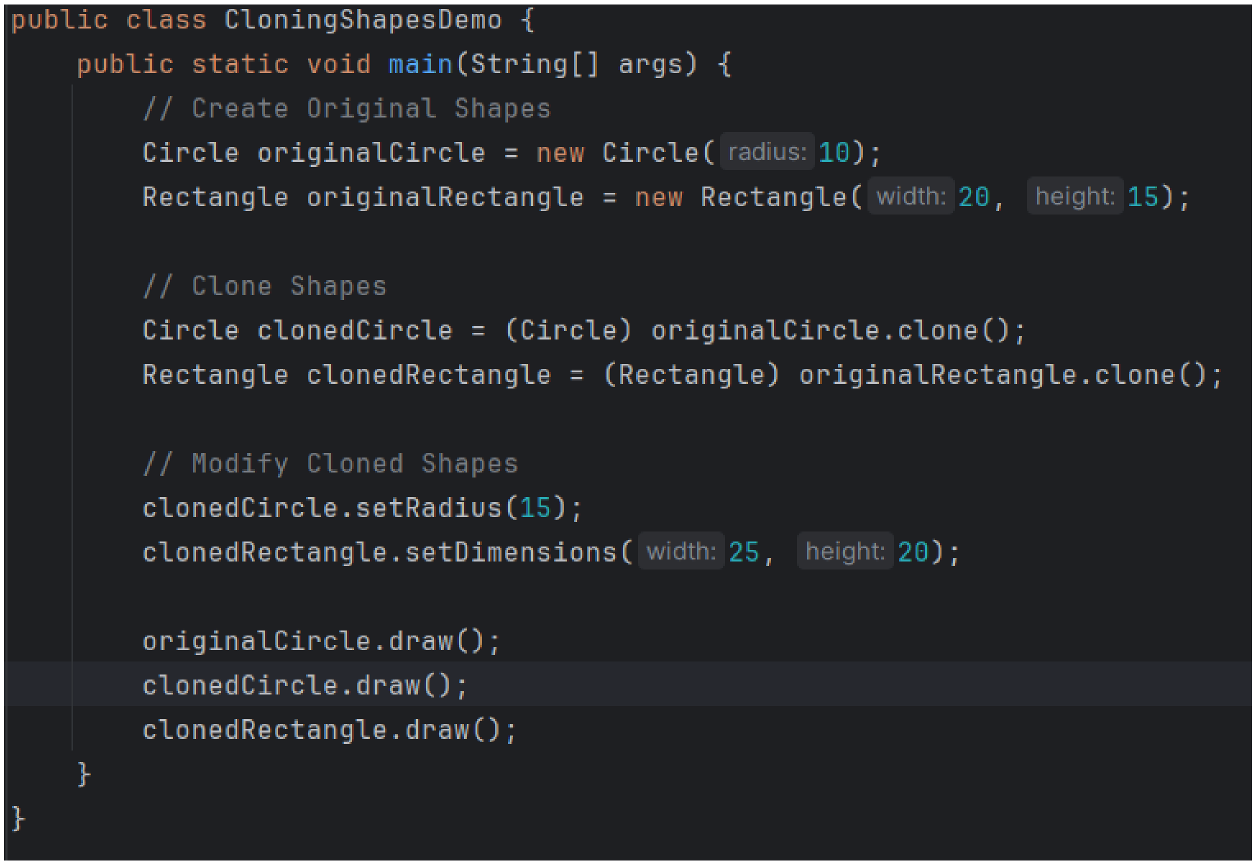
Code



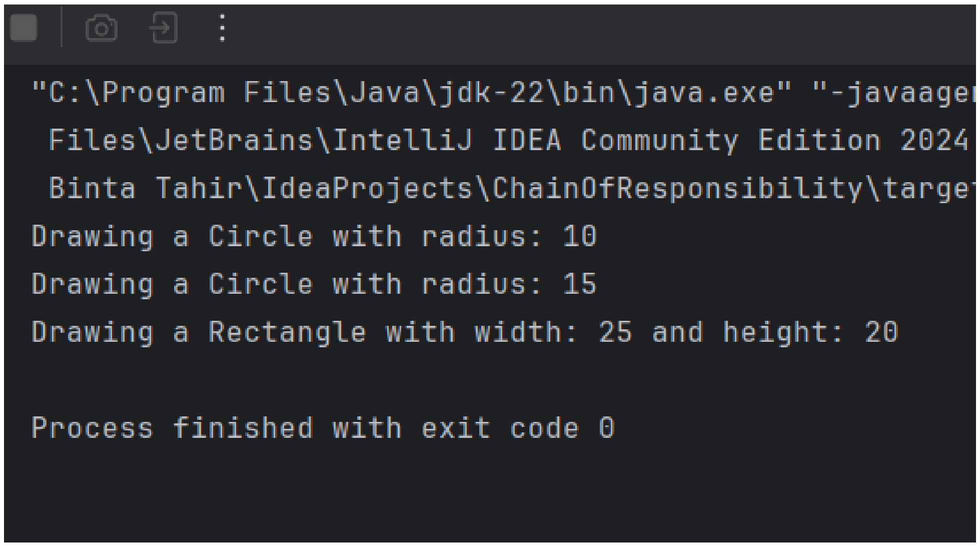
```java
package org.example;
// Step 1: Create the Prototype Interface
interface Shape extends Cloneable {
    Shape clone();
    void draw();
}

// Step 2: Implement Concrete Prototypes
class Circle implements Shape {
    private int radius;

    public Circle(int radius) {
        this.radius = radius;
    }

    @Override
    public Shape clone() {
        return new Circle(this.radius);
    }

    @Override
    public void draw() {
        System.out.println("Drawing a Circle with radius: " + radius);
    }

    public void setRadius(int radius) {
        this.radius = radius;
    }
}
```



```java
class Rectangle implements Shape {
    private int width, height;

    public Rectangle(int width, int height) {
        this.width = width;
        this.height = height;
    }

    @Override
    public Shape clone() {
        return new Rectangle(this.width, this.height);
    }

    @Override
    public void draw() {
        System.out.println("Drawing a Rectangle with width: " + width + " and height: " + height);
    }

    public void setDimensions(int width, int height) {
        this.width = width;
        this.height = height;
    }
}
```

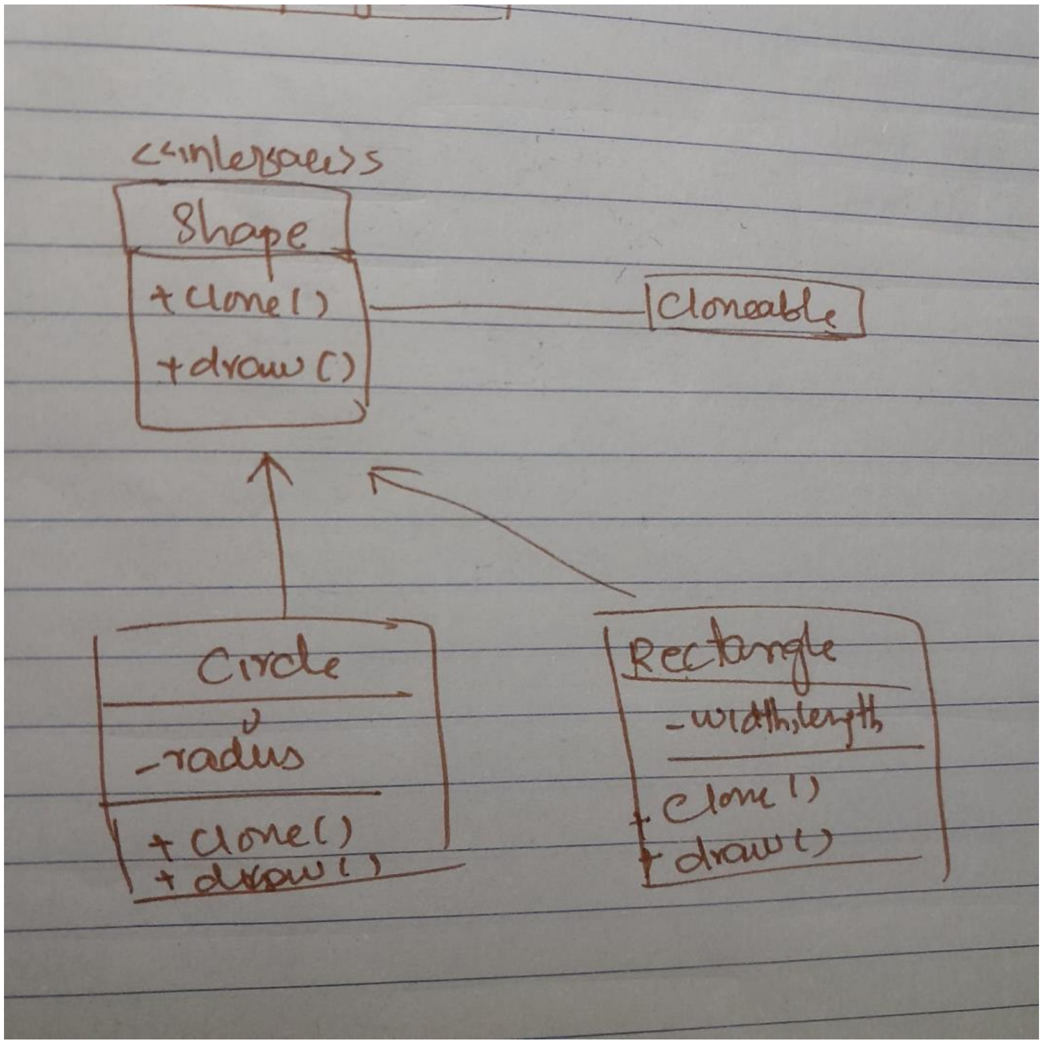


```java
public class CloningShapesDemo {
    public static void main(String[] args) {
        // Create Original Shapes
        Circle originalCircle = new Circle( radius: 10);
        Rectangle originalRectangle = new Rectangle( width: 20,  height: 15);

        // Clone Shapes
        Circle clonedCircle = (Circle) originalCircle.clone();
        Rectangle clonedRectangle = (Rectangle) originalRectangle.clone();

        // Modify Cloned Shapes
        clonedCircle.setRadius(15);
        clonedRectangle.setDimensions( width: 25,  height: 20);

        originalCircle.draw();
        clonedCircle.draw();
        clonedRectangle.draw();
    }
}
```



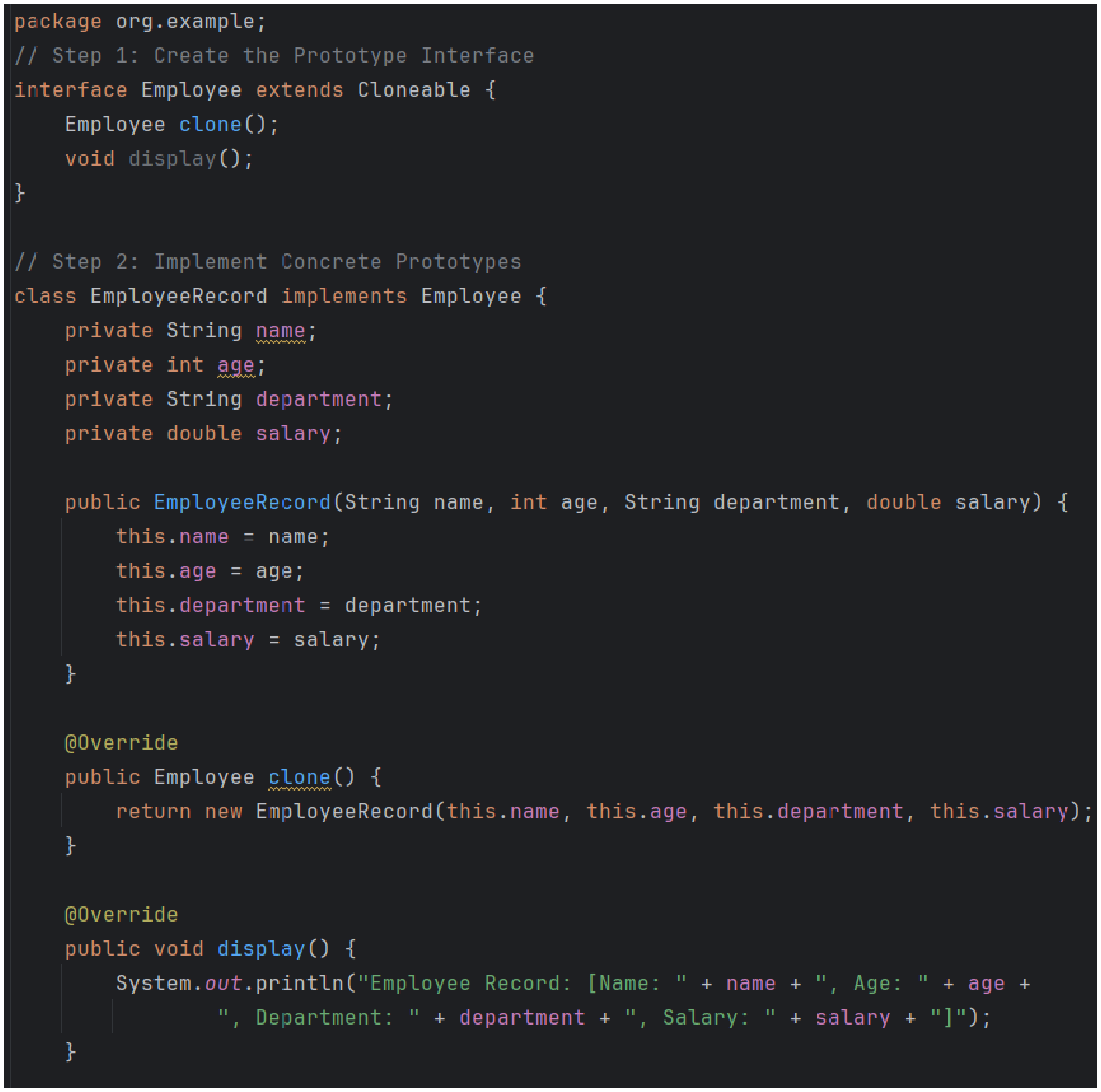
```
"C:\Program Files\Java\jdk-22\bin\java.exe" "-javaage
 Files\JetBrains\IntelliJ IDEA Community Edition 2024
 Binta Tahir\IdeaProjects\ChainOfResponsibility\targe
Drawing a Circle with radius: 10
Drawing a Circle with radius: 15
Drawing a Rectangle with width: 25 and height: 20

Process finished with exit code 0
```

Class Diagram

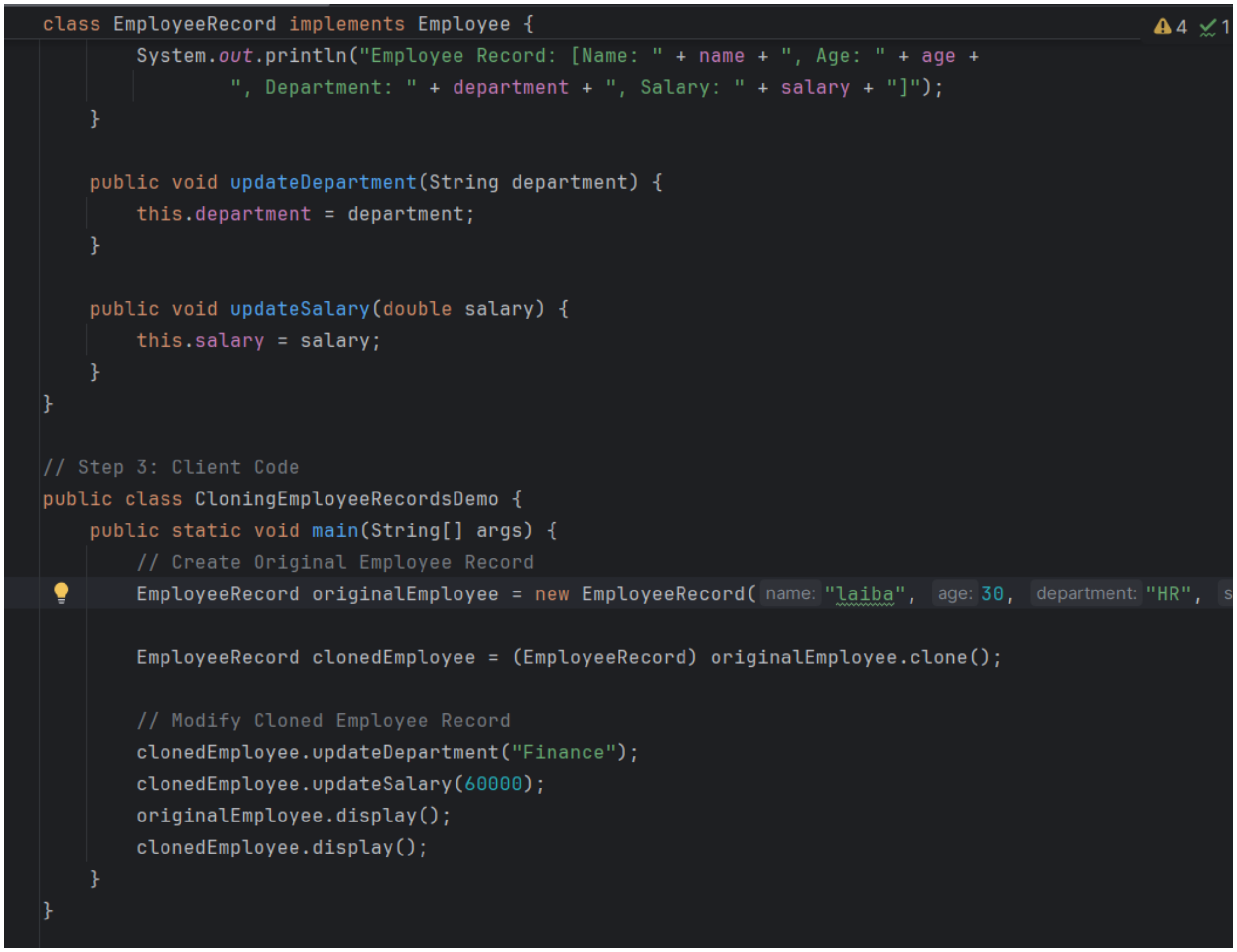


Task 2: Cloning Employee Records

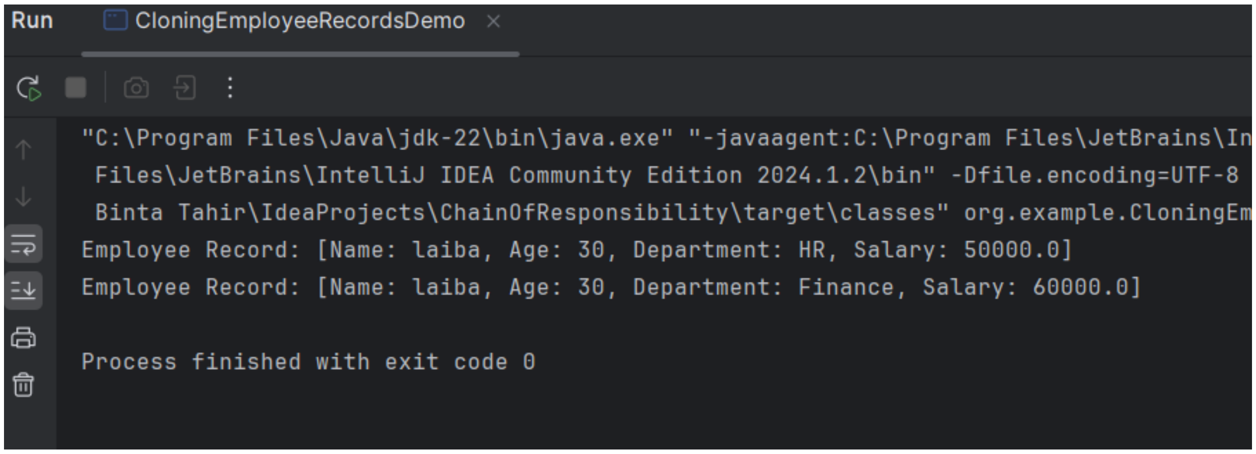
Code



```java
package org.example;
// Step 1: Create the Prototype Interface
interface Employee extends Cloneable {
    Employee clone();
    void display();
}

// Step 2: Implement Concrete Prototypes
class EmployeeRecord implements Employee {
    private String name;
    private int age;
    private String department;
    private double salary;

    public EmployeeRecord(String name, int age, String department, double salary) {
        this.name = name;
        this.age = age;
        this.department = department;
        this.salary = salary;
    }

    @Override
    public Employee clone() {
        return new EmployeeRecord(this.name, this.age, this.department, this.salary);
    }

    @Override
    public void display() {
        System.out.println("Employee Record: [Name: " + name + ", Age: " + age +
                ", Department: " + department + ", Salary: " + salary + "]");
    }
```



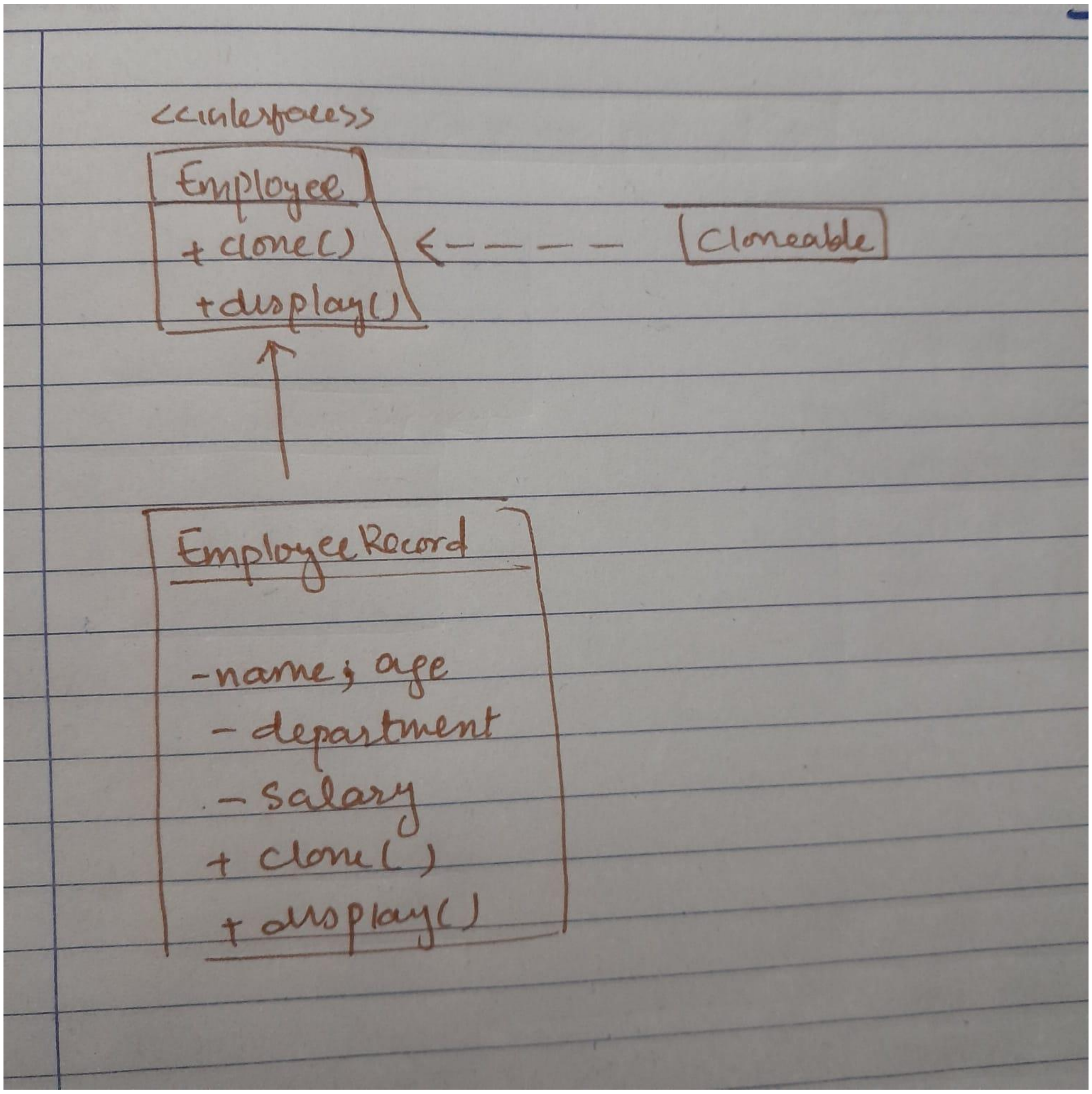
```
class EmployeeRecord implements Employee {
            System.out.println("Employee Record: [Name: " + name + ", Age: " + age +
                    ", Department: " + department + ", Salary: " + salary + "]");
        }

        public void updateDepartment(String department) {
            this.department = department;
        }

        public void updateSalary(double salary) {
            this.salary = salary;
        }
}

// Step 3: Client Code
public class CloningEmployeeRecordsDemo {
    public static void main(String[] args) {
        // Create Original Employee Record
        EmployeeRecord originalEmployee = new EmployeeRecord( name: "laiba",  age: 30,  department: "HR",  s

        EmployeeRecord clonedEmployee = (EmployeeRecord) originalEmployee.clone();

        // Modify Cloned Employee Record
        clonedEmployee.updateDepartment("Finance");
        clonedEmployee.updateSalary(60000);
        originalEmployee.display();
        clonedEmployee.display();
    }
}
```



Run   CloningEmployeeRecordsDemo

```
"C:\Program Files\Java\jdk-22\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\In
 Files\JetBrains\IntelliJ IDEA Community Edition 2024.1.2\bin" -Dfile.encoding=UTF-8
 Binta Tahir\IdeaProjects\ChainOfResponsibility\target\classes" org.example.CloningEm
Employee Record: [Name: laiba, Age: 30, Department: HR, Salary: 50000.0]
Employee Record: [Name: laiba, Age: 30, Department: Finance, Salary: 60000.0]

Process finished with exit code 0
```

Class diagram



<<interface>>
Employee
+ clone()
+display()
Cloneable
EmployeeRecord
-name; age
- department
- salary
+ clone( )
+ display()