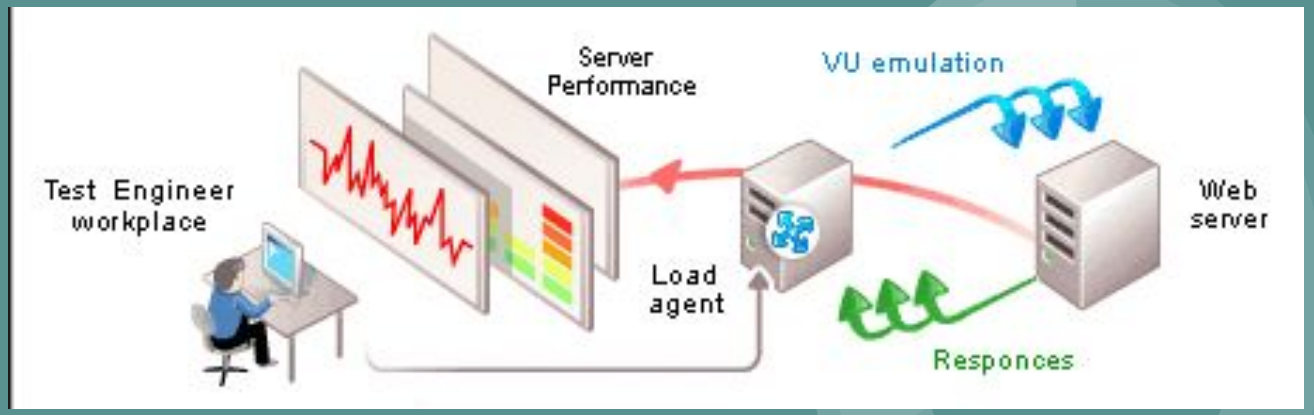

Load, Performance, and Reliability Testing

An Introduction

Introduction

- No matter how many tests you've run, once your application is nearly complete, there's really only one way to know whether or not your software **can handle the actual demands** your army of end users will soon be placing on it.
- It's called ***load testing***, and you can use a tool like Load Testing Tool to get the job done.

Introduction …2

- *“Load testing is the process of putting simulated demand on software, an application or website in a way that tests or demonstrates it's behavior under various conditions”*

- It is a sub-type of **Performance Testing**
- But different than **Stress Testing**

What can a Load Test Measure?

- *A load test enables you to measure:*

- *response times,*
- *throughput rates,*
- *and resource-utilization levels,*
- *and to identify your application's breaking point*

Example: Performance testing for a web app:

1

Define Testing Goals:
1 - response times.
2 - concurrent users,
3 - throughput,
4 - resource utilization,
5 - and scalability.

2

Identify Key Scenarios: Select the critical user scenarios for test. For a web application, these could include user registration, login, searching for items, adding items to a shopping cart, and checking out.

Example: Performance testing for web app..cont.

3

Choose Performance Testing Tools:
Select appropriate tools for performance testing. Popular choices include JMeter, LoadRunner, Gatling, and Apache Benchmark. These tools help simulate user interactions with your web application.

4

Create Test Scripts:
Develop test scripts that replicate user actions. These scripts simulate user behavior on the web application, such as navigating through pages, submitting forms, and making API requests.

Example: Performance testing for web app..cont.

5

Configure Test Environment:
Set up a test environment that closely resembles the production environment. This includes hardware, software, and network configurations. Ensure that the environment is isolated from the production system to prevent interference.

6

Create Test Scripts:
Develop test scripts that replicate user actions. These scripts simulate user behavior on the web application, such as navigating through pages, submitting forms, and making API requests.

Example: Performance testing for web app..cont.

7

Execute Tests:
Run performance tests with different scenarios and gradually increase the load to observe how the application behaves. This could involve a load test, stress test, and scalability test.

8

Collect Metrics and Analyze Results:
collect key performance metrics such as response times, throughput, error rates, and resource utilization. Monitor server health, database performance, and application server performance. Analyze the results to identify bottlenecks and areas for improvement.

Load Testing with MS Visual Studio 2018

- Visual Studio and Team Services provide a wide range of tools and capabilities for performance testing your websites and applications.

- You can scale your tests to **hundreds of thousands of concurrent users** and generate load from **multiple regions** worldwide.

Load Testing with VS 2018: Load Test Your App in the CLOUD

- **VS 2018** enables test engineers to load test their web sites in the cloud
- **20,000** virtual user minutes for free
- **Test where your users are**: Test from different locations to reduce latency and simulate local conditions.
- **Get deeper insights about performance**: Get full server and client diagnostics when you simulate real-life load patterns for your app.

Analyzing Load Test Results in VS 2018

- Top 5 Slowest Pages
  - The URL and the average page load time are displayed for each page
- Top 5 Slowest Tests
  - The name of the test and the average test time are displayed for each test.
- Top 5 Slowest SQL Operations
  - The name of the operation and the duration are displayed for each test

Best Practices to Load Testing

- **Page Load Time:**
  - If one page takes long time to load, you need to first analyze it's delivered content (eg. Loading a video?)
- **Ramping Up Virtual Users:**
  - Do not start with all virtual users artificial bottlenecks.
- **Setting Load Duration:**
  - Depends on scenario's duration
- **Parameterizing Tests:**
  - For example a search page should not be tested all time with same search data
- **Emulating Browser Speed and Bandwidth**
  - You need to be aware that connection speeds vary significantly
- **Simulating Concurrent Requests:**
  - Web browsers send multiple requests for each page content

Reliability Testing

- Reliability testing in software applications is performed to assess the software's ability to consistently and accurately perform its intended functions over an extended period of time.

- It aims to identify and measure the system's reliability, including its stability, availability, and robustness.
- Reliability testing is essential for ensuring that a software application meets user expectations and can handle real-world usage scenarios without unexpected failure.
- It helps build trust in the software's stability and performance, making it a critical part of the software development and quality assurance process.

Reliability Testing Process

1 - Define Reliability Requirements:

Start by defining the reliability requirements and objectives for your software application. These requirements should be based on the expectations of your users and the specific needs of your application.

2 - Identify Critical Scenarios:

Identify critical use cases or scenarios that are representative of how the software will be used in the real world. These scenarios should cover a range of inputs, conditions, and user interactions.

Reliability Testing Process

3 - Test Data and Environment Setup:

Prepare the necessary test data, including both typical and edge cases. Set up the test environment to closely resemble the production environment in terms of hardware, software, and configurations.

Types of Reliability Testing Methods:

a. Stress Testing:

Apply heavy loads and conditions to the software to assess how it performs under extreme circumstances. This can include testing with a high volume of concurrent users or excessive data loads.

b. Load Testing:

Measure the software's performance under expected loads to ensure it can handle the anticipated user base without degrading performance.

c. Volume Testing:

Verify that the application can handle large volumes of data without performance degradation or data corruption.

Types of Reliability Testing Methods:..2

d. Failover and Recovery Testing:

Evaluate the software's ability to gracefully handle failures, such as server crashes, network outages, and database failures, and recover without data loss.

e. Endurance Testing:

Continuously run the software under normal usage conditions for an extended period to identify memory leaks, resource leaks, and other issues that may degrade performance over time.

Automate Tests:

Automation tools can be used to run reliability tests for an extended period, allowing for consistent and repeatable testing.