Name: Tural Mehtiyev
Project Title: Scalability experiment of microservice architecture on an online bookstore application
Date: 11.07.2023
Report: Midterm Report

Revision of the project Roadmap

1. **Definition the Project Concept:** Done

- Identification of the purpose and goals of microservice architecture.
- Determination of the key functionalities and services that the architecture will provide.
- Understanding the expected performance requirements, such as response time, throughput, and scalability.

Description of work:

Microservices for online bookstore application.

Product Catalog Microservice:

- Responsible for managing the product inventory, details, and availability.
- Provides APIs for retrieving product information, searching for products, and updating inventory.

Shopping Cart Microservice:

- Handles the management of customer shopping carts.
- Provides APIs for adding items to the cart, removing items, and updating quantities.

Order Management Microservice:

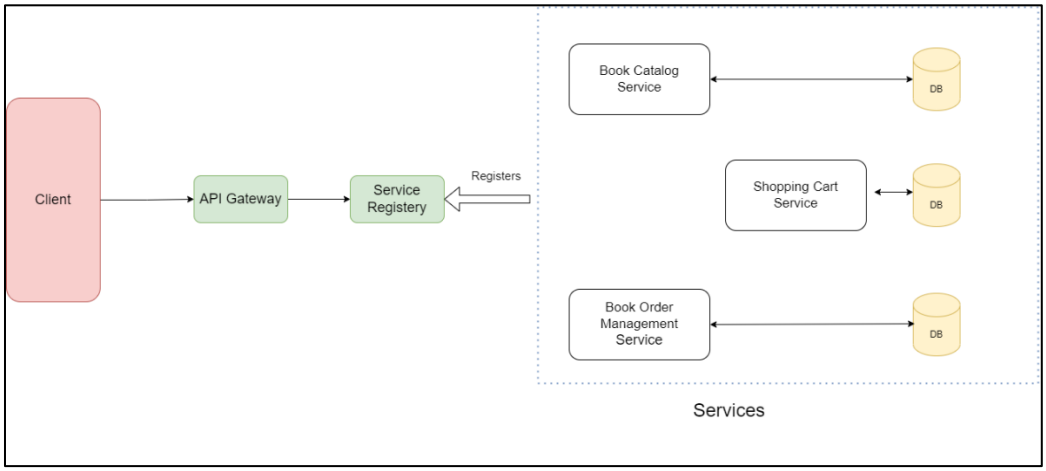
- Manages the processing of customer orders.
- Handles order placement, order fulfillment, and order status updates.
- Provides APIs for creating new orders, retrieving order details, and updating order status.

2. **System Design:** In Progress

- Definition of the overall architecture of microservices, including their relationships and interactions.
- Determination of the flow of the process and the sequence of microservice invocations.
- Identification of the inputs and outputs of each microservice.
- Decision on the communication protocols, such as REST, messaging queues, or event-driven architectures.
- Determination of the data storage requirements and select appropriate databases or data stores.

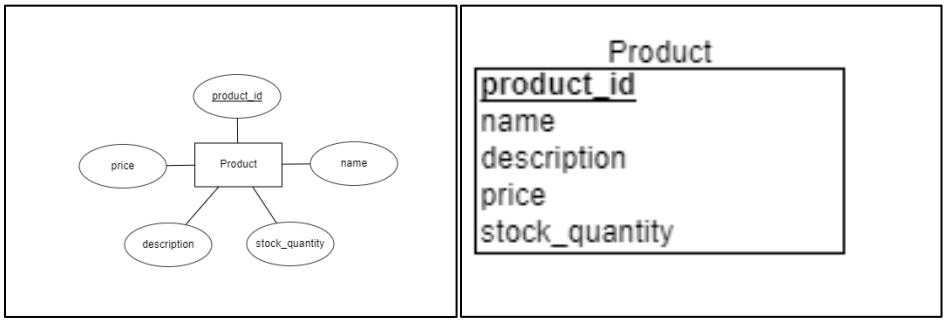
Description of work:

Draft Architecture

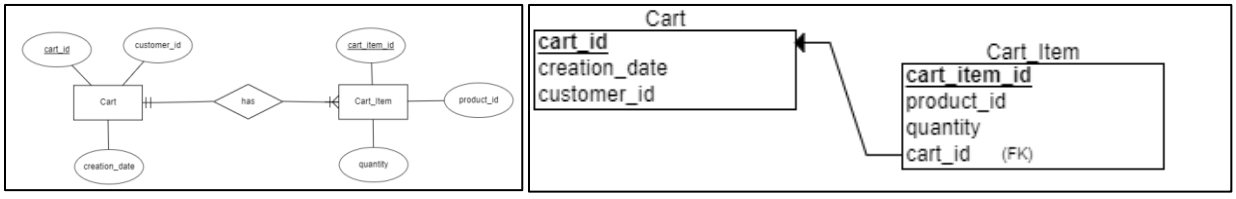


Entity Relationship Diagram and Schema for Microservice Databases

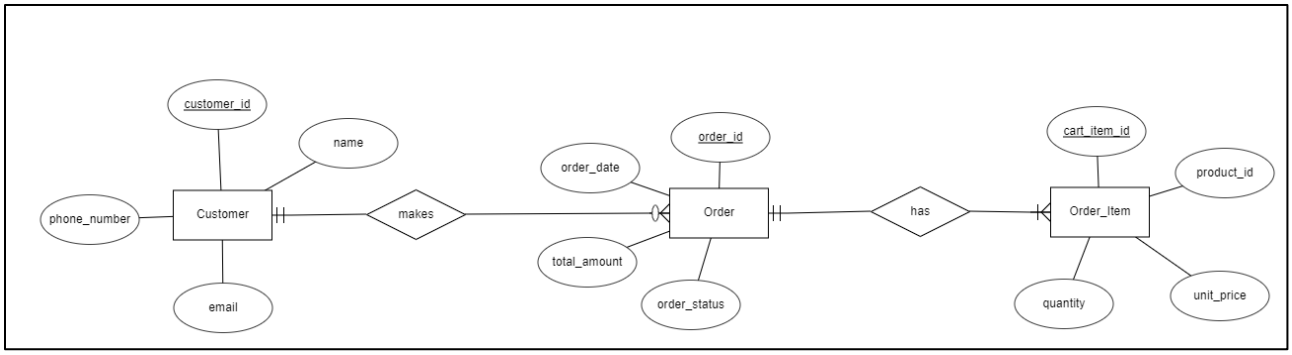
Book Catalog Service:

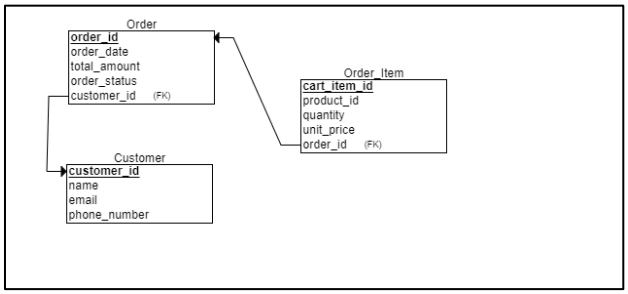


Shopping Cart Service:



Book Order Management:





Despite the given progress above, I am still working on the architecture of the application on the following issues:

- Determination of the flow of the process and the sequence of microservice invocations.
- Identification of the inputs and outputs of each microservice.
- Decision on the communication protocols, such as REST, messaging queues, or event-driven architectures.
- Determination of the data storage requirements and select appropriate databases or data stores.

3. **Identification of Performance Metrics and Measurement Strategy:** Done

- Determination of the performance metrics that you want to measure during the performance test.
- Aligning the chosen performance metrics with the project goals and requirements.

Description of work:

To measure the effect of resource scaling on a microservice application's performance, I will employ several measurement strategies.

Load Testing:

Characteristics: Simulation of real-world loads and measure the system's response.
Variables: Concurrent users/requests, response time, throughput, CPU and memory usage.
Roles:

- **Concurrent users/requests**: Represents the number of users or requests accessing the system simultaneously.
- **Response time**: Measures the time taken for the system to respond to a request.
- **Throughput**: Indicates the number of requests the system can handle per unit of time.

Key Assumption: The load test environment closely resembles the production environment, and the user behavior is representative.

Expected Outcome: Increasing the number of concurrent users/requests will negatively impact response time and throughput.

- **Null Hypothesis**: Increasing the number of concurrent users/requests does not impact response time and throughput.

- **Alternative Hypothesis**: Increasing the number of concurrent users/requests does affect impact response time and throughput.

Identification of bottlenecks

To identify the main bottleneck in the overall application, a manual approach will be adopted whereby each application will be individually scaled and the response rate of the entire application will be measured.

Scalability Testing:

Characteristics: Assess the system's ability to handle increased workloads with growing resources.
Variables: Workload, scalability indicators (e.g., response time, throughput).
Roles:

- Workload: Represents the amount of work imposed on the system.
- System resources: Measure the allocation and utilization of resources necessary for the system's operation.
- Scalability indicators: Quantify the system's performance, such as response time and throughput.

Data Normality Assurance:

The principles of the Central Limit Theorem (CLT) and the Law of Large Numbers (LLN) will be applied in my experiment to achieve normalization with the following steps.

- **Determining the number of samples and the sample size:**

  I will have at least 50 observations in each sample with total 100 samples for each scenario.

- **Collection of Sample Data:**

  I will perform multiple runs experiment for each experiment scenario, where each run represents a sample. For example, I will collect response time and throughput data for each samples in each scenario.

- **Calculation of Sample Means:**

  I will calculate the mean of each sample (e.g., average response time and throughput for each sample) for 2 scenarios. These sample means will serve as data points for normalization. As a result, I will have a distribution of sample means which will guarantee the normal distribution.

Expected Outcome: Increasing system resources (e.g., adding more instances, nodes, or containers) will result in improved performance.

- **Null Hypothesis**: Increasing system resources does not affect scalability, resulting in the same response time and throughput.
- **Alternative Hypothesis**: Increasing system resources does affect system's scalability, resulting in the improved response time and increased throughput.

Accepting or Rejecting Hypothesis:

To determine whether to accept or reject your hypothesis, compare the performance metrics obtained from different test scenarios (e.g., with varying numbers of instances or resources) will be compared and the collected data will be assessed if there is a noticeable improvement in performance and throughput as the resources scale. Statistical analysis techniques such as t-tests will be used to evaluate the significance of the results and determine if they support your hypothesis.

4. **Application Development:** In Progress

- Implementation of the application with the planned architecture
- Setting up the test environment

Description of work:

I am still watching several online resources and learning about the effective design of microservice applications.

5. **Test Scenario Design:** Done

- Definition of realistic test scenarios that mimic the expected usage patterns of microservices.
- Determination of the workload, including the number of concurrent users, request rates, and data volumes.
- Designing test cases to cover different types of interactions and inputs for microservices.
- Consideration of both normal and peak load scenarios to assess performance under various conditions.

Description of work:

Test Scenario

- Scenario 1: Single Node Configuration

  Microservice 1: 1 instance
  Microservice 2: 1 instance
  Microservice 3 (Bottleneck): 1 instance

- Scenario 2: Multiple Node Configuration (Scaling the Bottleneck Microservice)

  Microservice 1: 1 instance
  Microservice 2: 1 instance
  Microservice 3 (Bottleneck): 2 or more instances (scaled)

Design of Load Tests

Below is an initial draft outline of the load tests that will be performed using Apache JMeter:

Test 1: Baseline Performance Test (Low Load)

- ✓ Concurrent Users: 50
- ✓ Ramp-up Period: 10 seconds
- ✓ Test Duration: 5 minutes

Test 2: Scalability Test (Medium Load)

- ✓ Concurrent Users: 100
- ✓ Ramp-up Period: 10 seconds
- ✓ Test Duration: 5 minutes

Test 3: Scalability Test (High Load)

- ✓ Concurrent Users: 200
- ✓ Ramp-up Period: 5 seconds
- ✓ Test Duration: 10 minutes

Note: The given all mentioned cases above will be applied to both scenario (with and without scaled versions)

6. **Load Generation:** In Progress

- ➢ Selection of a suitable load testing tool, such as Apache JMeter, Gatling, or Locust.
- ➢ Configuration of the load testing tool to generate the desired workload based on the defined test scenarios.
- ➢ Generation of realistic user behavior by incorporating think times, session management, and user profiles.

Description of work:

I have chosen Apache JMeter tool for the performance of testing and am currently watching a tutorial series for the configuration and effective usage of the tool.

7. **Performance Test Execution:** Not Started

- ➢ Execution the performance tests using the defined test scenarios and load profiles.
- ➢ Monitoring and collecting performance metrics during the test execution.
- ➢ Gathering relevant data on response times, throughput, error rates, and resource utilization.
- ➢ Ensuring that the tests run for a sufficient duration to capture stable performance measurements.

Description of work:

Since I have not fully set up Apache JMeter and I do not have the application ready, this part is still not started at this point.

8. **Analyze and Interpret Results:** In Progress

- Analysis of the collected performance data and metrics.
- Identification of any performance bottlenecks, such as slow microservices, high response times, or resource limitations.
- Interpretation of the results in the context of the project goals and performance requirements.
- Comparison of the observed performance against the defined benchmarks or thresholds.

Description of work:

I am analyzing similar papers for potential methodology of analysis and the visuals used to describe the results of analysis. I have described the sample visuals (histograms, radar plots) that I am also going to use in report 3.

9. **Documentation and Reporting:** Not Started

- Documenting the performance test procedures, configurations, and results.
- Creation of a detailed report summarizing the performance findings, optimizations, and recommendations.
- Sharing the report with relevant stakeholders and project teams.

Description of work:

I am planning to start this stage once I am fully done with the experiment and the results are ready. The expected date for this stage is the last week of the project.