1. Performance Testing Plan for a Web Application

Objective:
To evaluate the web application's performance in terms of response time, stability, and scalability, ensuring it meets user expectations under normal and peak load conditions.

Key Components

Scope:
The performance testing will cover the web application’s core functionalities, including:

- User login and registration
- Product search and filtering
- Add to cart and checkout processes
- Payment gateway and confirmation page
- Account management features

Testing Goals:

- Determine the application’s response time under varying loads.
- Identify bottlenecks and performance degradation points.
- Ensure the application maintains acceptable performance under peak loads.

Tools:

- **K6**: For API performance testing (ensuring backend APIs can handle load).
- **Blazemeter**: For load testing and simulating large numbers of concurrent users on the front end.
- **Apache JMeter**: For testing the application’s response under various conditions (stress and endurance testing).

Metrics to Measure:

1. **Response Time**: Average time for requests to return results.
2. **Throughput**: Number of transactions processed per second.
3. **Error Rate**: Percentage of failed requests.
4. **Peak Load**: Maximum concurrent users the application can handle.
5. **Resource Utilization**: CPU, memory, and network usage during tests.
6. **Latency**: Time taken for data to travel between client and server.

Performance Testing Scenarios

1. **Load Testing:**

   - Objective: Test if the application can handle expected user loads.
   - Scenario: Simulate typical user load (e.g., 500-1000 concurrent users) performing a series of actions.
2. **Stress Testing:**
   - Objective: Identify the breaking point by applying higher-than-normal loads.
   - Scenario: Incrementally increase load until performance degrades or the system fails.
3. **Endurance Testing (Soak Testing):**
   - Objective: Assess stability over extended periods.
   - Scenario: Maintain a steady load for several hours to detect potential memory leaks or other issues.
4. **Spike Testing:**
   - Objective: Evaluate how the system handles sudden traffic spikes.
   - Scenario: Simulate a large number of users logging in at once.

Execution Plan:

1. Set up environments (test servers, databases, etc.).
2. Define user scenarios in K6, Blazemeter, and JMeter based on the key functionalities.
3. Run tests, collect data, and monitor resource usage.
4. Analyze results and identify performance bottlenecks.
5. Document findings and share recommendations for optimization.