Exam : **MCIA-Level-1**

Title : MuleSoft Certified Integration Architect - Level 1

Vendor : MuleSoft

Version : DEMO

NO.1 An Order microservice and a Fulfillment microservice are being designed to communicate with their dients through message-based integration (and NOT through API invocations).
The Order microservice publishes an Order message (a kind of command message) containing the details of an order to be fulfilled. The intention is that Order messages are only consumed by one Mute application, the Fulfillment microservice.
The Fulfilment microservice consumes Order messages, fulfills the order described therein, and then publishes an OrderFulfilted message (a kind of event message). Each OrderFulfilted message can be consumed by any interested Mule application, and the Order microservice is one such Mute application.
What is the most appropriate choice of message broker(s) and message destination(s) in this scenario?

A. Order messages are sent to an Anypoint MQ exchange OrderFulfilled messages are sent to an Anypoint MQ queue Both microservices interact with Anypoint MQ as the message broker, which must therefore scale to support the load of both microservices

B. Order messages are sent to a JMS queue. OrderFulfilled messages are sent to a JMS topic Both microservices interact with the same JMS provider (message broker) instance, which must therefore scale to support the load of both microservices

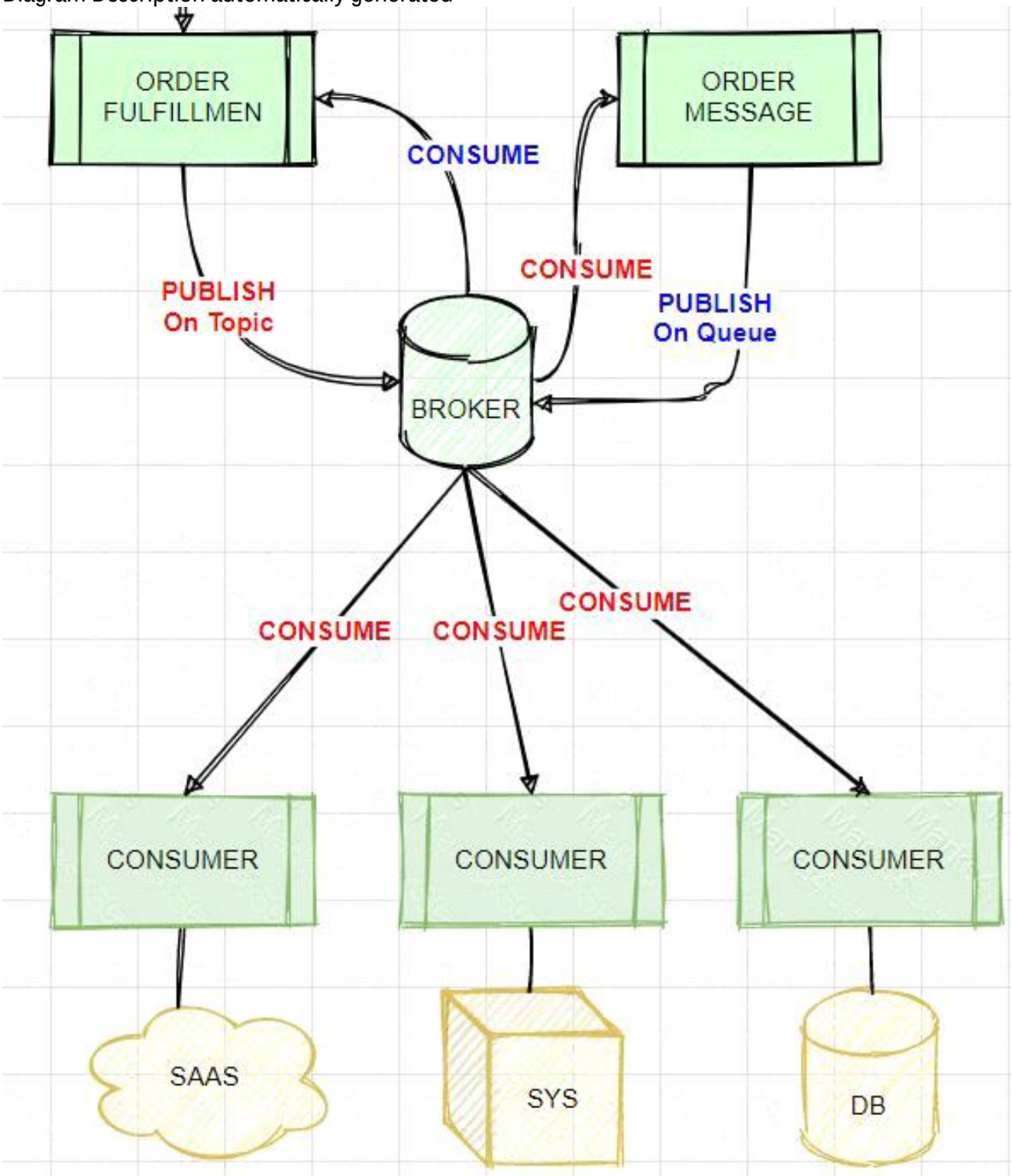
C. Order messages are sent directly to the Fulfillment microservices. OrderFulfilled messages are sent directly to the Order microservice The Order microservice interacts with one AMQP-compatible message broker and the Fulfillment microservice interacts with a different AMQP-compatible message broker, so that both message brokers can be chosen and scaled to best support the load of each microservice

D. Order messages are sent to a JMS queue. OrderFulfilled messages are sent to a JMS topic The Order microservice interacts with one JMS provider (message broker) and the Fulfillment microservice interacts with a different JMS provider, so that both message brokers can be chosen and scaled to best support the load of each microservice

Answer: B

Explanation:
* If you need to scale a JMS provider/ message broker, - add nodes to scale it horizontally or - ad d memory to scale it vertically * Cons of adding another JMS provider/ message broker: - adds cost. - adds complexity to use two JMS brokers - adds Operational overhead if we use two brokers, say , ActiveMQ and IBM MQ * So Two options that mention to use two brokers are not best choice. * It's mentioned that "The Fulfillment microservice consumes Order messages, fulfills the order described therein, and then publishes an OrderFulfilled message. Each OrderFulfilled message can be consumed by any interested Mule application." - When you publish a message on a topic, it goes to all the subscribers who are interested - so zero to many subscribers will receive a copy of the message. - When you send a message on a queue, it will be received by exactly one consumer. * As we need multiple consumers to consume the message below option is not valid choice: "Order messages are sent to an Anypoint MQ exchange. OrderFulfilled messages are sent to an Anypoint MQ queue. Both microservices interact with Anypoint MQ as the message broker, which must therefore scale to support the load of both microservices" * Order messages are only consumed by one Mule application, the Fulfillment microservice, so we will publish it on queue and OrderFulfilled message can be consumed by any interested Mule application so it need to be published on Topic using same broker. * Correct answer: Best choice in this scenario is: "Order messages are sent to a JMS queue. OrderFulfilled messages are sent to a JMS topic. Both microservices interact with the same JMS

provider (message broker) instance, which must therefore scale to support the load of both microservices" Tried to depict scenario in diagram:
Diagram Description automatically generated



NO.2 An organization has decided on a cloudhub migration strategy that aims to minimize the organizations own IT resources. Currently, the organizational has all of its Mule applications running on its own premises and uses an premises load balancer that exposes all APIs under the base URL https://api.acme.com As part of the migration strategy, the organization plans to migrate all of its Mule applications and load balancer to cloudhub What is the most straight-forward and cost effective

approach to the Mule applications deployment and load balancing that preserves the public URLs?

A. Deploy the Mule applications to Cloudhub
Update the CNAME record for an api.acme.com in the organizations DNS server pointing to the A record of a cloudhub dedicated load balancer(DLB) Apply mapping rules in the DLB to map URLs to their corresponding Mule applications

B. For each migrated Mule application, deploy an API proxy Mule application to Cloudhub with all applications under the control of a dedicated load balancer(CLB) Update the CNAME record for api.acme.com in the organization DNS server pointing to the A record of a cloudhub dedicated load balancer(DLB) Apply mapping rules in the DLB to map each API proxy application to its corresponding Mule applications

C. Deploy the Mule applications to Cloudhub
Create CNAME record for api.acme.com in the Cloudhub Shared load balancer (SLB) pointing to the A record of the on-premise load balancer Apply mapping rules in the SLB to map URLs to their corresponding Mule applications

D. Deploy the Mule applications to Cloudhub Update the CNAME record for api.acme.com in the organization DNS server pointing to the A record of the cloudhub shared load balancer(SLB) Apply mapping rules in the SLB to map URLs to their corresponding Mule applications.

Answer: A
Explanation:
https://help.mulesoft.com/s/feed/0D52T000055pzgsSAA.

NO.3 An organization has an HTTPS-enabled Mule application named Orders API that receives requests from another Mule application named Process Orders.
The communication between these two Mule applications must be secured by TLS mutual authentication (two-way TLS).
At a minimum, what must be stored in each truststore and keystore of these two Mule applications to properly support two-way TLS between the two Mule applications while properly protecting each Mule application's keys?

A. Orders API truststore: The Orders API public key
Process Orders keystore: The Process Orders private key and public key

B. Orders API truststore: The Orders API private key and public key
Process Orders keystore: The Process Orders private key public key

C. Orders API truststore: The Process Orders public key
Orders API keystore: The Orders API private key and public key
Process Orders truststore: The Orders API public key
Process Orders keystore: The Process Orders private key and public key

D. Orders API truststore: The Process Orders public key
Orders API keystore: The Orders API private key
Process Orders truststore: The Orders API public key
Process Orders keystore: The Process Orders private key

Answer: C

NO.4 What requirement prevents using Anypoint MQ as the messaging broker for a Mule application?

A. When the payload sent through the message broker must use XML format

B. When the payload sent through the message broker must be encrypted
C. When the messaging broker must support point-to-point messaging
D. When the messaging broker must be deployed on-premises
Answer: D
Explanation:
Anypoint MQ is a cloud-based messaging service provided by MuleSoft, and it cannot be deployed on-premises. If there is a requirement for the messaging broker to be deployed on-premises, Anypoint MQ would not be suitable. Other considerations such as payload format (XML), encryption, and point-to-point messaging are supported by Anypoint MQ, but the deployment environment requirement is a critical constraint.
References
* MuleSoft Anypoint MQ Documentation
* Cloud vs. On-Premises Messaging Solutions

NO.5 A manufacturing company is planning to deploy Mule applications to its own Azure Kubernetes Service infrastructure.
The organization wants to make the Mule applications more available and robust by deploying each Mule application to an isolated Mule runtime in a Docker container while managing all the Mule applications from the MuleSoft-hosted control plane.
What is the most idiomatic (used for its intended purpose) choice of runtime plane to meet these organizational requirements?
A. Anypoint Platform Private Cloud Edition
B. Anypoint Runtime Fabric
C. CloudHub
D. Anypoint Service Mesh
Answer: B

NO.6 A Mule application is running on a customer-hosted Mule runtime in an organization's network. The Mule application acts as a producer of asynchronous Mule events. Each Mule event must be broadcast to all interested external consumers outside the Mule application. The Mule events should be published in a way that is guaranteed in normal situations and also minimizes duplicate delivery in less frequent failure scenarios.
The organizational firewall is configured to only allow outbound traffic on ports 80 and 443. Some external event consumers are within the organizational network, while others are located outside the firewall.
What Anypoint Platform service is most idiomatic (used for its intended purpose) for publishing these Mule events to all external consumers while addressing the desired reliability goals?
A. CloudHub VM queues
B. Anypoint MQ
C. Anypoint Exchange
D. CloudHub Shared Load Balancer
Answer: B
Explanation:
Set the Anypoint MQ connector operation to publish or consume messages, or to accept (ACK) or not accept (NACK) a message.

NO.7 According to MuleSoft, what Action should an IT organization take regarding its technology assets in order to close the IT delivery.

A. Make assets easily discoverable via a central repository

B. Focus project delivery efforts on custom assets that meet the specific requirements of each individual line of business

C. Create weekly meetings that all members of IT attend to present justification and request approval to use existing assets

D. Hire additional staff to meet the demand for asset creation required for approved projects and timelines

Answer: A

Explanation:

To close the IT delivery gap, MuleSoft recommends that IT organizations make their technology assets easily discoverable via a central repository. This approach ensures that existing assets, such as APIs, connectors, templates, and other reusable components, are accessible to all teams within the organization. By having a central repository, teams can quickly find and leverage these assets, reducing duplication of effort, speeding up project delivery, and fostering a culture of reuse and collaboration. This strategy aligns with the principles of an API-led connectivity approach, where reusable assets and APIs form the foundation for scalable and efficient IT delivery.

References

* MuleSoft Documentation on API-led Connectivity
* Anypoint Platform Best Practices for Asset Management

NO.8 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

Answer: A

NO.9 A mule application uses an HTTP request operation to involve an external API.

The external API follows the HTTP specification for proper status code usage.

What is possible cause when a 3xx status code is returned to the HTTP Request operation from the external API?

A. The request was not accepted by the external API

B. The request was Redirected to a different URL by the external API

C. The request was NOT RECEIVED by the external API

D. The request was ACCEPTED by the external API

Answer: B
Explanation:
3xx HTTP status codes indicate a redirection that the user agent (a web browser or a crawler) needs to take further action when trying to access a particular resource.

NO.10 What limits if a particular Anypoint Platform user can discover an asset in Anypoint Exchange?
A. Design Center and RAML were both used to create the asset
B. The existence of a public Anypoint Exchange portal to which the asset has been published
C. The type of the asset in Anypoint Exchange
D. The business groups to which the user belongs
Answer: D
Explanation:
* "The existence of a public Anypoint Exchange portal to which the asset has been published" - question does not mention anything about the public portal. Beside the public portal is open to the internet, to anyone. * If you cannot find an asset in the current business group scopes, search in other scopes. In the left navigation bar click All assets (assets provided by MuleSoft and your own master organization), Provided by MuleSoft, or a business group scope. User belonging to one Business Group can see assets related to his group only Reference:
https://docs.mulesoft.com/exchange/to-find-info https://docs.mulesoft.com/exchange/asset-details
Correct answer is The business groups to which the user belongs

NO.11 An organization is using Mulesoft cloudhub and develops API's in the latest version. As a part of requirements for one of the API's, third party API needs to be called. The security team has made it clear that calling any external API needs to have include listing As an integration architect please suggest the best way to accomplish the design plan to support these requirements?
A. Implement includelist IP on the cloudhub VPC firewall to allow the traffic
B. Implement the validation of includelisted IP operation
C. Implement the Any point filter processor to implement the include list IP
D. Implement a proxy for the third party API and enforce the IPinclude list policy and call this proxy from the flow of the API
Answer: D
Explanation:
* Requirement Analysis: The security team requires any external API call to be restricted by an IP include list. This ensures that only specified IP addresses can access the third-party API.
* Design Plan: To fulfill this requirement, implementing a proxy for the third-party API is the best approach. This proxy can enforce the IP include list policy.
* Implementation Steps:
* Create a Proxy API: Set up a new API proxy in Anypoint Platform to act as an intermediary for the third-party API.
* Configure IP Include List Policy: Within the Anypoint API Manager, apply the IP whitelist policy to the proxy API. This policy will ensure that only requests from specified IP addresses are allowed to reach the third-party API.
* Modify Main API Flow: Update the flow of your main API to call the newly created proxy API instead of directly calling the third-party API.

* Testing: Conduct thorough testing to ensure that the proxy API correctly forwards requests to the third-party API only if the requests originate from IPs in the include list.
* Advantages:
* Security: This method ensures that only approved IPs can access the third-party API, adhering to the security team's requirements.
* Manageability: Centralizes the IP restriction management within the API Manager, simplifying maintenance and updates.
References
* MuleSoft Documentation on API Proxies
* MuleSoft Documentation on IP Whitelist Policy

NO.12 An external web UI application currently accepts occasional HTTP requests from client web browsers to change (insert, update, or delete) inventory pricing information in an inventory system's database. Each inventory pricing change must be transformed and then synchronized with multiple customer experience systems in near real-time (in under 10 seconds). New customer experience systems are expected to be added in the future.
The database is used heavily and limits the number of SELECT queries that can be made to the database to 10 requests per hour per user.
What is the most scalable, idiomatic (used for its intended purpose), decoupled. reusable, and maintainable integration mechanism available to synchronize each inventory pricing change with the various customer experience systems in near real-time?
A. Write a Mule application with a Database On Table Row event source configured for the inventory pricing database, with the watermark attribute set to an appropriate database column In the same now, use a Scatter-Gather to call each customer experience system's REST API with transformed inventory-pricing records
B. Add a trigger to the inventory-pricing database table so that for each change to the inventory pricing database, a stored procedure is called that makes a REST call to a Mule application Write the Mule application to publish each Mule event as a message to an Anypoint MQ exchange Write other Mule applications to subscribe to the Anypoint MQ exchange, transform each received message, and then update the Mule application's corresponding customer experience system(s)
C. Replace the external web UI application with a Mule application to accept HTTP requests from client web browsers In the same Mule application, use a Batch Job scope to test if the database request will succeed, aggregate pricing changes within a short time window, and then update both the inventory pricing database and each customer experience system using a Parallel For Each scope
D. Write a Mule application with a Database On Table Row event source configured for the inventory pricing database, with the ID attribute set to an appropriate database column In the same flow, use a Batch Job scope to publish transformed Inventory-pricing records to an Anypoint MQ queue Write other Mule applications to subscribe to the Anypoint MQ queue, transform each received message, and then update the Mule application's corresponding customer experience system(s)
Answer: B

NO.13 An organization is in the process of building automated deployments using a CI/CD process. As a part of automated deployments, it wants to apply policies to API Instances.
What tool can the organization use to promote and deploy API Manager policies?
A. Anypoint CLI

B. MUnit Maven plugin

C. Mule Maven plugin

D. Runtime Manager agent

Answer: A

Explanation:

To automate the promotion and deployment of API Manager policies as part of a CI/CD process, an organization can use the Anypoint CLI (Command Line Interface). The Anypoint CLI provides a set of commands to interact with various components of the Anypoint Platform, including API Manager.

Using the Anypoint CLI, you can:

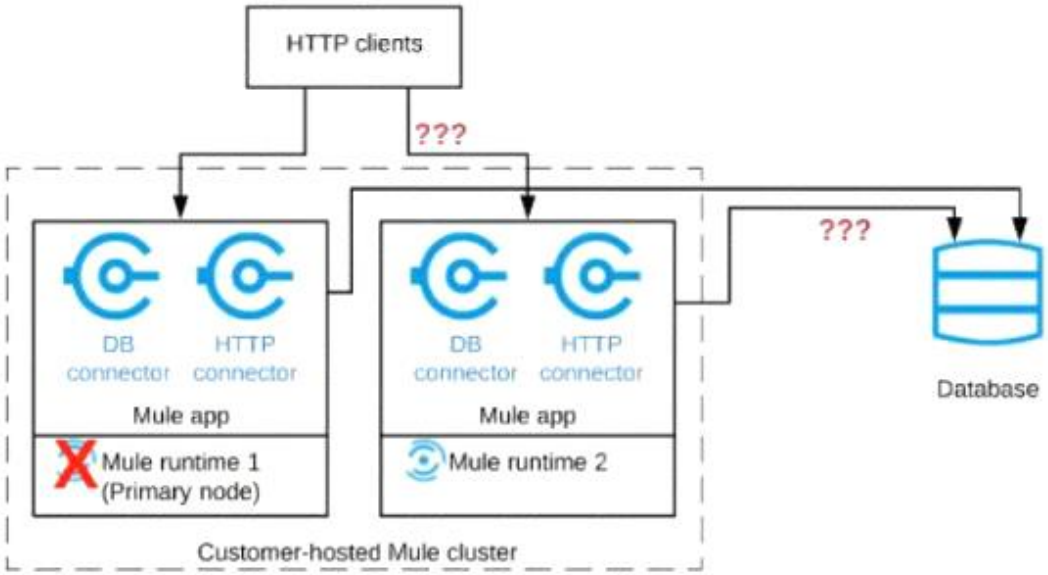
* Apply policies to API instances.
* Promote APIs and their associated policies across different environments.
* Script these actions as part of the CI/CD pipeline to ensure consistency and automation in managing API policies.

Other tools like the MUnit Maven plugin, Mule Maven plugin, and Runtime Manager agent do not provide direct capabilities to manage API Manager policies.

References

* MuleSoft Anypoint CLI Documentation
* Automating API Management with Anypoint CLI

NO.14 Refer to the exhibit.



A Mule application is deployed to a cluster of two customer-hosted Mute runtimes. The Mute application has a flow that polls a database and another flow with an HTTP Listener.

HTTP clients send HTTP requests directly to individual cluster nodes.

What happens to database polling and HTTP request handling in the time after the primary (master) node of the cluster has railed, but before that node is restarted?

A. Database polling continues Only HTTP requests sent to the remaining node continue to be accepted

B. Database polling stops All HTTP requests continue to be accepted

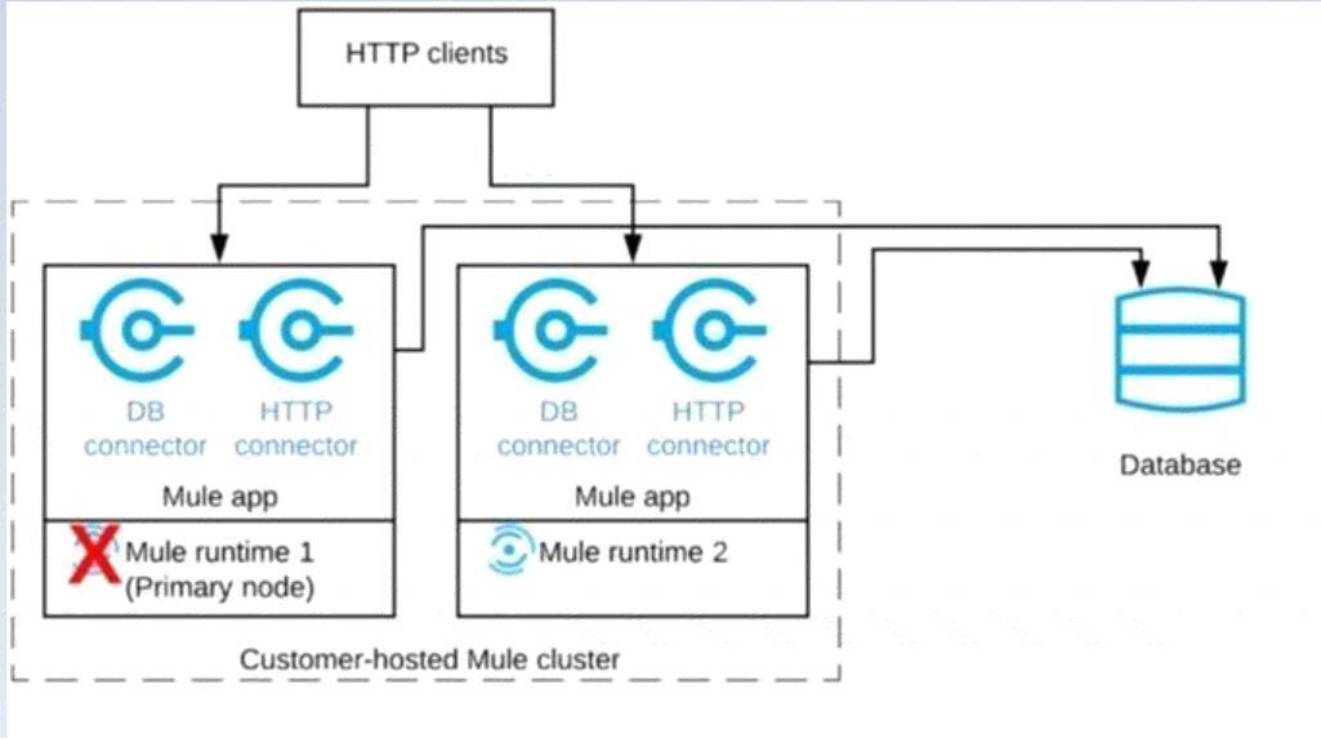
C. Database polling continues All HTTP requests continue to be accepted, but requests to the failed node Incur increased latency

D. Database polling stops All HTTP requests are rejected

Answer: A

Explanation:

Correct answer is Database polling continues Only HTTP requests sent to the remaining node continue to be accepted. Explanation : Architecture descripted in the question could be described as follows.When node 1 is down , DB polling will still continue via node 2 . Also requests which are coming directly to node 2 will also be accepted and processed in BAU fashion. Only thing that wont work is when requests are sent to Node 1 HTTP connector. The flaw with this architecture is HTTP clients are sending HTTP requests directly to individual cluster nodes. By default, clustering Mule runtime engines ensures high system availability. If a Mule runtime engine node becomes unavailable due to failure or planned downtime, another node in the cluster can assume the workload and continue to process existing events and messages Diagram Description automatically generated



NO.15 According to MuleSoft, which deployment characteristic applies to a microservices application architecture?

A. Services exist as independent deployment artifacts and can be scaled -independently of other services

B. All services of an application can be deployed together as single Java WAR file

C. A deployment to enhance one capability requires a redeployment of all capabilities

D. Core business capabilities are encapsulated in a single, deployable application

Answer: A

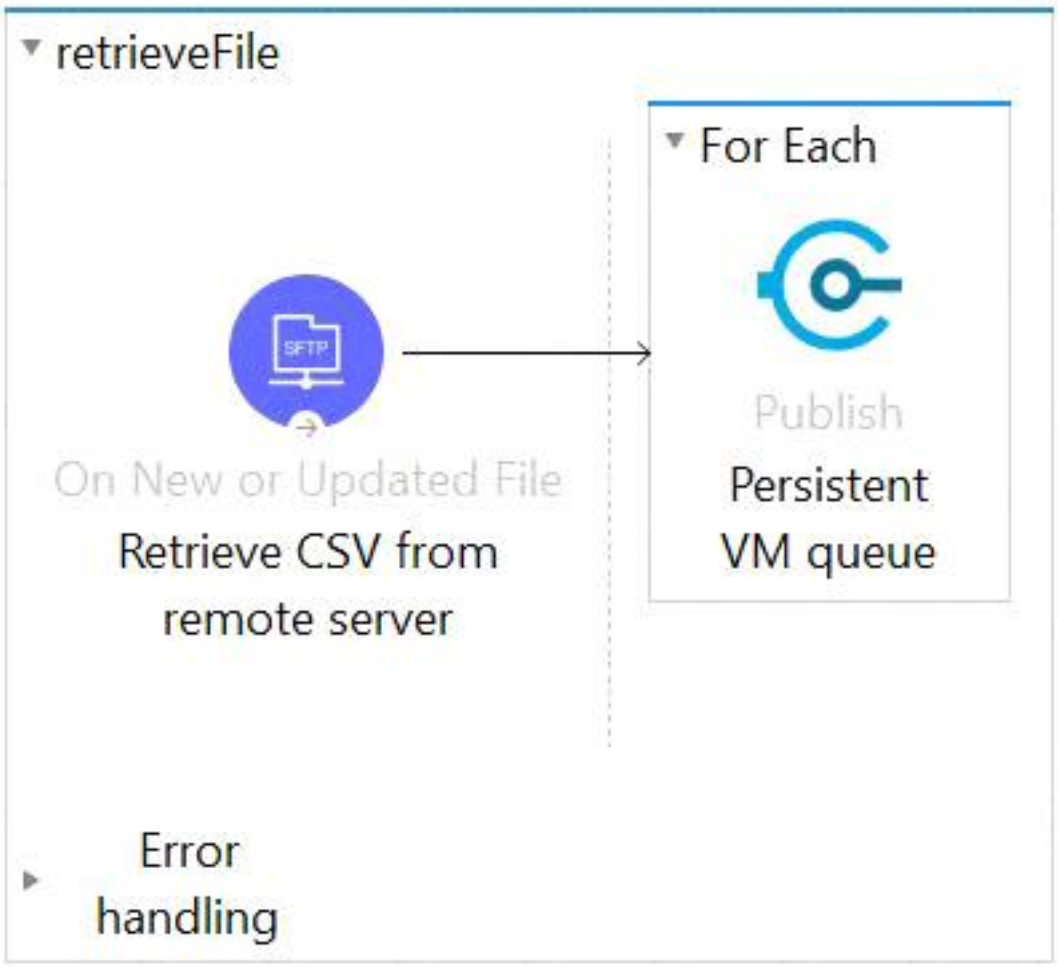
Explanation:

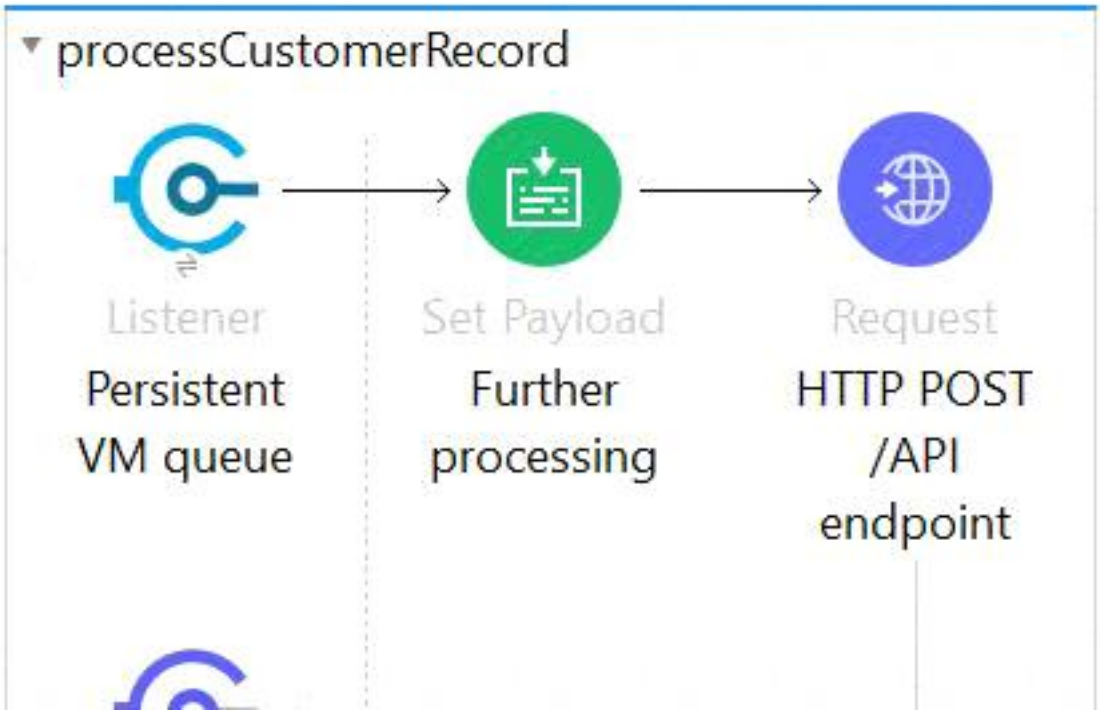
In a microservices application architecture, each service is designed to be an independent deployment artifact.

This means that services can be deployed, updated, and scaled independently of one another. This characteristic allows for greater flexibility and agility in managing applications, as individual services can be scaled up or down based on demand without impacting other services. It also enhances fault

isolation, as issues in one service do not necessarily affect the entire application.
This is in contrast to monolithic architectures, where all components are packaged and deployed together, often resulting in a single point of failure and difficulties in scaling and updating specific parts of the application.
References
* MuleSoft Documentation on Microservices Architecture
* Principles of Microservices Design

NO.16 Refer to the exhibit.





This Mule application is deployed to multiple Cloudhub workers with persistent queue enabled. The retrievefile flow event source reads a CSV file from a remote SFTP server and then publishes each record in the CSV file to a VM queue. The processCustomerRecords flow's VM Listner receives messages from the same VM queue and then processes each message separately.

How are messages routed to the cloudhub workers as messages are received by the VM Listener?
A. Each message is routed to ONE of the Cloudhub workers in a DETERMINSTIC round robin fashion thereby EXACTLY BALANCING messages among the cloudhub workers
B. Each messages routes to ONE of the available Clouhub workers in a NON- DETERMINSTIC non round-robin fashion thereby APPROXIMATELY BALANCING messages among the cloudhub workers
C. Each message is routed to the SAME Cloudhub worker that retrieved the file, thereby BINDING ALL messages to ONLY that ONE Cloudhub worker
D. Each message is duplicated to ALL of the Cloudhub workers, thereby SHARING EACH message with ALL the Cloudhub workers.
Answer: B

NO.17 Which type of communication is managed by a service mesh in a microservices architecture?
A. Communication between microservices runtime administrators
B. Communication between microservices developers
C. Communication between microservices
D. Communication between trading partner services
Answer: C
Explanation:
In a microservices architecture, a service mesh manages the communication between microservices. This involves handling service discovery, load balancing, failure recovery, metrics, and monitoring. Service meshes also provide more complex operational requirements like A/B testing, canary releases, rate limiting, access control, and end-to-end authentication. By abstracting these functionalities away from individual microservices, a service mesh allows developers to focus on business logic while ensuring reliable and secure inter-service communication.
References:
* Understanding Service Mesh
* Service Mesh for Microservices

NO.18 An organization is building out a test suite for their application using MUnit.
The Integration Architect has recommended using Test Recorder in Anypoint Studio to record the processing flows and then configure unit tests based on the captured events.
What Is a core consideration that must be kept In mind while using Test Recorder?
A. The Recorder supports loops where the structure of the data being tested changes inside the Iteration
B. Mocking values resulting from parallel processes are possible and will not affect theexecution of the processors that follow in the test
C. The Recorder supports mocking a message before or inside a Foreach processor
D. Tests for flows cannot be created if Mule errors are raised Inside the flows, even if the errors are handled by On-Error Continue error handlers
Answer: D
Explanation:
* MUnit and Test Recorder:
* MUnit is MuleSoft's testing framework designed to create automated tests for Mule applications.
* Test Recorder in Anypoint Studio allows capturing live processing flows and events to automatically

generate test cases.
* Core Consideration:
* While using Test Recorder, it is essential to consider how errors within the flows are handled.
* If Mule errors are raised within the flows, Test Recorder will not be able to create tests for these flows, even if the errors are managed by On-Error Continue error handlers.
* This limitation requires careful design and error handling strategies to ensure test cases can be recorded and executed effectively.
* Error Handling in MUnit:
* MUnit allows simulating error scenarios using mocks and spies.
* Proper error handling within the application flow ensures robustness and reliable test creation.
* However, the inability to create tests when errors are raised during recording indicates a need for alternative approaches to manage errors during test case generation.
References:
* MuleSoft Documentation on MUnit: MUnit Documentation
* MuleSoft Blog on MUnit Test Recorder: MUnit Test Recorder

NO.19 According to MuleSoft, which system integration term describes the method, format, and protocol used for communication between two system?
A. Component
B. interaction
C. Message
D. Interface
Answer: D
Explanation:
According to MuleSoft, the term "interface" describes the method, format, and protocol used for communication between two systems. An interface defines how systems interact, specifying the data formats (e.g., JSON, XML), protocols (e.g., HTTP, FTP), and methods (e.g., GET, POST) that are used to exchange information. Properly designed interfaces ensure compatibility and seamless communication between integrated systems.
References:
* MuleSoft Glossary of Integration Terms
* System Interfaces and APIs

NO.20 A mule application designed to fulfil two requirements
a) Processing files are synchronously from an FTPS server to a back-end database using VM intermediary queues for load balancing VM events b) Processing a medium rate of records from a source to a target system using batch job scope Considering the processing reliability requirements for FTPS files, how should VM queues be configured for processing files as well as for the batch job scope if the application is deployed to Cloudhub workers?
A. Use Cloud hub persistent queues for FTPS files processing
There is no need to configure VM queues for the batch jobs scope as it uses by default the worker' s disc for VM queueing
B. Use Cloud hub persistent VM queue for FTPS file processing
There is no need to configure VM queues for the batch jobs scope as it uses by default the worker' s JVM memory for VM queueing

C. Use Cloud hub persistent VM queues for FTPS file processing
Disable VM queue for the batch job scope
D. Use VM connector persistent queues for FTPS file processing Disable VM queue for the batch job scope
Answer: A
Explanation:
When processing files synchronously from an FTPS server to a back-end database using VM intermediary queues for load balancing VM events on CloudHub, reliability is critical. CloudHub persistent queues should be used for FTPS file processing to ensure that no data is lost in case of worker failure or restarts. These queues provide durability and reliability since they store messages persistently.
For the batch job scope, it is not necessary to configure additional VM queues. By default, batch jobs on CloudHub use the worker's disk for VM queueing, which is reliable for handling medium-rate records processing from a source to a target system. This approach ensures that both FTPS file processing and batch job processing meet reliability requirements without additional configuration for batch job scope.
References
* MuleSoft Documentation on CloudHub and VM Queues
* Anypoint Platform Best Practices

NO.21 When using Anypoint Platform across various lines of business with their own Anypoint Platform business groups, what configuration of Anypoint Platform is always performed at the organization level as opposed to at the business group level?
A. Environment setup
B. Identity management setup
C. Role and permission setup
D. Dedicated Load Balancer setup
Answer: B
Explanation:
* Roles are business group specific. Configure identity management in the Anypoint Platform master organization. As the Anypoint Platform organization administrator, you can configure identity management in Anypoint Platform to set up users for single sign-on (SSO). * Roles and permissions can be set up at business group and organization level also. But Identity Management setup is only done at Organization level * Business groups are self-contained resource groups that contain Anypoint Platform resources such as applications and APIs. Business groups provide a way to separate and control access to Anypoint Platform resources because users have access only to the busine

NO.22 How are the API implementation , API client, and API consumer combined to invoke and process an API ?
A. The API consumer creates an API implementation , which receives API invocations from an API such that they are processed for an API client
B. The API consumer creates an API client which sends API invocations to an API such that they are processed by an API implementation
C. An API client creates an API consumer, which receives API invocation from an API such that they

are processed for an API implementation

D. The API client creates an API consumer which sends API invocations to an API such that they are processed by API implementation

Answer: C

Explanation:

The API consumer creates an API client which sends API invocations to an API such that they are processed by an API implementation This is based on below definitions API client * An application component * that accesses a service * by invoking an API of that service - by definition of the term API over HTTP API consumer * A business role, which is often assigned to an individual * that develops API clients, i.e., performs the activities necessary for enabling an API client to invoke APIs API implementation * An application component * that implements the functionality

NO.23 A Mule application is synchronizing customer data between two different database systems. What is the main benefit of using eXtended Architecture (XA) transactions over local transactions to synchronize these two different database systems?

A. An XA transaction synchronizes the database systems with the least amount of Mule configuration or coding

B. An XA transaction handles the largest number of requests in the shortest time

C. An XA transaction automatically rolls back operations against both database systems if any operation falls

D. An XA transaction writes to both database systems as fast as possible

Answer: B

NO.24 An organization's governance process requires project teams to get formal approval from all key stakeholders for all new Integration design specifications. An integration Mule application Is being designed that interacts with various backend systems. The Mule application will be created using Anypoint Design Center or Anypoint Studio and will then be deployed to a customer-hosted runtime.

What key elements should be included in the integration design specification when requesting approval for this Mule application?

A. SLAs and non-functional requirements to access the backend systems

B. Snapshots of the Mule application's flows, including their error handling

C. A list of current and future consumers of the Mule application and their contact details

D. The credentials to access the backend systems and contact details for the administrator of each system

Answer: A

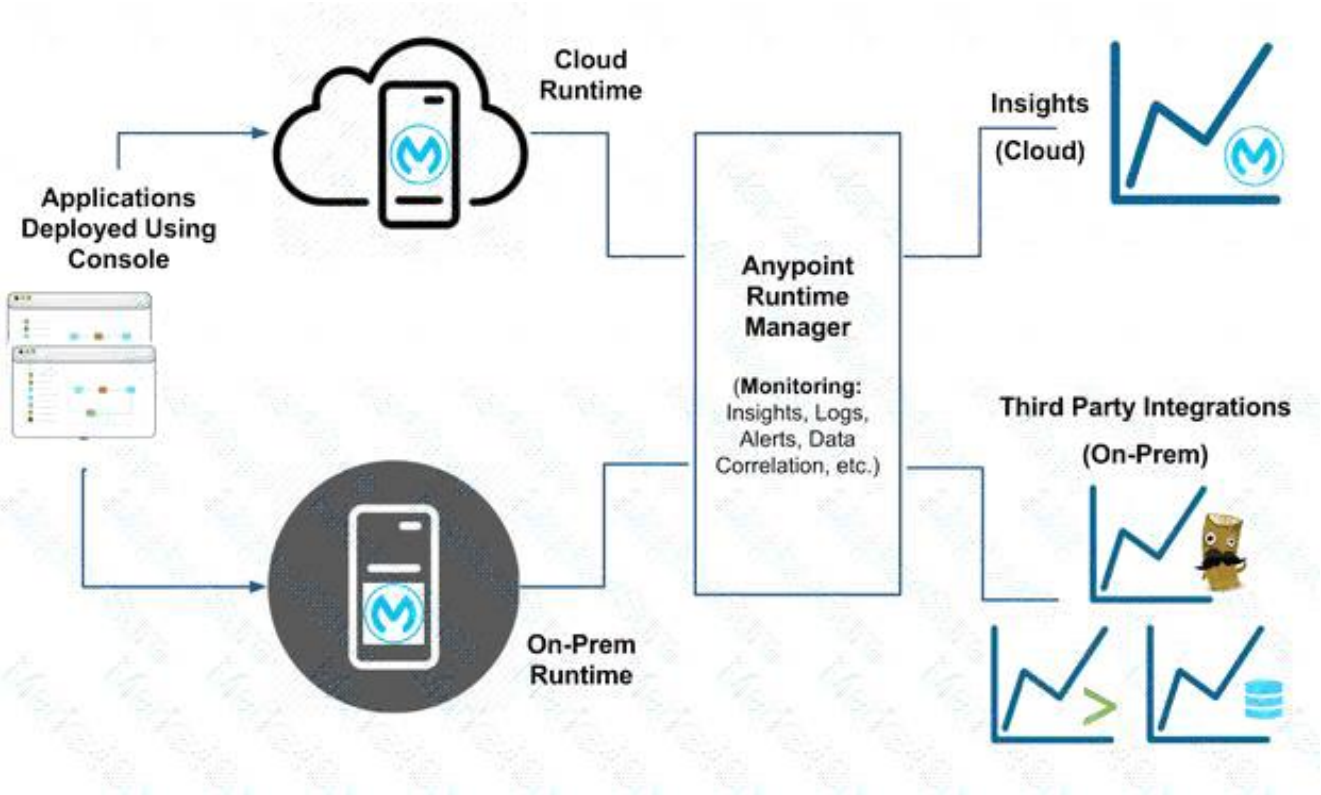
Explanation:

SLAs and non-functional requirements to access the backend systems. Only this option actually speaks to design parameters and reqs. * Below two are technical implementations and not the part of design: - Snapshots of the Mule application's flows, including their error handling - The credential s to access the backend systems and contact details for the administrator of each system * List of consumers is not relevant to the design

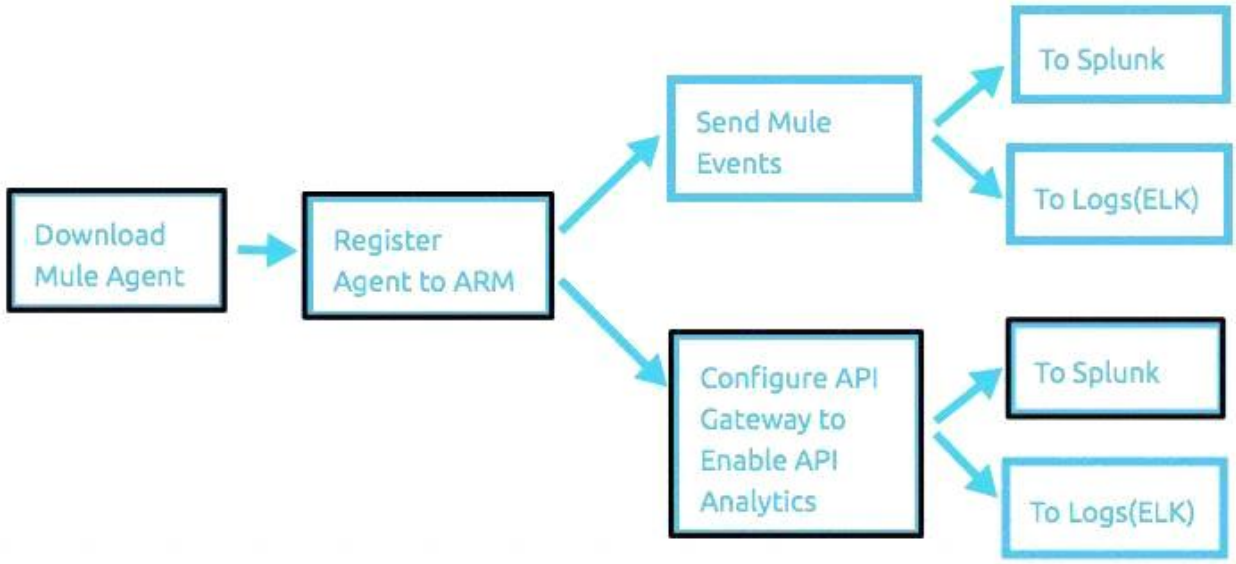
NO.25 In which order are the API Client, API Implementation, and API interface components called in a typical REST request?

A. API Client > API implementation > API Interface
B. API interface > API Client > API Implementation
C. API Client > API Interface > API implementation
D. API Implementation > API Interface > API Client
Answer: C
Explanation:
In a typical REST request, the order of interaction is:
* API Client: The client initiates the request to access data or functionality exposed by the API.
* API Interface: This represents the contract or the definition of the API, often specified in OpenAPI or RAML. It defines the endpoints, request/response formats, and other API details.
* API Implementation: This is the actual backend logic that processes the request and returns the response. It interacts with databases, other services, or performs business logic to fulfill the request.
References:
* Understanding REST APIs
* API Design and Implementation

NO.26 An organization uses Mule runtimes which are managed by Anypoint Platform - Private Cloud Edition. What MuleSoft component is responsible for feeding analytics data to non-MuleSoft analytics platforms?
A. Anypoint Exchange
B. The Mule runtimes
C. Anypoint API Manager
D. Anypoint Runtime Manager
Answer: D
Explanation:
*Explanation
Correct answer is Anypoint Runtime Manager
MuleSoft Anypoint Runtime Manager (ARM) provides connectivity to Mule Runtime engines deployed across your organization to provide centralized management, monitoring and analytics reporting. However, most enterprise customers find it necessary for these on-premises runtimes to integrate with their existing non MuleSoft analytics / monitoring systems such as Splunk and ELK to support a single pane of glass view across the infrastructure.
* You can configure the Runtime Manager agent to export data to external analytics tools.
Using either the Runtime Manager cloud console or Anypoint Platform Private Cloud Edition, you can :
--> Send Mule event notifications, including flow executions and exceptions, to Splunk or ELK.
--> Send API Analytics to Splunk or ELK. Sending data to third-party tools is not supported for applications deployed on CloudHub.
You can use the CloudHub custom log appender to integrate with your logging system. Reference:
https://docs.mulesoft.com/runtime-manager/
https://docs.mulesoft.com/release-notes/runtime-manager-agent/runtime-manager-agent-release-notes Diagram Description automatically generated



Additional Info:
It can be achieved in 3 steps:
1) register an agent to a runtime manager,
2) configure a gateway to enable API analytics to be sent to non MuleSoft analytics platform (Splunk for ex.)
- as highlighted in the following diagram and
3) setup dashboards.
Diagram Description automatically generated



NO.27 A company is planning to extend its Mule APIs to the Europe region. Currently all new applications are deployed to Cloudhub in the US region following this naming convention

{API name}-{environment}. for example, Orders-SAPI-dev, Orders-SAPI-prod etc.
Considering there is no network restriction to block communications between API's, what strategy should be implemented in order to apply the same new API's running in the EU region of CloudHub as well to minimize latency between API's and target users and systems in Europe?

A. Set region property to Europe (eu-de) in API manager for all the mule application No need to change the naming convention

B. Set region property to Europe (eu-de) in API manager for all the mule application Change the naming convention to {API name}-{environment}-{region} and communicate this change to the consuming applications and users

C. Set region property to Europe (eu-de) in runtime manager for all the mule application No need to change the naming convention

D. Set region property to Europe (eu-de) in runtime manager for all the mule application Change the naming convention to {API name}-{environment}-{region} and communicate this change to the consuming applications and users

Answer: D

Explanation:
To extend Mule APIs to the Europe region and ensure minimal latency between APIs and target users and systems in Europe, it is important to set the region property correctly and update the naming convention.
Here's the detailed approach:
* Setting the Region Property:
* Runtime Manager Configuration: In MuleSoft's Runtime Manager, you can deploy Mule applications to specific regions. For the European deployment, set the region property to Europe (eu-de) for all Mule applications. This ensures that the applications are physically hosted in the European data centers, reducing latency for European users.
* Go to Runtime Manager in Anypoint Platform.
* Select the application to deploy.
* Choose the Region setting and select Europe (eu-de).
* Updating Naming Convention:
* Naming Convention Update: Change the naming convention to include the region in the application names. The new convention should be {API name}-{environment}-{region} (e.g., Orders-SAPI-dev-eu, Orders-SAPI-prod-eu). This helps in distinguishing applications deployed in different regions.
* Update the naming convention during deployment.
* Ensure the new naming convention is documented and communicated to all stakeholders.
* Communicating the Change:
* Stakeholder Communication: Inform all consuming applications and users about the updated naming convention. This ensures that they update their references and configurations accordingly to avoid any disruptions in service.
* Documentation: Update any internal documentation to reflect the new naming convention and deployment regions.
* Benefits:
* Reduced Latency: By deploying the applications in the Europe region, the data proximity to European users will be improved, resulting in lower latency.
* Clear Identification: The updated naming convention helps in easily identifying the environment
* and region of deployment, making management and troubleshooting easier.

References:
* MuleSoft Documentation on Deploying to CloudHub
* MuleSoft Documentation on Managing Applications in Runtime Manager

NO.28 What best describes the Fully Qualified Domain Names (FQDNs), also known as DNS entries, created when a Mule application is deployed to the CloudHub Shared Worker Cloud?

A. A fixed number of FQDNs are created, IRRESPECTIVE of the environment and VPC design

B. The FQDNs are determined by the application name chosen, IRRESPECTIVE of the region

C. The FQDNs are determined by the application name, but can be modified by an administrator after deployment

D. The FQDNs are determined by both the application name and the region

Answer: D

Explanation:
Every Mule application deployed to CloudHub receives a DNS entry pointing to the CloudHub. The DNS entry is a CNAME for the CloudHub Shared Load Balancer in the region to which the Mule application is deployed. When we deploy the application on CloudHub, we get a generic url to access the endpoints. Generic URL looks as below:
<application-name>.<region>.cloudhub.io <application-name> is the deployed application name which is unique across all the MuleSoft clients. <region> is the region name in which an application is deployed.
The
public CloudHub (shared) load balancer already redirects these requests, where myApp is the name of the Mule application deployment to CloudHub: HTTP requests to http://myApp.
<region>.cloudhub.io redirects to
http://mule-worker-myApp.<region>.cloudhub.io:8081
HTTPS traffic to https://myApp.<region>.cloudhub.io redirects to
https://mule-worker-myApp.<region>.cloudhub.io:8082