Name: Arshiya Tabassum A

Superset ID : 6424209

**WEEK 1**

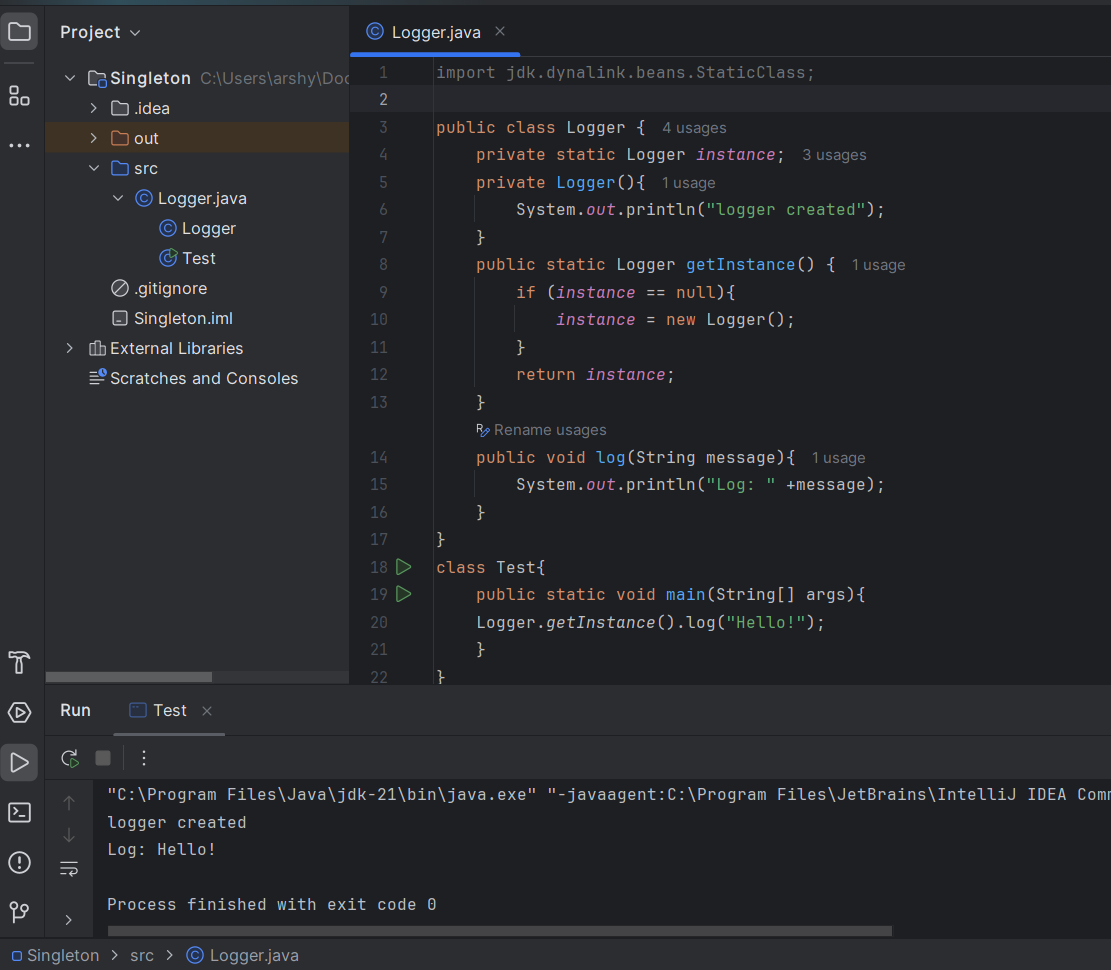
**Design Pattern and Principles**

**Module 1**

Exercise 1: Implementing the Singleton Pattern

Scenario:

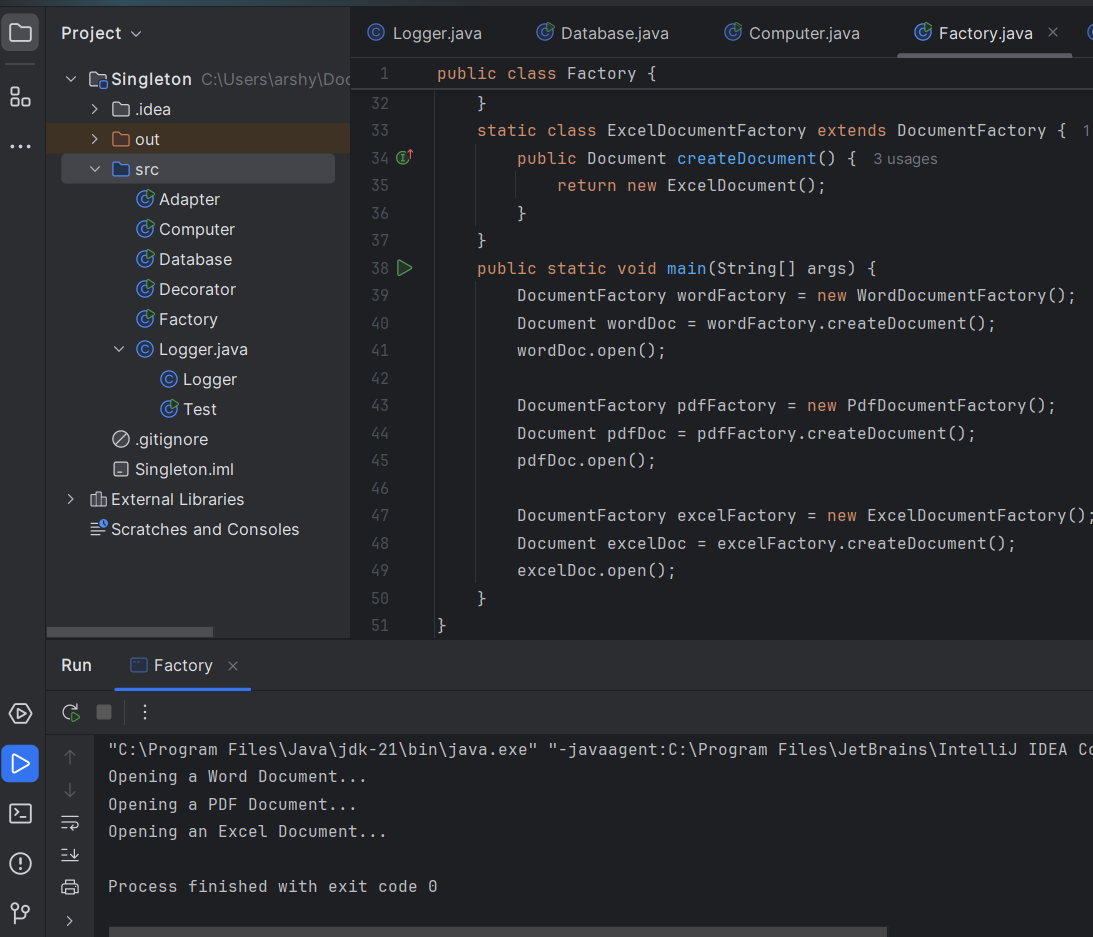
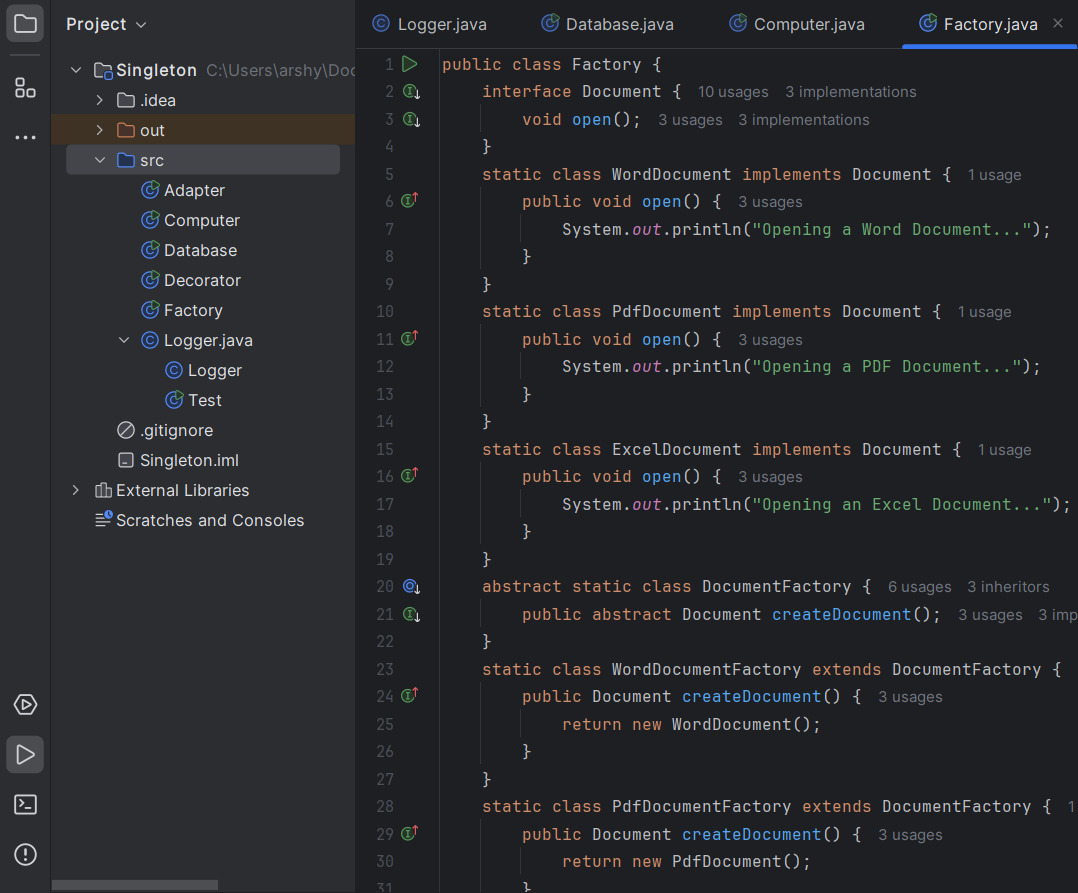
You need to ensure that a logging utility class in your application has only one instance throughout the application lifecycle to ensure consistent logging.

****

Exercise 2: Implementing the Factory Method Pattern

Scenario:

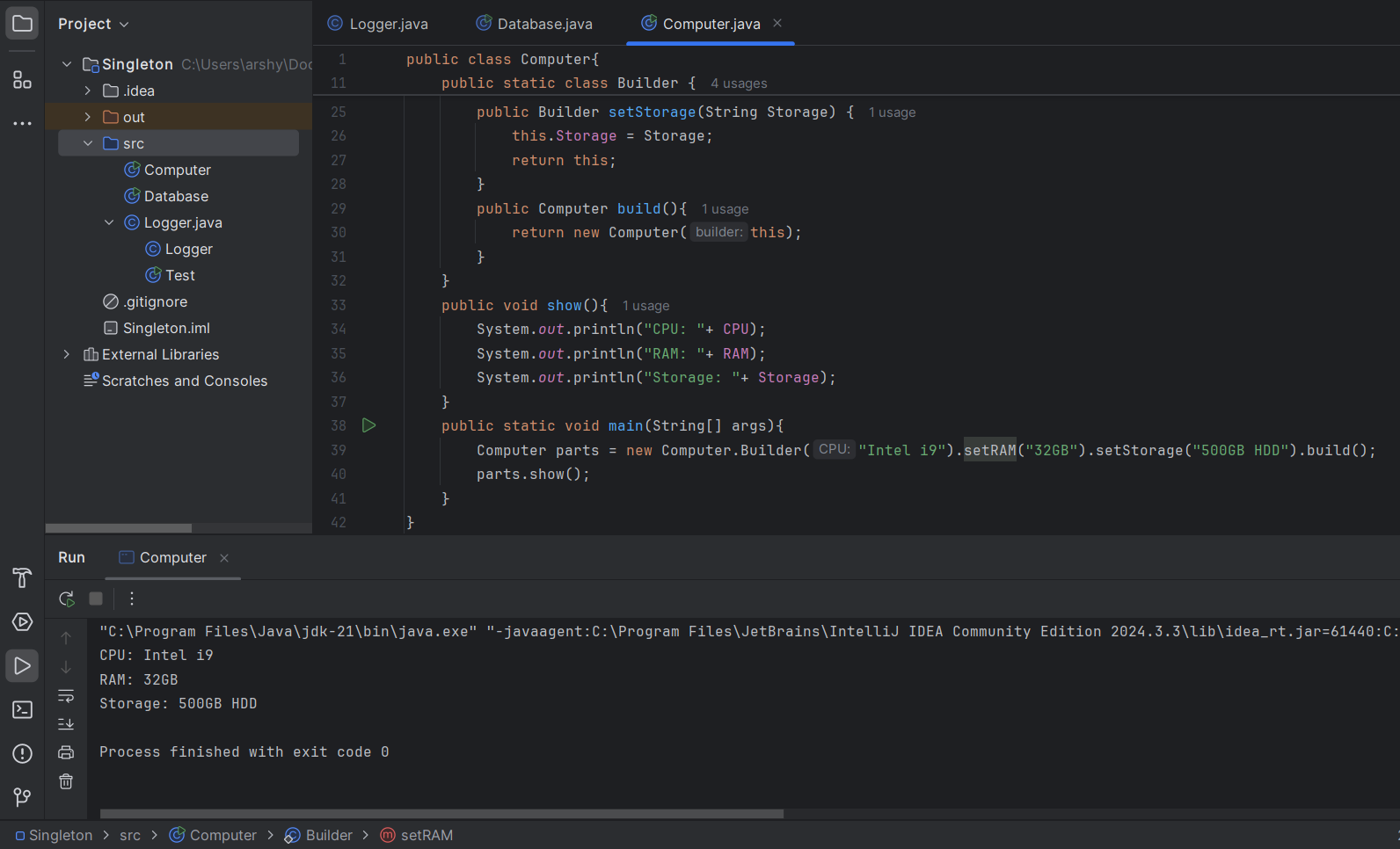
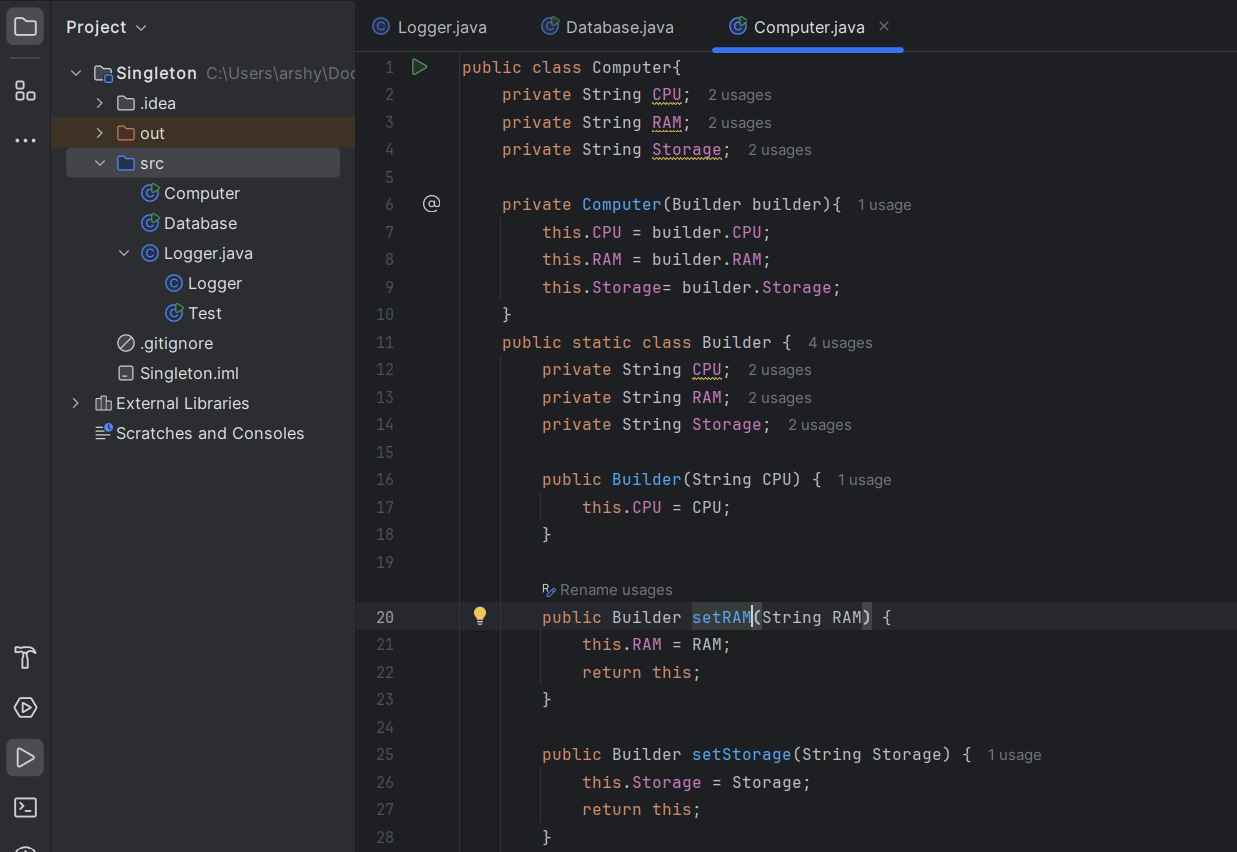
You are developing a document management system that needs to create different types of documents (e.g., Word, PDF, Excel). Use the Factory Method Pattern to achieve this.



Exercise 3: Implementing the Builder Pattern

Scenario:

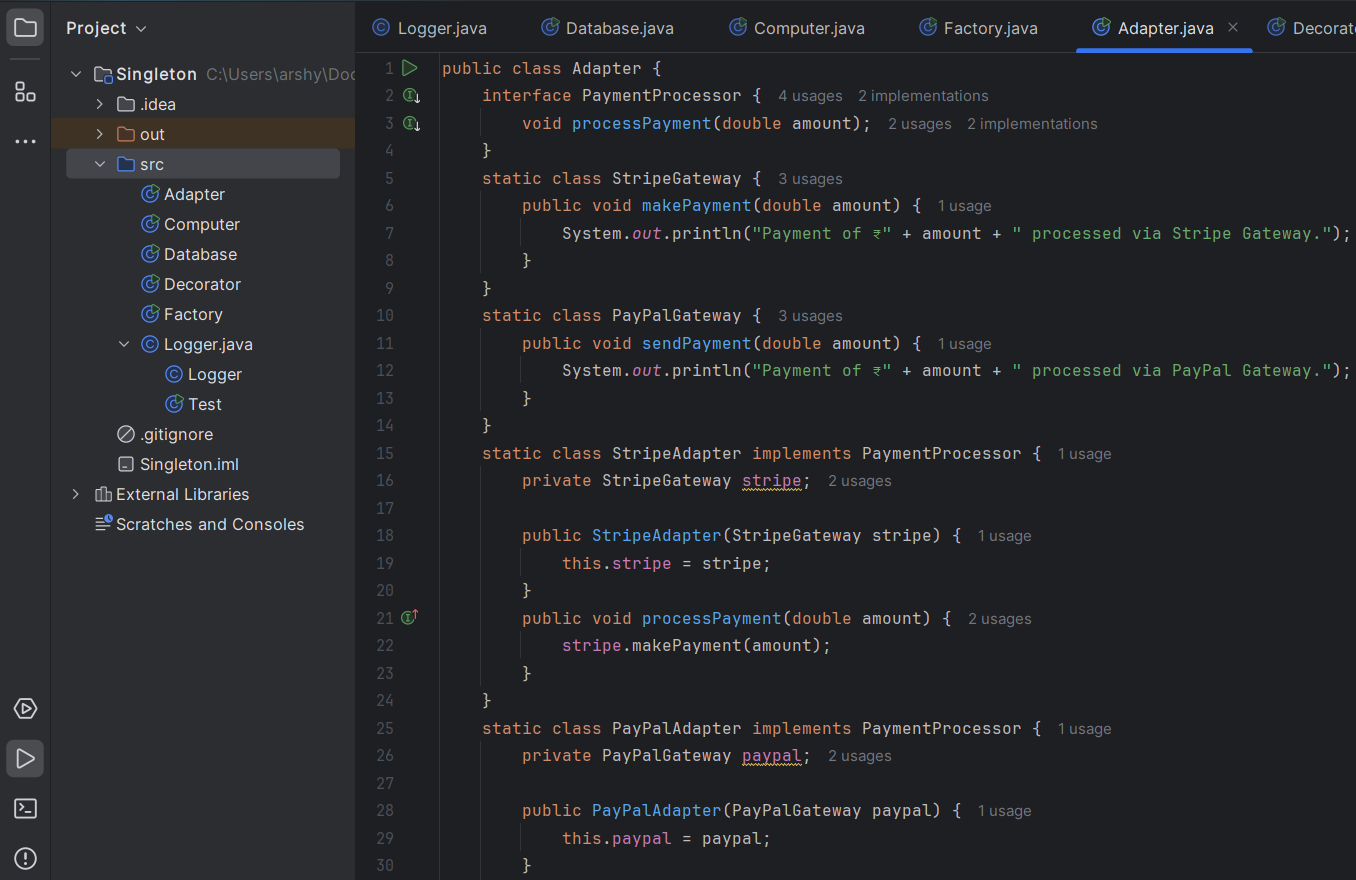
You are developing a system to create complex objects such as a Computer with multiple optional parts. Use the Builder Pattern to manage the construction process.

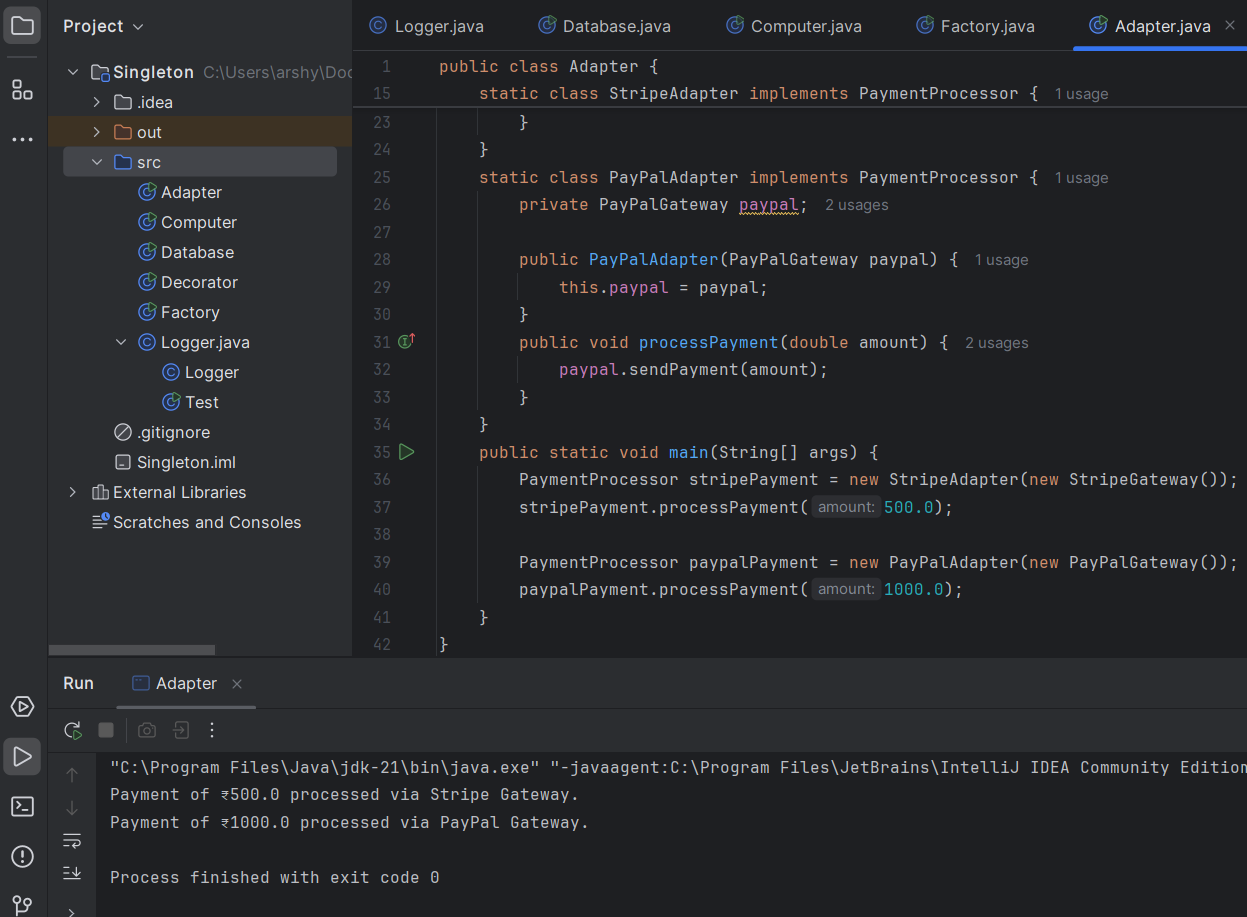


Exercise 4: Implementing the Adapter Pattern

Scenario:

You are developing a payment processing system that needs to integrate with multiple third-party payment gateways with different interfaces. Use the Adapter Pattern to achieve this.

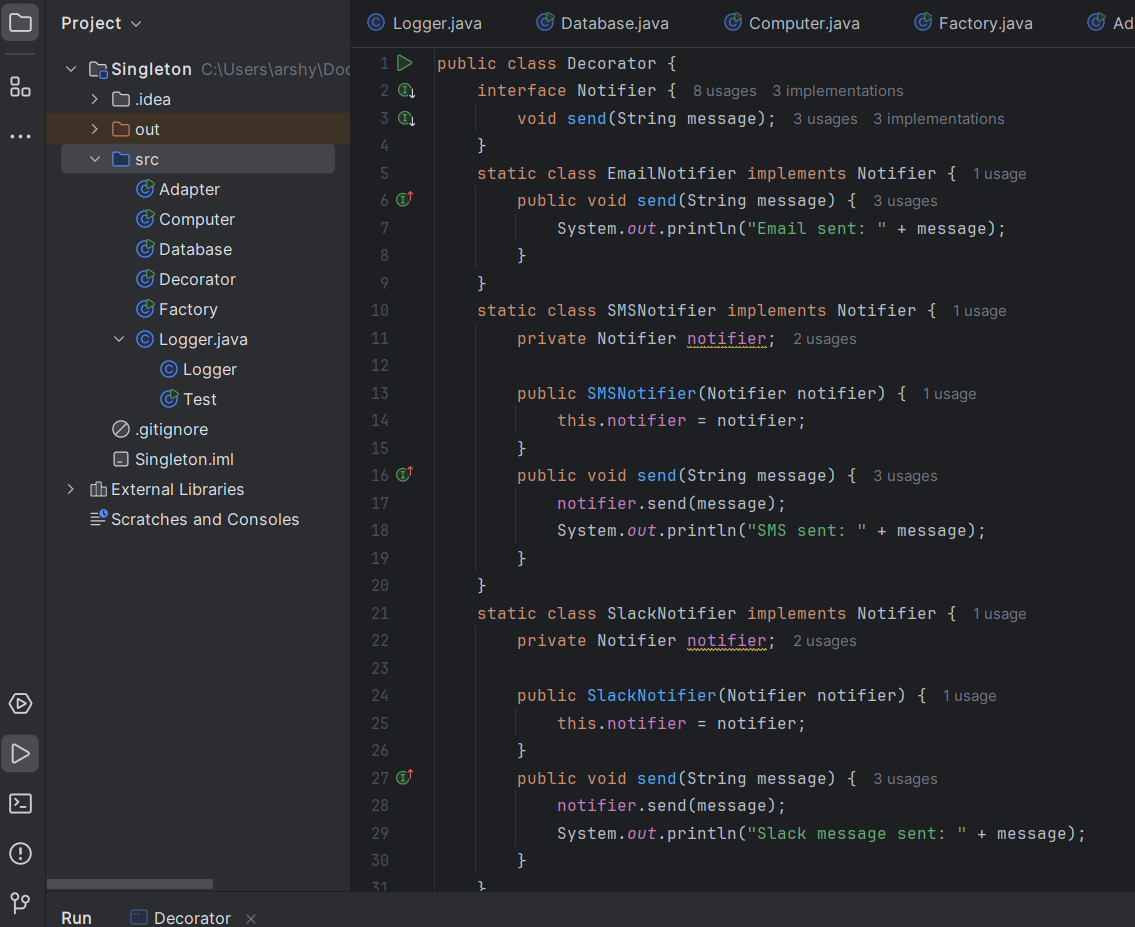


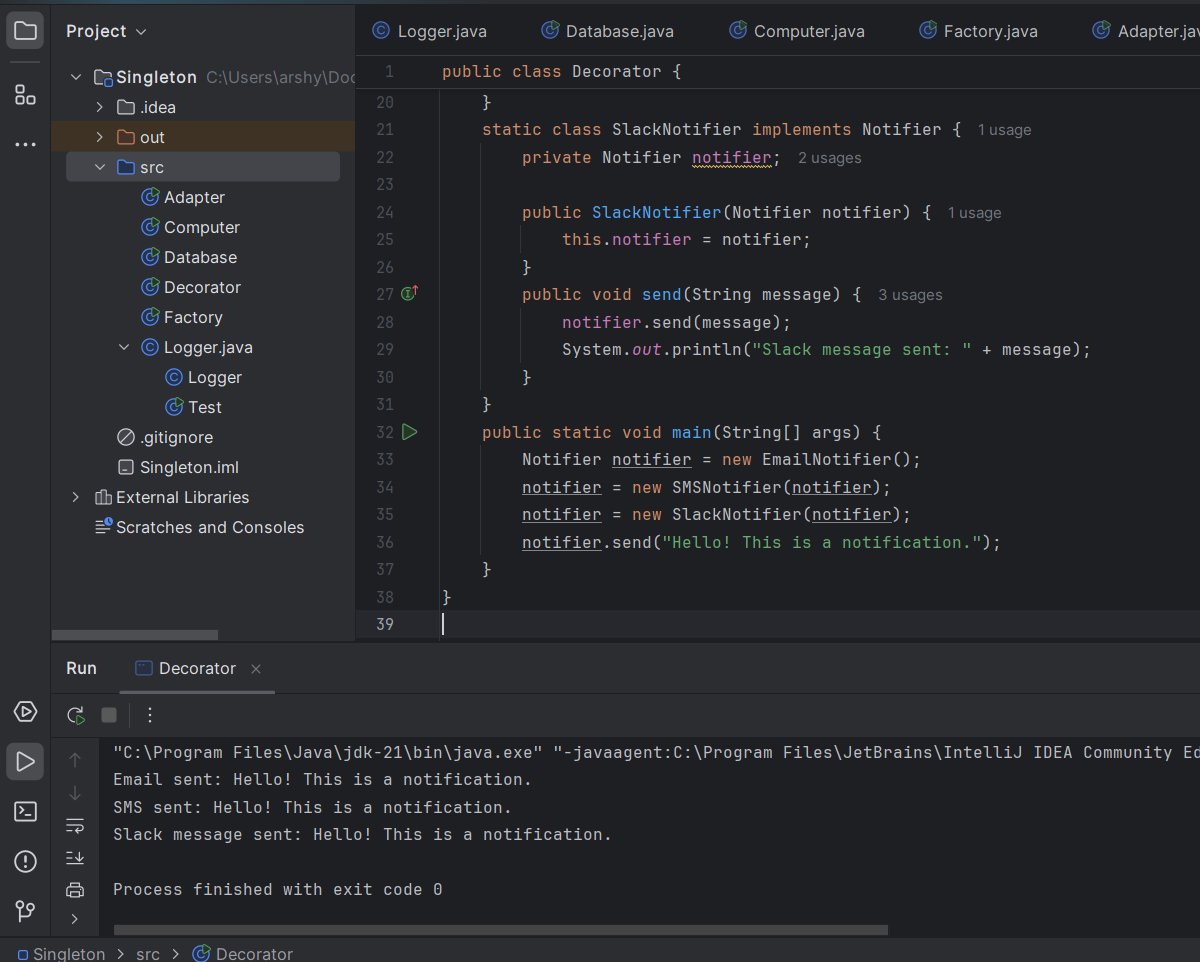


Exercise 5: Implementing the Decorator Pattern

Scenario:

You are developing a notification system where notifications can be sent via multiple channels (e.g., Email, SMS). Use the Decorator Pattern to add functionalities dynamically.

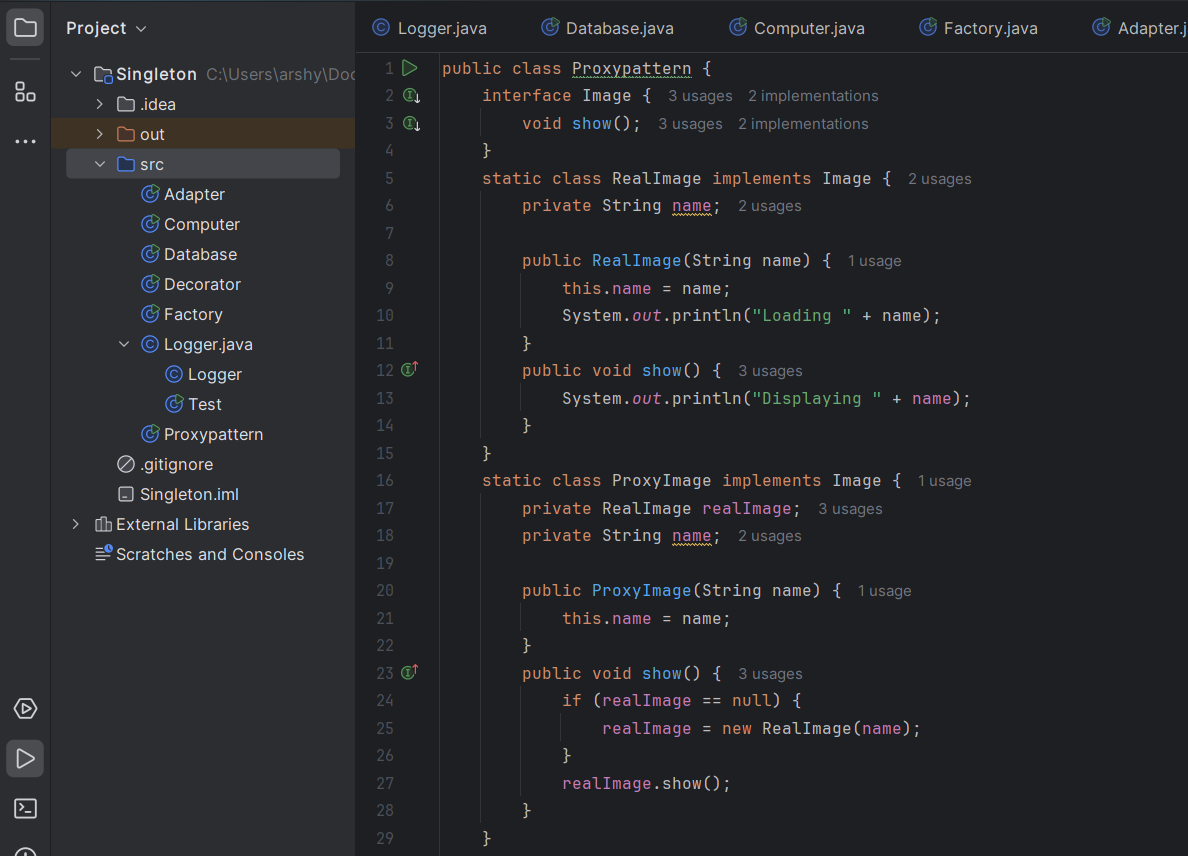


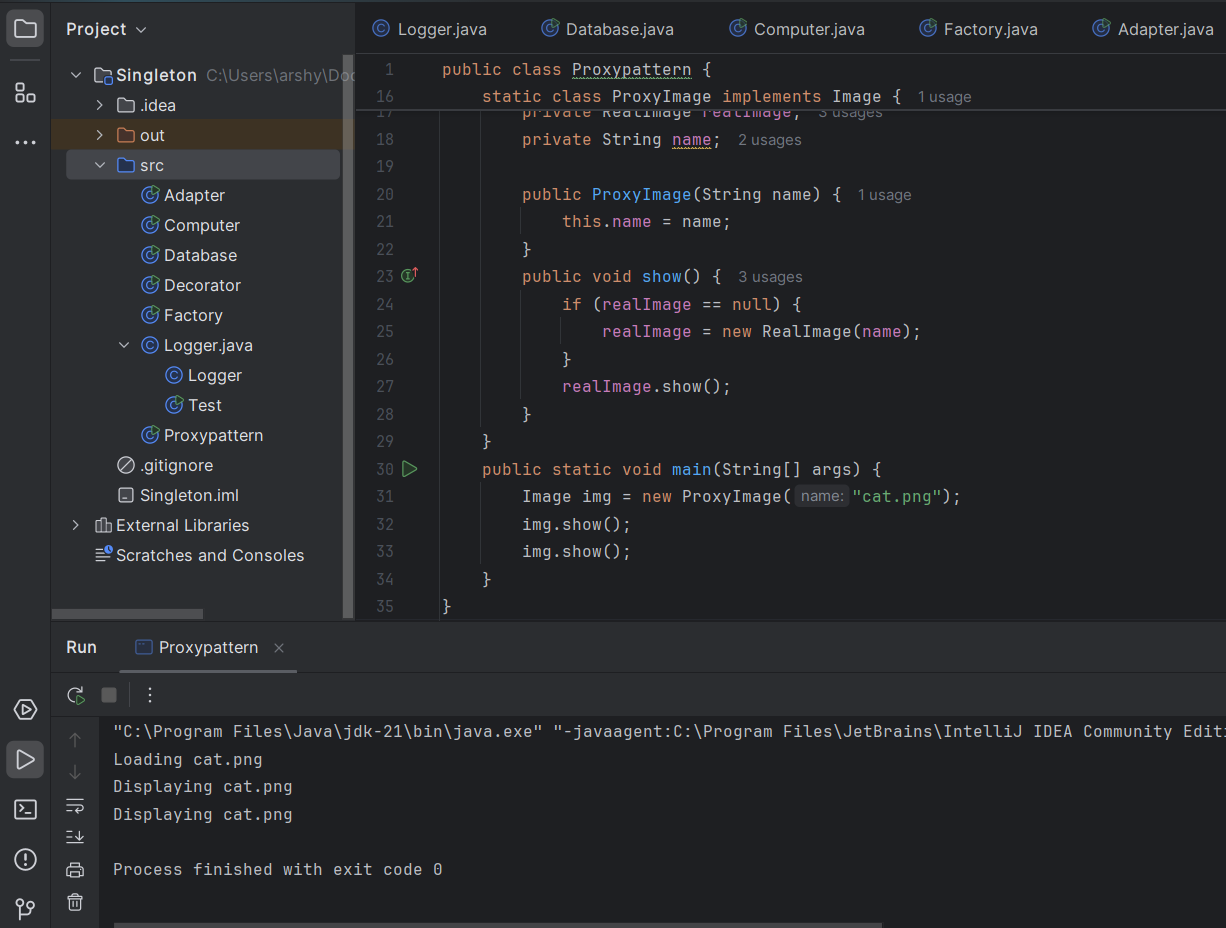


Exercise 6: Implementing the Proxy Pattern

Scenario:

You are developing an image viewer application that loads images from a remote server. Use the Proxy Pattern to add lazy initialization and caching.

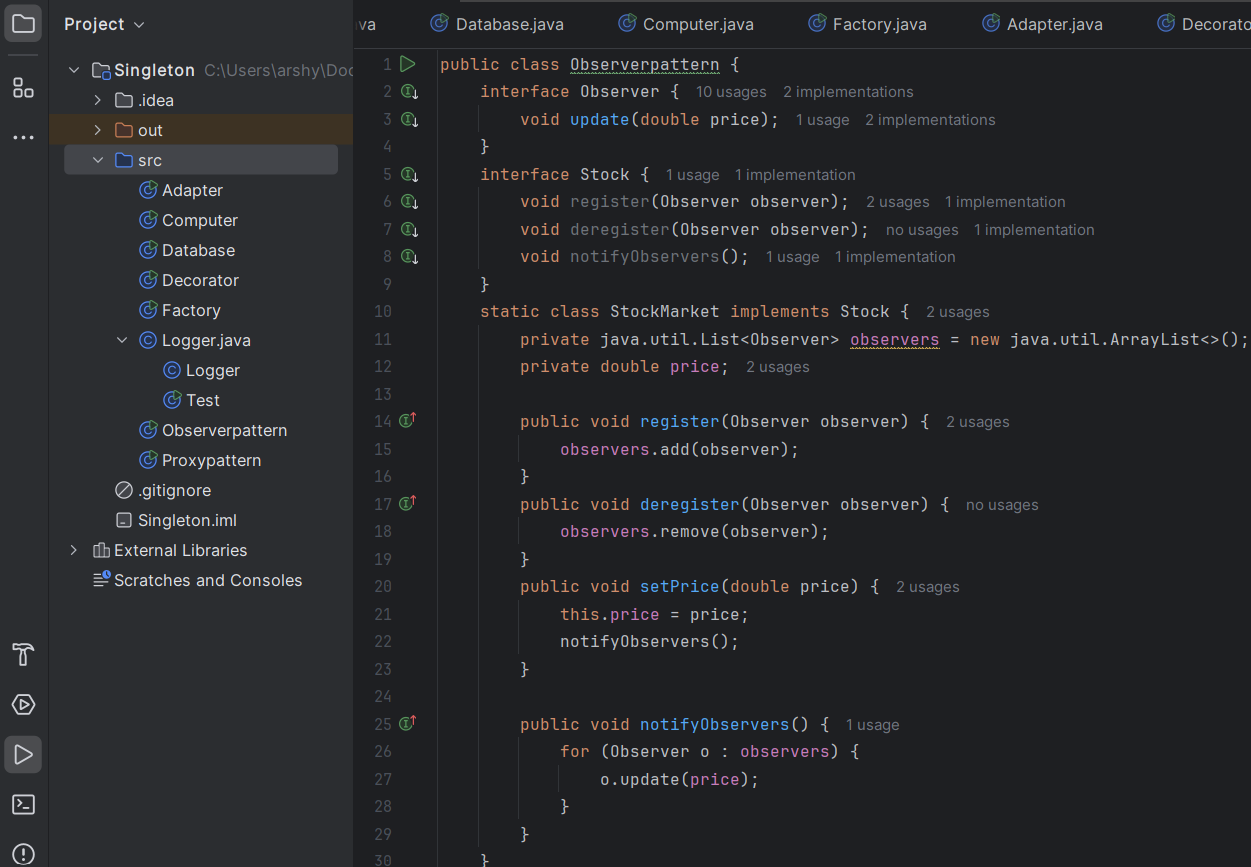


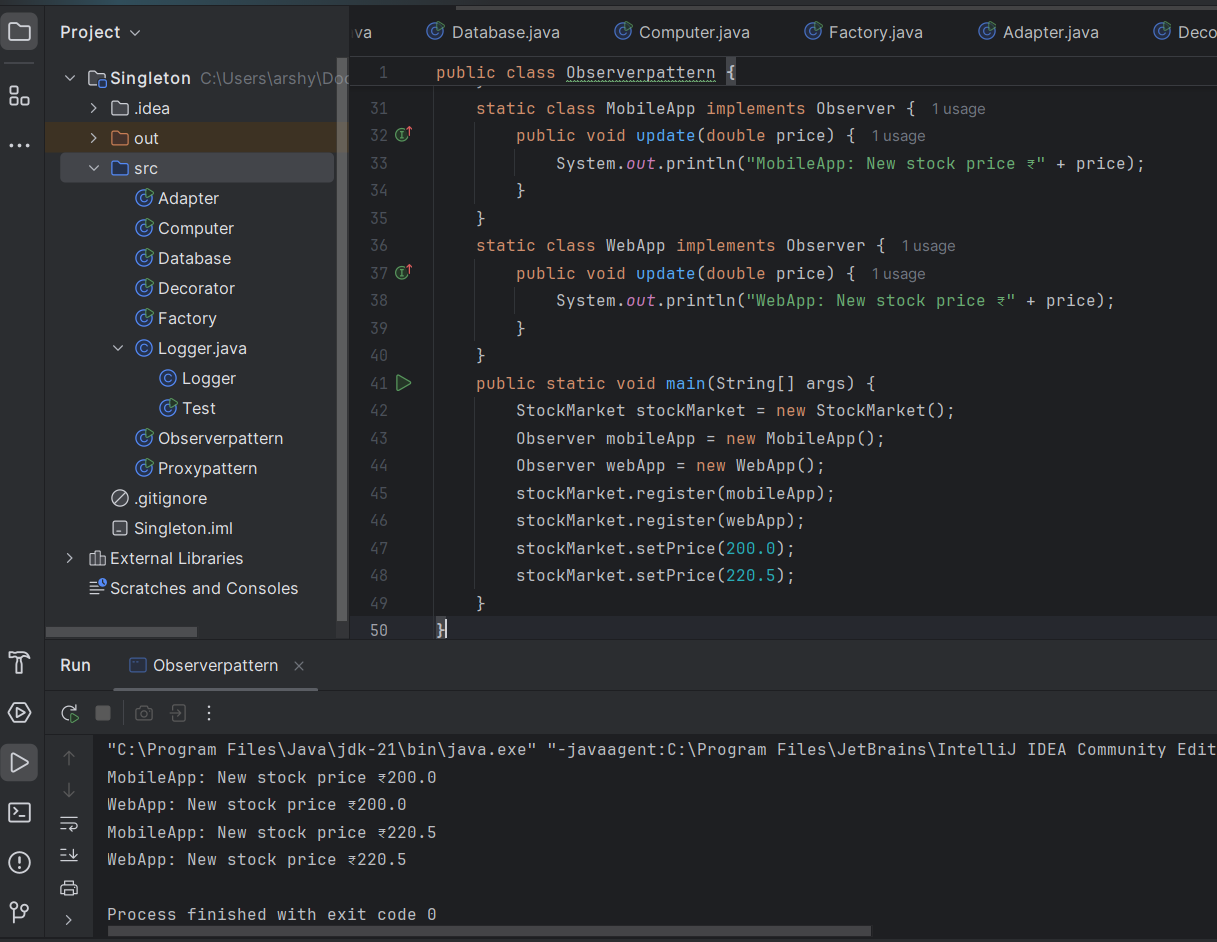


Exercise 7: Implementing the Observer Pattern

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.

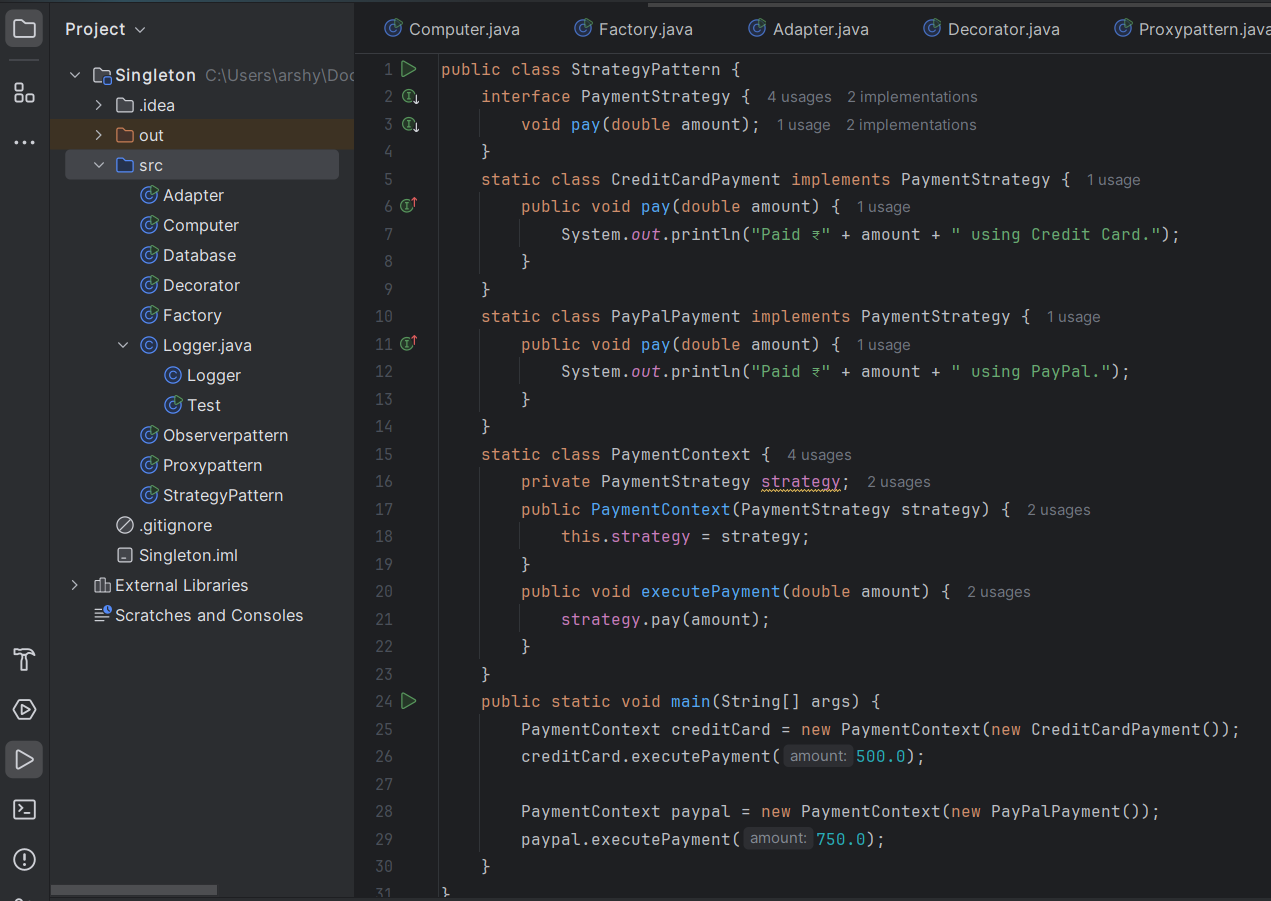


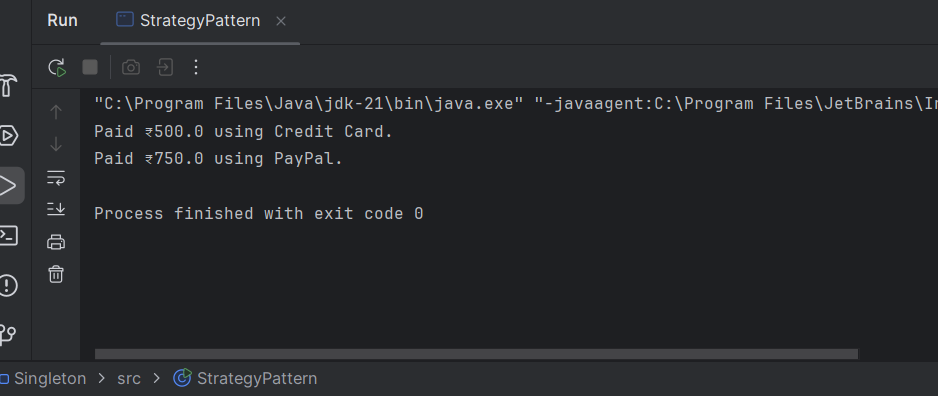


Exercise 8: Implementing the Strategy Pattern

Scenario:

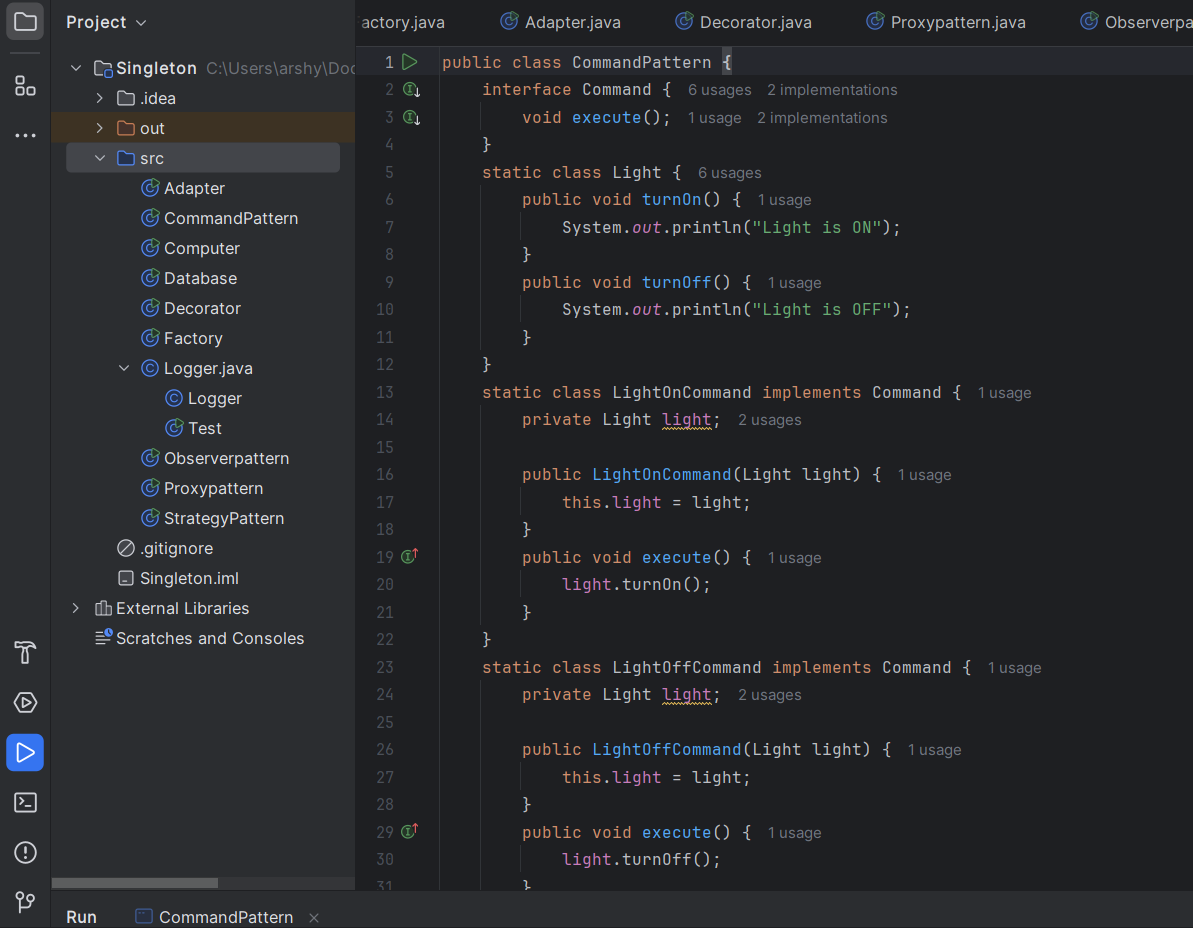
You are developing a payment system where different payment methods (e.g., Credit Card, PayPal) can be selected at runtime. Use the Strategy Pattern to achieve this.

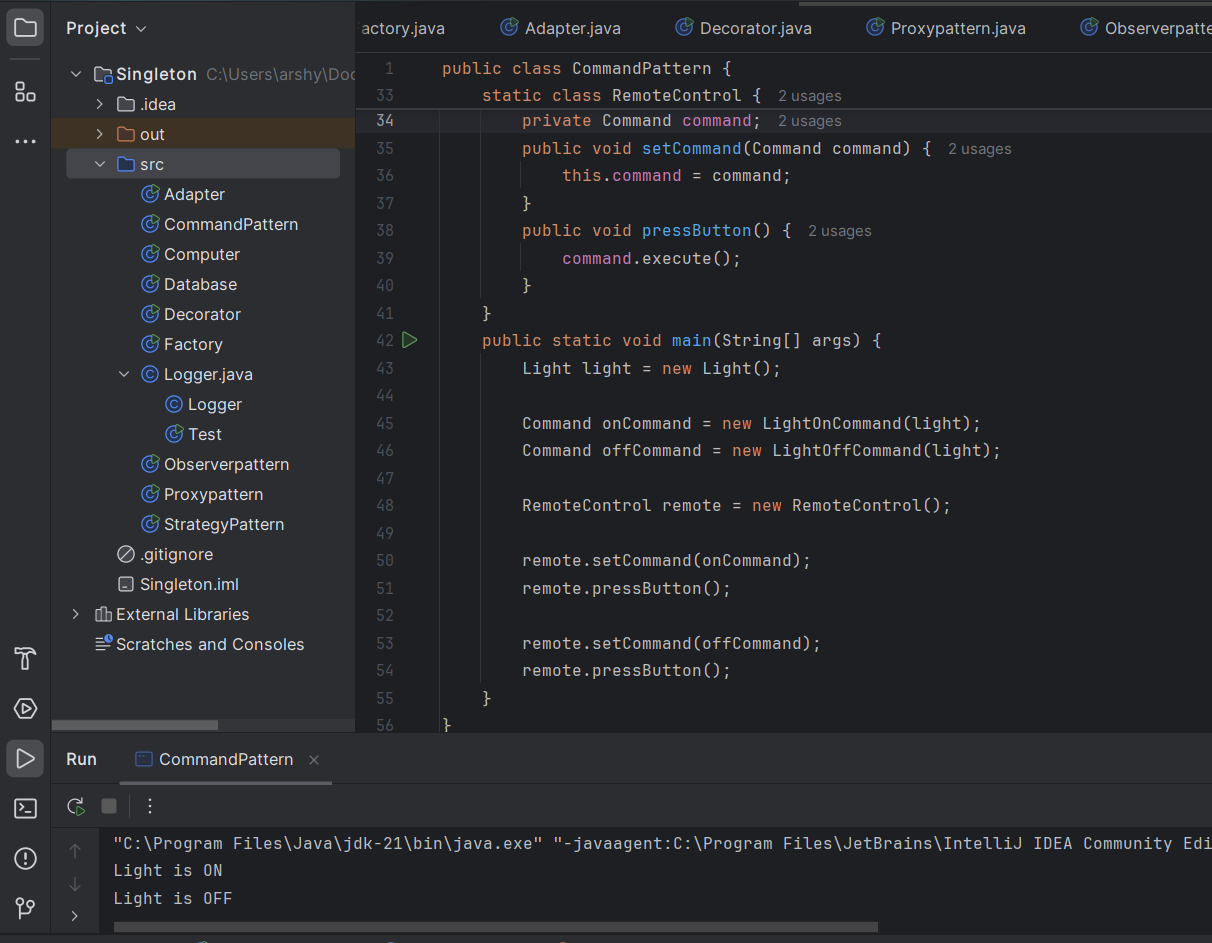




Exercise 9: Implementing the Command Pattern

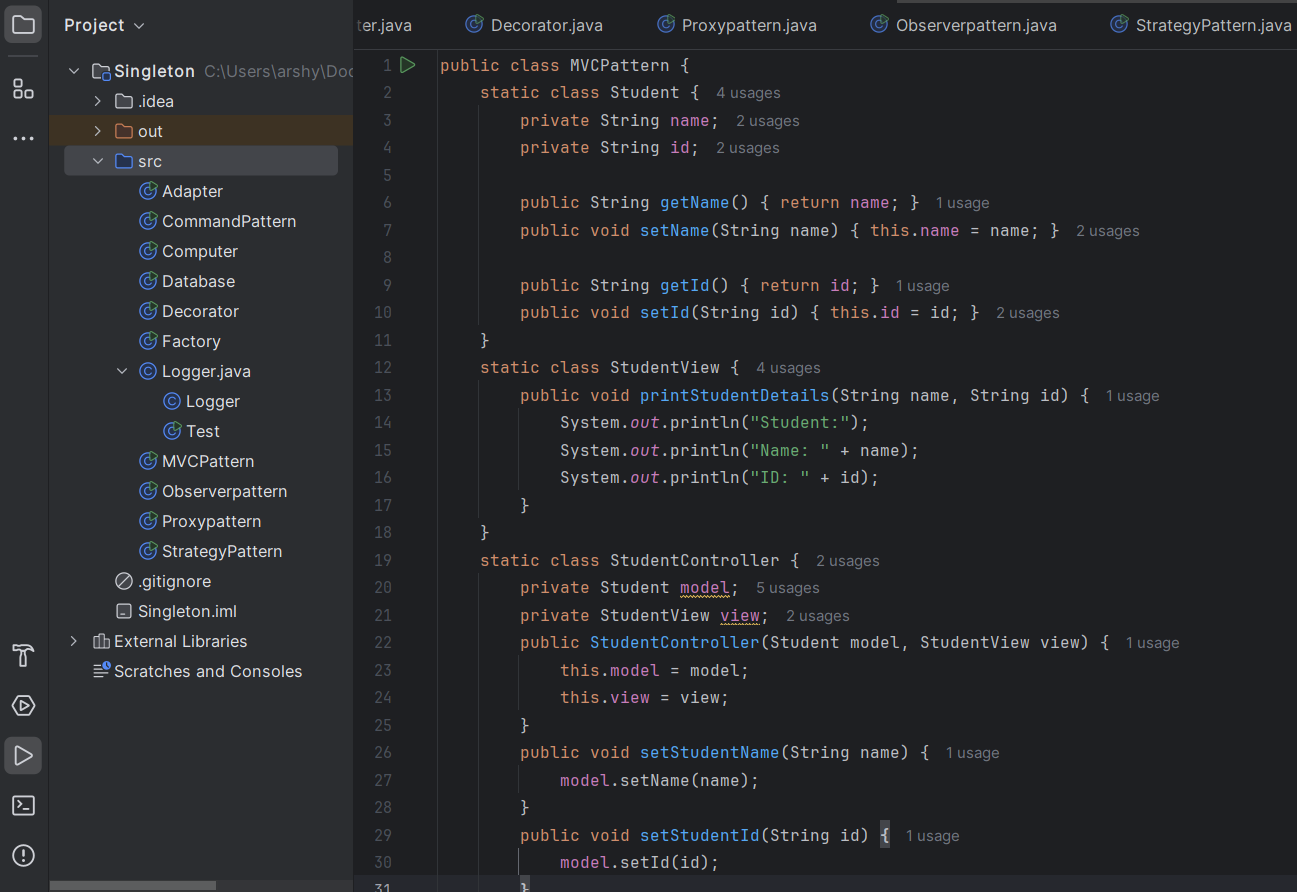
Scenario: You are developing a home automation system where commands can be issued to turn devices on or off. Use the Command Pattern to achieve this.

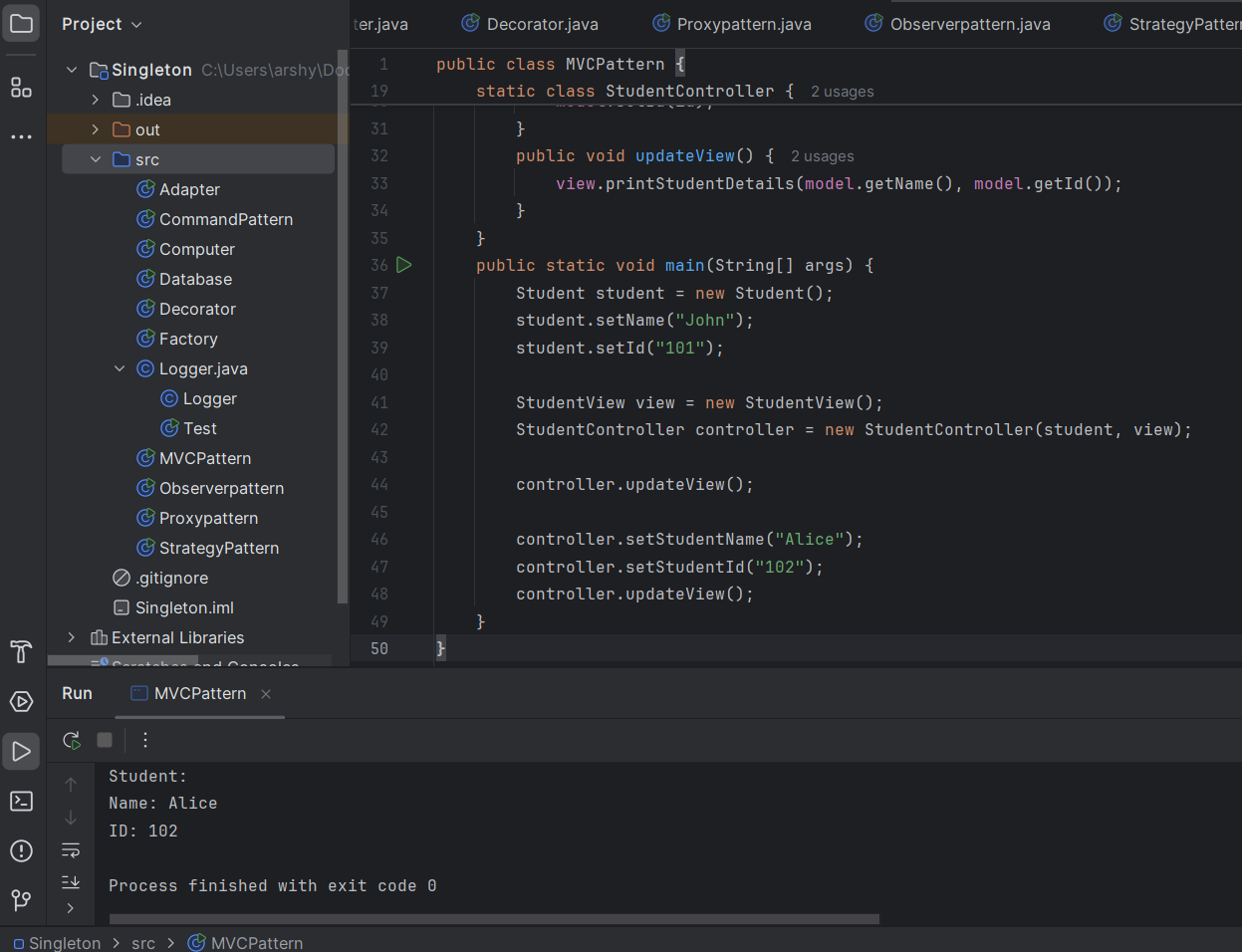




Exercise 10: Implementing the MVC Pattern

Scenario: You are developing a simple web application for managing student records using the MVC pattern.





Exercise 11: Implementing Dependency Injection

Scenario: You are developing a customer management application where the service class depends on a repository class. Use Dependency Injection to manage these dependencies.

