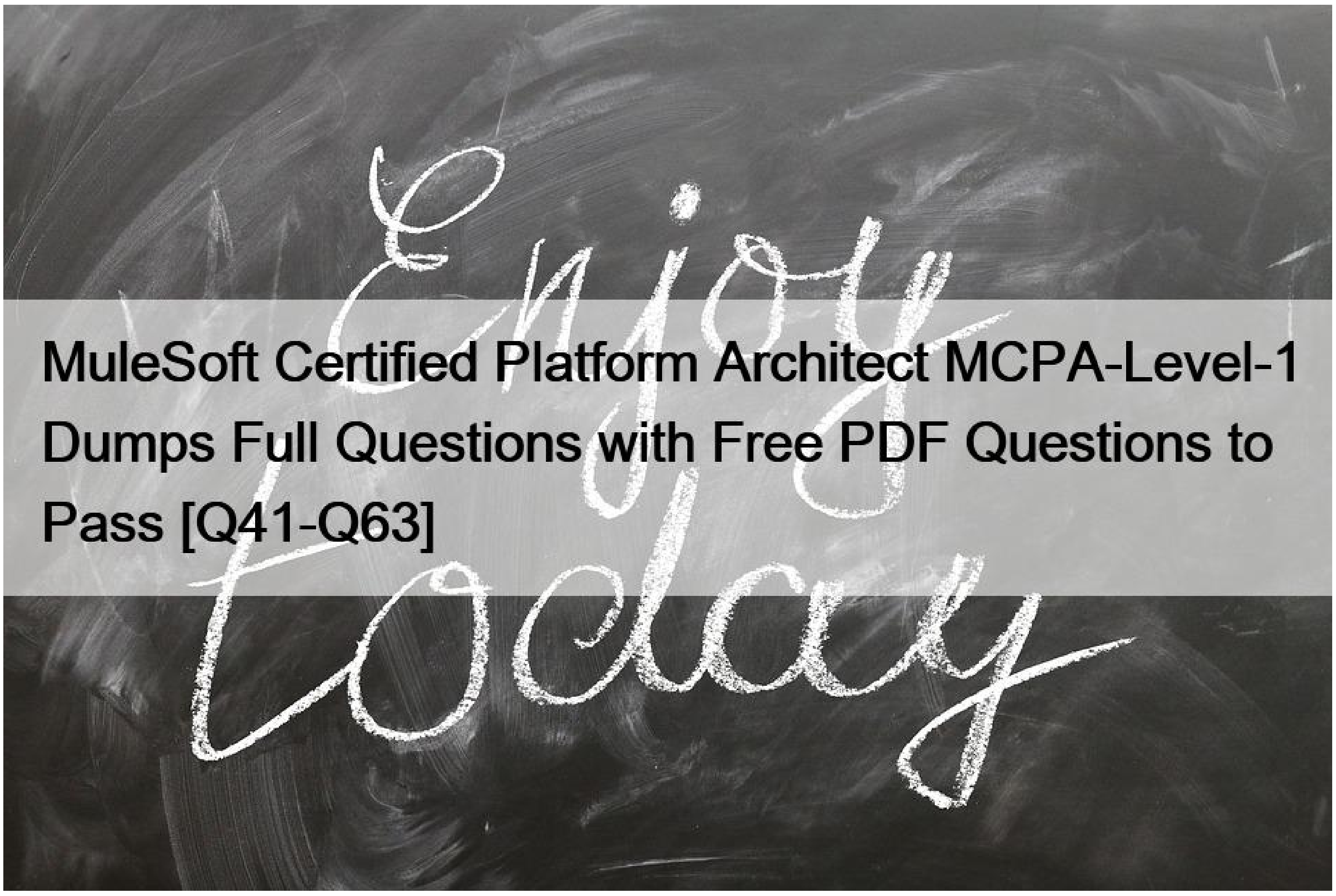
MuleSoft Certified Platform Architect MCPA-Level-1 Dumps Full Questions with Free PDF Questions to Pass [Q41-Q63


Enjoy today
MuleSoft Certified Platform Architect MCPA-Level-1 Dumps Full Questions with Free PDF Questions to Pass [Q41-Q63]

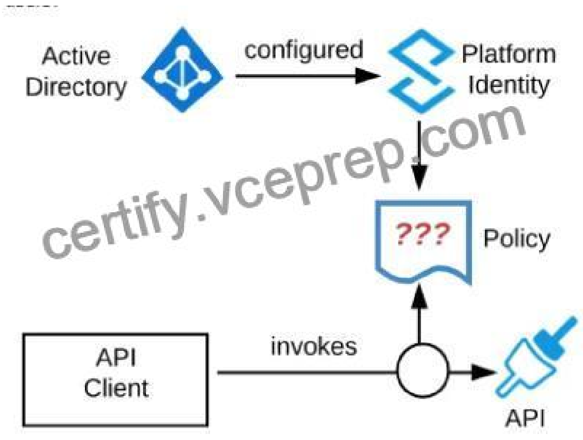
MuleSoft Certified Platform Architect MCPA-Level-1 Dumps Full Questions with Free PDF Questions to Pass
100% Updated MuleSoft MCPA-Level-1 Enterprise PDF Dumps

NEW QUESTION 41

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 fixed number of FQDNs are created, IRRESPECTIVE of the environment and VPC design
* The FQDNs are determined by the application name chosen, IRRESPECTIVE of the region
* The FQDNs are determined by the application name, but can be modified by an administrator after deployment
* The FQDNs are determined by both the application name and the Anypoint Platform organization

NEW QUESTION 42

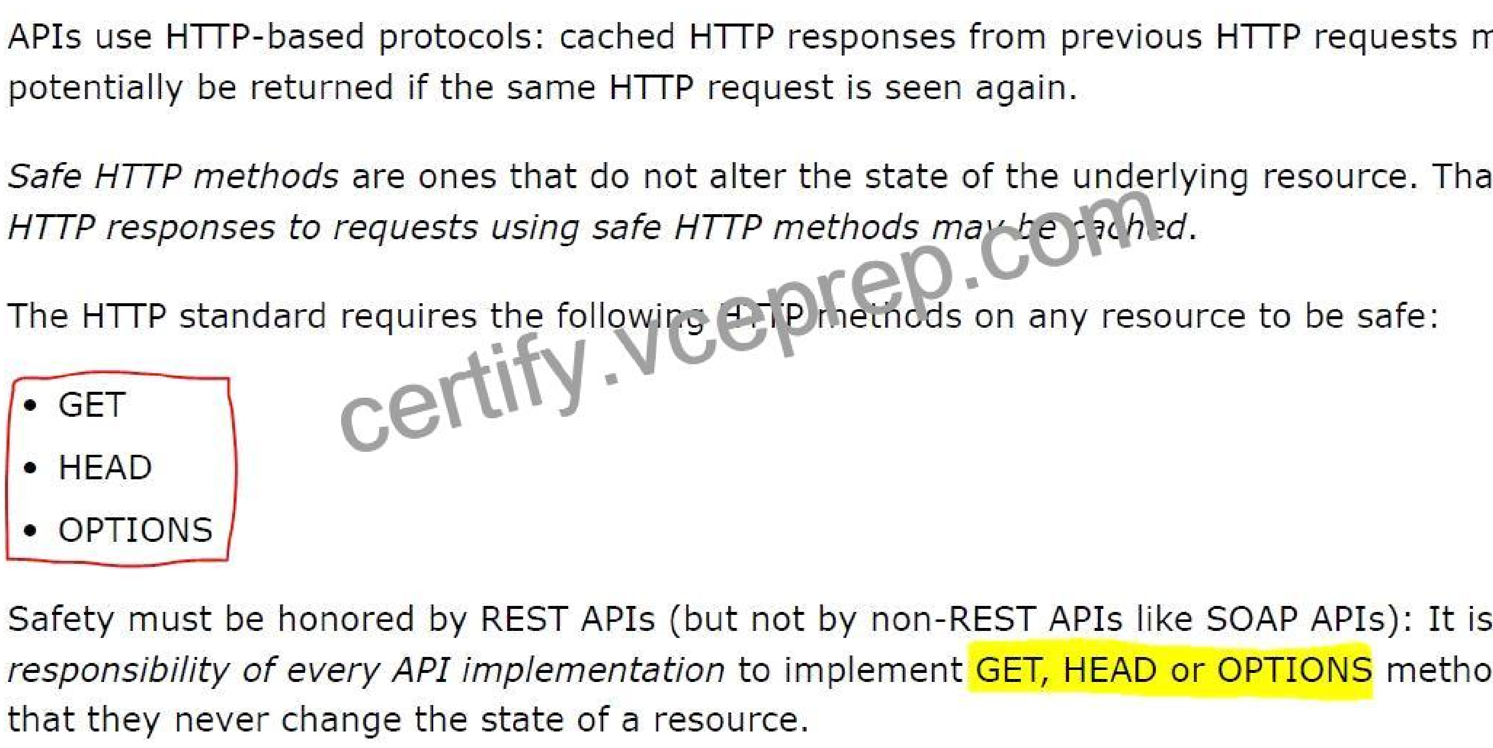
Refer to the exhibit. An organization is running a Mule standalone runtime and has configured Active Directory as the Anypoint Platform external Identity Provider. The organization does not have budget for other system components.



What policy should be applied to all instances of APIs in the organization to most effecuvelyKestrict access to a specific group of internal users?
* Apply a basic authentication &#8211; LDAP policy; the internal Active Directory will be configured as the LDAP source for authenticating users
* Apply a client ID enforcement policy; the specific group of users will configure their client applications to use their specific client credentials
* Apply an IP whitelist policy; only the specific users&#8217; workstations will be in the whitelist
* Apply an OAuth 2.0 access token enforcement policy; the internal Active Directory will be configured as the OAuth server

NEW QUESTION 43

The responses to some HTTP requests can be cached depending on the HTTP verb used in the request. According to the HTTP specification, for what HTTP verbs is this safe to do?
* PUT, POST, DELETE
* GET, HEAD, POST
* GET, PUT, OPTIONS
* GET, OPTIONS, HEAD
Correct answer: GET, OPTIONS, HEAD



APIs use HTTP-based protocols: cached HTTP responses from previous HTTP requests m
potentially be returned if the same HTTP request is seen again.

Safe HTTP methods are ones that do not alter the state of the underlying resource. Tha
HTTP responses to requests using safe HTTP methods may be cached.

The HTTP standard requires the following HTTP methods on any resource to be safe:

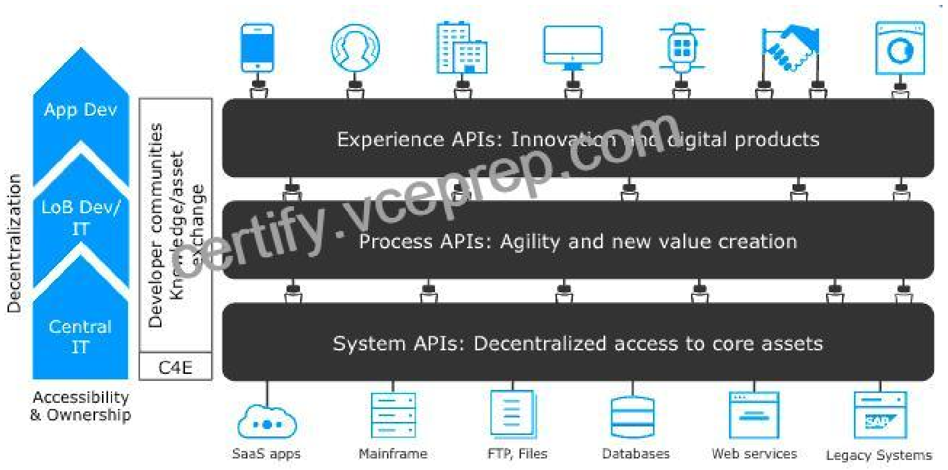
- GET
- HEAD
- OPTIONS

Safety must be honored by REST APIs (but not by non-REST APIs like SOAP APIs): It is
responsibility of every API implementation to implement GET, HEAD or OPTIONS metho
that they never change the state of a resource.

http://restcookbook.com/HTTP%20Methods/idempotency/

NEW QUESTION 44

Which layer in the API-led connectivity focuses on unlocking key systems, legacy systems, data sources etc and exposes the functionality?
* Experience Layer
* Process Layer
* System Layer
Correct answer: System Layer


App Dev
LoB Dev/ IT
Central IT
Decentralization
Accessibility & Ownership
Developer communities
Knowledge/asset exchange
C4E
Experience APIs: Innovation and digital products
Process APIs: Agility and new value creation
System APIs: Decentralized access to core assets
SaaS apps
Mainframe
FTP, Files
Databases
Web services
Legacy Systems
SAP

The APIs used in an API-led approach to connectivity fall into three categories:

System APIs &#8211; these usually access the core systems of record and provide a means of insulating the user from the complexity or any changes to the underlying systems. Once built, many users, can access data without any need to learn the underlying systems and can reuse these APIs in multiple projects.

Process APIs &#8211; These APIs interact with and shape data within a single system or across systems (breaking down data silos) and are created here without a dependence on the source systems from which that data originates, as well as the target channels through which that data is delivered.

Experience APIs &#8211; Experience APIs are the means by which data can be reconfigured so that it is most easily consumed by its intended audience, all from a common data source, rather than setting up separate point-to-point integrations for each channel. An Experience API is usually created with API-first design principles where the API is designed for the specific user experience in mind.

NEW QUESTION 45

A system API is deployed to a primary environment as well as to a disaster recovery (DR) environment, with different DNS names in each environment. A process API is a client to the system API and is being rate limited by the system API, with different limits in each of the environments. The system API&#8217;s DR environment provides only 20% of the rate limiting offered by the primary environment. What is the best API fault-tolerant invocation strategy to reduce overall errors in the process API, given these conditions and constraints?
* Invoke the system API deployed to the primary environment; add timeout and retry logic to the process API to avoid intermittent failures; if it still fails, invoke the system API deployed to the DR environment
* Invoke the system API deployed to the primary environment; add retry logic to the process API to handle intermittent failures by invoking the system API deployed to the DR environment
* In parallel, invoke the system API deployed to the primary environment and the system API deployed to the DR environment; add timeout and retry logic to the process API to avoid intermittent failures; add logic to the process API to combine the results
* Invoke the system API deployed to the primary environment; add timeout and retry logic to the process API to avoid intermittent failures; if it still fails, invoke a copy of the process API deployed to the DR environment
Correct answer: Invoke the system API deployed to the primary environment; add timeout and retry logic to the process API to avoid intermittent failures; if it still fails, invoke the system API deployed to the DR environment

****

There is one important consideration to be noted in the question which is &#8211; System API in DR environment provides only 20% of the rate limiting offered by the primary environment. So, comparitively, very less calls will be allowed into the DR environment API opposed to its primary environment. With this in mind, lets analyse what is the right and best fault-tolerant invocation strategy.

1. Invoking both the system APIs in parallel is definitely NOT a feasible approach because of the 20% limitation we have on DR environment. Calling in parallel every time would easily and quickly exhaust the rate limits on DR environment and may not give chance to genuine intermittent error scenarios to let in during the time of need.

2. Another option given is suggesting to add timeout and retry logic to process API while invoking primary environment&#8217;s system API. This is good so far. However, when all retries failed, the option is suggesting to invoke the copy of process API on DR environment which is not right or recommended. Only system API is the one to be considered for fallback and not the whole process API. Process APIs usually have lot of heavy orchestration calling many other APIs which we do not want to repeat again by calling DR&#8217;s process API. So this option is NOT right.

3. One more option given is suggesting to add the retry (no timeout) logic to process API to directly retry on DR environment&#8217;s system API instead of retrying the primary environment system API first. This is not at all a proper fallback. A proper fallback should occur only after all retries are performed and exhausted on Primary environment first. But here, the option is suggesting to directly retry fallback API on first failure itself without trying main API. So, this option is NOT right too.

This leaves us one option which is right and best fit.

&#8211; Invoke the system API deployed to the primary environment

&#8211; Add Timeout and Retry logic on it in process API

&#8211; If it fails even after all retries, then invoke the system API deployed to the DR environment.

NEW QUESTION 46

An organization wants to make sure only known partners can invoke the organization&#8217;s APIs. To achieve this security goal, the organization wants to enforce a Client ID Enforcement policy in API Manager so that only registered partner applications can invoke the organization&#8217;s APIs. In what type of API implementation does MuleSoft recommend adding an API proxy to enforce the Client ID Enforcement policy, rather than embedding the policy directly in the application&#8217;s JVM?
* A Mule 3 application using APIkit
* A Mule 3 or Mule 4 application modified with custom Java code
* A Mule 4 application with an API specification
* A Non-Mule application
Correct answer: A Non-Mule application

****

>> All type of Mule applications (Mule 3/ Mule 4/ with APIkit/ with Custom Java Code etc) running on Mule Runtimes support the Embedded Policy Enforcement on them.

>> The only option that cannot have or does not support embedded policy enforcement and must have API Proxy is for Non-Mule Applications.

So, Non-Mule application is the right answer.

NEW QUESTION 47

An API experiences a high rate of client requests (TPS) vwth small message paytoads. How can usage limits be imposed on the API based on the type of client application?
* Use an SLA-based rate limiting policy and assign a client application to a matching SLA tier based on its type
* Use a spike control policy that limits the number of requests for each client application type
* Use a cross-origin resource sharing (CORS) policy to limit resource sharing between client applications, configured by the client application type
* Use a rate limiting policy and a client ID enforcement policy, each configured by the client application type
Correct answer: Use an SLA-based rate limiting policy and assign a client application to a matching SLA tier based on its type.

****

>> SLA tiers will come into play whenever any limits to be imposed on APIs based on client type

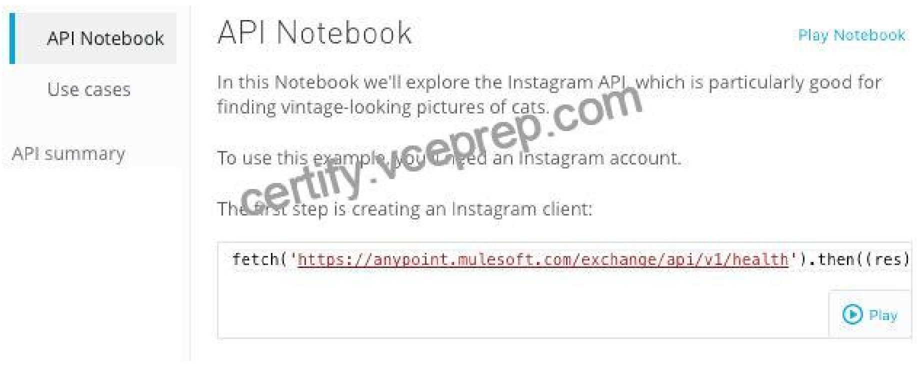
NEW QUESTION 48

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



API Notebook

In this Notebook we'll explore the Instagram API, which is particularly good for finding vintage-looking pictures of cats.

To use this example, you'll need an Instagram account.

The first step is creating an Instagram client:

```
fetch('https://anypoint.mulesoft.com/exchange/api/v1/health').then((res)
```

NEW QUESTION 49

An Order API must be designed that contains significant amounts of integration logic and involves the invocation of the Product API.

The power relationship between Order API and Product API is one of &#8220;Customer/Supplier&#8221;, because the Product API is used heavily throughout the organization and is developed by a dedicated development team located in the office of the CTO.

What strategy should be used to deal with the API data model of the Product API within the Order API?
* Convince the development team of the Product API to adopt the API data model of the Order API such that the integration logic of the Order API can work with one consistent internal data model
* Work with the API data types of the Product API directly when implementing the integration logic of the Order API such that the Order API uses the same (unchanged) data types as the Product API
* Implement an anti-corruption layer in the Order API that transforms the Product API data model into internal data types of the Order API
* Start an organization-wide data modeling initiative that will result in an Enterprise Data Model that will then be used in both the Product API and the Order API
Correct answer: Convince the development team of the product API to adopt the API data model of the Order API such that integration logic of the Order API can work with one consistent internal data model

****

Key details to note from the given scenario:

>> Power relationship between Order API and Product API is customer/supplier So, as per below rules of &#8220;Power Relationships&#8221;, the caller (in this case Order API) would request for features to the called (Product API team) and the Product API team would need to accomodate those requests.

NEW QUESTION 50

Mule applications that implement a number of REST APIs are deployed to their own subnet that is inaccessible from outside the organization.

External business-partners need to access these APIs, which are only allowed to be invoked from a separate subnet dedicated to partners &#8211; called Partner-subnet. This subnet is accessible from the public internet, which allows these external partners to reach it.

Anypoint Platform and Mule runtimes are already deployed in Partner-subnet. These Mule runtimes can already access the APIs.

What is the most resource-efficient solution to comply with these requirements, while having the least impact on other applications that are currently using the APIs?
* Implement (or generate) an API proxy Mule application for each of the APIs, then deploy the API proxies to the Mule runtimes
* Redeploy the API implementations to the same servers running the Mule runtimes
* Add an additional endpoint to each API for partner-enablement consumption
* Duplicate the APIs as Mule applications, then deploy them to the Mule runtimes

NEW QUESTION 51

What best explains the use of auto-discovery in API implementations?
* It makes API Manager aware of API implementations and hence enables it to enforce policies
* It enables Anypoint Studio to discover API definitions configured in Anypoint Platform
* It enables Anypoint Exchange to discover assets and makes them available for reuse
* It enables Anypoint Analytics to gain insight into the usage of APIs
Explanation

https://docs.mulesoft.com/api-manager/2.x/api-auto-discovery-new-concept

NEW QUESTION 52

In an organization, the InfoSec team is investigating Anypoint Platform related data traffic.

From where does most of the data available to Anypoint Platform for monitoring and alerting originate?
* From the Mule runtime or the API implementation, depending on the deployment model
* From various components of Anypoint Platform, such as the Shared Load Balancer, VPC, and Mule runtimes
* From the Mule runtime or the API Manager, depending on the type of data
* From the Mule runtime irrespective of the deployment model
Correct answer: From the Mule runtime irrespective of the deployment model

****

>> Monitoring and Alerting metrics are always originated from Mule Runtimes irrespective of the deployment model.

>> It may seems that some metrics (Runtime Manager) are originated from Mule Runtime and some are (API Invocations/ API Analytics) from API Manager. However, this is realistically NOT TRUE. The reason is, API manager is just a management tool for API instances but all policies upon applying on APIs eventually gets executed on Mule Runtimes only (Either Embedded or API Proxy).

>> Similarly all API Implementations also run on Mule Runtimes.

So, most of the day required for monitoring and alerts are originated fron Mule Runtimes only irrespective of whether the deployment model is MuleSoft-hosted or Customer-hosted or Hybrid.

NEW QUESTION 53

What do the API invocation metrics provided by Anypoint Platform provide?
* ROI metrics from APIs that can be directly shared with business users
* Measurements of the effectiveness of the application network based on the level of reuse
* Data on past API invocations to help identify anomalies and usage patterns across various APIs
* Proactive identification of likely future policy violations that exceed a given threat threshold
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&#8230;

>> 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 54

How are an API implementation, API client, and API consumer combined to invoke and process an API?
* The API consumer creates an API implementation, which receives API invocations from an API such that they are processed for an API client
* The API client creates an API consumer, which receives API invocations from an API such that they are processed for an API implementation
* The ApI consumer creates an API client, which sends API invocations to an API such that they are processed by an API implementation
* The ApI client creates an API consumer, which sends API invocations to an API such that they are processed by an API implementation

NEW QUESTION 55

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?

* Implement a heartbeat/health check within the API and invoke it from outside the Anypoint Platform and alert when the heartbeat does not respond
* Configure a &#8220;worker not responding&#8221; alert in Anypoint Runtime Manager
* Handle API invocation exceptions within the calling API client and raise an alert from that API client when the API Is unavailable
* Create an alert for when the API receives no requests within a specified time period

NEW QUESTION 56

Mule applications that implement a number of REST APIs are deployed to their own subnet that is inaccessible from outside the organization.

External business-partners need to access these APIs, which are only allowed to be invoked from a separate subnet dedicated to partners &#8211; called Partner-subnet. This subnet is accessible from the public internet, which allows these external partners to reach it.

Anypoint Platform and Mule runtimes are already deployed in Partner-subnet. These Mule runtimes can already access the APIs.

What is the most resource-efficient solution to comply with these requirements, while having the least impact on other applications that are currently using the APIs?
* Implement (or generate) an API proxy Mule application for each of the APIs, then deploy the API proxies to the Mule runtimes
* Redeploy the API implementations to the same servers running the Mule runtimes
* Add an additional endpoint to each API for partner-enablement consumption
* Duplicate the APIs as Mule applications, then deploy them to the Mule runtimes

NEW QUESTION 57

An API implementation is being designed that must invoke an Order API, which is known to repeatedly experience downtime.

For this reason, a fallback API is to be called when the Order API is unavailable.

What approach to designing the invocation of the fallback API provides the best resilience?
* Search Anypoint Exchange for a suitable existing fallback API, and then implement invocations to this fallback API in addition to the Order API
* Create a separate entry for the Order API in API Manager, and then invoke this API as a fallback API if the primary Order API is unavailable
* Redirect client requests through an HTTP 307 Temporary Redirect status code to the fallback API whenever the Order API is unavailable
* Set an option in the HTTP Requester component that invokes the Order API to instead invoke a fallback API whenever an HTTP 4xx or 5xx response status code is returned from the Order API

NEW QUESTION 58

When using CloudHub with the Shared Load Balancer, what is managed EXCLUSIVELY by the API implementation (the Mule application) and NOT by Anypoint Platform?
* The assignment of each HTTP request to a particular CloudHub worker
* The logging configuration that enables log entries to be visible in Runtime Manager
* The SSL certificates used by the API implementation to expose HTTPS endpoints
* The number of DNS entries allocated to the API implementation

NEW QUESTION 59

What is true about where an API policy is defined in Anypoint Platform and how it is then applied to API instances?
* The API policy Is defined In Runtime Manager as part of the API deployment to a Mule runtime, and then ONLY applied to the specific API Instance
* The API policy Is defined In API Manager for a specific API Instance, and then ONLY applied to the specific API instance
* The API policy Is defined in API Manager and then automatically applied to ALL API instances
* The API policy is defined in API Manager, and then applied to ALL API instances in the specified environment

NEW QUESTION 60

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?
* Bundle properties files for each environment into the Mule API implementation&#8217;s deployable archive, then promote the Mule API implementation to the Production environment using Anypoint CLI or the Anypoint Platform REST APIsB.
* Modify the Mule API implementation&#8217;s properties in the API Manager Properties tab, then promote the Mule API implementation to the Production environment using API Manager
* Modify the Mule API implementation&#8217;s properties in Anypoint Exchange, then promote the Mule API implementation to the Production environment using Runtime Manager
* 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
Correct answer: Bundle properties files for each environment into the Mule API implementation&#8217;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.

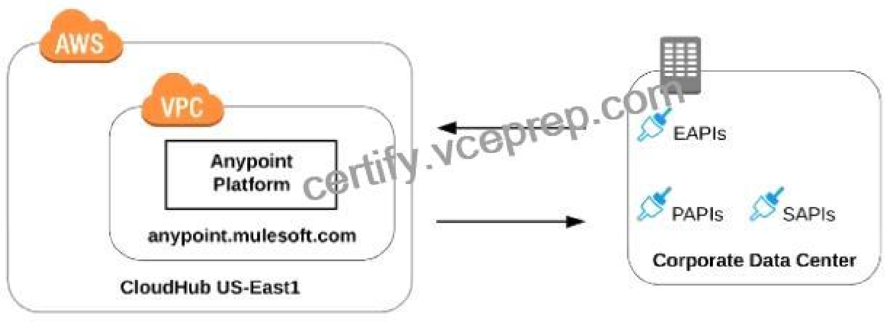
>> API Manager is for managing API instances, their contracts, policies and SLAs. It has also got no provision to modify the properties of API implementations.

>> 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 61

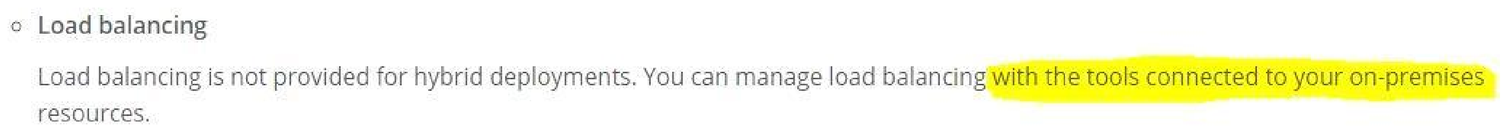
Refer to the exhibit.



what is true when using customer-hosted Mule runtimes with the MuleSoft-hosted Anypoint Platform control plane (hybrid deployment)?
* Anypoint Runtime Manager initiates a network connection to a Mule runtime in order to deploy Mule applications
* The MuleSoft-hosted Shared Load Balancer can be used to load balance API invocations to the Mule runtimes
* API implementations can run successfully in customer-hosted Mule runtimes, even when they are unable to communicate with the control plane
* Anypoint Runtime Manager automatically ensures HA in the control plane by creating a new Mule runtime instance in case of a node failure
Correct answer: API implementations can run successfully in customer-hosted Mule runtimes, even when they are unable to communicate with the control plane.

****

>> We CANNOT use Shared Load balancer to load balance APIs on customer hosted runtimes


- Load balancing

  Load balancing is not provided for hybrid deployments. You can manage load balancing with the tools connected to your on-premises resources.

>> For Hybrid deployment models, the on-premises are first connected to Runtime Manager using Runtime Manager agent. So, the connection is initiated first from On-premises to Runtime Manager. Then all control can be done from Runtime Manager.

>> Anypoint Runtime Manager CANNOT ensure automatic HA. Clusters/Server Groups etc should be configured before hand.

Only TRUE statement in the given choices is, API implementations can run successfully in customer-hosted Mule runtimes, even when they are unable to communicate with the control plane. There are several references below to justify this statement.

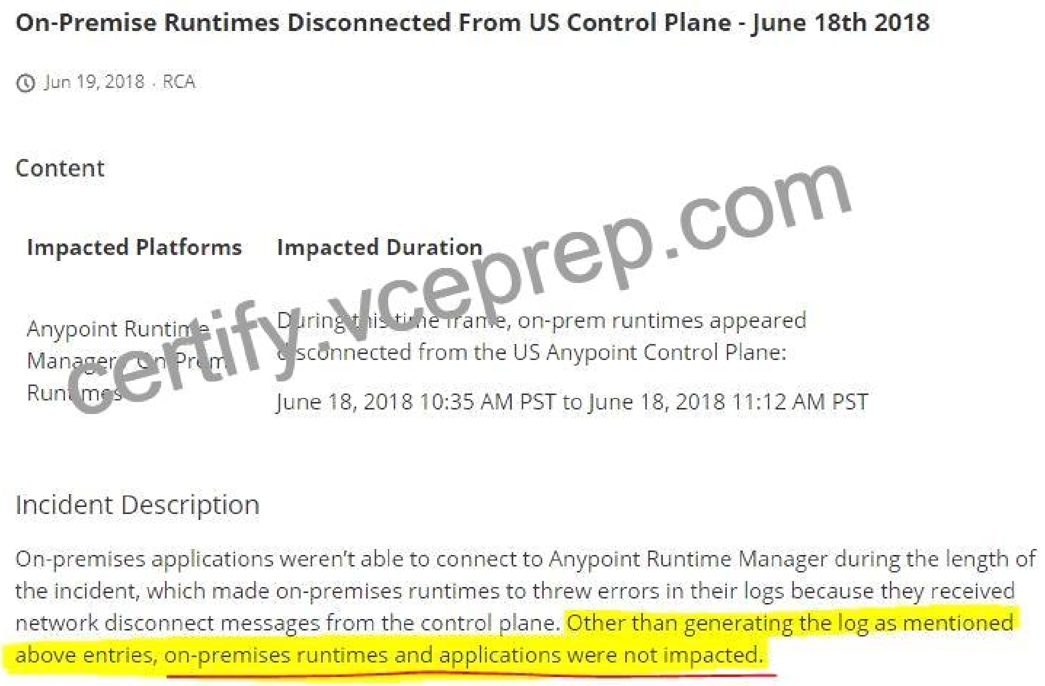
References:

https://docs.mulesoft.com/runtime-manager/deployment-strategies#hybrid-deployments

https://help.mulesoft.com/s/article/On-Premise-Runtimes-Disconnected-From-US-Control-Plane-June-18th-2018

https://help.mulesoft.com/s/article/Runtime-Manager-cannot-manage-On-Prem-Applications-and-Servers-from-US-Control-Plane-June-25th-2019

https://help.mulesoft.com/s/article/On-premise-Runtimes-Appear-Disconnected-in-Runtime-Manager-May-29th-2018



On-Premise Runtimes Disconnected From US Control Plane - June 18th 2018

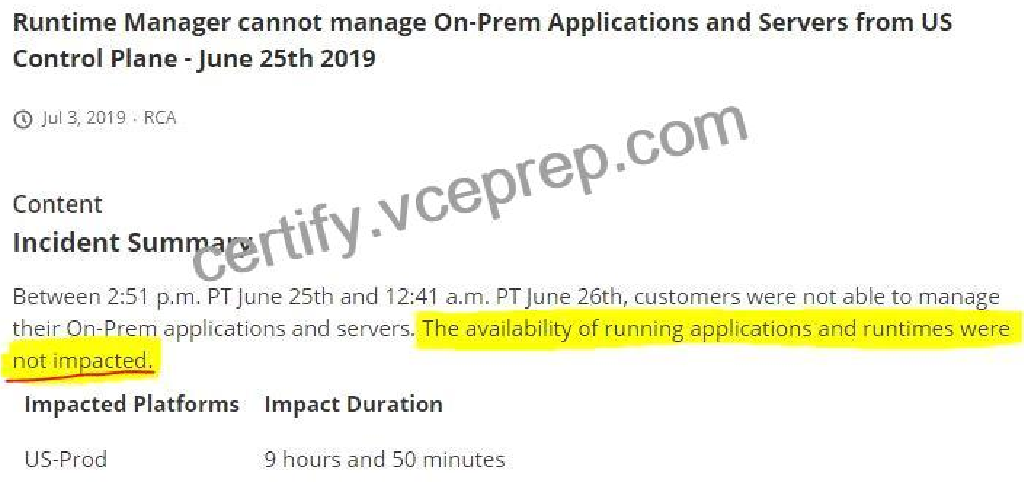
Jun 19, 2018 · RCA

Content

Impacted Platforms	Impacted Duration
Anypoint Runtime Manager - On Prem Runtimes	During this time frame, on-prem runtimes appeared disconnected from the US Anypoint Control Plane: June 18, 2018 10:35 AM PST to June 18, 2018 11:12 AM PST

Incident Description

On-premises applications weren't able to connect to Anypoint Runtime Manager during the length of the incident, which made on-premises runtimes to threw errors in their logs because they received network disconnect messages from the control plane. Other than generating the log as mentioned above entries, on-premises runtimes and applications were not impacted.



Runtime Manager cannot manage On-Prem Applications and Servers from US Control Plane - June 25th 2019

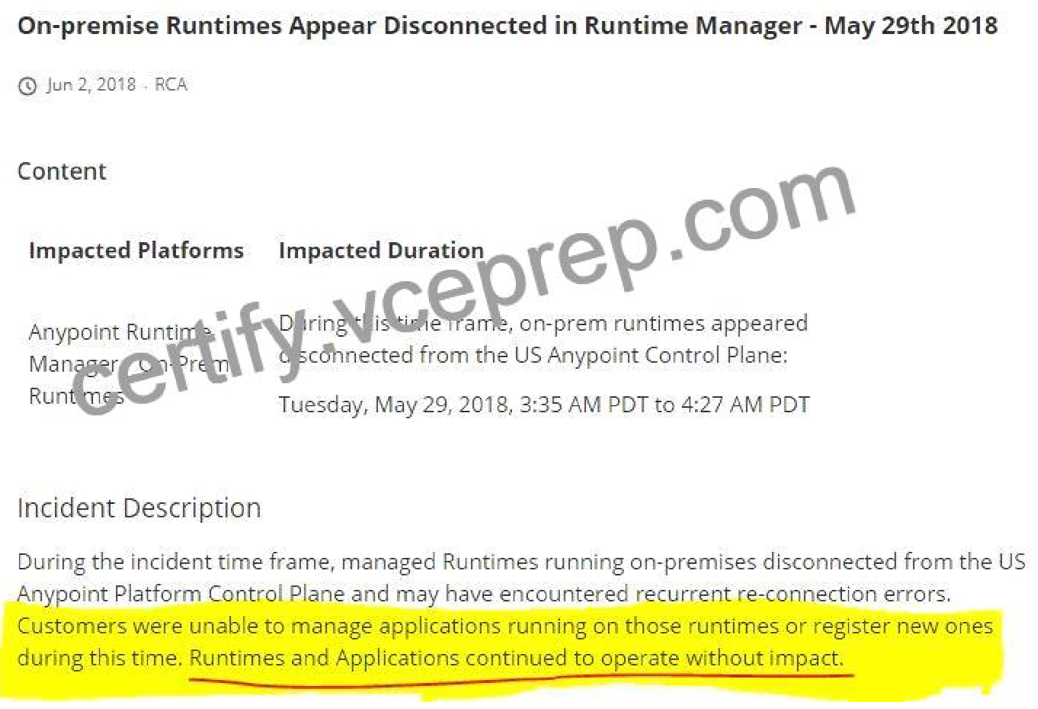
Jul 3, 2019 · RCA

Content

Incident Summary

Between 2:51 p.m. PT June 25th and 12:41 a.m. PT June 26th, customers were not able to manage their On-Prem applications and servers. The availability of running applications and runtimes were not impacted.

Impacted Platforms	Impact Duration
US-Prod	9 hours and 50 minutes



On-premise Runtimes Appear Disconnected in Runtime Manager - May 29th 2018

Jun 2, 2018 · RCA

Content

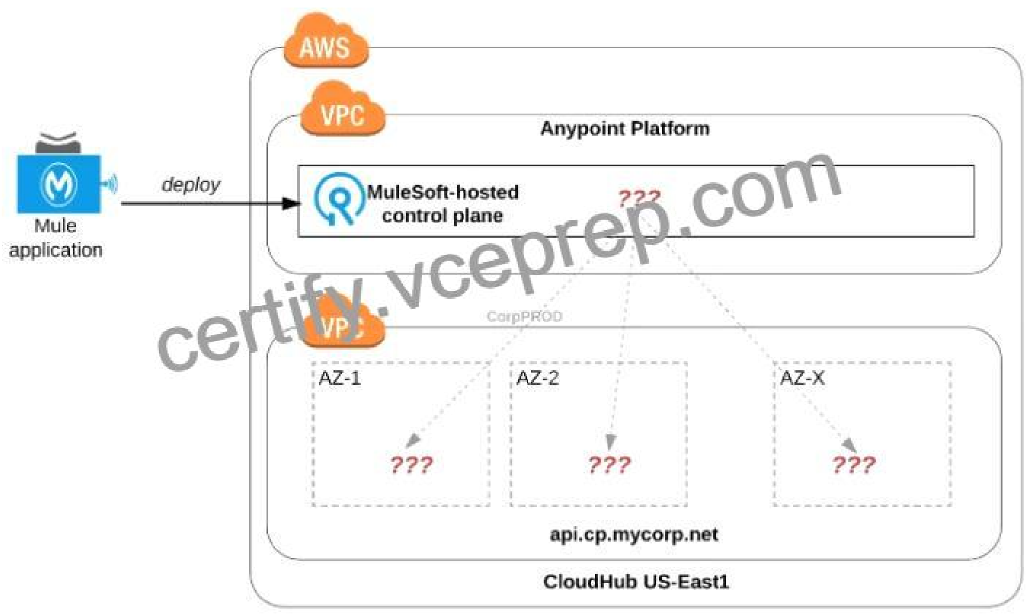
Impacted Platforms	Impacted Duration
Anypoint Runtime Manager - On-Prem Runtimes	During this time frame, on-prem runtimes appeared disconnected from the US Anypoint Control Plane: Tuesday, May 29, 2018, 3:35 AM PDT to 4:27 AM PDT

Incident Description

During the incident time frame, managed Runtimes running on-premises disconnected from the US Anypoint Platform Control Plane and may have encountered recurrent re-connection errors. Customers were unable to manage applications running on those runtimes or register new ones during this time. Runtimes and Applications continued to operate without impact.

NEW QUESTION 62

Refer to the exhibit.



An organization uses one specific CloudHub (AWS) region for all CloudHub deployments.

How are CloudHub workers assigned to availability zones (AZs) when the organization&#8217;s Mule applications are deployed to CloudHub in that region?

* Workers belonging to a given environment are assigned to the same AZ within that region
* AZs are selected as part of the Mule application&#8217;s deployment configuration
* Workers are randomly distributed across available AZs within that region
* An AZ is randomly selected for a Mule application, and all the Mule application&#8217;s CloudHub workers are assigned to that one AZ

Correct answer: Workers are randomly distributed across available AZs within that region.

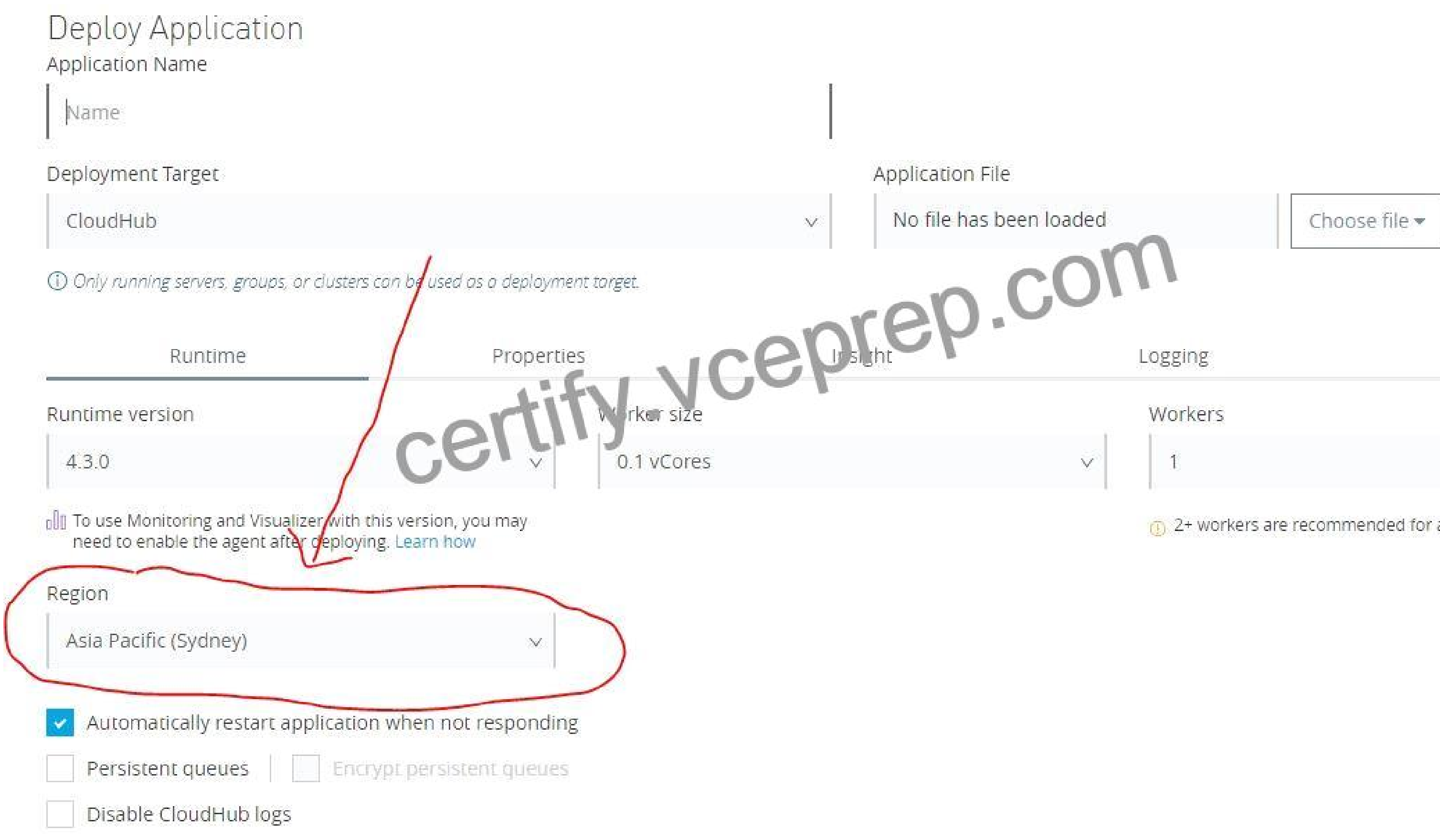
****

>> Currently, we only have control to choose which AWS Region to choose but there is no control at all using any configurations or deployment options to decide what Availability Zone (AZ) to assign to what worker.

>> There are NO fixed or implicit rules on platform too w.r.t assignment of AZ to workers based on environment or application.

>> They are completely assigned in random. However, cloudhub definitely ensures that HA is achieved by assigning the workers to more than on AZ so that all workers are not assigned to same AZ for same application.

Reference:


Deploy Application

Application Name: Name

Deployment Target: CloudHub

Application File: No file has been loaded

Only running servers, groups, or clusters can be used as a deployment target.

Runtime | Properties | Insight | Logging

Runtime version: 4.3.0

Worker size: 0.1 vCores

Workers: 1

To use Monitoring and Visualizer with this version, you may need to enable the agent after deploying. Learn how

2+ workers are recommended for a

Region: Asia Pacific (Sydney)

- [x] Automatically restart application when not responding
- [ ] Persistent queues
- [ ] Encrypt persistent queues
- [ ] Disable CloudHub logs

NEW QUESTION 63

What are 4 important Platform Capabilities offered by Anypoint Platform?

* API Versioning, API Runtime Execution and Hosting, API Invocation, API Consumer Engagement
* API Design and Development, API Runtime Execution and Hosting, API Versioning, API Deprecation

* API Design and Development, API Runtime Execution and Hosting, API Operations and Management, API Consumer Engagement
* API Design and Development, API Deprecation, API Versioning, API Consumer Engagement

Correct answer: API Design and Development, API Runtime Execution and Hosting, API Operations and Management, API Consumer Engagement

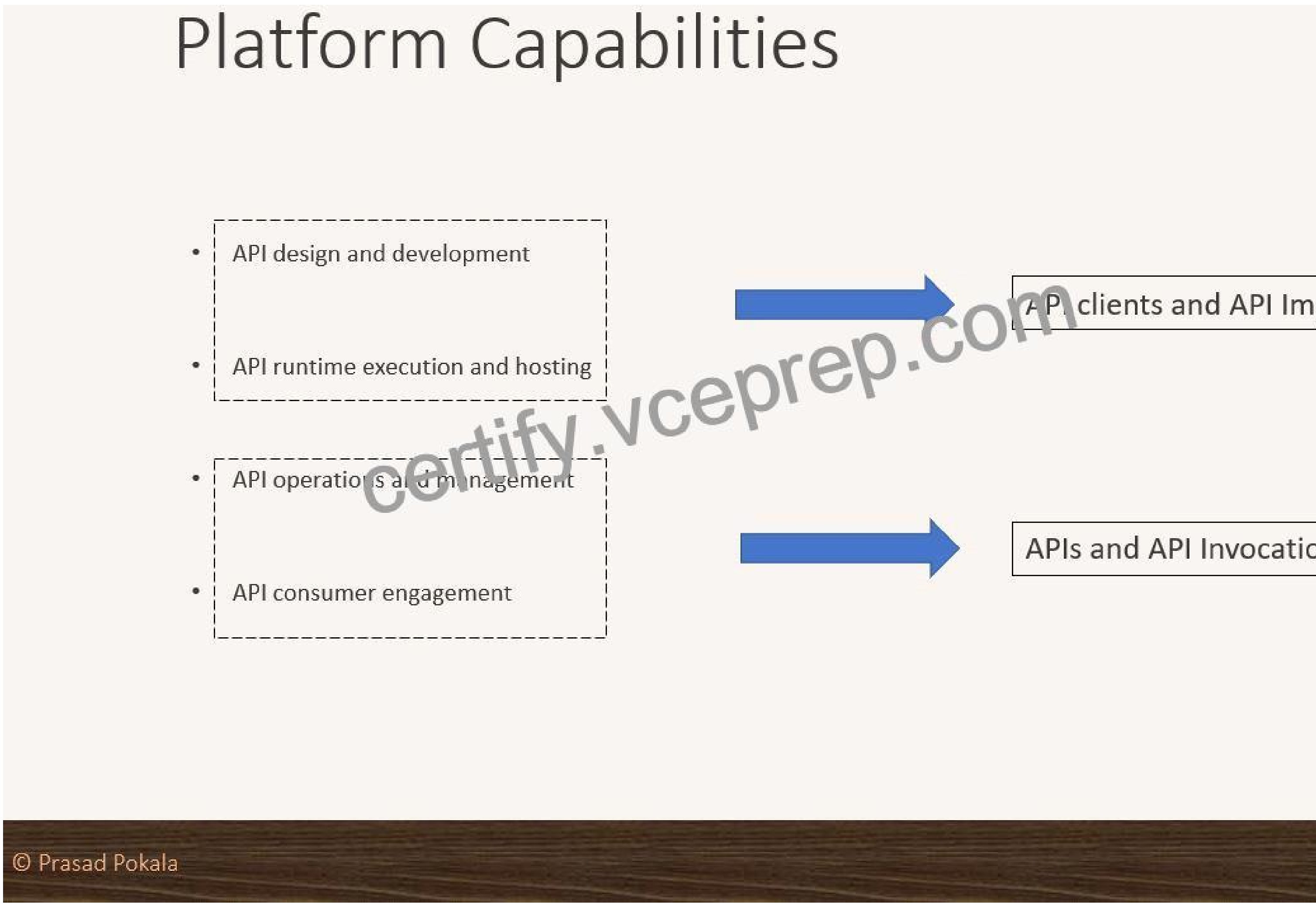
****

>> API Design and Development &#8211; Anypoint Studio, Anypoint Design Center, Anypoint Connectors

>> API Runtime Execution and Hosting &#8211; Mule Runtimes, CloudHub, Runtime Services

>> API Operations and Management &#8211; Anypoint API Manager, Anypoint Exchange

>> API Consumer Management &#8211; API Contracts, Public Portals, Anypoint Exchange, API Notebooks



Platform Capabilities

- API design and development
- API runtime execution and hosting
- API operations and management
- API consumer engagement

API clients and API Im

APIs and API Invocatic

© Prasad Pokala

Use Valid Exam MCPA-Level-1 by VCEPrep Books For Free Website:
https://www.vceprep.com/MCPA-Level-1-latest-vce-prep.html]