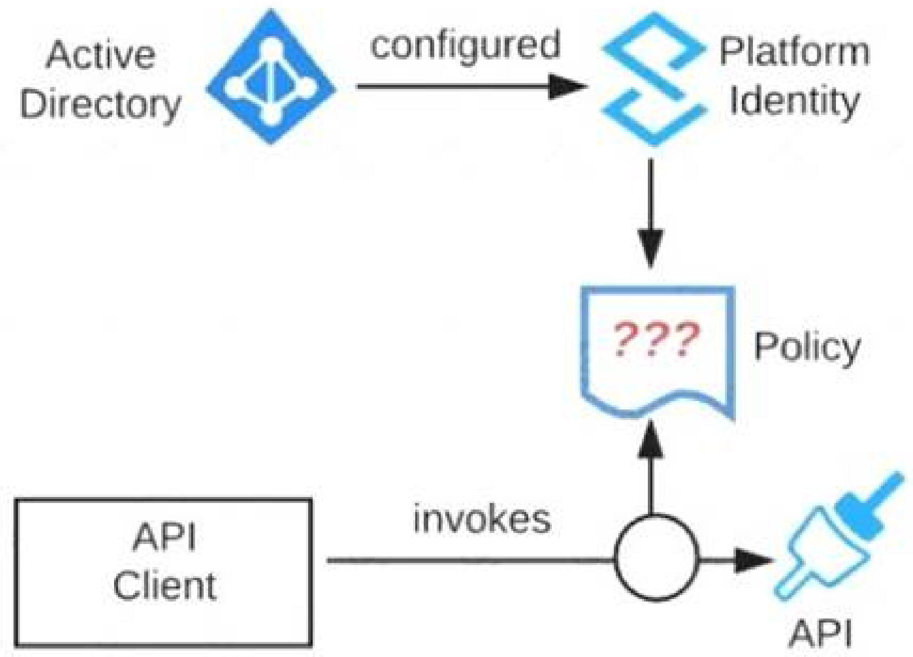
MuleSoft

Exam Questions MCPA-Level-1

MuleSoft Certified Platform Architect - Level 1

NEW QUESTION 1
Refer to the exhibit. An organization is running a Mule standalone runtime and has configured Active Directory as the Anypoint Platform external Identity Provider. The organization does not have budget for other system components.



What policy should be applied to all instances of APIs in the organization to most effecuvelyKestrict access to a specific group of internal users?

A. Apply a basic authentication - LDAP policy; the internal Active Directory will be configured as the LDAP source for authenticating users
B. Apply a client ID enforcement policy; the specific group of users will configure their client applications to use their specific client credentials
C. Apply an IP whitelist policy; only the specific users' workstations will be in the whitelist
D. Apply an OAuth 2.0 access token enforcement policy; the internal Active Directory will be configured as the OAuth server

Answer: A

Explanation:
Correct Answer
Apply a basic authentication - LDAP policy; the internal Active Directory will be configured as the LDAP source for authenticating users.
****
>> IP Whitelisting does NOT fit for this purpose. Moreover, the users workstations may not necessarily have static IPs in the network.
>> OAuth 2.0 enforcement requires a client provider which isn't in the organizations system components.
>> It is not an effective approach to let every user create separate client credentials and configure those for their usage.
The effective way it to apply a basic authentication - LDAP policy and the internal Active Directory will be configured as the LDAP source for authenticating users.

NEW QUESTION 2
How are an 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 client creates an API consumer, which receives API invocations from an API such that they are processed for an API implementation
C. The ApI consumer creates an API client, which sends API invocations to an API such that they are processed by an API implementation
D. The ApI client creates an API consumer, which sends API invocations to an API such that they are processed by an API implementation

Answer: C

Explanation:

Correct Answer
The API consumer creates an API client, which sends API invocations to an API such that they are processed by an API implementation
*** Terminology:
>> API Client - It is a piece of code or program the is written to invoke an API
>> API Consumer - An owner/entity who owns the API Client. API Consumers write API clients.
>> API - The provider of the API functionality. Typically an API Instance on API Manager where they are managed and operated.
>> API Implementation - The actual piece of code written by API provider where the functionality of the API is implemented. Typically, these are Mule Applications running on Runtime Manager.

NEW QUESTION 3
In which layer of API-led connectivity, does the business logic orchestration reside?

A. System Layer
B. Experience Layer
C. Process Layer

Answer: C

Explanation:

Correct Answer
Process Layer
****
>> Experience layer is dedicated for enrichment of end user experience. This layer is to meet the needs of different API clients/ consumers.
>> System layer is dedicated to APIs which are modular in nature and implement/ expose various individual functionalities of backend systems
>> Process layer is the place where simple or complex business orchestration logic is written by invoking one or many System layer modular APIs

So, Process Layer is the right answer.

NEW QUESTION 4
What is most likely NOT a characteristic of an integration test for a REST API implementation?

A. The test needs all source and/or target systems configured and accessible
B. The test runs immediately after the Mule application has been compiled and packaged
C. The test is triggered by an external HTTP request
D. The test prepares a known request payload and validates the response payload

Answer: B

Explanation:

Correct Answer
The test runs immediately after the Mule application has been compiled and packaged
****
>> Integration tests are the last layer of tests we need to add to be fully covered.
>> These tests actually run against Mule running with your full configuration in place and are tested from external source as they work in PROD.
>> These tests exercise the application as a whole with actual transports enabled. So, external systems are affected when these tests run.
So, these tests do NOT run immediately after the Mule application has been compiled and packaged.
FYI... Unit Tests are the one that run immediately after the Mule application has been compiled and packaged.

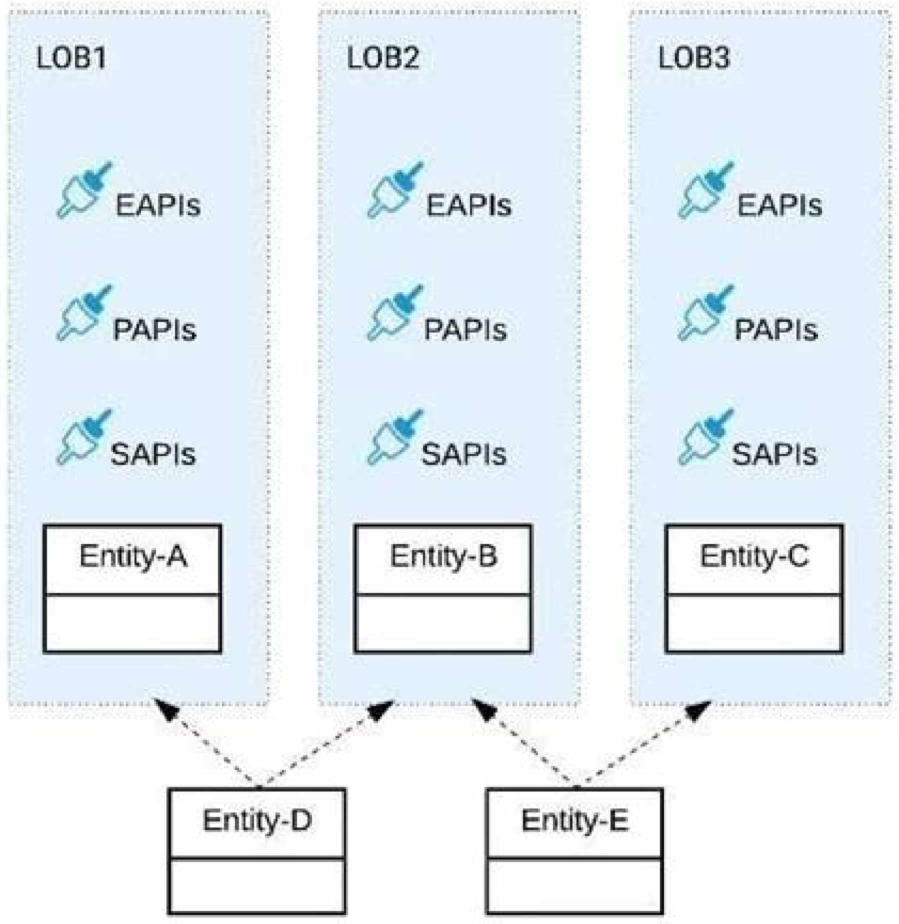
NEW QUESTION 5
Say, there is a legacy CRM system called CRM-Z which is offering below functions:
* 1. Customer creation
* 2. Amend details of an existing customer
* 3. Retrieve details of a customer
* 4. Suspend a customer

A. Implement a system API named customerManagement which has all the functionalities wrapped in it asvarious operations/resources
B. Implement different system APIs named createCustomer, amendCustomer, retrieveCustomer and suspendCustomer as they are modular and has seperation of concerns
C. Implement different system APIs named createCustomerInCRMZ, amendCustomerInCRMZ, retrieveCustomerFromCRMZ and suspendCustomerInCRMZ as they are modular and has seperation of concerns

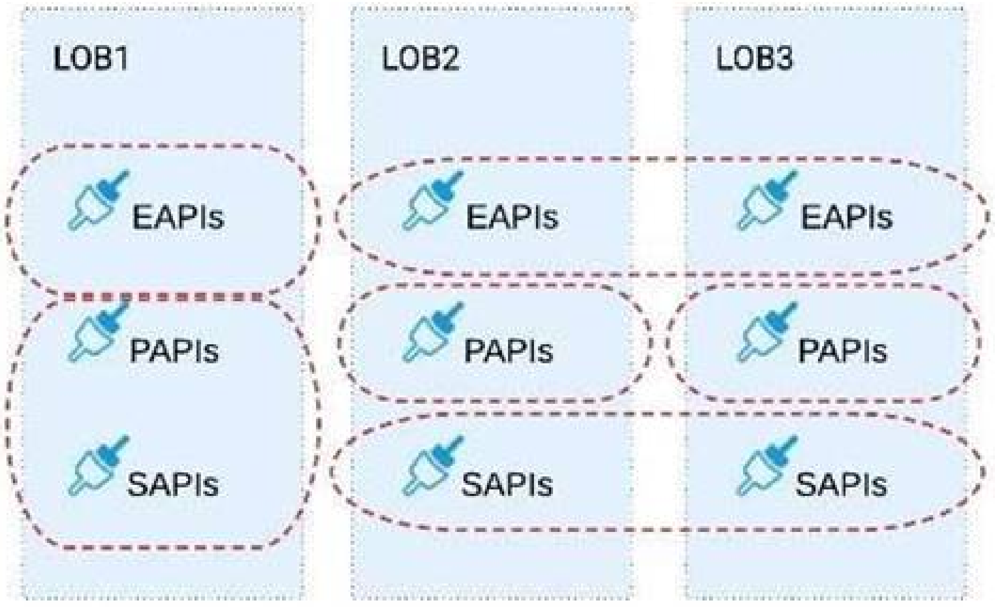
Answer: B

Explanation:
Correct Answer
Implement different system APIs named createCustomer, amendCustomer, retrieveCustomer and suspendCustomer as they are modular and has seperation of concerns
****
>> It is quite normal to have a single API and different Verb + Resource combinations. However, this fits well for an Experience API or a Process API but not a best architecture style for System APIs. So, option with just one customerManagement API is not the best choice here.
>> The option with APIs in createCustomerInCRMZ format is next close choice w.r.t modularization and less maintenance but the naming of APIs is directly coupled with the legacy system. A better foreseen approach would be to name your APIs by abstracting the backend system names as it allows seamless replacement/migration of any backend system anytime. So, this is not the correct choice too.
>> createCustomer, amendCustomer, retrieveCustomer and suspendCustomer is the right approach and is the best fit compared to other options as they are both modular and same time got the names decoupled from backend system and it has covered all requirements a System API needs.

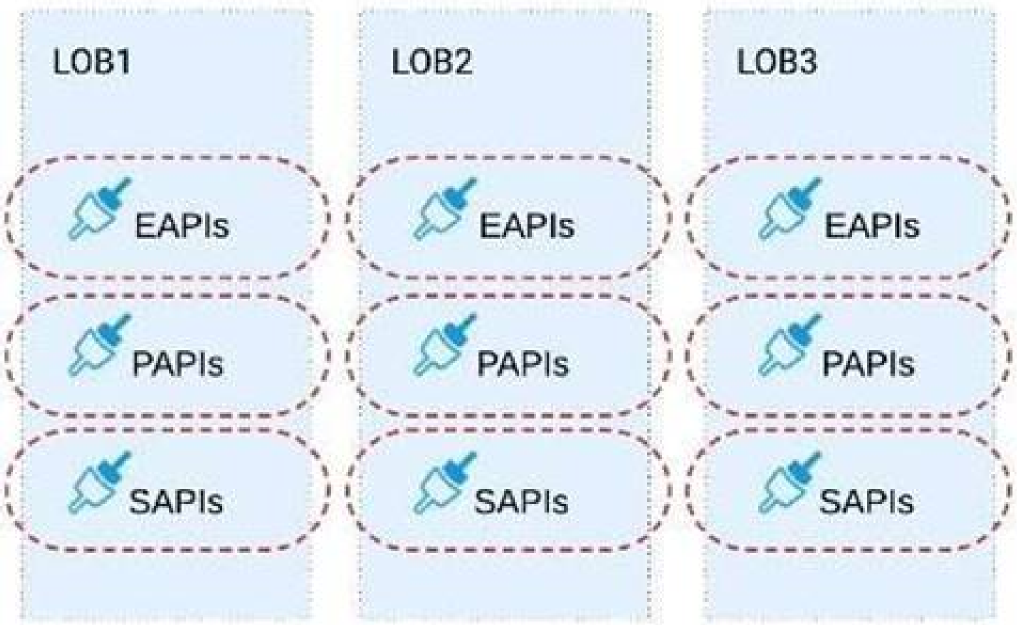
NEW QUESTION 6
Refer to the exhibit.



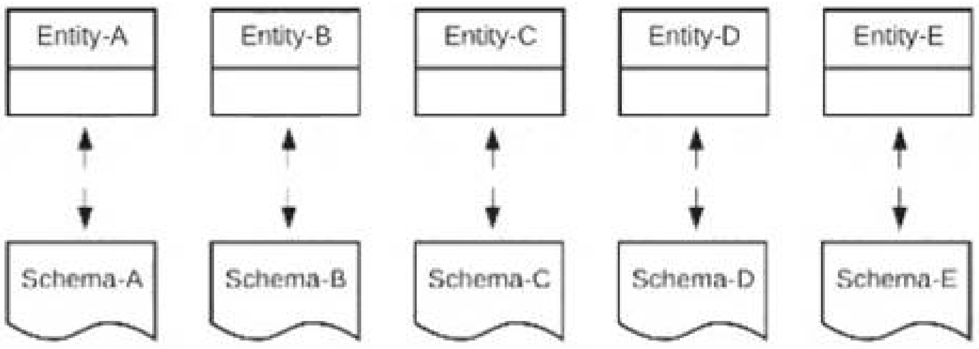
Three business processes need to be implemented, and the implementations need to communicate with several different SaaS applications.
These processes are owned by separate (siloed) LOBs and are mainly independent of each other, but do share a few business entities. Each LOB has one development team and their own budget
In this organizational context, what is the most effective approach to choose the API data models for the APIs that will implement these business processes with minimal redundancy of the data models?
A) Build several Bounded Context Data Models that align with coherent parts of the business processes and the definitions of associated business entities



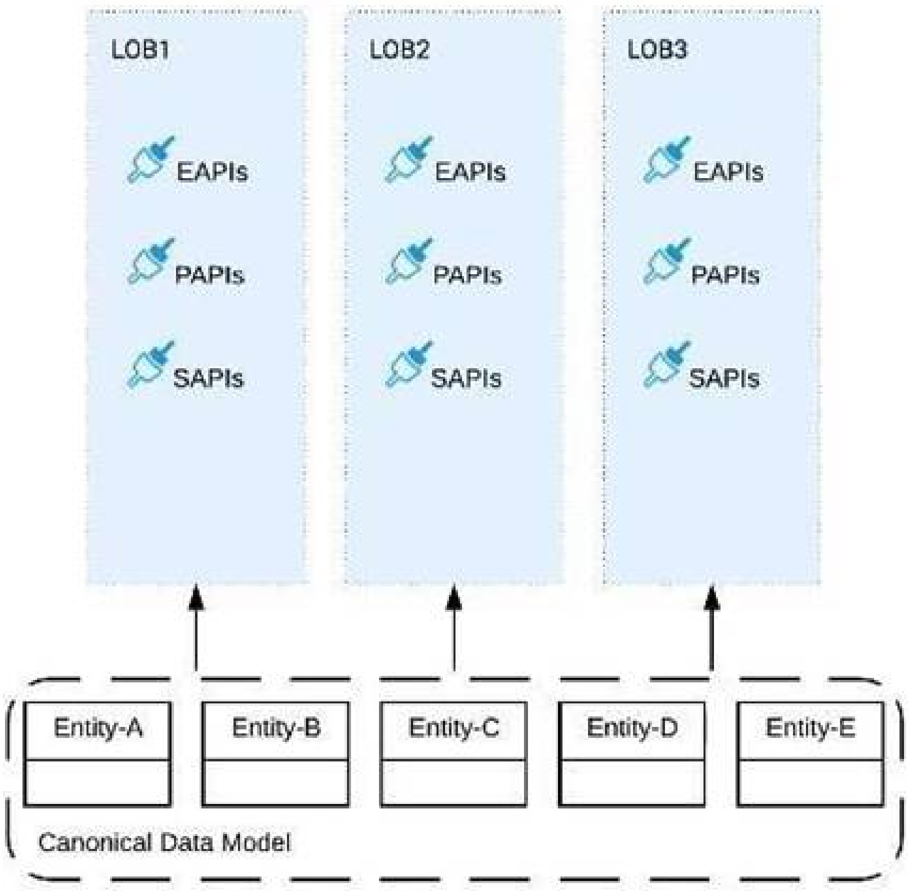
B) Build distinct data models for each API to follow established micro-services and Agile API-centric practices



C) Build all API data models using XML schema to drive consistency and reuse across the organization



D) Build one centralized Canonical Data Model (Enterprise Data Model) that unifies all the data types from all three business processes, ensuring the data model is consistent and non-redundant



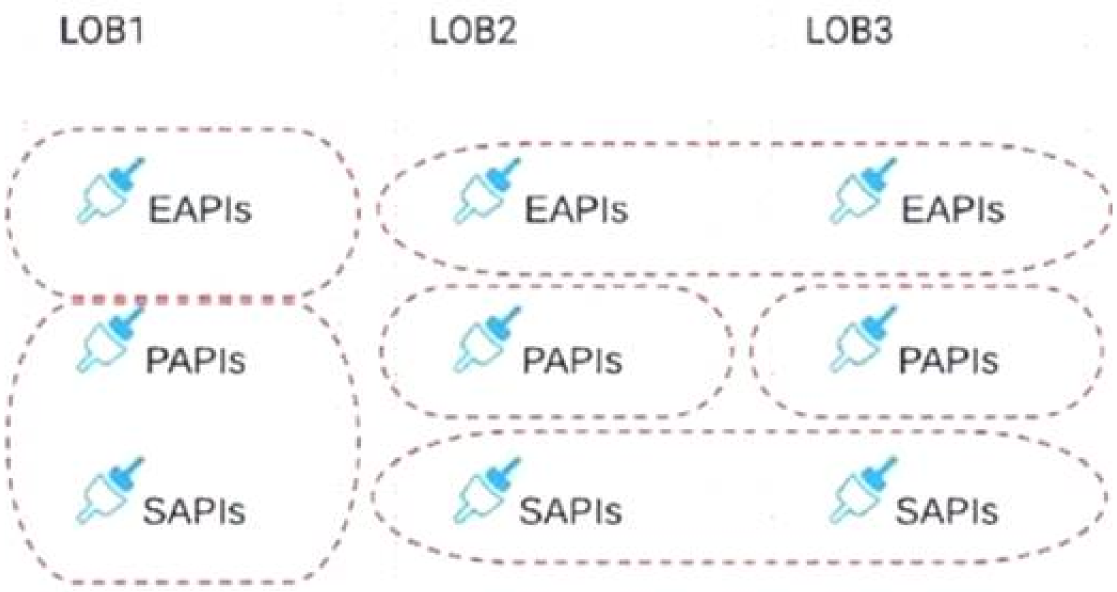
A. Option A
B. Option B
C. Option C
D. Option D

Answer: A

Explanation:

Correct Answer
Build several Bounded Context Data Models that align with coherent parts of the business processes and the definitions of associated business entities.
****

>> The options w.r.t building API data models using XML schema/ Agile API-centric practices are irrelevant to the scenario given in the question. So these two are INVALID.
>> Building EDM (Enterprise Data Model) is not feasible or right fit for this scenario as the teams and LOBs work in silo and they all have different initiatives, budget etc.. Building EDM needs intensive coordination among all the team which evidently seems not possible in this scenario.
So, the right fit for this scenario is to build several Bounded Context Data Models that align with coherent parts of the business processes and the definitions of associated business entities.



NEW QUESTION 7
A code-centric API documentation environment should allow API consumers to investigate and execute API client source code that demonstrates invoking one or more APIs as part of representative scenarios.
What is the most effective way to provide this type of code-centric API documentation environment using Anypoint Platform?

A. Enable mocking services for each of the relevant APIs and expose them via their Anypoint Exchange entry
B. Ensure the APIs are well documented through their Anypoint Exchange entries and API Consoles and share these pages with all API consumers
C. Create API Notebooks and include them in the relevant Anypoint Exchange entries
D. Make relevant APIs discoverable via an Anypoint Exchange entry

Answer: C

Explanation:

Correct Answer
Create API Notebooks and Include them in the relevant Anypoint exchange entries

****
>> API Notebooks are the one on Anypoint Platform that enable us to provide code-centric API documentation

NEW QUESTION 8
A company uses a hybrid Anypoint Platform deployment model that combines the EU control plane with customer-hosted Mule runtimes. After successfully testing a Mule API implementation in the Staging environment, the Mule API implementation is set with environment-specific properties and must be promoted to the Production environment. What is a way that MuleSoft recommends to configure the Mule API implementation and automate its promotion to the Production environment?

A. Bundle properties files for each environment into the Mule API implementation's deployable archive, then promote the Mule API implementation to the Production environment using Anypoint CLI or the Anypoint Platform REST APIsB.
B. Modify the Mule API implementation's properties in the API Manager Properties tab, then promote the Mule API implementation to the Production environment using API Manager
C. Modify the Mule API implementation's properties in Anypoint Exchange, then promote the Mule API implementation to the Production environment using Runtime Manager
D. Use an API policy to change properties in the Mule API implementation deployed to the Staging environment and another API policy to deploy the Mule API implementation to the Production environment

Answer: A

Explanation:

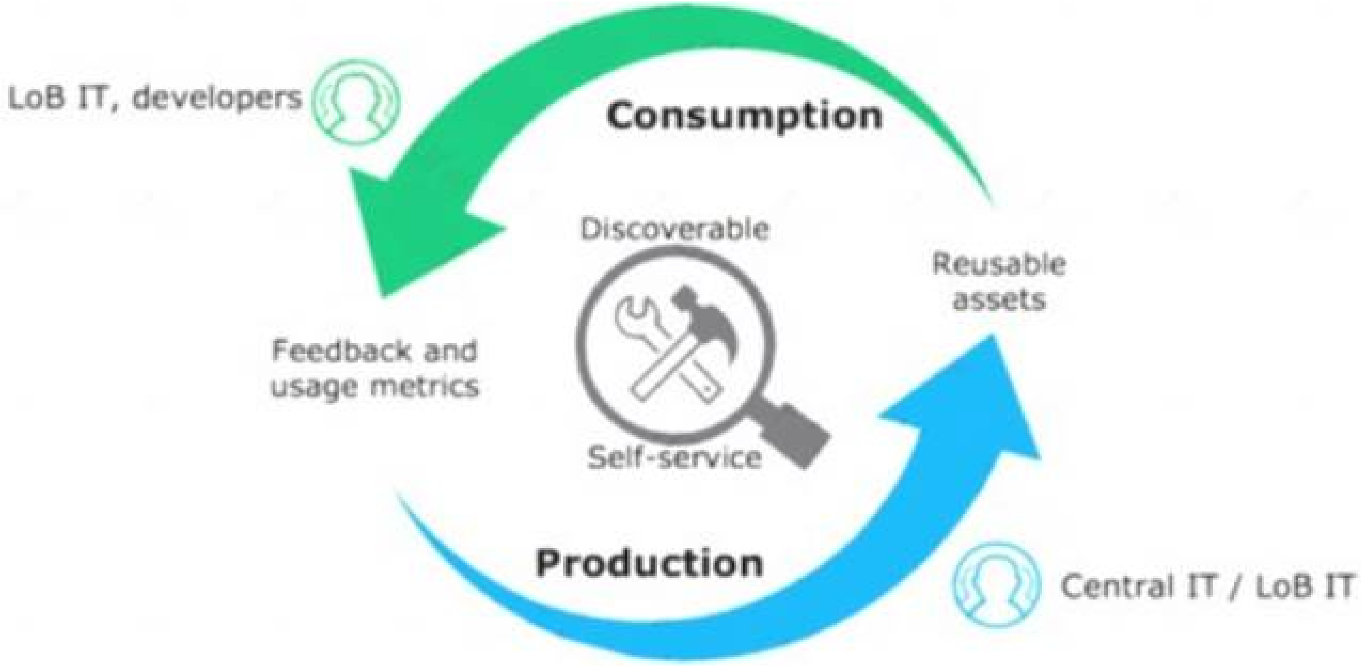
Correct Answer
Bundle properties files for each environment into the Mule API implementation's deployable archive, then promote the Mule API implementation to the Production environment using Anypoint CLI or the Anypoint Platform REST APIs
****
>> Anypoint Exchange is for asset discovery and documentation. It has got no provision to modify the properties of Mule API implementations at all.
>> API Manager is for managing API instances, their contracts, policies and SLAs. It has also got no provision to modify the properties of API implementations.
>> API policies are to address Non-functional requirements of APIs and has again got no provision to modify the properties of API implementations.
So, the right way and recommended way to do this as part of development practice is to bundle properties files for each environment into the Mule API implementation and just point and refer to respective file per environment.

NEW QUESTION 9
Which of the below, when used together, makes the IT Operational Model effective?

A. Create reusable assets, Do marketing on the created assets across organization, Arrange time to time LOB reviews to ensure assets are being consumed or not
B. Create reusable assets, Make them discoverable so that LOB teams can self-serve and browse the APIs, Get active feedback and usage metrics
C. Create resuable assets, make them discoverable so that LOB teams can self-serve and browse the APIs

Answer: C

Explanation:
Correct Answer
Create reusable assets, Make them discoverable so that LOB teams can self-serve and browse the APIs, Get active feedback and usage metrics.
** Diagram, arrow Description automatically generated



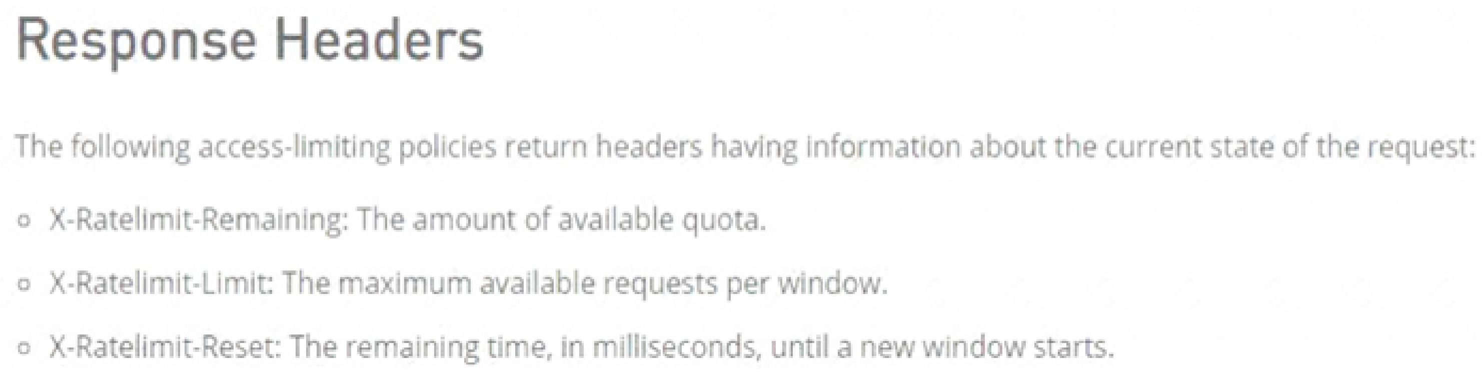
NEW QUESTION 10
How can the application of a rate limiting API policy be accurately reflected in the RAML definition of an API?

A. By refining the resource definitions by adding a description of the rate limiting policy behavior
B. By refining the request definitions by adding a remaining Requests query parameter with description, type, and example
C. By refining the response definitions by adding the out-of-the-box Anypoint Platform rate-limit-enforcement securityScheme with description, type, and example
D. By refining the response definitions by adding the x-ratelimit-* response headers with description, type, and example

Answer: D

Explanation:
Correct Answer
By refining the response definitions by adding the x-ratelimit-* response headers with description, type, and example

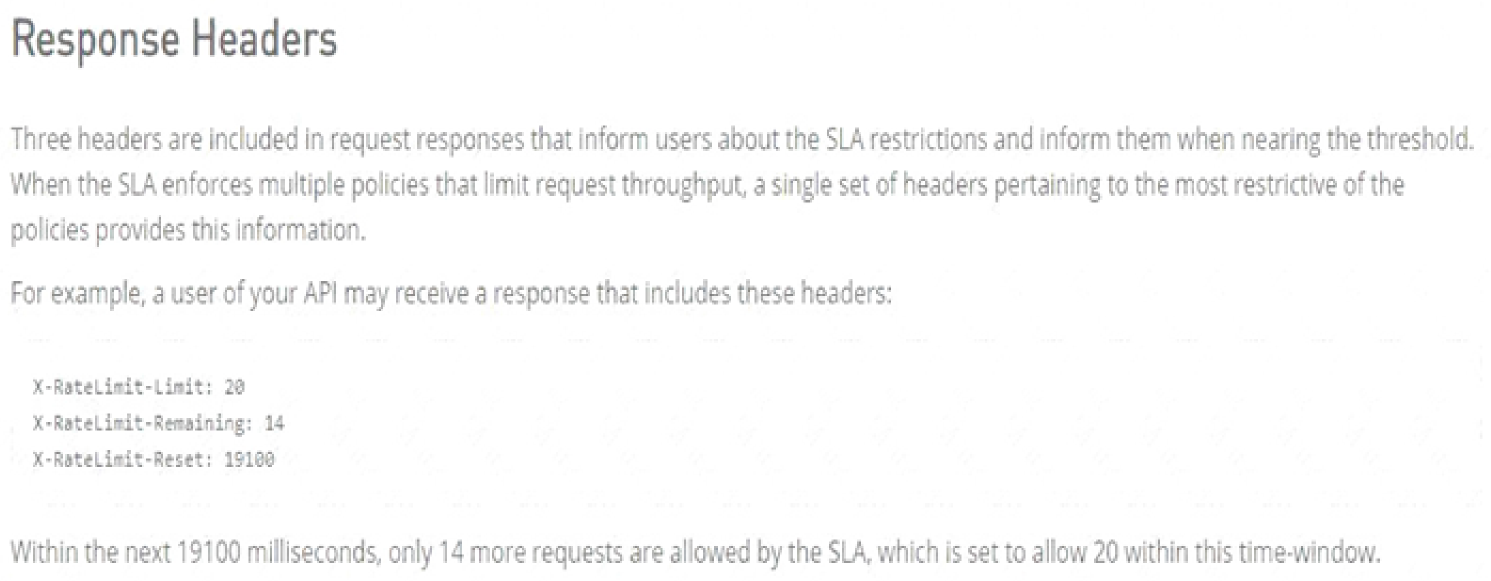
****



Response Headers

The following access-limiting policies return headers having information about the current state of the request:

- X-Ratelimit-Remaining: The amount of available quota.
- X-Ratelimit-Limit: The maximum available requests per window.
- X-Ratelimit-Reset: The remaining time, in milliseconds, until a new window starts.



Response Headers

Three headers are included in request responses that inform users about the SLA restrictions and inform them when nearing the threshold. When the SLA enforces multiple policies that limit request throughput, a single set of headers pertaining to the most restrictive of the policies provides this information.

For example, a user of your API may receive a response that includes these headers:

```
X-RateLimit-Limit: 20
X-RateLimit-Remaining: 14
X-RateLimit-Reset: 19100
```

Within the next 19100 milliseconds, only 14 more requests are allowed by the SLA, which is set to allow 20 within this time-window.

References:
https://docs.mulesoft.com/api-manager/2.x/rate-limiting-and-throttling#response-headers https://docs.mulesoft.com/api-manager/2.x/rate-limiting-and-throttling-sla-based-policies#response-headers

NEW QUESTION 10
What is a best practice when building System APIs?

A. Document the API using an easily consumable asset like a RAML definition
B. Model all API resources and methods to closely mimic the operations of the backend system
C. Build an Enterprise Data Model (Canonical Data Model) for each backend system and apply it to System APIs
D. Expose to API clients all technical details of the API implementation's interaction wifch the backend system

Answer: B

Explanation:

Correct Answer
Model all API resources and methods to closely mimic the operations of the backend system.
****
>> There are NO fixed and straight best practices while opting data models for APIs. They are completly contextual and depends on number of factors. Based upon those factors, an enterprise can choose if they have to go with Enterprise Canonical Data Model or Bounded Context Model etc.
>> One should NEVER expose the technical details of API implementation to their API clients. Only the API interface/ RAML is exposed to API clients.
>> It is true that the RAML definitions of APIs should be as detailed as possible and should reflect most of the documentation. However, just that is NOT enough to call your API as best documented API. There should be even more documentation on Anypoint Exchange with API Notebooks etc. to make and create a developer friendly API and repository..
>> The best practice always when creating System APIs is to create their API interfaces by modeling their resources and methods to closely reflect the operations and functionalities of that backend system.

NEW QUESTION 12
An API implementation is deployed to CloudHub.
What conditions can be alerted on using the default Anypoint Platform functionality, where the alert conditions depend on the end-to-end request processing of the API implementation?

A. When the API is invoked by an unrecognized API client
B. When a particular API client invokes the API too often within a given time period
C. When the response time of API invocations exceeds a threshold
D. When the API receives a very high number of API invocations

Answer: C

Explanation:

Correct Answer
When the response time of API invocations exceeds a threshold
****
>> Alerts can be setup for all the given options using the default Anypoint Platform functionality
>> However, the question insists on an alert whose conditions depend on the end-to-end request processing of the API implementation.
>> Alert w.r.t "Response Times" is the only one which requires end-to-end request processing of API implementation in order to determine if the threshold is exceeded or not.

NEW QUESTION 17

An API client calls one method from an existing API implementation. The API implementation is later updated. What change to the API implementation would require the API client's invocation logic to also be updated?

A. When the data type of the response is changed for the method called by the API client
B. When a new method is added to the resource used by the API client
C. When a new required field is added to the method called by the API client
D. When a child method is added to the method called by the API client

Answer: C

Explanation:

Correct Answer
When a new required field is added to the method called by the API client
****
>> Generally, the logic on API clients need to be updated when the API contract breaks.
>> When a new method or a child method is added to an API , the API client does not break as it can still continue to use its existing method. So these two options are out.
>> We are left for two more where "datatype of the response if changed" and "a new required field is added".
>> Changing the datatype of the response does break the API contract. However, the question is insisting on the "invocation" logic and not about the response handling logic. The API client can still invoke the API successfully and receive the response but the response will have a different datatype for some field.
>> Adding a new required field will break the API's invocation contract. When adding a new required field, the API contract breaks the RAML or API spec agreement that the API client/API consumer and API provider has between them. So this requires the API client invocation logic to also be updated.

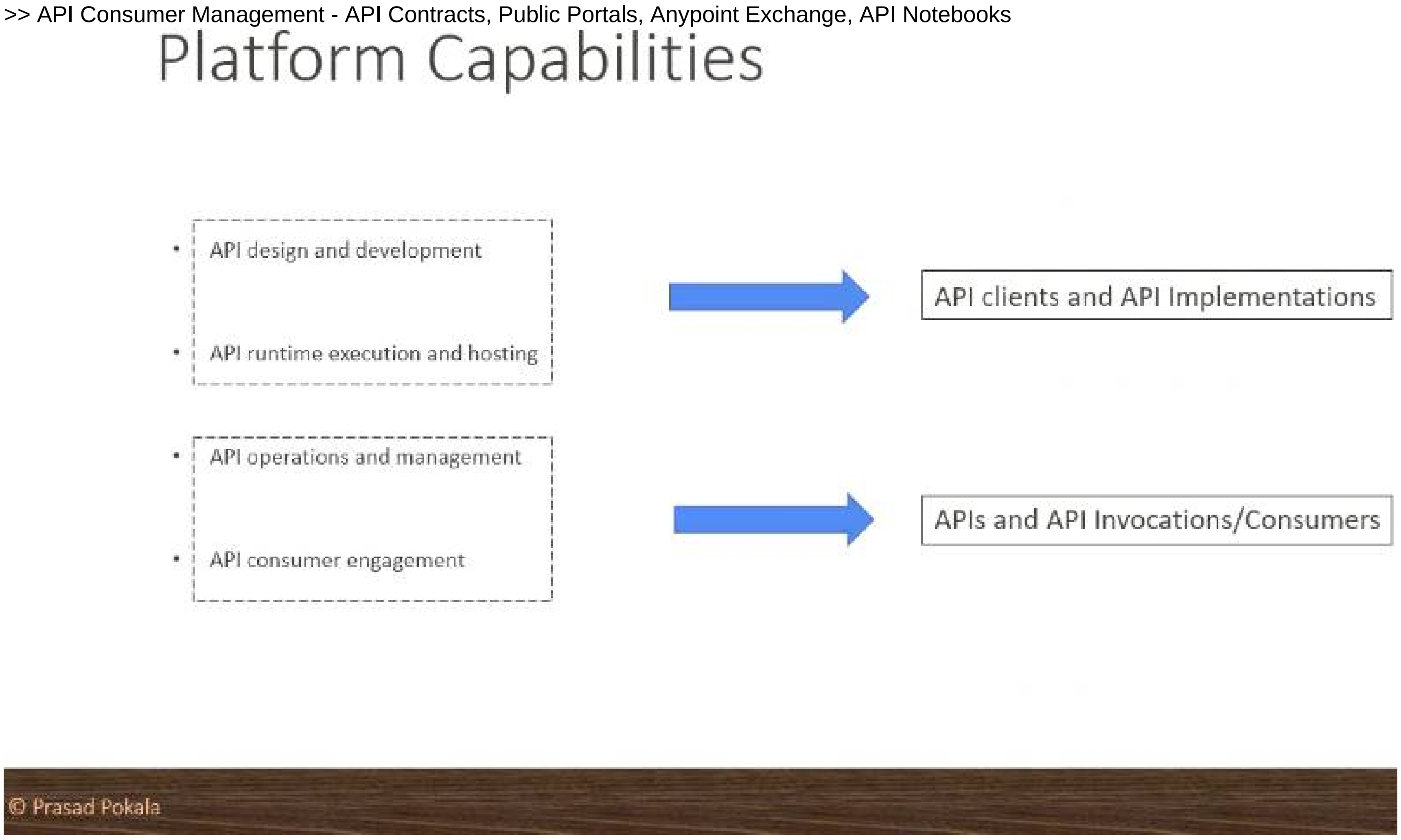
NEW QUESTION 21
What are 4 important Platform Capabilities offered by Anypoint Platform?

A. API Versioning, API Runtime Execution and Hosting, API Invocation, API Consumer Engagement
B. API Design and Development, API Runtime Execution and Hosting, API Versioning, API Deprecation
C. API Design and Development, API Runtime Execution and Hosting, API Operations and Management, API Consumer Engagement
D. API Design and Development, API Deprecation, API Versioning, API Consumer Engagement

Answer: C

Explanation:

Correct Answer
API Design and Development, API Runtime Execution and Hosting, API Operations and Management, API Consumer Engagement
****
>> API Design and Development - Anypoint Studio, Anypoint Design Center, Anypoint Connectors
>> API Runtime Execution and Hosting - Mule Runtimes, CloudHub, Runtime Services
>> API Operations and Management - Anypoint API Manager, Anypoint Exchange
>> API Consumer Management - API Contracts, Public Portals, Anypoint Exchange, API Notebooks



Platform Capabilities

- API design and development
- API runtime execution and hosting

API clients and API Implementations

- API operations and management
- API consumer engagement

APIs and API Invocations/Consumers

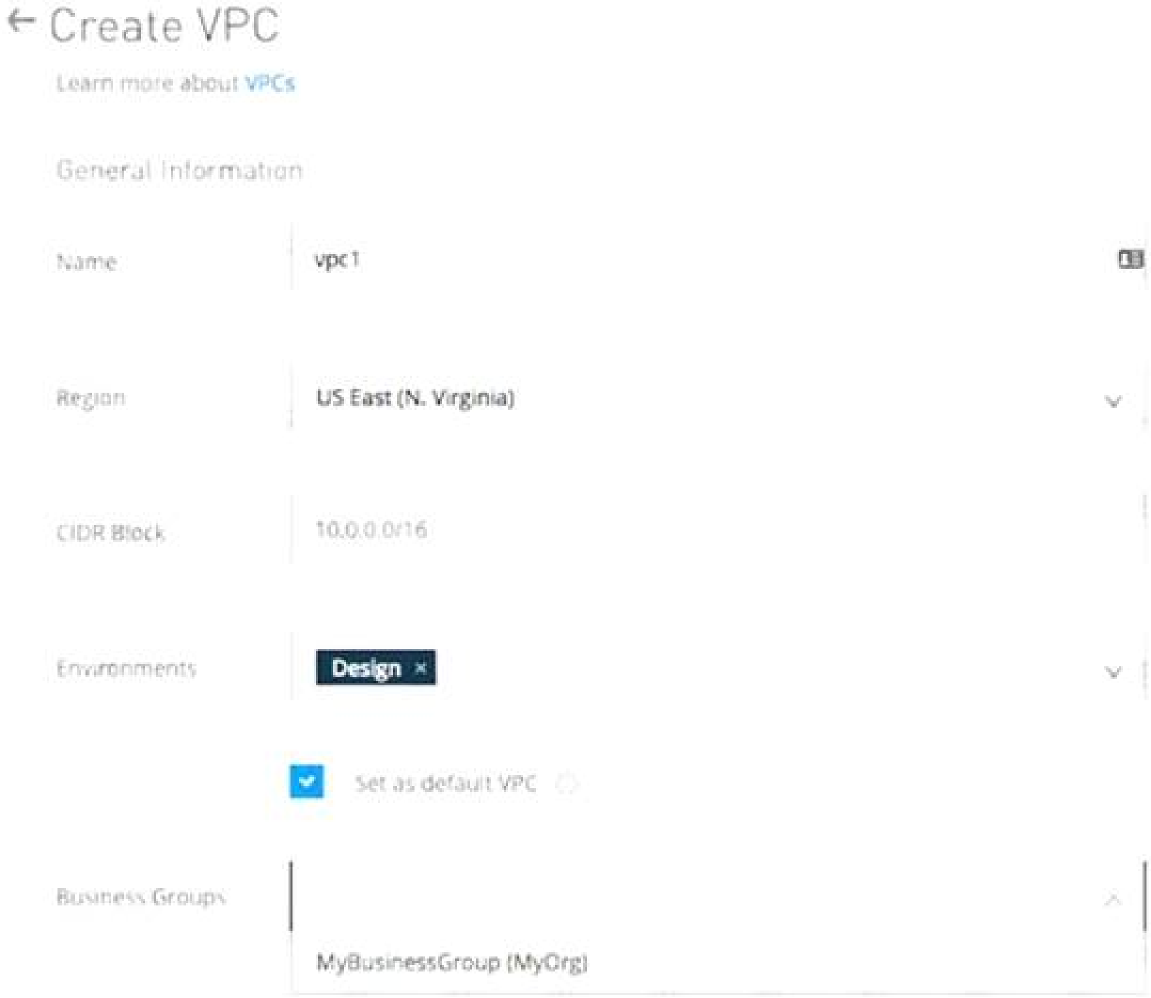
© Prasad Pokala

NEW QUESTION 22
What is true about the technology architecture of Anypoint VPCs?

A. The private IP address range of an Anypoint VPC is automatically chosen by CloudHub
B. Traffic between Mule applications deployed to an Anypoint VPC and on-premises systems can staywithin a private network
C. Each CloudHub environment requires a separate Anypoint VPC
D. VPC peering can be used to link the underlying AWS VPC to an on-premises (non AWS) private network

Answer: B

Explanation:
Correct Answer
Traffic between Mule applications deployed to an Anypoint VPC and on-premises systems can stay within a private network
****
>> The private IP address range of an Anypoint VPC is NOT automatically chosen by CloudHub. It is chosen by us at the time of creating VPC using thr CIDR blocks.
CIDR Block: The size of the Anypoint VPC in Classless Inter-Domain Routing (CIDR) notation.
For example, if you set it to 10.111.0.0/24, the Anypoint VPC is granted 256 IP addresses from 10.111.0.0 to 10.111.0.255.
Ideally, the CIDR Blocks you choose for the Anypoint VPC come from a private IP space, and should not overlap with any other Anypoint VPC's CIDR Blocks, or any CIDR Blocks in use in your corporate network.



← Create VPC

Learn more about VPCs

General Information

Name: vpc1

Region: US East (N. Virginia)

CIDR Block: 10.0.0.0/16

Environments: Design ×

☑ Set as default VPC

Business Groups: MyBusinessGroup (MyOrg)

that each CloudHub environment requires a separate Anypoint VPC. Once an Anypoint VPC is created, we can choose a same VPC by multiple environments. However, it is generally a best and recommended practice to always have seperate Anypoint VPCs for Non-Prod and Prod environments.
>> We use Anypoint VPN to link the underlying AWS VPC to an on-premises (non AWS) private network. NOT VPC Peering.

NEW QUESTION 24
Due to a limitation in the backend system, a system API can only handle up to 500 requests per second. What is the best type of API policy to apply to the system API to avoid overloading the backend system?

A. Rate limiting
B. HTTP caching
C. Rate limiting - SLA based
D. Spike control

Answer: D

Explanation:
Correct Answer
Spike control
****
>> First things first, HTTP Caching policy is for purposes different than avoiding the backend system from overloading. So this is OUT.
>> Rate Limiting and Throttling/ Spike Control policies are designed to limit API access, but have different intentions.
>> Rate limiting protects an API by applying a hard limit on its access.
>> Throttling/ Spike Control shapes API access by smoothing spikes in traffic. That is why, Spike Control is the right option.

NEW QUESTION 27
Traffic is routed through an API proxy to an API implementation. The API proxy is managed by API Manager and the API implementation is deployed to a CloudHub VPC using Runtime Manager. API policies have been applied to this API. In this deployment scenario, at what point are the API policies enforced on incoming API client requests?

A. At the API proxy
B. At the API implementation
C. At both the API proxy and the API implementation
D. At a MuleSoft-hosted load balancer

Answer: A

Explanation:
Correct Answer
At the API proxy

****

>> API Policies can be enforced at two places in Mule platform.
>> One - As an Embedded Policy enforcement in the same Mule Runtime where API implementation is running.
>> Two - On an API Proxy sitting in front of the Mule Runtime where API implementation is running.
>> As the deployment scenario in the question has API Proxy involved, the policies will be enforced at the API Proxy.

NEW QUESTION 31
A set of tests must be performed prior to deploying API implementations to a staging environment. Due to data security and access restrictions, untested APIs cannot be granted access to the backend systems, so instead mocked data must be used for these tests. The amount of available mocked data and its contents is sufficient to entirely test the API implementations with no active connections to the backend systems. What type of tests should be used to incorporate this mocked data?

A. Integration tests
B. Performance tests
C. Functional tests (Blackbox)
D. Unit tests (Whitebox)

Answer: D

Explanation:

Correct Answer
Unit tests (Whitebox)
****

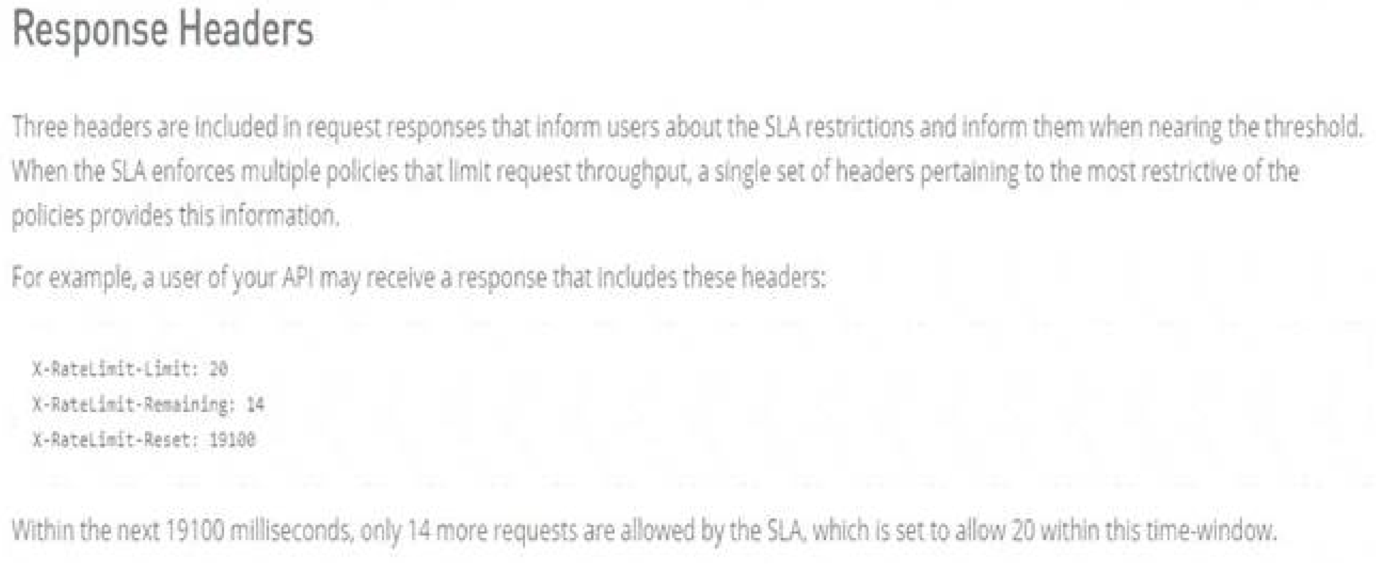
NEW QUESTION 32
Question 10: Skipped
An API implementation returns three X-RateLimit-* HTTP response headers to a requesting API client. What type of information do these response headers indicate to the API client?

A. The error codes that result from throttling
B. A correlation ID that should be sent in the next request
C. The HTTP response size
D. The remaining capacity allowed by the API implementation

Answer: D

Explanation:

Correct Answer
The remaining capacity allowed by the API implementation.
****
>> Reference:
https://docs.mulesoft.com/api-manager/2.x/rate-limiting-and-throttling-sla-based-policies#response-headers



Response Headers

Three headers are included in request responses that inform users about the SLA restrictions and inform them when nearing the threshold. When the SLA enforces multiple policies that limit request throughput, a single set of headers pertaining to the most restrictive of the policies provides this information.

For example, a user of your API may receive a response that includes these headers:

```
X-RateLimit-Limit: 20
X-RateLimit-Remaining: 14
X-RateLimit-Reset: 19100
```

Within the next 19100 milliseconds, only 14 more requests are allowed by the SLA, which is set to allow 20 within this time-window.

NEW QUESTION 36
An API implementation is deployed on a single worker on CloudHub and invoked by external API clients (outside of CloudHub). How can an alert be set up that is guaranteed to trigger AS SOON AS that API implementation stops responding to API invocations?

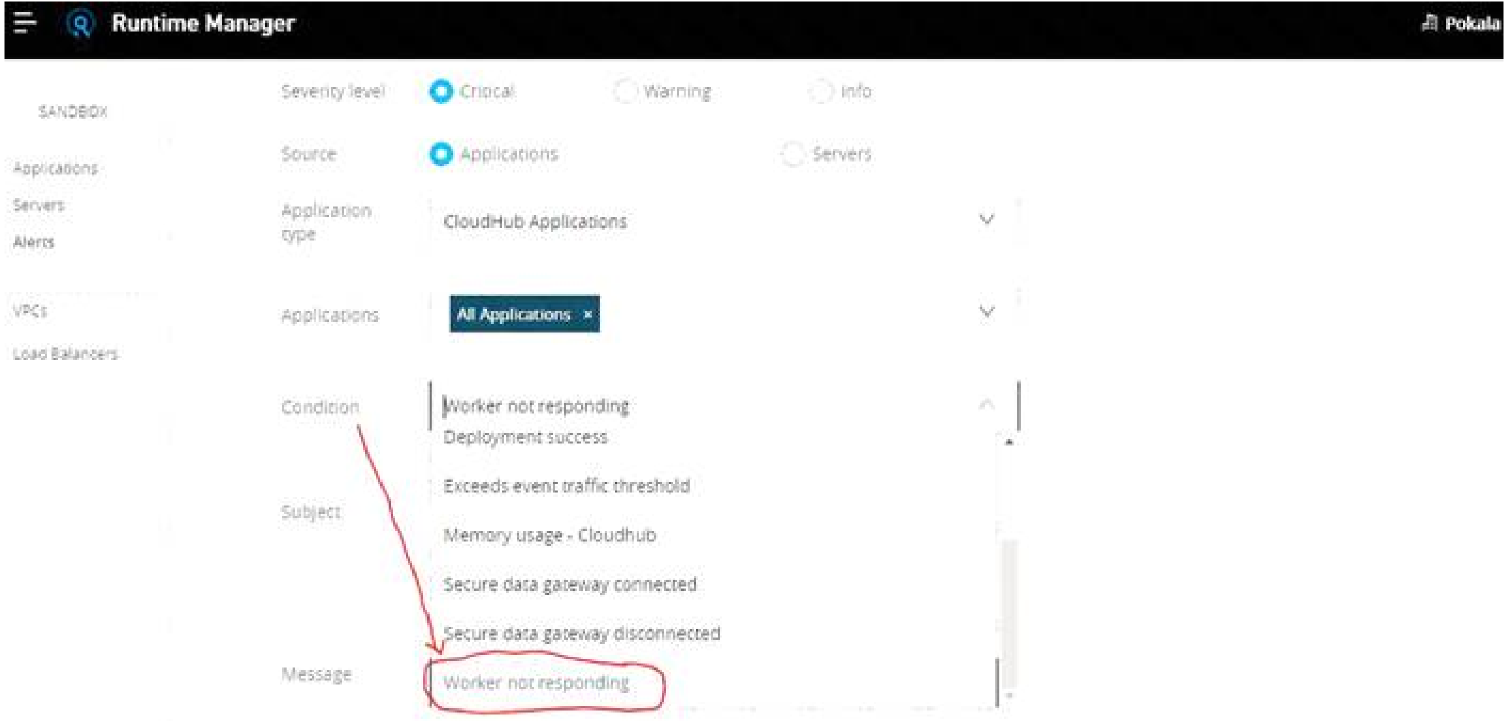
A. Implement a heartbeat/health check within the API and invoke it from outside the Anypoint Platform and alert when the heartbeat does not respond
B. Configure a "worker not responding" alert in Anypoint Runtime Manager
C. Handle API invocation exceptions within the calling API client and raise an alert from that API client when the API Is unavailable
D. Create an alert for when the API receives no requests within a specified time period

Answer: B

Explanation:

Correct Answer
Configure a “Worker not responding” alert in Anypoint Runtime Manager.
****
>> All the options eventually helps to generate the alert required when the application stops responding.

>> However, handling exceptions within calling API and then raising alert from API client is inappropriate and silly. There could be many API clients invoking the API implementation and it is not ideal to have this setup consistently in all of them. Not a realistic way to do.
>> Implementing a health check/ heartbeat with in the API and calling from outside to detmine the health sounds OK but needs extra setup for it and same time there are very good chances of generating false alarms when there are any intermittent network issues between external tool calling the health check API on API implementation. The API implementation itself may not have any issues but due to some other factors some false alarms may go out.
>> Creating an alert in API Manager when the API receives no requests within a specified time period would actually generate realistic alerts but even here some false alarms may go out when there are genuinely no
requests from API clients.
The best and right way to achieve this requirement is to setup an alert on Runtime Manager with a condition "Worker not responding". This would generate an alert AS SOON AS the workers become unresponsive.



Severity level ● Critical ○ Warning ○ Info

Source ● Applications ○ Servers

Application type: CloudHub Applications

Applications: All Applications ×

Condition: Worker not responding

Deployment success
Exceeds event traffic threshold
Memory usage - Cloudhub
Secure data gateway connected
Secure data gateway disconnected
Worker not responding

Subject

Message

Bottom of Form Top of Form

NEW QUESTION 38
What do the API invocation metrics provided by Anypoint Platform provide?

A. ROI metrics from APIs that can be directly shared with business users
B. Measurements of the effectiveness of the application network based on the level of reuse
C. Data on past API invocations to help identify anomalies and usage patterns across various APIs
D. Proactive identification of likely future policy violations that exceed a given threat threshold

Answer: C

Explanation:

Correct Answer
Data on past API invocations to help identify anomalies and usage patterns across various APIs
****
API Invocation metrics provided by Anypoint Platform:
>> Does NOT provide any Return Of Investment (ROI) related information. So the option suggesting it is OUT.
>> Does NOT provide any information w.r.t how APIs are reused, whether there is effective usage of APIs or not etc...
>> Does NOT prodive any prediction information as such to help us proactively identify any future policy violations.
So, the kind of data/information we can get from such metrics is on past API invocations to help identify anomalies and usage patterns across various APIs.

NEW QUESTION 39
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 API's 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 update should be identified as a project risk and full regression testing of the functionality that uses this API should be run
B. The API producer should be contacted to understand the change to existing functionality
C. The API producer should be requested to run the old version in parallel with the new one
D. The API client code ONLY needs to be changed if it needs to take advantage of new features

Answer: D

NEW QUESTION 40
What CANNOT be effectively enforced using an API policy in Anypoint Platform?

A. Guarding against Denial of Service attacks
B. Maintaining tamper-proof credentials between APIs
C. Logging HTTP requests and responses
D. Backend system overloading

Answer: A

Explanation:
Correct Answer
Guarding against Denial of Service attacks

****
>> Backend system overloading can be handled by enforcing "Spike Control Policy"
>> Logging HTTP requests and responses can be done by enforcing "Message Logging Policy"
>> Credentials can be tamper-proofed using "Security" and "Compliance" Policies
However, unfortunately, there is no proper way currently on Anypoint Platform to guard against DOS attacks.

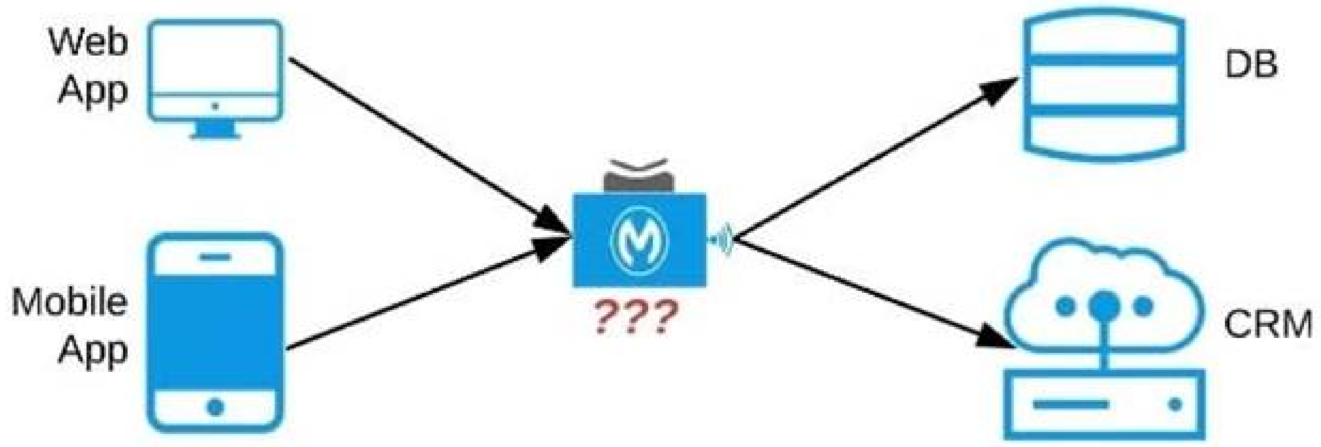
NEW QUESTION 43
What API policy would be LEAST LIKELY used when designing an Experience API that is intended to work with a consumer mobile phone or tablet application?

A. OAuth 2.0 access token enforcement
B. Client ID enforcement
C. JSON threat protection
D. IPwhitellst

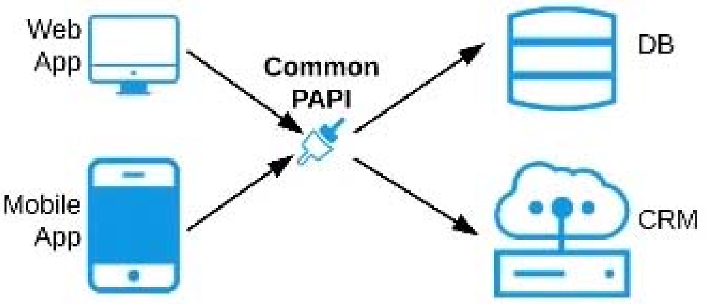
Answer: D

Explanation:
Correct Answer
IP whitelist
****
>> OAuth 2.0 access token and Client ID enforcement policies are VERY common to apply on Experience APIs as API consumers need to register and access the APIs using one of these mechanisms
>> JSON threat protection is also VERY common policy to apply on Experience APIs to prevent bad or suspicious payloads hitting the API implementations.
>> IP whitelisting policy is usually very common in Process and System APIs to only whitelist the IP range inside the local VPC. But also applied occassionally on some experience APIs where the End User/ API Consumers are FIXED.
>> When we know the API consumers upfront who are going to access certain Experience APIs, then we can request for static IPs from such consumers and whitelist them to prevent anyone else hitting the API.
However, the experience API given in the question/ scenario is intended to work with a consumer mobile phone or tablet application. Which means, there is no way we can know all possible IPs that are to be whitelisted as mobile phones and tablets can so many in number and any device in the city/state/country/globe.
So, It is very LEAST LIKELY to apply IP Whitelisting on such Experience APIs whose consumers are typically Mobile Phones or Tablets.

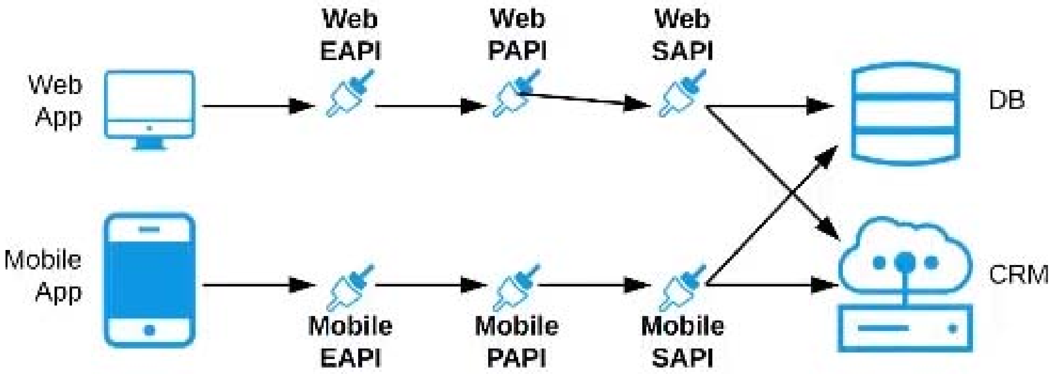
NEW QUESTION 44
Refer to the exhibit. An organization needs to enable access to their customer data from both a mobile app and a web application, which each need access to common fields as well as certain unique fields.
The data is available partially in a database and partially in a 3rd-party CRM system.
What APIs should be created to best fit these design requirements?



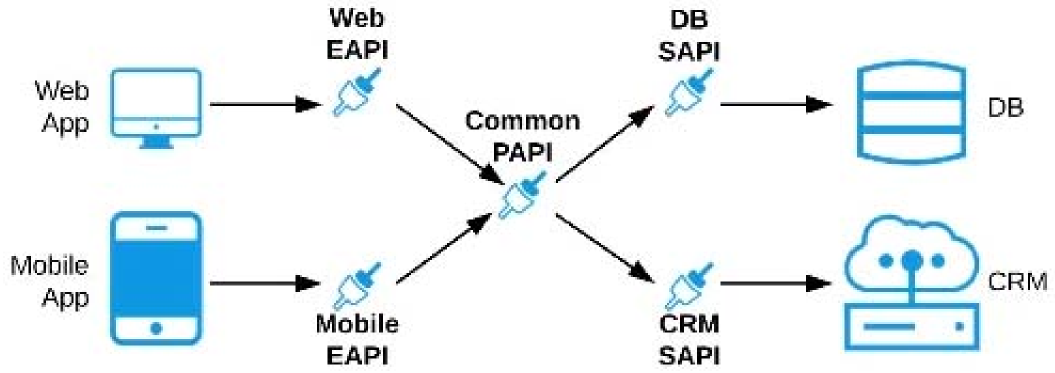
A) A Process API that contains the data required by both the web and mobile apps, allowing these applications to invoke it directly and access the data they need thereby providing the flexibility to add more fields in the future without needing API changes



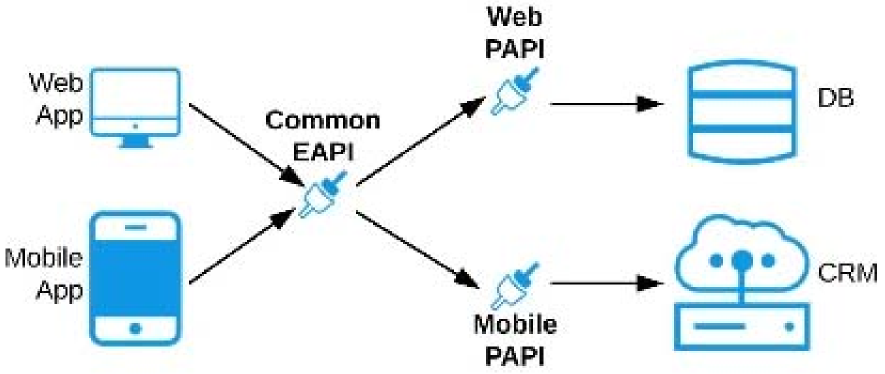
B) One set of APIs (Experience API, Process API, and System API) for the web app, and another set for the mobile app



C) Separate Experience APIs for the mobile and web app, but a common Process API that invokes separate System APIs created for the database and CRM system



D) A common Experience API used by both the web and mobile apps, but separate Process APIs for the web and mobile apps that interact with the database and the CRM System



A. Option A
B. Option B
C. Option C
D. Option D

Answer: C

Explanation:

Correct Answer
Separate Experience APIs for the mobile and web app, but a common Process API that invokes separate System APIs created for the database and CRM system
*** As per MuleSoft's API-led connectivity:
>> Experience APIs should be built as per each consumer needs and their experience.
>> Process APIs should contain all the orchestration logic to achieve the business functionality.
>> System APIs should be built for each backend system to unlock their data.

NEW QUESTION 49
Once an API Implementation is ready and the API is registered on API Manager, who should request the access to the API on Anypoint Exchange?

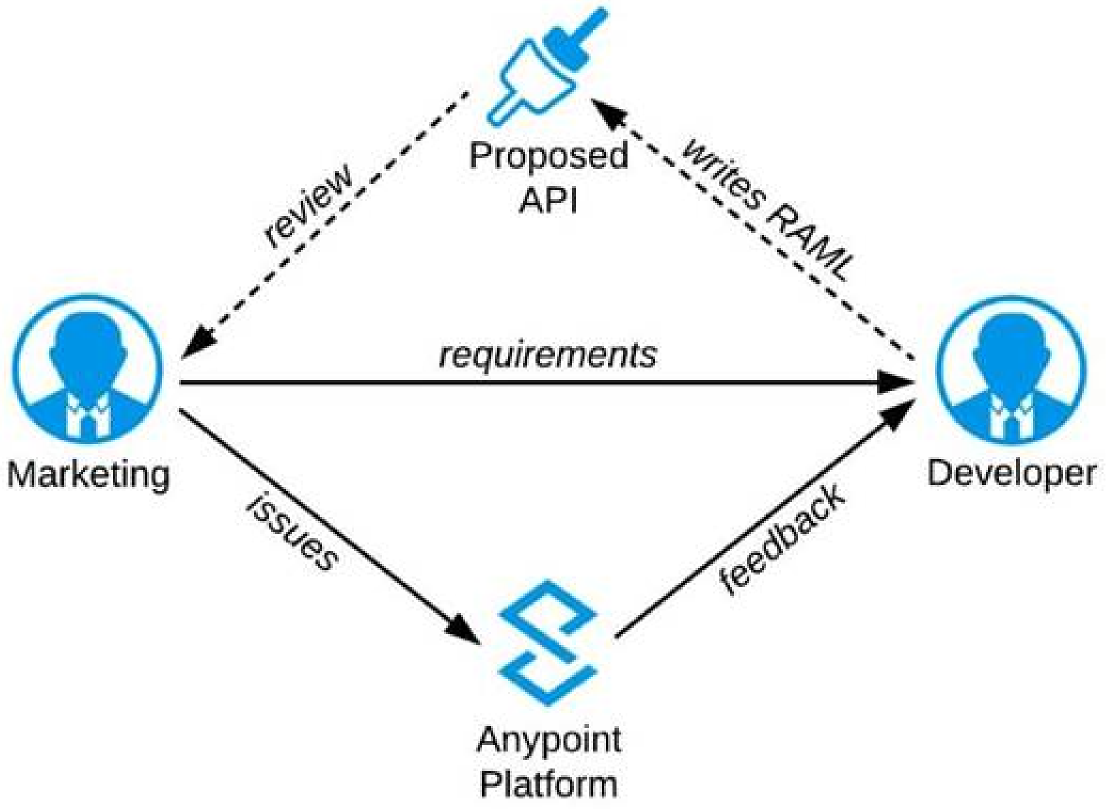
A. None
B. Both
C. API Client
D. API Consumer

Answer: D

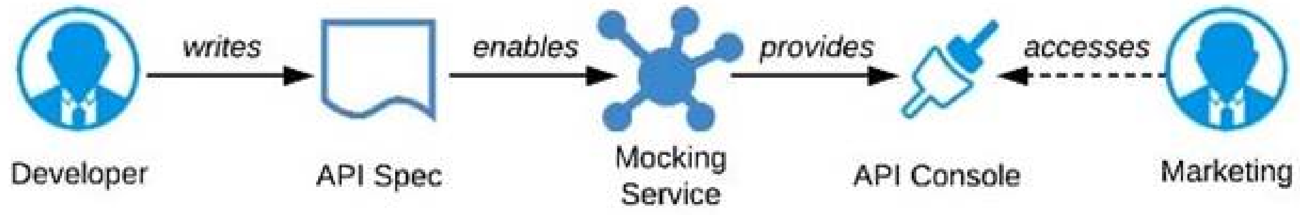
Explanation:

Correct Answer
API Consumer
****
>> API clients are piece of code or programs that use the client credentials of API consumer but does not directly interact with Anypoint Exchange to get the access
>> API consumer is the one who should get registered and request access to API and then API client needs to use those client credentials to hit the APIs
So, API consumer is the one who needs to request access on the API from Anypoint Exchange

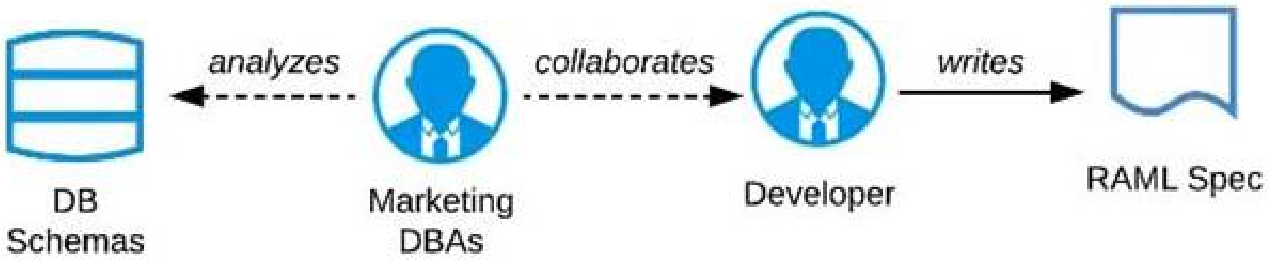
NEW QUESTION 53
Refer to the exhibit.



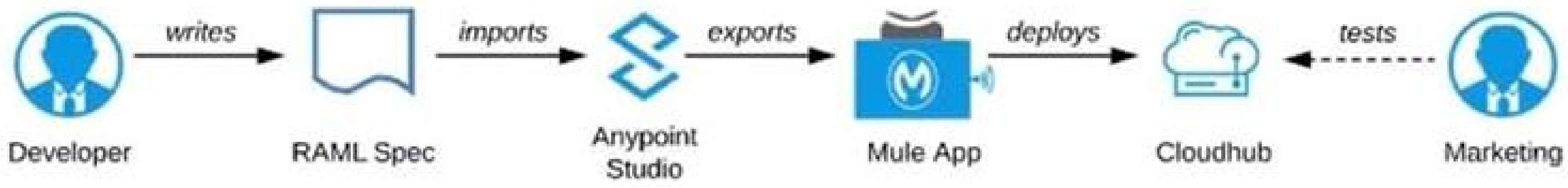
A RAML definition has been proposed for a new Promotions Process API, and has been published to
Anypoint Exchange.
The Marketing Department, who will be an important consumer of the Promotions API, has important requirements and expectations that must be met.
What is the most effective way to use Anypoint Platform features to involve the Marketing Department in this early API design phase?
A) Ask the Marketing Department to interact with a mocking implementation of the API using the automatically generated API Console



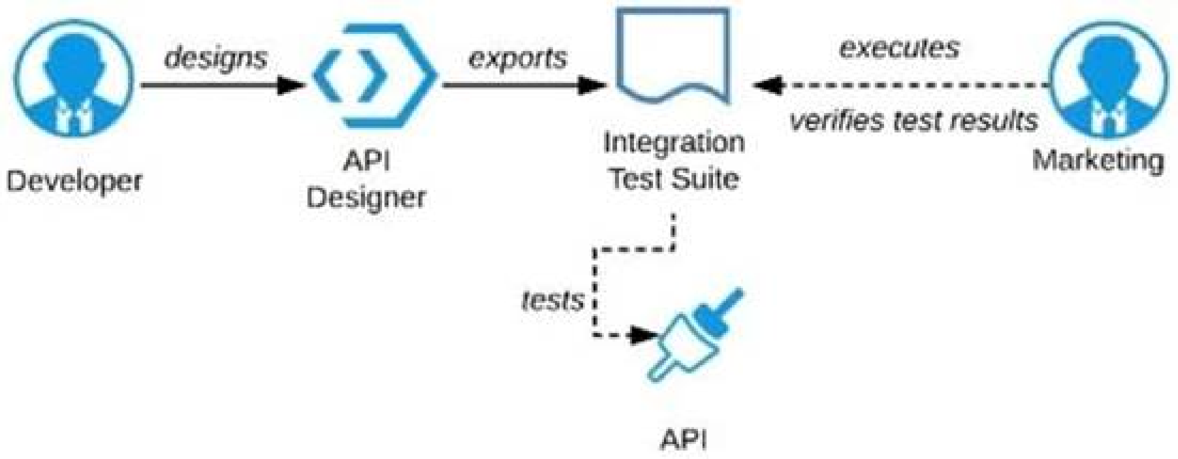
B) Organize a design workshop with the DBAs of the Marketing Department in which the database schema of the Marketing IT systems is translated into RAML



C) Use Anypoint Studio to Implement the API as a Mule application, then deploy that API implementation to CloudHub and ask the Marketing Department to interact with it



D) Export an integration test suite from API designer and have the Marketing Department execute the tests In that suite to ensure they pass



A. Option A
B. Option B
C. Option C
D. Option D

Answer: A

Explanation:
Correct Answer
Ask the Marketing Department to interact with a mocking implementation of the API using the automatically generated API Console.
** As per MuleSoft's IT Operating Model:
>> API consumers need NOT wait until the full API implementation is ready.
>> NO technical test-suites needs to be shared with end users to interact with APIs.

>> Anypoint Platform offers a mocking capability on all the published API specifications to Anypoint Exchange which also will be rich in documentation covering all details of API functionalities and working nature.
>> No needs of arranging days of workshops with end users for feedback.
API consumers can use Anypoint Exchange features on the platform and interact with the API using its mocking feature. The feedback can be shared quickly on the same to incorporate any changes.

NEW QUESTION 58
A REST API is being designed to implement a Mule application.
What standard interface definition language can be used to define REST APIs?

A. Web Service Definition Language(WSDL)
B. OpenAPI Specification (OAS)
C. YAML
D. AsyncAPI Specification

Answer: B

NEW QUESTION 59
Version 3.0.1 of a REST API implementation represents time values in PST time using ISO 8601 hh:mm:ss format. The API implementation needs to be changed to instead represent time values in CEST time using ISO 8601 hh:mm:ss format. When following the semver.org semantic versioning specification, what version should be assigned to the updated API implementation?

A. 3.0.2
B. 4.0.0
C. 3.1.0
D. 3.0.1

Answer: B

Explanation:

Correct Answer 4.0.0
*** As per semver.org semantic versioning specification:
Given a version number MAJOR.MINOR.PATCH, increment the:
- MAJOR version when you make incompatible API changes.
- MINOR version when you add functionality in a backwards compatible manner.
- PATCH version when you make backwards compatible bug fixes.
As per the scenario given in the question, the API implementation is completely changing its behavior. Although the format of the time is still being maintained as hh:mm:ss and there is no change in schema w.r.t format, the API will start functioning different after this change as the times are going to come completely different.
Example: Before the change, say, time is going as 09:00:00 representing the PST. Now on, after the change, the same time will go as 18:00:00 as Central European Summer Time is 9 hours ahead of Pacific Time.
>> This may lead to some uncertain behavior on API clients depending on how they are handling the times in the API response. All the API clients need to be informed that the API functionality is going to change and will return in CEST format. So, this considered as a MAJOR change and the version of API for this new change would be 4.0.0

NEW QUESTION 63
What Anypoint Connectors support transactions?

A. Database, JMS, VM
B. Database, 3MS, HTTP
C. Database, JMS, VM, SFTP
D. Database, VM, File

Answer: A

NEW QUESTION 67
A company has started to create an application network and is now planning to implement a Center for Enablement (C4E) organizational model. What key factor would lead the company to decide upon a federated rather than a centralized C4E?

A. When there are a large number of existing common assets shared by development teams
B. When various teams responsible for creating APIs are new to integration and hence need extensive training
C. When development is already organized into several independent initiatives or groups
D. When the majority of the applications in the application network are cloud based

Answer: C

Explanation:
Correct Answer
When development is already organized into several independent initiatives or groups
****
>> It would require lot of process effort in an organization to have a single C4E team coordinating with multiple already organized development teams which are into several independent initiatives. A single C4E works well with different teams having at least a common initiative. So, in this scenario, federated C4E works well instead of centralized C4E.

NEW QUESTION 68
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

Answer: C

NEW QUESTION 71
An organization is deploying their new implementation of the OrderStatus System API to multiple workers in CloudHub. This API fronts the organization's on-premises Order Management System, which is accessed by the API implementation over an IPsec tunnel.
What type of error typically does NOT result in a service outage of the OrderStatus System API?

A. A CloudHub worker fails with an out-of-memory exception
B. API Manager has an extended outage during the initial deployment of the API implementation
C. The AWS region goes offline with a major network failure to the relevant AWS data centers
D. The Order Management System is Inaccessible due to a network outage in the organization's on-premises data center

Answer: A

Explanation:

Correct Answer
A CloudHub worker fails with an out-of-memory exception.
****
>> An AWS Region itself going down will definitely result in an outage as it does not matter how many workers are assigned to the Mule App as all of those in that region will go down. This is a complete downtime and outage.
>> Extended outage of API manager during initial deployment of API implementation will of course cause issues in proper application startup itself as the API Autodiscovery might fail or API policy templates and polices may not be downloaded to embed at the time of applicaiton startup etc... there are many reasons that could cause issues.
>> A network outage onpremises would of course cause the Order Management System not accessible and it does not matter how many workers are assigned to the app they all will fail and cause outage for sure.
The only option that does NOT result in a service outage is if a cloudhub worker fails with an out-of-memory exception. Even if a worker fails and goes down, there are still other workers to handle the requests and keep the API UP and Running. So, this is the right answer.

NEW QUESTION 73
A company wants to move its Mule API implementations into production as quickly as possible. To protect access to all Mule application data and metadata, the company requires that all Mule applications be deployed to the company's customer-hosted infrastructure within the corporate firewall. What combination of runtime plane and control plane options meets these project lifecycle goals?

A. Manually provisioned customer-hosted runtime plane and customer-hosted control plane
B. MuleSoft-hosted runtime plane and customer-hosted control plane
C. Manually provisioned customer-hosted runtime plane and MuleSoft-hosted control plane
D. iPaaS provisioned customer-hosted runtime plane and MuleSoft-hosted control plane

Answer: A

Explanation:
Correct Answer
Manually provisioned customer-hosted runtime plane and customer-hosted control plane
****
There are two key factors that are to be taken into consideration from the scenario given in the question.
>> Company requires both data and metadata to be resided within the corporate firewall
>> Company would like to go with customer-hosted infrastructure.
Any deployment model that is to deal with the cloud directly or indirectly (Mulesoft-hosted or Customer's own cloud like Azure, AWS) will have to share atleast the metadata.
Application data can be controlled inside firewall by having Mule Runtimes on customer hosted runtime plane. But if we go with Mulsoft-hosted/ Cloud-based control plane, the control plane required atleast some minimum level of metadata to be sent outside the corporate firewall.
As the customer requirement is pretty clear about the data and metadata both to be within the corporate firewall, even though customer wants to move to production as quickly as possible, unfortunately due to the nature of their security requirements, they have no other option but to go with manually provisioned customer-hosted runtime plane and customer-hosted control plane.

NEW QUESTION 77
An organization makes a strategic decision to move towards an IT operating model that emphasizes consumption of reusable IT assets using modern APIs (as defined by MuleSoft).
What best describes each modern API in relation to this new IT operating model?

A. Each modern API has its own software development lifecycle, which reduces the need for documentation and automation
B. Each modem API must be treated like a product and designed for a particular target audience (for instance, mobile app developers)
C. Each modern API must be easy to consume, so should avoid complex authentication mechanisms such as SAML or JWT D
D. Each modern API must be REST and HTTP based

Answer: B

Explanation:

Correct Answers
* 1. Each modern API must be treated like a product and designed for a particular target audience (for instance mobile app developers)
****

Bottom of Form Top of Form

NEW QUESTION 82
A company requires Mule applications deployed to CloudHub to be isolated between non-production and production environments. This is so Mule applications deployed to non-production environments can only access backend systems running in their customer-hosted non-production environment, and so Mule applications deployed to production environments can only access backend systems running in their customer-hosted production environment. How does MuleSoft recommend modifying Mule applications,
configuring environments, or changing infrastructure to support this type of per-environment isolation between Mule applications and backend systems?

A. Modify properties of Mule applications deployed to the production Anypoint Platform environments to prevent access from non-production Mule applications
B. Configure firewall rules in the infrastructure inside each customer-hosted environment so that only IP addresses from the corresponding Anypoint Platform environments are allowed to communicate with corresponding backend systems
C. Create non-production and production environments in different Anypoint Platform business groups
D. Create separate Anypoint VPCs for non-production and production environments, then configure connections to the backend systems in the corresponding customer-hosted environments

Answer: D

Explanation:
Correct Answer
Create separate Anypoint VPCs for non-production and production environments, then configure connections to the backend systems in the corresponding customer-hosted environments.
****
>> Creating different Business Groups does NOT make any difference w.r.t accessing the non-prod and prod customer-hosted environments. Still they will be accessing from both Business Groups unless process network restrictions are put in place.
>> We need to modify or couple the Mule Application Implementations with the environment. In fact, we should never implements application coupled with environments by binding them in the properties. Only basic things like endpoint URL etc should be bundled in properties but not environment level access restrictions.
>> IP addresses on CloudHub are dynamic until unless a special static addresses are assigned. So it is not possible to setup firewall rules in customer-hosted infrastrcture. More over, even if static IP addresses are assigned, there could be 100s of applications running on cloudhub and setting up rules for all of them would be a hectic task, non-maintainable and definitely got a good practice.
>> Thbeest practice recommended
by MulesoftIn(fact any cloud provider), is to have your Anypoint VPCs
seperated for Prod and Non-Prod and perform the VPC peering or VPN tunneling for these Anypoint VPCs to respective Prod and Non-Prod customer-hosted environment networks.

NEW QUESTION 87
The implementation of a Process API must change.
What is a valid approach that minimizes the impact of this change on API clients?

A. Update the RAML definition of the current Process API and notify API client developers by sending them links to the updated RAML definition
B. Postpone changes until API consumers acknowledge they are ready to migrate to a new Process API or API version
C. Implement required changes to the Process API implementation so that whenever possible, the Process API's RAML definition remains unchanged
D. Implement the Process API changes in a new API implementation, and have the old API implementation return an HTTP status code 301 - Moved Permanently to inform API clients they should be calling the new API implementation

Answer: C

Explanation:
Correct Answer
Implement required changes to the Process API implementation so that, whenever possible, the Process API's RAML definition remains unchanged.
*** Key requirement in the question is:
>> Approach that minimizes the impact of this change on API clients Based on above:
>> Updating the RAML definition would possibly impact the API clients if the changes require any thing mandatory from client side. So, one should try to avoid doing that until really necessary.
>> Implementing the changes as a completely different API and then redirectly the clients with 3xx status code is really upsetting design and heavily impacts the API clients.
>> Organisations and IT cannot simply postpone the changes required until all API consumers acknowledge they are ready to migrate to a new Process API or API version. This is unrealistic and not possible.
The best way to handle the changes always is to implement required changes to the API implementations so that, whenever possible, the API's RAML definition remains unchanged.

NEW QUESTION 90
An organization has implemented a Customer Address API to retrieve customer address information. This API has been deployed to multiple environments and has been configured to enforce client IDs everywhere.
A developer is writing a client application to allow a user to update their address. The developer has found the Customer Address API in Anypoint Exchange and wants to use it in their client application.
What step of gaining access to the API can be performed automatically by Anypoint Platform?

A. Approve the client application request for the chosen SLA tier
B. Request access to the appropriate API Instances deployed to multiple environments using the client application's credentials
C. Modify the client application to call the API using the client application's credentials
D. Create a new application in Anypoint Exchange for requesting access to the API

Answer: A

Explanation:

Correct Answer
Approve the client application request for the chosen SLA tier

****
>> Only approving the client application request for the chosen SLA tier can be automated
>> Rest of the provided options are not valid

NEW QUESTION 91
An organization has created an API-led architecture that uses various API layers to integrate mobile clients with a backend system. The backend system consists of a number of specialized components and can be accessed via a REST API. The process and experience APIs share the same bounded-context model that is different from the backend data model. What additional canonical models, bounded-context models, or anti-corruption layers are best added to this architecture to help process data consumed from the backend system?

A. Create a bounded-context model for every layer and overlap them when the boundary contexts overlap, letting API developers know about the differences between upstream and downstream data models
B. Create a canonical model that combines the backend and API-led models to simplify and unify data models, and minimize data transformations.
C. Create a bounded-context model for the system layer to closely match the backend data model, and add an anti-corruption layer to let the different bounded contexts cooperate across the system and process layers
D. Create an anti-corruption layer for every API to perform transformation for every data model to match each other, and let data simply travel between APIs to avoid the complexity and overhead of building canonical models

Answer: C

Explanation:
Correct Answer
Create a bounded-context model for the system layer to closely match the backend data model, and add an anti-corruption layer to let the different bounded contexts cooperate across the system and process layers
****
>> Canonical models are not an option here as the organization has already put in efforts and created bounded-context models for Experience and Process APIs.
>> Anti-corruption layers for ALL APIs is unnecessary and invalid because it is mentioned that experience and process APIs share same bounded-context model. It is just the System layer APIs that need to choose their approach now.
>> So, having an anti-corruption layer just between the process and system layers will work well. Also to speed up the approach, system APIs can mimic the backend system data model.

NEW QUESTION 94
......