MCPA-LEVEL-1-MAINTENANCE Q&As

MuleSoft Certified Platform Architect - Level 1 MAINTENANCE

Pass Mulesoft MCPA-LEVEL-1-MAINTENANCE Exam with 100% Guarantee

Free Download Real Questions & Answers **PDF** and **VCE** file from:

https://www.pass4itsure.com/mcpa-level-1-maintenance.html

100% Passing Guarantee
100% Money Back Assurance

Following Questions and Answers are all new published by Mulesoft Official Exam Center

QUESTION 1

What is a typical result of using a fine-grained rather than a coarse-grained API deployment model to implement a given business process?

A. A decrease in the number of connections within the application network supporting the business process

B. A higher number of discoverable API-related assets in the application network

C. A better response time for the end user as a result of the APIs being smaller in scope and complexity

D. An overall tower usage of resources because each fine-grained API consumes less resources

Correct Answer: B

A higher number of discoverable API-related assets in the application network.

****

>> We do NOT get faster response times in fine-grained approach when compared to coarse-grained approach.

>> In fact, we get faster response times from a network having coarse-grained APIs compared to a network having fine-grained APIs model. The reasons are below.

Fine-grained approach:

1.

will have more APIs compared to coarse-grained

2.

So, more orchestration needs to be done to achieve a functionality in business process.

3.

Which means, lots of API calls to be made. So, more connections will needs to be established. So, obviously more hops, more network i/o, more number of integration points compared to coarse-grained approach where fewer APIs with

bulk functionality embedded in them.

4.

That is why, because of all these extra hops and added latencies, fine-grained approach will have bit more response times compared to coarse-grained.

5.

Not only added latencies and connections, there will be more resources used up in fine- grained approach due to more number of APIs.

That\\'s why, fine-grained APIs are good in a way to expose more number of resuable assets in your network and make

them discoverable. However, needs more maintenance, taking care of integration points, connections, resources with a

little compromise w.r.t network hops and response times.

QUESTION 2

A new upstream API Is being designed to offer an SLA of 500 ms median and 800 ms maximum (99th percentile) response time. The corresponding API implementation needs to sequentially invoke 3 downstream APIs of very similar

complexity.

The first of these downstream APIs offers the following SLA for its response time: median:

100 ms, 80th percentile: 500 ms, 95th percentile: 1000 ms. If possible, how can a timeout be set in the upstream API for the invocation of the first downstream API to meet the new upstream API\\'s desired SLA?

A. Set a timeout of 50 ms; this times out more invocations of that API but gives additional room for retries

B. Set a timeout of 100 ms; that leaves 400 ms for the other two downstream APIs to complete

C. No timeout is possible to meet the upstream API\\'s desired SLA; a different SLA must be negotiated with the first downstream API or invoke an alternative API

D. Do not set a timeout; the Invocation of this API Is mandatory and so we must wait until it responds

Correct Answer: B

Set a timeout of 100ms; that leaves 400ms for other two downstream APIs to complete

****

Key details to take from the given scenario:

>> Upstream API\\'s designed SLA is 500ms (median). Lets ignore maximum SLA response times.

>> This API calls 3 downstream APIs sequentially and all these are of similar complexity. >> The first downstream API is offering median SLA of 100ms, 80th percentile: 500ms; 95th percentile: 1000ms.

Based on the above details:

>> We can rule out the option which is suggesting to set 50ms timeout. Because, if the median SLA itself being offered is 100ms then most of the calls are going to timeout and time gets wasted in retried them and eventually gets exhausted

with all retries. Even if some retries gets successful, the remaining time wont leave enough room for 2nd and 3rd downstream APIs to respond within time.

>> The option suggesting to NOT set a timeout as the invocation of this API is mandatory and so we must wait until it responds is silly. As not setting time out would go against the good implementation pattern and moreover if the first API is

not responding within its offered median SLA 100ms then most probably it would either respond in 500ms (80th percentile) or 1000ms (95th percentile). In BOTH cases, getting a successful response from 1st downstream API does NO

GOOD because already by this time the Upstream API SLA of 500 ms is breached. There is no time left to call 2nd and 3rd downstream APIs. >> It is NOT true that no timeout is possible to meet the upstream APIs desired SLA. As 1st

downstream API is offering its median SLA of 100ms, it means MOST of the time we would get the responses within that time. So, setting a timeout of 100ms would be ideal for MOST calls as it leaves enough room of 400ms for remaining 2

downstream API calls.

QUESTION 3

An API has been updated in Anypoint exchange by its API producer from version 3.1.1 to 3.2.0 following accepted semantic versioning practices and the changes have been communicated via the APIs public portal. The API endpoint does NOT change in the new version. How should the developer of an API client respond to this change?

A. The API producer should be requested to run the old version in parallel with the new one

B. The API producer should be contacted to understand the change to existing functionality

C. The API client code only needs to be changed if it needs to take advantage of the new features

D. The API clients need to update the code on their side and need to do full regression

Correct Answer: C

QUESTION 4

What is a key requirement when using an external Identity Provider for Client Management in Anypoint Platform?

A. Single sign-on is required to sign in to Anypoint Platform

B. The application network must include System APIs that interact with the Identity Provider

C. To invoke OAuth 2.0-protected APIs managed by Anypoint Platform, API clients must submit access tokens issued by that same Identity Provider

D. APIs managed by Anypoint Platform must be protected by SAML 2.0 policies

Correct Answer: C

https://www.folkstalk.com/2019/11/mulesoft-integration-and-platform.html To invoke OAuth 2.0-protected APIs managed by Anypoint Platform, API clients must submit access tokens issued by that same Identity Provider

****

>> It is NOT necessary that single sign-on is required to sign in to Anypoint Platform because we are using an external Identity Provider for Client Management >> It is NOT necessary that all APIs managed by Anypoint Platform must be

protected by SAML 2.0 policies because we are using an external Identity Provider for Client Management

>> Not TRUE that the application network must include System APIs that interact with the Identity Provider because we are using an external Identity Provider for Client Management Only TRUE statement in the given options is - "To invoke

OAuth 2.0-protected APIs managed by Anypoint Platform, API clients must submit access tokens issued by that same Identity Provider"

References:

https://docs.mulesoft.com/api-manager/2.x/external-oauth-2.0-token-validation-policy https://blogs.mulesoft.com/dev/api-dev/api-security-ways-to-authenticate-and-authorize/

QUESTION 5

A Mule application exposes an HTTPS endpoint and is deployed to three CloudHub workers that do not use static IP addresses. The Mule application expects a high volume of client requests in short time periods. What is the most cost-effective infrastructure component that should be used to serve the high volume of client requests?

A. A customer-hosted load balancer

B. The CloudHub shared load balancer

C. An API proxy

D. Runtime Manager autoscaling

Correct Answer: B

The CloudHub shared load balancer

****

The scenario in this question can be split as below:

>> There are 3 CloudHub workers (So, there are already good number of workers to handle high volume of requests)

>> The workers are not using static IP addresses (So, one CANNOT use customer load- balancing solutions without static IPs)

>> Looking for most cost-effective component to load balance the client requests among the workers.

Based on the above details given in the scenario:

>> Runtime autoscaling is NOT at all cost-effective as it incurs extra cost. Most over, there are already 3 workers running which is a good number. >> We cannot go for a customer-hosted load balancer as it is also NOT most cost-effective

(needs custom load balancer to maintain and licensing) and same time the Mule App is not having Static IP Addresses which limits from going with custom load balancing. >> An API Proxy is irrelevant there as it has no role to play w.r.t

handling high volumes or load balancing.

So, the only right option to go with and fits the purpose of scenario being most cost- effective is - using a CloudHub Shared Load Balancer.

Latest MCPA-LEVEL-1-MAINTENANCE Dumps
MCPA-LEVEL-1-MAINTENANCE PDF Dumps
MCPA-LEVEL-1-MAINTENANCE Exam Questions