MCIA-LEVEL-1 Q&As

MuleSoft Certified Integration Architect - Level 1

Pass Mulesoft MCIA-LEVEL-1 Exam with 100% Guarantee

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

https://www.passapply.com/mcia-level-1.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

What is true about the network connections when a Mule application uses a JMS connector to interact with a JMS provider (message broker)?

A. To complete sending a JMS message, the JMS connector must establish a network connection with the JMS message recipient

B. To receive messages into the Mule application, the JMS provider initiates a network connection to the JMS connector and pushes messages along this connection

C. The JMS connector supports both sending and receiving of JMS messages over the protocol determined by the JMS provider

D. The AMQP protocol can be used by the JMS connector to portably establish connections to various types of JMS providers

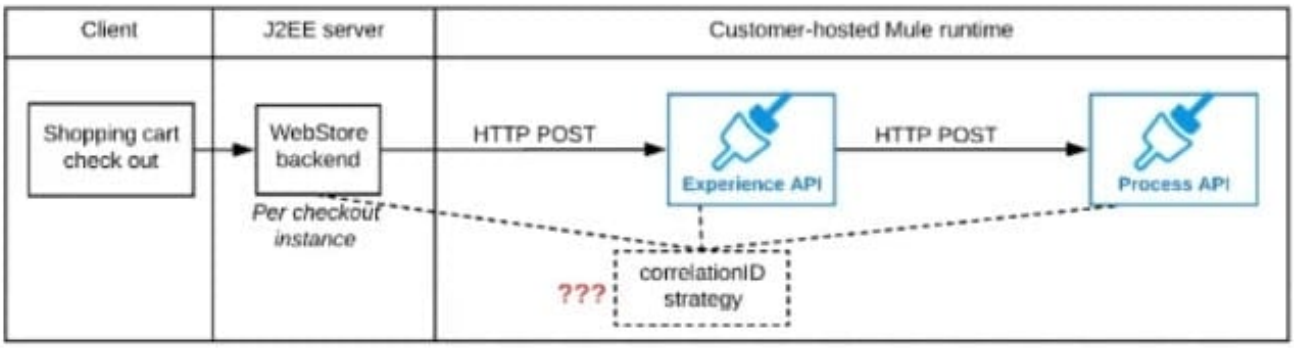
Correct Answer: C

*

To send message or receive JMS (Java Message Service) message no separate network connection need to be established. So option A, C and D are ruled out.

QUESTION 2

Refer to the exhibit.



A shopping cart checkout process consists of a web store backend sending a sequence of API invocations to an Experience API, which in turn invokes a Process API. All API invocations are over HTTPS POST. The Java web store backend executes in a Java EE application server, while all API implementations are Mule applications executing in a customer -hosted Mule runtime.

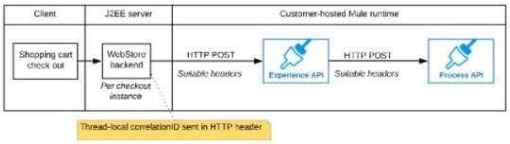
End-to-end correlation of all HTTP requests and responses belonging to each individual checkout Instance is required. This is to be done through a common correlation ID, so that all log entries written by the web store backend, Experience API implementation, and Process API implementation include the same correlation ID for all requests and responses belonging to the same checkout instance.

What is the most efficient way (using the least amount of custom coding or configuration) for the web store backend and the implementations of the Experience API and Process API to participate in end-to-end correlation of the API

invocations for each checkout instance?

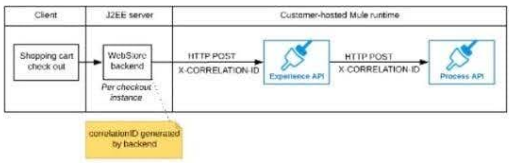
A. The web store backend, being a Java EE application, automatically makes use of the thread-local correlation ID generated by the Java EE application server and automatically transmits that to the Experience API using HTTP-standard headers

No special code or configuration is included in the web store backend, Experience API, and Process API implementations to generate and manage the correlation ID



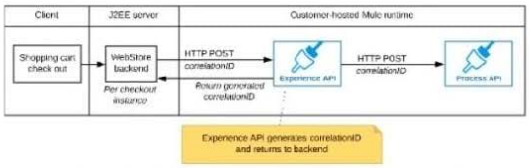
B. The web store backend generates a new correlation ID value at the start of checkout and sets it on the X-CORRELATION-It HTTP request header In each API invocation belonging to that checkout

No special code or configuration is included in the Experience API and Process API implementations to generate and manage the correlation ID



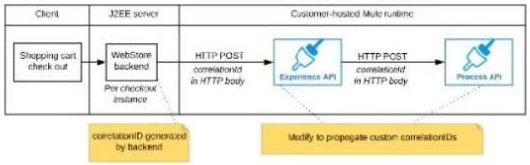
C. The Experience API implementation generates a correlation ID for each incoming HTTP request and passes it to the web store backend in the HTTP response, which includes it in all subsequent API invocations to the Experience API.

The Experience API implementation must be coded to also propagate the correlation ID to the Process API in a suitable HTTP request header



D. The web store backend sends a correlation ID value in the HTTP request body In the way required by the Experience API

The Experience API and Process API implementations must be coded to receive the custom correlation ID In the HTTP requests and propagate It in suitable HTTP request headers



A. Option A

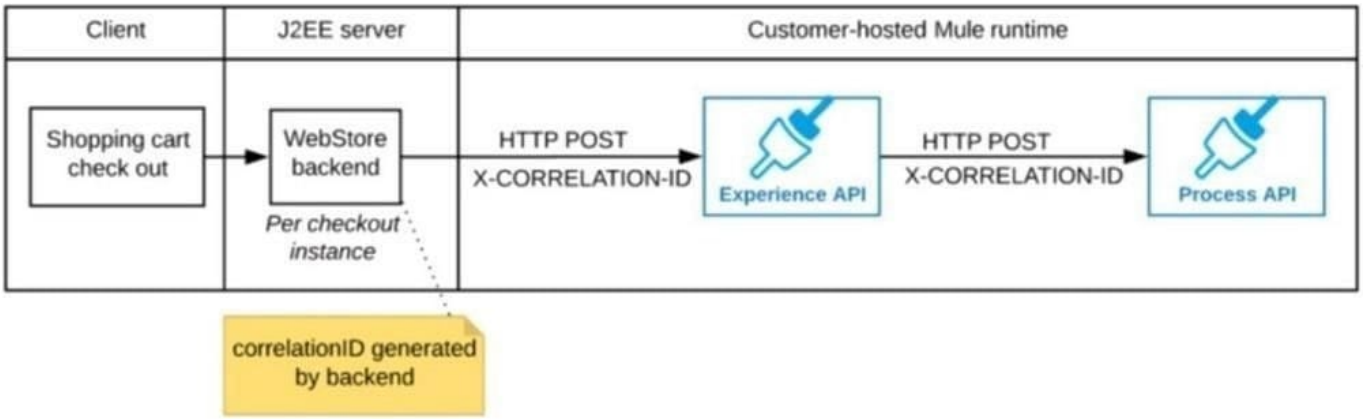
B. Option B

C. Option C

D. Option D

Correct Answer: B

Explanation: Correct answer is "The web store backend generates a new correlation ID value at the start of checkout and sets it on the X?ORRELATION-ID HTTP request header in each API invocation belonging to that checkout No special code or configuration is included in the Experience API and Process API implementations to generate and manage the correlation ID" : By design, Correlation Ids cannot be changed within a flow in Mule 4 applications and can be set only at source. This ID is part of the Event Context and is generated as soon as the message is received by the application. When a HTTP Request is received, the request is inspected for "X-Correlation-Id" header. If "X-Correlation-Id" header is present, HTTP connector uses this as the Correlation Id. If "X-Correlation-Id" header is NOT present, a Correlation Id is randomly generated. For Incoming HTTP Requests: In order to set a custom Correlation Id, the client invoking the HTTP request must set "X-Correlation-Id" header. This will ensure that the Mule Flow uses this Correlation Id. For Outgoing HTTP Requests: You can also propagate the existing Correlation Id to downstream APIs. By default, all outgoing HTTP Requests send "X- Correlation-Id" header. However, you can choose to set a different value to "X-Correlation- Id" header or set "Send Correlation Id" to NEVER. Mulesoft Reference: https://help.mulesoft.com/s/article/How-to-Set-CustomCorrelation-Id- for-Flows-with-HTTP-Endpoint-in-Mule-4



QUESTION 3

A company is designing a mule application to consume batch data from a partner\\'s ftps server The data files have been compressed and then digitally signed using PGP.

What inputs are required for the application to securely consumed these files?

A. ATLS context Key Store requiring the private key and certificate for the company PGP public key of partner PGP private key for the company

B. ATLS context first store containing a public certificate for partner ftps server and the PGP public key of the partner TLS contact Key Store containing the FTP credentials

C. TLS context trust or containing a public certificate for the ftps server The FTP username and password The PGP public key of the partner

D. The PGP public key of the partner The PGP private key for the company The FTP username and password

Correct Answer: D

QUESTION 4

What are two reasons why a typical MuleSoft customer favors a MuleSoft-hosted Anypoint Platform runtime plane over a customer-hosted runtime for its Mule application deployments? (Choose two.)

A. Reduced application latency

B. Increased application isolation

C. Reduced time-to-market for the first application

D. Increased application throughput

E. Reduced IT operations effort

Correct Answer: CE

QUESTION 5

An organization has defined a common object model in Java to mediate the communication between different Mule applications in a consistent way. A Mule application is being built to use this common object model to process responses from

a SOAP API and a REST API and then write the processed results to an order management system.

The developers want Anypoint Studio to utilize these common objects to assist in creating mappings for various transformation steps in the Mule application.

What is the most idiomatic (used for its intended purpose) and performant way to utilize these common objects to map between the inbound and outbound systems in the Mule application?

A. Use JAXB (XML) and Jackson (JSON) data bindings

B. Use the WSS module

C. Use the Java module

D. Use the Transform Message component

Correct Answer: A

Reference: https://docs.mulesoft.com/mule-runtime/3.9/understanding-mule-configuration