Project Title:

“Performance Testing on JPetStore Web Application using OctoPerf”

Website/Endpoint:

- **Website URL:** https://petstore.octoperf.com/
- **Pages for Testing:** Focus on key areas of the website:
    - Home Page: /
    - Product Category: /catalogue
    - Search Functionality: /search
    - Shopping Cart: /cart
    - Checkout Process: /checkout

Project Overview:

This project involves performing a comprehensive performance test on the **JPetStore** web application. The objective is to assess the web application's ability to handle various levels of user traffic, simulate real-world customer interactions, and ensure that the site remains responsive under load.

Project Scope:

The performance testing will cover the following:

1. **Load Testing**: Evaluate how the application handles typical traffic patterns from users browsing products, adding them to the cart, and completing orders.
2. **Stress Testing**: Push the application beyond its limits by increasing the load to identify breaking points or bottlenecks.
3. **Spike Testing**: Simulate sudden bursts of traffic, such as a flash sale, and examine the recovery time and behavior.
4. **Endurance Testing**: Assess the application's stability under prolonged user activity (e.g., sustained heavy browsing and purchases over several hours).
5. **Scalability Testing**: Determine how well the application scales with increasing users and requests.

Project Requirements:

1. Setup and Access:

- Access the **JPetStore** web application at https://petstore.octoperf.com/.
- Identify the core user journeys to be tested (e.g., browsing products, adding to cart, and completing a purchase).

2. Tools Required:

- **OctoPerf** for creating and executing performance tests.
- **Apache JMeter** for generating traffic and simulating user behavior.
- **GitHub** for storing test plans and scripts.
- **Jenkins** for continuous integration and running automated performance tests.

Detailed Benchmarks to Measure:

A. Response Time Metrics:

1. **Average Response Time**:
   - **Description**: Measure the average page load time for key user actions like browsing, adding to cart, and checkout.
   - **Benchmark**: Page load time should be < 2 seconds for 90% of requests.
2. **Response Time (95th Percentile)**:
   - **Description**: Measure the response time for the slowest 5% of requests.
   - **Benchmark**: Should be < 3 seconds for 95% of requests.

B. Throughput and Load Handling:

3. **Requests per Second (Throughput)**:
   - **Description**: Measure the number of successful requests the web application can handle per second.
   - **Benchmark**: The site should handle at least 1000 requests per second under normal load.
4. **Concurrent Users**:
   - **Description**: Measure how many users can simultaneously browse and shop on the website.
   - **Benchmark**: The application should support at least 300 concurrent users without performance degradation.

C. Error Rate:

5. **Error Rate under Load**:
   - **Description**: Track the percentage of failed requests under normal and stressed conditions.
   - **Benchmark**: Error rate should remain < 2% during normal load and < 5% under stress conditions.

D. Network Performance:

6. **Latency**:
   - **Description**: Measure the network latency between the user and the web server.
   - **Benchmark**: Should be < 300ms for most requests.

Test Scenarios to Execute:

1. **Scenario 1: Load Testing**
   - Simulate 300 concurrent users browsing products, adding items to the cart, and completing a purchase for 30 minutes.
2. **Scenario 2: Stress Testing**

    - Increase the user load beyond the expected limit (e.g., 500 concurrent users) to determine the breaking point.
3. **Scenario 3: Spike Testing**
    - Introduce a sudden increase in traffic (e.g., 1000 users accessing the site within a short span of 10 seconds) to observe recovery time.
4. **Scenario 4: Endurance Testing**
    - Simulate 200 users continuously interacting with the site over a period of 2 hours to test for long-term stability.

P.S. Run all .JMX file from Jenkins, configured with Git.

Deliverables:

1. **Performance Test Plan:**
    - Document outlining the tools used, strategy, benchmarks, and test execution timeline.
2. **Performance Testing Report:**
    - **Summary of all tests performed.**
    - **Graphs** showing response times, throughput, errors, and network latency.
    - **Bottleneck identification** and suggestions for improvement.
    - **Recommendations** for enhancing the performance and scalability of the web application.
3. **JMeter Test Scripts:**
    - Submit all JMeter .jmx files with the configured test scenarios.

Report Submission Template:

1. **Introduction:**
    - Overview of **JPetStore** and the purpose of the performance test.
    - Performance goals (e.g., response time, throughput, error rates).
2. **Test Setup:**
    - Tools: **OctoPerf**, **JMeter**, **Jenkins**, **GitHub**.
    - Environment setup: Hardware (CPU, RAM), network, etc.
    - Key user actions tested (e.g., browsing, checkout).
3. **Test Results:**
    - **Response Time Analysis**: Average, 95th percentile, and 99th percentile response times.
    - **Throughput**: Requests per second under different loads.
    - **Error Rates**: Error rates and HTTP status code distribution.
    - **Network Latency**: Impact of network conditions on performance.
4. **Conclusion:**
    - Summary of findings: Whether the web application met performance expectations.
    - Insights into bottlenecks and areas for improvement.

    - Recommendations for optimization
5. **Appendix:**
    - JMeter .jmx files.
    - Screenshots and graphs from performance tests.
    - Detailed explanation of tools and configurations used.

Additional Notes:

- **Project Duration:** To be completed within 5 days.
    - **Days 1-2:** Set up tools and environment.
    - **Day 3:** Develop JMeter scripts and OctoPerf scenarios.
    - **Days 4-5:** Execute tests and generate reports.