



MCD-LEVEL-2^{Q&As}

MuleSoft Certified Developer - Level 2 (Mule 4)





Pass Mulesoft MCD-LEVEL-2 Exam with 100% Guarantee

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

<https://www.geekcert.com/mcd-level-2.html>

100% Passing Guarantee
100% Money Back Assurance

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

-  **Instant Download** After Purchase
-  **100% Money Back** Guarantee
-  **365 Days** Free Update
-  **800,000+** Satisfied Customers





QUESTION 1

Which pattern can a web API use to notify its client of state changes as soon as they occur?

- A. HTTP Webhook
- B. Shared database trigger
- C. Schedule Event Publisher
- D. ETL data load

Correct Answer: A

A web API can use HTTP Webhook to notify its client of state changes as soon as they occur. A webhook is an HTTP callback that allows an API to send real-time notifications to another system or application when an event happens. The client registers a URL with the API where it wants to receive notifications, and then the API sends an HTTP request to that URL with information about the event. References:<https://docs.mulesoft.com/connectors/webhook/webhook-connector>

QUESTION 2

A developer has created the first version of an API designed for business partners to work commodity prices.

What should developer do to allow more than one major version of the same API to be exposed by the implementation?

- A. In Design Center, open the RAML and modify each operation to include the major version number
- B. In Anypoint Studio, generate scaffolding from the RAML, and then modify the in the generated flows to include a parameter to replace the version number
- C. In Design Center, open the RAML and modify baseUri to include a variable that indicates the version number
- D. In Anypoint Studio, generate scaffolding from the RAML, and then modify the flow names generated by APIKit to include a variable with the major version number

Correct Answer: C

To allow more than one major version of the same API to be exposed by the implementation, the developer should modify the baseUri property in the RAML file to include a variable that indicates the version number. The baseUri property defines the base URL of the API and can include variables that are replaced with actual values when mocking or deploying the API. By using a variable for the version number, the developer can expose different versions of the API using different base URLs and avoid conflicts or confusion. References: <https://docs.mulesoft.com/api-designer/design-modify-raml-specs#baseuri> <https://docs.mulesoft.com/api-manager/2.x/api-versioning>

QUESTION 3

Refer to the exhibit.



Project Settings
Create a Mule project in the workspace or in an external location.

Project Name:

Runtime
Mule Server 4.4.0 EE
Mule Server 4.3.0 EE
[Install Runtimes](#)

API Implementation
Add an API implementation to your project to automatically set up an APIkit router and create placeholder flows for each resource method

Import a published API | Import RAML from local file | Download RAML from Design Center

Start building API implementations by importing the specification here. [Learn more](#)

Name	Version
------	---------

When creating a new project, which API implementation allows for selecting the correct API version and scaffolding the flows from the API specification?

- A. Import a published API
- B. Generate a local RAML from anypoint Studio
- C. Download RAML from Design Center
- D. Import RAML from local file

Correct Answer: A

To create a new project that selects the correct API version and scaffolds the flows from the API specification, the developer should import a published API. This option allows importing an API specification that has been published to Anypoint Exchange or Design Center, and selecting a specific version of that API specification. The developer can also choose to scaffold flows based on that API specification. References:<https://docs.mulesoft.com/apikit/4.x/apikit-4-new-project-task>

QUESTION 4

Two APIs are deployed to a two-node on-prem cluster. Due to a requirements change, the two APIs must communicate to exchange data asynchronously.

- A. If the two APIs use the same domain, the VM Connector can be leveraged
- B. The VM Connector is used to inter-application communication, so it is not possible to use the VM Connector
- C. Instead of using the VM Connector use directly
- D. It is not possible to use the VM Connector since the APIs are running in a cluster mode and each mode has its own



set of VM Queues

Correct Answer: A

To communicate asynchronously between two APIs deployed to a two-node on-prem cluster, the developer can use the VM Connector if the two APIs use the same domain. The VM Connector allows passing messages between different Mule applications within a single Mule runtime instance or across different instances using shared memory or persistent storage. If two APIs are deployed under the same domain, they can share resources such as VM queues and communicate asynchronously using VM Connector operations.

References:<https://docs.mulesoft.com/mule-runtime/4.3/vm-connector> <https://docs.mulesoft.com/mule-runtime/4.3/shared-resources>

QUESTION 5

A developer is working on a project that requires encrypting all data before sending it to a backend application. To accomplish this, the developer will use PGP encryption in the Mule 4 Cryptography module.

What is required to encrypt the data before sending it to the backend application?

- A. The application needs to configure HTTPS TLS context information to encrypt the data
- B. The application needs to both the private and public keys to encrypt the data
- C. The application needs the public key from the backend service to encrypt the data
- D. The application needs the private key from the backend service to encrypt the data

Correct Answer: C

To encrypt the data before sending it to the backend application using PGP encryption, the application needs the public key from the backend service. PGP encryption uses a public-key cryptography system, which means that each party has a pair of keys: a public key and a private key. The public key is used to encrypt data, and the private key is used to decrypt data. Therefore, to encrypt data for a specific recipient (the backend service), the application needs to use the recipient's public key. The recipient can then use its own private key to decrypt the data.

References:<https://docs.mulesoft.com/mule-runtime/4.3/cryptography-pgp>

[MCD-LEVEL-2 PDF Dumps](#) [MCD-LEVEL-2 VCE Dumps](#)

[MCD-LEVEL-2 Exam Questions](#)