Your computer's time zone America/Chicago (GMT -6:00) does not match your account's time zone of US/Eastern (GMT -5:00). Click here to update your time zone preferences.

CS590DE-2020-02A-05D >
Tests & Quizzes

Tests & Quizzes

Final exam

Return to Assessment List

Part 1 of 1 - 93.0 / 100.0 Points

Question 1 of 15 8.0 10.0 Points

[10 minutes]

For which of the following architecture requirements would you use **CQRS**?
Select all architectural requirements so that if this architectural requirements applies, you would apply the CQRS pattern.

- [ ] A. When you need future proof systems
- [ ] B. When you build a distributed system
- [x] C. When read performance is critical and need to be extremely fast for your microservice
- [x] D. When you design your microservice using Domain Driven Design
- [x] E. When queries and commands have different scaling requirements for your microservice
- [ ] F. When your screens start to look very different then your tables in your microservice
- [ ] G. When you have an event driven architecture
- [ ] H. When you need load balancing for your microservice
- [ ] I. When you use a NoSQL database for your microservice
- [ ] J. When strict consistency is required for your microservice
- [ ] K. When security is important for your microservice

Answer Key: C, E, F

Question 2 of 15 10.0 10.0 Points

[10 minutes]

Explain clearly the difference(s) between a **Service Oriented Architecture (SOA)** and a **Microservice architecture.**

Service-Oriented Architecture:
a. Course grained services
b. Different services or Component Connected through ESB
c. Complex ESB is used
d. Service in SOA itself a Monolith
e. Single point of failure (if service bus fails)

Microservice Architecture:
a. Fine-grained services
b. Independent services can run independently (run as a different process)
c. Simple and lightweight services
d. Not Technology Agnostic (free to choose any language for different services)
e. Even though one service fails others can still run

In Microservice Architecture, Services are independently deployable, decoupled, and ensures high availability using different techniques.

Question 3 of 15 5.0 5.0 Points

[10 minutes]

a. Explain how **orchestration** works in a distributed system

b. Give the advantages and disadvantages of **orchestration**

Orchestration is the idea the same as Orchestra where one person leads others. In the distributed system one service will be primary i.e. central command.

For example:
=> Product service (Order service, review service, payment service)

in this example, Product service is commanding other services.

Advantages:
a. Central Command
b. Easy to use
c. Simple

Disadvantages:
a. single point of failure: If Command service fails all the system goes down.

In particular, unlike choreography there is no coordination. The main problem is a single point of failure in Orchestration. In the distributed system, Choreography is mostly preferred over Orchestration.

Question 4 of 15 5.0 5.0 Points

[10 minutes]

a. Explain how **choreography** works in a distributed system

b. Give the advantages and disadvantages of **choreography**

Choreography is the idea where Services work in groups same like Choreography in real life. For example:
Product Service---> Review Service
Product Service---> Order Service

In this example there is no central command like Orchestration.

Advantages:
a. Adding a new service is easy.
b. Works collaboratively.
c. No single point of failure.

Disadvantages:
a. Applies mostly in complex system
b. if not handled properly can have a performance impact

In particular, the distributed system prefers Choreography over Orchestration.

Question 5 of 15 6.0 10.0 Points

[10 minutes]

a. Explain clearly **Conways law**

b. Explain clearly what Conways law means for a microservice architecture.

Conway's Laws basically addressed to manage microservices. Organizing microservice is very crucial. There are different teams in the organization. The developer has its own perspective, the Business team has there own perspective and Operation people has their own perspective.

Build, test, release these are the essential things in the fast delivery. Basically, all the code will manged in the

GIT repository. Jenkins plays a vital role to pull the code and build the script. Script help to make artifact in Nexus. From there based on our environment we can deploy our product to different environments (eg. production, Test).

Conway's law basically talks about managing the microservices in an organization.

Question 6 of 15 10.0 10.0 Points

[15 minutes]

a. Explain clearly why local logging does not work in a microservice architecture

b. Explain clearly how logging should be implemented in a microservice architecture.

In Microservice Architecture, logs will come from different services. Unlike Monolith where logs come from one Monolith service.

For example in Microservice Architecture:
Microservice 1----> log
Microservice 2---> log
.....
.....
Microservice n ---> log

logs are coming from different services. Maintaining the log differently does not help. hence, local logging does not work here.

SOLUTION:
In a microservice architecture, we can use ELK where the log is centralized, searched, and monitored as per our requirement.
a. Logs can be centralized or collected using Logstash.
b. Collected data can be searched using Elastic search
c And Kivana can be used to visualize the log

Question 7 of 15 2.0 2.0 Points

[2 minutes]

In one short sentence explain the problem that **Zipkin** solves

Tracing the microservice becomes essential when we have multiple services running simultaneously. Zipkin is used to tracing the services. Based on the trace id given by sleuth, Zipkin helps to trace the microservices.

steps:
simply add the dependency of the sleuth and Zipkin's and you can even visualize the service trace in UI tools given by Zipkins. Also, we can view the dependency of the services using Zipkin.

Our service may be doing back and forth with various internal and external services. Tracing becomes vital to know about service communication. Nowadays, Organizations are using many other tools as well, however whole idea is to trace the services.

Question 8 of 15 2.0 2.0 Points

[2 minutes]

In one short sentence explain the problem that **Ribbon** solves

The ribbon is a client-side load balancer. To know about RIbbon and working mechanism I did experiment using lab exercise. I am going to write the same experience:
=>, for example, we have two instances of the service, and one instance is failed or not available. The ribbon helps to find other instances of the service. It follows the Round Robbin algorithm and service instance becomes available.

The ribbon helps for the client-side load balancing of the application.

Question 9 of 15 2.0 2.0 Points

[2 minutes]

In one short sentence explain the problem that **Zuul** solves

Zuul provides a proxy API gateway to access other service or applications. It helps to make every application call loosely coupled by using a service registry (i.e Eureka).

Zuul is an API gateway where the client can make requests and Zuul help to call the service from the service registry. Usually we call using service Id or name.

Question 10 of 15 2.0 2.0 Points

[2 minutes]

In one short sentence explain the problem that **Eureka** solves

Eureka is the service registry like our phone book. Eureka helps to register the services and makes them available whenever we want them to call. It makes application call loosely coupled, which means we don't need to know the physical location of the services. Also, Eureka knows when service is not available. Eureka continuously monitors the services.

Question 11 of 15 2.0 2.0 Points

[2 minutes]

In one short sentence explain the problem that **Hystrix** solves

Hystrix makes our application Resilient. By providing a fallback method Hystrix helps to solve the problem wherein the case suppose our method fails or application unable to get the expected results. The very useful case could be the timeout. For example service unable to respond because of network error or other external factors.

When something unusual happens during service call it will direct to fallback method which makes our application resilient.

Question 12 of 15 9.0 10.0 Points

[10 minutes]

Explain clearly how a **circuit breaker** works in a microservice.

Explain the 3 techniques that a circuit breaker uses.

The circuit breaker is the mechanism to make our service resilient. When some exceptional event that is not in our expectation happens then it breaks the circuit. For example, One service is slow because of a network error, we are not supposed to make our system wait for long or forever instead we need to know why that is happening.

Three techniques circuit breaker uses:
a. Timeout: when the service not able to respond within a specified time slot, a timeout occurs and breaks the circuit.
b. Circuit breakers: when undesirable state service goes through then it breaks the circuit (one possible scenario could be timeout)
c. Bulkhead: It maintains the Thread pool and thread queue. We and define thread we can handle and queue to make the wait for the incoming thread. If the thread pool is over the queue is full then circuit breaker might happen.

Question 13 of 15 10.0 10.0 Points

[15 minutes]

Kafka uses **event sourcing** for its messages.

a. Explain clearly what this means.

b. Explain clearly why kafka uses event sourcing. What are the advantages.

Events are immutable. Kafka handles the real-time stream of data and it acts as a database to store the events. Different events are stored in the event stream. It has topics and brokers, messages coming in different events are added in the partition. We can have multiple partitions in the broker as well.

Kafka solves the problem of blackboard where the issue was synchronization. In Kafka no synchronization issue because messages are added but no modification of existing messages. Event sourcing gives more knowledge about the system and nowadays it is becoming more popular. Storing events and that leads to application states helps for more analytical information about the business.

Advantages of Kafka:
a. No synchronization issues like blackboard
b. Can store messages even the receiver can take it later
c. Supports event sourcing which helps to get more analytical information in future
d. works in real-time streaming data.
e. it can solve the problem to maintain the state and caching of the application. No matter how many consumers we added, it is a stable and fine way to consume published messages.

Question 14 of 15 15.0 15.0 Points

[10 minutes]

For securing a microservice architecture we looked at **Oauth2** and **JWT**.

a. Explain the difference between Oauth2 and JWT.

b. What does JWT add to Oauth2/ In other words, why do we need JWT if we have Oauth2?

a. Oauth2 is a framework where JWT is a standard token. Oauth2 is an authorization server that gives a token to access the service if the user is valid. JWT provides more information about users and gives standard token working on top of OAUTH2.

The primary difference between JWT and OAUTH2 is, OAUTH2 is a framework (Authorization Server) and

JSON Web Token (JWT) is a standard token service.

b.
If we use OAUTH2 only to secure our services, we need to call our authorization server every time we need to call our resource. JWT provides a standard way with more information so that when OAUTH2 gives the token it stores along with user information and we don't need to call Auth2 to verify again. It is a more secure and standard way to maintain our authentication and secure REST API. Using Oauth2 and JWT is the most preferred way to secure endpoints and easy to understand the flow of execution. JWT standardizes our token.

Question 15 of 15 5.0 5.0 Points

[10 minutes]

Describe how we can relate the **API Gateway** to one or more principles of SCI. Your answer should be about 2 to 3 paragraphs. The number of points you get for this questions depends how well you explain the relationship between the **API Gateway** to one or more principles of SCI.

API gateway is the main entry point of our application which is similar to transcendental consciousness through which we can control and enter into our whole physical and mental system. There is one very popular SCI principle called, SEEK THE HIGHEST FIRST through which we can see the whole thing going on in our life. First know who you are and understand the value of yours. Experience the transcendental consciousness and know the value of your life. do not focus on unwanted things. Know the huge potential inside you which is in the hidden state; just by knowing the unified field. Similarly, just access the whole system through the API gateway. Do not need to know whirling swirling details of the system, access the services even without knowing their physical address through API gateway.

Another very popular SCI principle I like is HARMONY EXISTS IN DIVERSITY. Through API gateway we can access the different services, those are responsible to be one complete application. Similarly, we meet with different people and share our knowledge and experience which helps to strengthen our knowledge. Different organs are functioning differently, Many thoughts are coming and going. Adding all these diverse things makes the whole system and we can control that diversity from one single point which is Transcendental consciousness. In the API gateway, many services are integrated or accessible through it that makes the whole system. May be one service is .NET, one is Python, and others in Java. These all help to make a single robust application. hence, Harmony exists in diversity.

In summary, like API gateway we know there is a tool to access all our internal and external services of the body i.e TM. Experience life and live happily. Blissful life is the best gift ever. API gateway is the way to access all the services of any enterprise application and Transcendental consciousness is the perfect gateway to explore happiness in our life.

- Gateway
- Accessibility Information
- The Sakai Project
-

MUM Global Online Education 19.3

Sun, 17 Jan 2021 16:21:26 CST
Your Preferred Time:
Sun, 17 Jan 2021 17:21:26 EST
Server:
sakai1
Build Info:
RELEASE

- Powered by Sakai

Change Profile Picture

Error removing image
Error uploading image
Upload Choose File No file chosen

Save Cancel Connections Remove
Search for people ...
View More

My Connections | Pending Connections

You don't have any connnections yet. Search for people above to get started.
You have no pending connections.
←Back to My Connections
Search for people ...
$({cmLoader.getString("connection_manager_no_results")}

Done