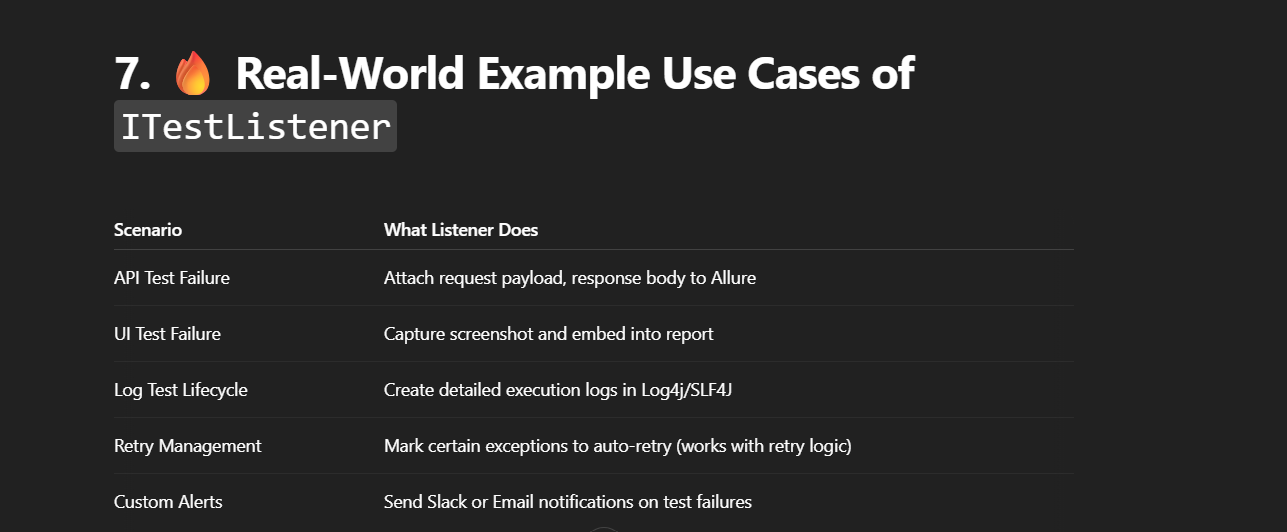
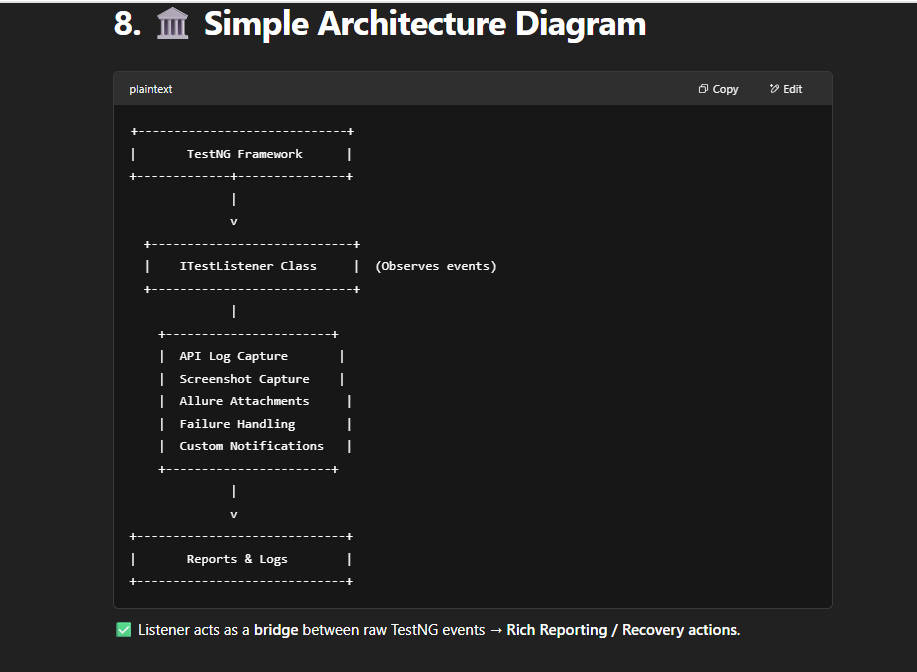
**ITestListener**

1. What is ITestListener?
   1. ITestListener is an interface provided by TestNG that lets you listen to important events during the execution of test methods.
   2. “Observer” for your test cases — you get notified automatically when tests start, succeed, fail, or get skipped.
2. Why is ITestListener Useful?
   1. **Centralized control for:**
      1. Handling test success, failure, skipped events.
      2. Logging, taking screenshots, attaching API logs, etc.
      3. Creating custom behaviors on test results without touching the test code.
      4. Automatically improving reports like Allure, ExtentReports, etc.
      5. Making tests more maintainable and DRY (Don't Repeat Yourself).
3. Where is ITestListener Used in a Project?
   1. Typically used in:
      1. Base Framework Layer (utilities, setup)
      2. Custom Reporting (Allure, Extent, HTML Reports)
      3. Failure Handling (taking screenshots, logs)
      4. Retry Handling (combine with IRetryAnalyzer)
      5. Test Lifecycle Management (when you want hooks on test events)
4. How is ITestListener Used?
   1. Step-by-Step:
      1. Implement the Interface in a class.
      2. Override methods like onTestSuccess(), onTestFailure(), etc.
      3. Register the Listener either:
      4. Using @Listeners annotation in your test classes.
      5. Or globally in testng.xml.
5. Key Methods of ITestListener
   1. onTestStart(ITestResult result)
      1. When a test method starts
   2. onTestSuccess(ITestResult result)
      1. When a test method passes
   3. onTestFailure(ITestResult result)
      1. When a test method fails
   4. onTestSkipped(ITestResult result)
      1. When a test is skipped
   5. onStart(ITestContext context)
      1. Before any test method in a test <test> tag starts
   6. onFinish(ITestContext context)
      1. After all test methods in a test <test> tag finish





10. 📈 Advantages of Using ITestListener

* No need to put repetitive log/screenshot/attach code inside every test.
* Cleaner, maintainable codebase.
* Makes Allure/Extent reports automatically richer.
* Easier debugging for failures — logs and API traces are already captured.

🚀 Summary: Big Picture

| **Key Concept** | **Summary** |
| --- | --- |
| What | Observer for your test cases |
| Why | Centralize logging, reporting, retries |
| Where | Framework core utilities |
| How | Implement, override methods, register listener |
| Real Use | API failure logs, retry on network glitches, Allure attachments |
|  |  |

**AI Promot**

**. Overview of ITestListener**

* The code should use **ITestListener** to listen to events during the execution of test methods such as test start, success, failure, and skip.

**2. Key Features to Implement:**

* **Logging**: Capture logs at various stages of test execution.
* **Screenshots**: Take screenshots when a test fails.
* **Retry Handling**: Use ITestListener in combination with IRetryAnalyzer to handle retries for failed tests.
* **Reporting**: Attach logs and screenshots to reporting tools like **Allure** or **ExtentReports** for richer test reports.
* **Custom Behaviors**: Implement custom behaviors such as sending notifications or alerts when a test fails.

**3. Code Structure**

* **Base Framework Layer**: Show how to implement ITestListener in the base framework of the project, such as utilities and setup.
* **Test Lifecycle Management**: Use ITestListener to manage hooks for test events (e.g., when a test method starts, passes, fails, etc.).
* **Custom Reporting**: Demonstrate how to attach logs, screenshots, and other artifacts (like API response logs) to reports.

**4. Methods to Implement**

The code should implement the following ITestListener methods and explain each one:

* **onTestStart(ITestResult result)**: Triggered when a test method starts.
* **onTestSuccess(ITestResult result)**: Triggered when a test method passes.
* **onTestFailure(ITestResult result)**: Triggered when a test method fails (capture screenshot, logs, etc.).
* **onTestSkipped(ITestResult result)**: Triggered when a test is skipped.
* **onStart(ITestContext context)**: Triggered before any test starts.
* **onFinish(ITestContext context)**: Triggered after all tests in a suite finish.

**5. Code Example:**

The AI should generate a complete code example that implements ITestListener, with the following sections:

1. **ITestListener Implementation**
   * Implement a class that listens to test events using ITestListener and overrides the necessary methods.
   * Use logging, error handling, and reporting tools like **Allure** or **ExtentReports** to attach relevant information (e.g., logs, screenshots).
2. **Test Class**
   * Register the listener using **@Listeners** annotation in the test class or globally in **testng.xml**.
   * Show an example of a failing, passing, and skipped test.
3. **Custom Reporting Integration**
   * Use Allure or ExtentReports to demonstrate how the listener can make reports richer and more informative.
4. **Example Code for Screenshot and Log Capturing on Failure**
   * Provide code to capture a screenshot if a test fails and save it with a meaningful name.
   * Log the error stack trace when a test fails.

✅ What **ITestListener should do** (Responsibilities)



