Exercise-6

**1. Introduction**

The Image Viewer Application is a Java-based program that demonstrates the implementation of the Proxy design pattern. It simulates an image viewing system that loads images from a remote server, implementing lazy loading and caching mechanisms. This application serves as an educational tool to understand how the Proxy pattern can be used in real-world scenarios.

**2. System Requirements**

* Java Development Kit (JDK) 8 or higher
* Active internet connection
* Integrated Development Environment (IDE) such as Eclipse or IntelliJ IDEA (optional)
* Command-line interface (if not using an IDE)

**3. Application Structure**

The application consists of five main Java files:

1. Image.java - Interface
2. RealImage.java - Concrete class implementing Image
3. ProxyImage.java - Proxy class implementing Image
4. ImageDatabase.java - Simulated database for image information
5. ImageViewerApp.java - Main application class

**4. Class Descriptions**

**4.1 Image Interface**

This interface defines the common method that both the RealImage and ProxyImage classes will implement.

**4.2 RealImage Class**

This class represents the actual image object. It's responsible for loading the image from the server and displaying it.

Key methods:

* RealImage(String filename): Constructor
* loadImageFromServer(): Simulates loading an image from a remote server
* display(): Displays the image

**4.3 ProxyImage Class**

This class acts as a proxy for the RealImage. It implements lazy loading and caching.

Key methods:

* ProxyImage(String filename): Constructor
* display(): Checks if RealImage exists, creates it if necessary, then displays it

**4.4 ImageDatabase Class**

Simulates a database of image filenames.

Key method:

* getImageFilename(int id): Returns a filename for a given image ID

**4.5 ImageViewerApp Class**

The main class that runs the application and provides a user interface.

Key method:

* main(String[] args): Entry point of the application

**5. Implementation Details**

Link: [For code click here.](https://github.com/Akashmondal55/COGNIZANT/tree/9195ddd4cc72a12a76445fa112ac913d84c5e049/Week-1/Design%20patern%20and%20princple/ProxyPatternExample)

1. Future Enhancements

Implement a graphical user interface (GUI)

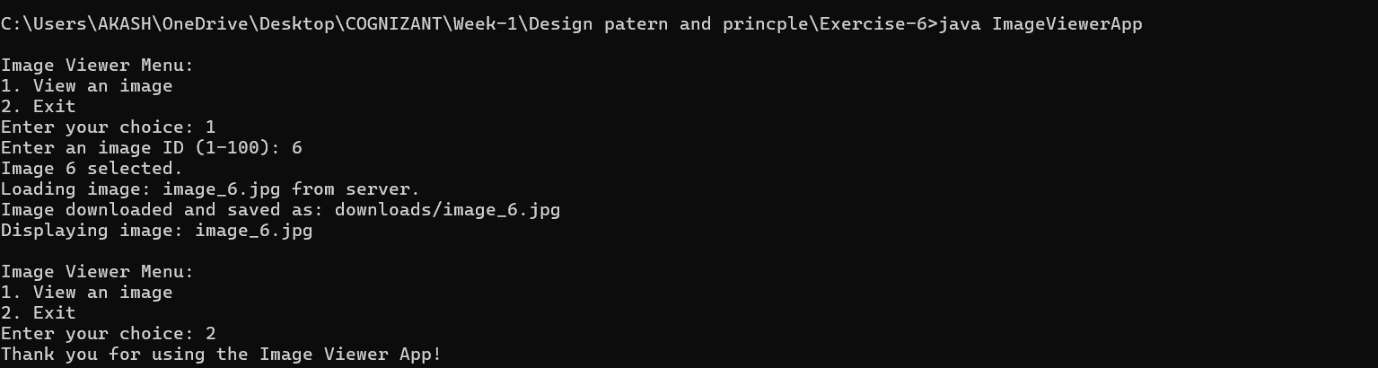
Add support for actual image rendering

Implement a more sophisticated caching mechanism

Add error handling for network issues

Expand the image database and add categorization

1. Output



9. Conclusion

This Image Viewer Application successfully demonstrates the use of the Proxy pattern to implement lazy loading and caching in a simulated image viewing scenario. It provides a foundation for more complex image handling applications and showcases effective use of design patterns in Java programming.