Code Review Checklist

1. Clean Code

* Code should be clean and follow design patterns wherever applicable
* High Cohesion and Loose Coupling
* **Design Patterns - DRY and SRP**(Don’t Repeat yourself) / Single Responsibility
* is there a potential for useful abstractions

1. Concurrency

* Thread-safety and concurrency related guidelines
* Do not create adhoc threads, instead use shared **Executors** wherever possible.
* Immutability for better thread-safety.
* **Synchronize access** to shared mutable data.
* Use non-blocking libraries and framework (webflux, kotlin coroutines), where needed.

1. Reusability

* Existing libraries and methods should be reused wherever possible. Do not reinvent the wheels.

1. Performance

* Time and Memory complexity of the code.
* Release resources after use (File Handles, db connections, etc)
* Avoid dynamic SQL, use prepared statements

1. Error Handling

* Proper error handling for negative scenarios, input should be validated before any processing.
* Use checked exceptions for recoverable conditions and runtime exceptions for programming errors

1. Readability

* Classes and methods should be small in terms of the number of lines.
* Explain yourself in the code.
* Use Guava or **Apache commons libraries** wherever possible.
* Did I grasp the concepts in a reasonable amount of time by looking at the code?

1. Code Convention & Consistency

* Meaningful naming convention, code should explain itself. Name of the variables, methods and classes should be intention revealing in nature.
* Standard Code formatting across project
* Code consistency with the existing culture of writing the code.
* Consistent method signatures (return optional instead of null for Java code)

1. Regression

* Side effect on existing code, does this change break backward compatibility? Versioing of endpoint!

1. Security

* OWASP standards for the security
* Do not log any sensitive information, neither put it into the query parameters.
* Inter service communication
* Whether synchronous inter service communication can be avoided?
* How to handle service invocation failures
* Can we use non-blocking mechanism for inter service communication?
* Using Pub/Sub or event driven approach instead of synchronous communication wherever possible

1. Unit Testing

* Proper unit tests that covers the functional requirements.

Non- Functional Reqiremnet Testing for quality Testing

Rsponse Time ,Reliablity

Performance Bottelenecks

Scalability - volume of User

Stability : Withstatnd the loads

Advantage of Testing

. Validate the features

Measure speed , accuracy and sstability

Keep users happy

Improve optimization

Identify discrepsncies

Tools for Performace Testing

Apache JMeter

Load View

Load Runner

Web Load

Neo Load

LoadUI

Types of Performace Testing

1. Load Testing - Ability , Performace Botteneck

2. Endurance Testing - continous Load increase

3. Volume Testing - increase DB volumes

4. Scalabiity Testing - Increasing user Load

5. Spike - Extreme and High Extreme descrese

6. Stress Testing - identifying the Break point in the application

Issue

1. Longer Loading time

2. Coding errors or hardware issues

Performace Bottlenecks

Network utilisation

CPU utilization

Memory utilisation

SLimitation

Disk Usage

Parameters considered for Performace Testing

Memory Usage

BandWidth

Memory Pages

Processor Usage

Response Time

CPU interruption

Commited Memory

Thread Counts

Network Output Queue Length

Top Waits

Factors for selecting Tools

1. Customer Prefered Tools

2. Avaialability of licence

3. Test region

License Cose

Additional Protocol Support

Tool Efficiency

Vendo Support

throughput --> Ampunt of data transferrred to Server according toe the client request in a given period of time .

Common mistakes by the users

1. Test results are note validated

2. Too small durations

3. Concurrent Users

4. Data not populated.

5. Netwrok Bankwidth not simulated

Phases of Performace Testing

1. Design / Palnning

2. Build

3.Execution

4. Analysing

Benchmark testing adn base line testing

BenchMar --> comparing the perfromace of the application as defined by the other companies

Baseline --> Any future changes , this inforamtion will be used as reference

Concurrent User Testing -> Different User --> same Operations

Protocols :

Set of rules for purpose of commuication between 2 or more systems

HHTP

HTTPS

FTP

WebSerivces

Citrix

Performance Tuning :

Improvement in system performance

Types of Performace Tuning

Hardware / Software Tuning

Hardware : Enhancing ,adding or supllanting the hardware components to increase teh systems perfromace .

Software : Profiing the Code ,Database

Why no manual Load testing ?

Complicated Procedure

Comples sync procedures

Infrastructure code

Activities in Performace Testing

1. Requirement Gathering

2. Tool Selection

3. Test plan

4. Test development

5. Test Modeling

6. Test Execution

7. Anaysis

8.Report

Soak Testing : Stability and Perfromace over extended period time

JMeter

JMeter is JAva tool- Load Testing - Opensource Software

perfromace of Application - PreferABLE Tool

How JMeter Works

Group of User sending request to targetr Server .--> Stats and Graphj

Regular Expresion

Samplers and Thread Groups :

Samplers are Specific type of Request to the server ,thread group decides which type of request it need to maek

thread Group : UIsed to set number of Users and time to load all the user s given in the thread group.

Procesors : to modify the sampler in scope

Pre and Post Processors

Pre --> Parse the link and return HTML

Post Processor : After making Sampler requesst. If server response is errors

Pre processor Eleements :

HTTP URL re-witing modifier

HTML link parser

HTTP user parameter modifier

Beanshell Preprocessor

Timer in JMeter

Get a pause between request

Constant Timer --> Delays in same amount of time

Gaussian Random Timer --> each user delays

Synchronising Timer --> sync the wait period

Unifirm Random Timer --> delay random delay

Test Fragment ?

Assertion in JMeter?

HTML (Syntax), REsponse , Duration ,size in bytes( response byts)

,XML (format ),

Execution Orders

1. Configuration Elements

2. Pre-processors

3. Timers

4. Sampleres

5. Post Processors

6. Assertions

7. Listeners

Reduce resource requirement

-Use Non GUI

-Use Sample Sampler in loop

Dont use Functonal Testing

SPIKE TESTING

JMeter Listeners

View Result Tree

Monitor Results

Summary Report

BeanShell Listeners

Aggregate Report

Contollers in JMeter

Samplers Controller --> get, post del to HHTP

Logical Controller -->

Distributed Load Testing

- Using Master Slave System

- Request from Slaves

Load Runner is Licences Tool

It has User Interface

More Technical capbabity

supports SAP , SIebel and Peoplesoft

Plug-ins in Jmeter

-thread group Plugin

Samplers Plugin like Web driver

Listener Plugin

JMeter / Soap UI

workbench

Storage Area - to add componenets to be added to test plan if required