Spring Professional Exam Tutorial v5.0

Question 05

Question 05 - Is REST scalable and/or interoperable?

Is REST scalable?

YES

Is REST interoperable?

YES

Question 05 - Is REST scalable and/or interoperable?

Scalability of RESTful Service is a result of developing software with following characteristics in mind:

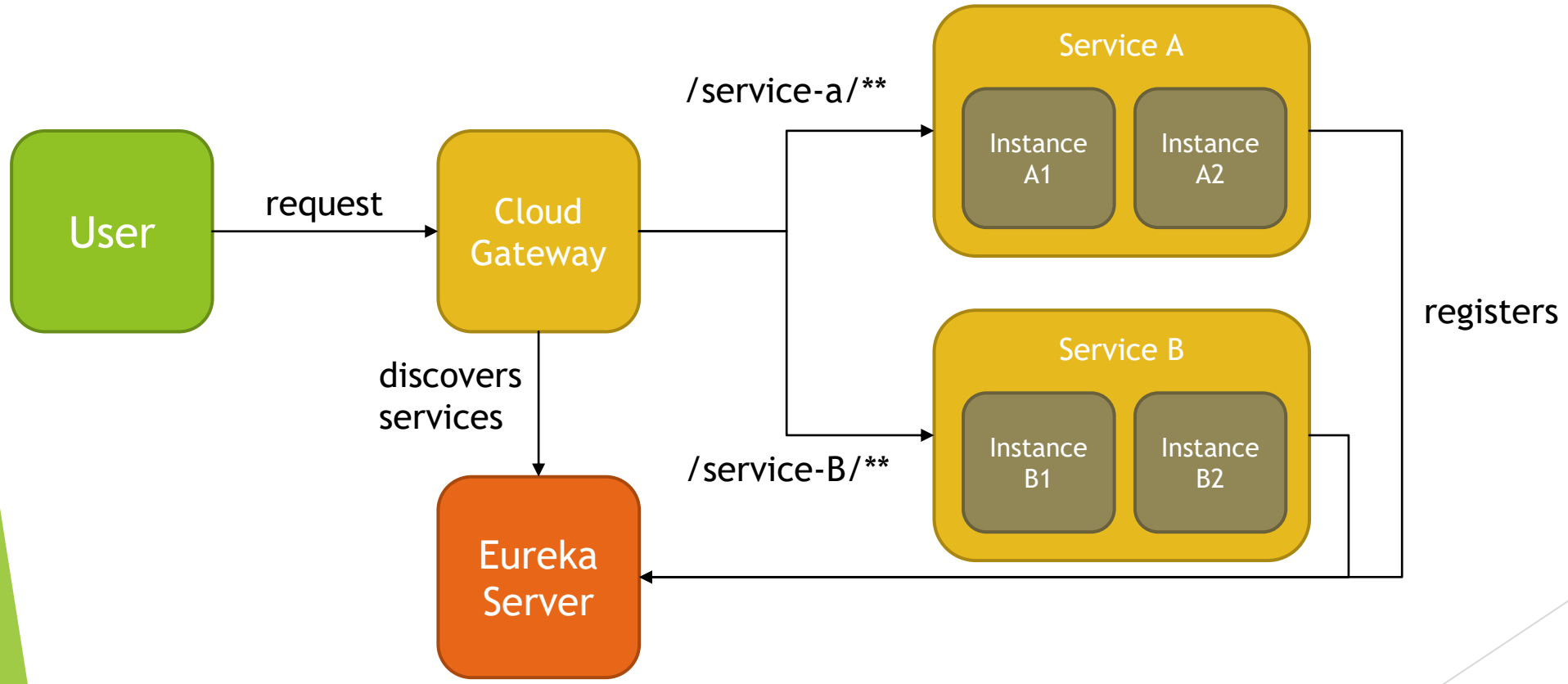
- **Statelessness -** each request to the system, should be design in a way, for it to be processed without having to keep any state at the backend side, for example, we want to avoid keeping information in HTTP Session related to user conversation with the system, this way we can delegate request to any backend node that can process the request without having to introduce share state between nodes

- **Layered Approach** – layered approach to the system design means that we can introduce new parts of the system in a way for it to be transparent to the client, resulting in ability to change system without having to modify client, example of this can be introduction of Application Load Balancer, API Gateway, Security Layers, Web Application Firewall without having to change client at all

- **Cacheability** – allows to create response for repeatable requests, without having to process them on service side, caching is introduced to improve response time and to reduce load on the service

Question 05 - Is REST scalable and/or interoperable?

Scalability that can be achieved by statelessness of RESTful Service is especially visible in Microservice Architecture, that can be created with usage of Spring Cloud Components. Layered Approach allows introduction of additional components without changing client application.



Question 05 - Is REST scalable and/or interoperable?

REST Service is interoperable because:

- Access to REST Service and resources available by URIs is standardized and not coupled with any specific technology, allowing you to consume REST Service in any technology of choice, like JavaScript, Python, Java, C++ etc.

- Data for the requested resource can be sent to client in different formats specified by the client, in case of HTTP protocol this can be done with usage of `Accept` header, for example `Accept: application/json` or `Accept: application/xml`

- All CRUD operations can be handled with standardized approach, in case of RESTful Service implemented with HTTP protocol, standardized HTTP methods `GET, PUT, PATCH, POST, DELETE` are used