Lab Assignment: Design Patterns in Java

Objective:
Understand and implement common design patterns like Singleton, Factory, and Observer.
Requirements:

1. Implement a `Singleton` class for a database connection manager.
2. Create a `Factory` design pattern for generating different types of shapes (`Circle`, `Rectangle`, `Square`).
3. Demonstrate the `Observer` design pattern by creating a simple weather monitoring system with `WeatherData` as the subject and `DisplayDevice` as observers.

Code Starter:

```
// Singleton Example
public class DatabaseConnection {
    private static DatabaseConnection instance;

    private DatabaseConnection() {}

    public static DatabaseConnection getInstance() {
        if (instance == null) {
            instance = new DatabaseConnection();
        }
        return instance;
    }
}
```

Exercises:

1. Extend the `Factory` pattern to include a new shape, `Triangle`.
2. Use `Collections.synchronizedList()` in the `Observer` example to make it thread-safe.
3. Implement a `Decorator` pattern to dynamically add features to a base class.

Bonus Tasks:

1. Combine the `Singleton` and `Factory` patterns to create a single instance of the `ShapeFactory`.
2. Use the `Strategy` pattern to allow different calculation algorithms in the `Observer` example.
3. Create a `Chain of Responsibility` example for handling user requests in a multi-tier application.

Submission:
Submit your code and screenshots of the outputs.