MuleSoft.MCPA-Level-1.v2025-03-11.q121

試験コード:	MCPA-Level-1
試験名称:	MuleSoft Certified Platform Architect - Level 1
認定資格:	MuleSoft
無料問題数:	121
バージョン:	v2025-03-11
アクセス数:	867
ページビュー数:	1210
https://www.jpnpdf.com/MuleSoft.MCPA-Level-1.v2025-03-11.q121-mondaishu.html	

最新問題: **1**
Mule アプリケーションが CloudHub Shared Worker Cloud にデプロイされるときに作成される、完全修飾ドメイン名 (FQDN) (DNS エントリとも呼ばれます) を最もよく表すものは何ですか?
A. 環境やVPCの設計に関係なく、固定数のFQDNが作成されます。
B. FQDNは、地域に関係なく、選択されたアプリケーション名によって決定されます。
C. FQDNはアプリケーション名によって決定されますが、展開後に管理者が変更できます。
D. FQDNはアプリケーション名とAnypoint Platform組織の両方によって決定されます。
Answer: (解答を表示する**)**
FQDNは、地域に関係なく、選択されたアプリケーション名によって決定されます。
****
>> Shared Worker Cloud にアプリケーションを展開する場合、FQDN は常に選択されたアプリケーション名によって決定されます。
>> アプリがどのリージョンにデプロイされるかは問題ではありません。
>> 生成された FQDN にはリージョンが含まれることは事実であり、真実です (例:
たとえば、exp-salesorder-api.au-s1.cloudhub.io など) の場合、別の CloudHub リージョンにデプロイするときに同じ名前を使用できるわけではありません。
>> アプリケーション名は、リージョンや組織に関係なく普遍的に一意である必要があり、共有ロード バランサーの FQDN のみを決定します。

最新問題: **2**
REST API 実装の統合テストの特性として最もありそうにないものは何ですか?
A. テストは既知のリクエストペイロードを準備し、レスポンスペイロードを検証します。
B. テストは、Mule アプリケーションがコンパイルされパッケージ化された直後に実行されます。
C. テストでは、すべてのソースシステムおよび/またはターゲットシステムが構成され、アクセス可能である必要があります。
D. テストは外部HTTPリクエストによってトリガーされます

Answer: (解答を表示する**)**

最新問題**: 3**
週に数回、API 実装により、Anypoint Monitoring ダッシュボードに 1 分あたり数千件のリクエストが表示されます。これらのバーストの間に、ダッシュボードには 1 分あたり 2 ～ 5 件のリクエストが表示されます。API 実装は、2 つの非クラスター化レプリカ、予約済み vCPU 1.0、および vCPU 制限 2.0 を使用して、Anypoint Runtime Fabric 上で実行されています。
API 消費者が応答時間が遅いと苦情を申し立てており、ダッシュボードでは苦情の時点で 99 パーセンタイルが 120 秒を超えていることが示されています。また、これらの期間中の CPU 使用率は 90% を超えていることも示されています。
QA 環境での手動テストでは、API コンシューマーは一貫して応答時間の遅さと CPU 使用率の高さを再現しており、この時点では他の API リクエストはありませんでした。ブレインストーミング セッションで、エンジニアリング チームはリクエストの応答時間を短縮するためのいくつかの提案を作成しました。
どの提案を最初に追求すべきでしょうか?
A. API実装のvCPUリソースを増やす
B. APIクライアントを変更して、問題のあるリクエストをより小さく、要求の少ないリクエストに分割します。
C. API実装のレプリカ数を増やす
D. APT クライアントをスロットルして、1 分あたりのリクエスト数を減らします。
Answer: A (メーッセージを残す**)**
* シナリオ分析:
* API 実装では、リクエストの集中時に CPU 使用率が高くなります (90% 以上)。これは、99 パーセンタイル応答時間が 120 秒を超えることで示されるように、応答時間が遅くなることに関連しています。
* API 実装は、2 つの非クラスター化レプリカを持つ Anypoint Runtime Fabric 上で実行されており、予約済みの vCPU は 1.0、vCPU 制限は 2.0 です。
* バースト時の CPU 使用率が高い場合、現在のリソースではピーク負荷を処理するのに十分でない可能性があります。
* オプションの評価:
* オプション A (正解): 各レプリカの vCPU リソースを増やすと、大量のトラフィックを処理するための処理能力が向上し、スパイク時の応答時間が短縮される可能性があります。バースト時には CPU 使用率が常に高くなるため、このオプションはリソースのボトルネックに直接対処します。
* オプション B: API クライアントを変更してリクエストを分割すると、個々のリクエストの負荷が軽減される可能性がありますが、クライアント側での実装が複雑になる可能性があり、CPU 使用率が高い問題が完全に解決されない可能性があります。
* オプション C: レプリカの数を増やすと負荷を分散できますが、各レプリカの CPU 負荷が高い場合、CPU リソースを増やさずにレプリカを追加しても問題が完全に解決されない可能性があります。
* オプション D: クライアントのスロットリングによりリクエスト数は減りますが、クライアントが高いリクエスト レートを維持する必要がある場合は、この方法は受け入れられない可能性があります。また、API 実装の CPU 制限に直接対処するものではありません。

* 結論：
* オプション A は、vCPU 割り当てを増やすことで CPU 使用率が高くなる根本的な原因に対処し、API がより多くのリクエストを効率的に処理できるようにするため、最適な選択肢です。他のオプションを検討する前に、まずこれを追求する必要があります。
需要の高い環境での API パフォーマンスの最適化の詳細については、MuleSoft の Runtime Fabric と vCPU リソース割り当てに関するドキュメントを参照してください。

最新問題: **4**
中央 IT チームの開発者が、OAuth 2.0 で保護されたシステム API の RAML 定義の初期バージョンをデザイン センターで作成し、Exchange に公開しました。LoB IT の別の開発者が Exchange でシステム API を発見し、それをプロセス API で活用したいと考えています。
プロセス API がシステム API を呼び出すための MuleSoft 推奨アプローチは何ですか?
A. Process APIは、まずExchangeからCAuth 2.0モジュールをインポートし、OAuthで更新する必要があります。
システムAPIを呼び出す前に2.0の資格情報が必要です
B. プロセスAPIはプロパティYAMLファイルを使用してシステムAPI URLを保存し、HTTPリクエストコネクタを使用してSysterm APIを呼び出します。
C. プロセスAPLは、システムAPI用にExchangeで自動生成されたREST接続コネクタを使用します。
D. プロセス API は、プロセス API POM ファイルを手動で更新して、システム API を依存関係として含めます。
Answer: C (メーッセージを残す**)**
MuleSoft のエコシステムでは、プロセス API がシステム API (Exchange に公開され、OAuth 2.0 によって保護されている) を使用する必要がある場合、REST Connect Connector を使用することをお勧めします。これがベスト プラクティスとどのように一致するかを以下に示します。
* 自動コネクタ生成:
* RAML または OAS 仕様が Exchange で公開されると、MuleSoft はその API 用の REST 接続コネクタを自動的に生成します。このコネクタは、HTTP リクエストの作成と OAuth 認証の処理の複雑さを抽象化するため、統合が簡素化されます。
* 合理化された統合:
* プロセス API は、生成されたこのコネクタを Exchange からインポートし、OAuth 資格情報を構成して、手動で HTTP を設定することなくシステム API への安全なアクセスを合理化できます。
* オプション C が正しい理由:
* REST Connect Connector を使用すると、MuleSoft の自動化ツールが直接活用され、手動構成が最小限に抑えられ、より保守しやすい統合が保証されます。
* 誤ったオプションの説明:
* オプション A (OAuth モジュールのインポート) は不要です。OAuth はコネクタの構成内で処理されます。
* オプション B (HTTP リクエストを含むプロパティ YAML ファイル) では手動セットアップが必要となり、エラーが発生しやすくなるため推奨されません。

* オプション D (POM ファイルを手動で更新する) は、Exchange を介して API を呼び出すのに直接役立ちません。
参考資料MuleSoft での REST Connect コネクタと OAuth 統合の使用に関する詳細については、API 管理とコネクタに関する MuleSoft のドキュメントを参照してください。

最新問題: **5**
既存の見積 API は RAML で定義されており、見積エンジンとのやり取りに REST クライアントによって使用されます。現在、見積の作成を可能にするリソースが RAML で定義されていますが、既存の見積の更新を可能にするための新しい要件が最近受信されました。
この変更を容易に処理できるようにするには、どの 2 つのアクションを実行する必要がありますか?
2つの回答を選択してください
A. 新しい更新リクエストに対応するためにAPI実装を更新します
B. 古いクライアントアプリケーションを削除し、変更に対応する新しいクライアントアプリケーションを作成します。
C. 更新リクエストの新しいメソッドの詳細で RAML を更新します。
D. Exchange の API の既存バージョンを廃止する
E. 更新されたエンドポイントへのアクセスを許可する新しいAPIポリシーをAPIマネージャーに追加します
Answer: A,C (メーッセージを残す**)**
既存の見積りの更新を許可するという新しい要件に対応するには、次のアクションを実行する必要があります。
* RAML定義を更新します オプションC）：
* The RAML specification defines the structure and behavior of the API. Adding a new method (such as PUT or PATCH) for updating quotes requires modifying the RAML to include this new endpoint. This ensures the API specification is up-to-date and accurately reflects the new functionality.
* Update the API Implementation (Option A):
* Once the RAML is updated, the backend API implementation must also be modified to handle the new update requests. This could involve adding logic to process and validate update requests, connect to necessary backend resources, and apply the changes to existing quotes.
* Explanation of Incorrect Options:
* Option B (removing and creating new clients) is unnecessary; client applications can remain as they are, with no need for complete replacement.
* Option D (deprecating existing versions) may not be required if backward compatibility is maintained.
* Option E (adding a new policy) does not facilitate functional changes and is unrelated to implementing the update feature.
ReferencesFor more details on updating RAML definitions and API implementations, refer to MuleSoft's API Design documentation on RAML and RESTful API practices.

最新問題: **6**
API 実装における自動検出の使用を最もよく説明するものは何ですか?
A. API ManagerがAPI実装を認識し、ポリシーを適用できるようにします。

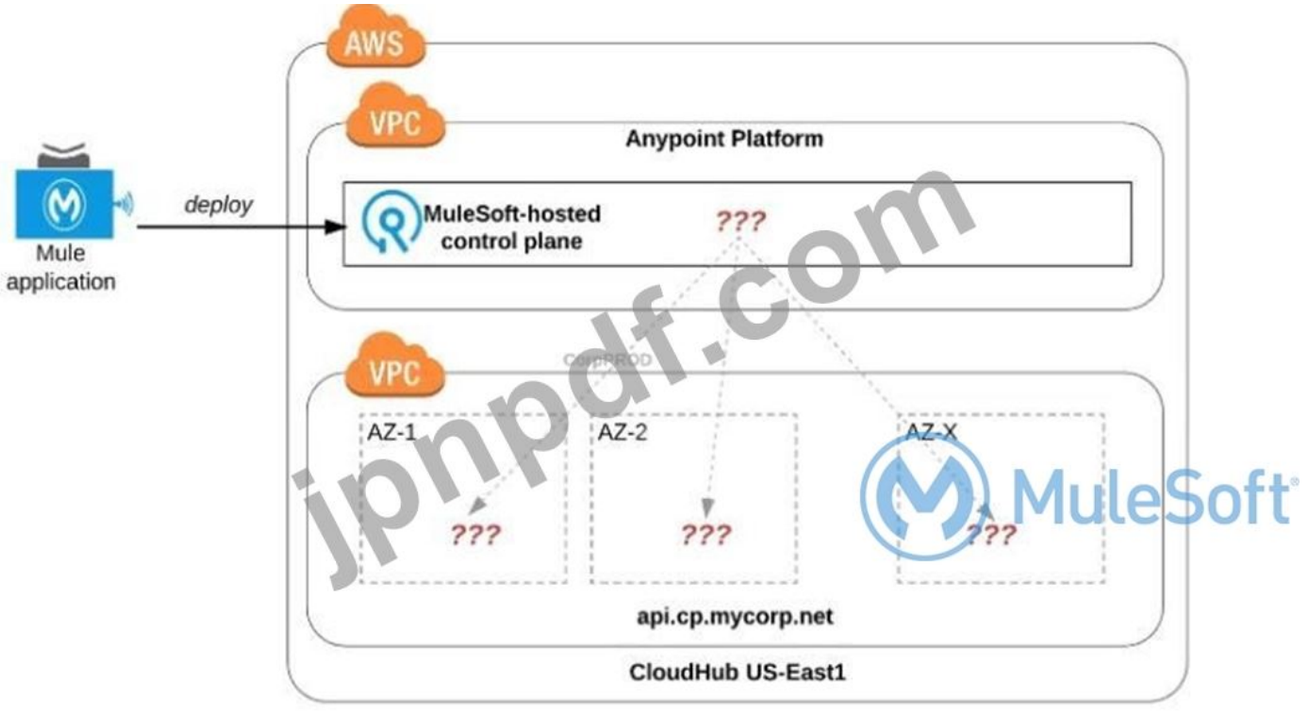
B. Anypoint StudioがAnypoint Platformで設定されたAPI定義を検出できるようにします。
C. Anypoint AnalyticsがAPIの使用状況を把握できるようになります。
D. Anypoint Exchangeが資産を発見し、再利用できるようにします
Answer: B (メーッセージを残す**)**

最新問題**: 7**
API 実装が CloudHub にデプロイされます。
デフォルトの Anypoint Platform 機能を使用すると、どのような条件でアラートを生成できますか。アラート条件は、API 実装のエンドツーエンドのリクエスト処理によって異なります。
A. APIが認識されないAPIクライアントによって呼び出された場合
B. 特定のAPIクライアントが一定期間内にAPIを頻繁に呼び出した場合
C. API呼び出しの応答時間がしきい値を超えた場合
D. APIが非常に多くのAPI呼び出しを受け取った場合
Answer: (解答を表示する**)**
正解: API呼び出しの応答時間がしきい値を超えた場合
****
>> デフォルトのAnypoint Platform機能を使用して、すべてのオプションに対してアラートを設定できます。
>> ただし、質問では、API 実装のエンドツーエンドのリクエスト処理に応じて条件が決まるアラートが求められています。
>> 応答時間」に関するアラートは、しきい値を超えているかどうかを判断するために、API 実装のエンドツーエンドのリクエスト処理を必要とする唯一のアラートです。

最新問題**: 8**
ある組織には、HTTP POST 経由で JSON データを受け入れる API がいくつかあります。API はすべて公開されており、いくつかのモバイル アプリケーションや Web アプリケーションに関連付けられています。
組織はこれらの API に対して認証やコンプライアンス ポリシーを使用することを望んでいませんが、同時に、悪意のある人物が API 実装を実行しているアプリケーションやサーバーを何らかの形で侵害する可能性のあるペイロードを送信する可能性があることを懸念しています。
この脅威への露出に対処できる、すぐに使用できる Anypoint Platform ポリシーは何ですか?
A. すべてのAPI呼び出しにHTTPS相互認証を使用して悪意のある行為者をシャットアウトします
B. すべてのAPIにIPブラックリストポリシーを適用します。ブラックリストにはすべての悪質な行為者が含まれます。
C. 悪意のあるデータが使用される前にそれを検出するヘッダー挿入および削除ポリシーを適用する
D. 潜在的な脅威ベクトルを検出するために、すべての API に JSON 脅威保護ポリシーを適用します。
Answer: (解答を表示する**)**
すべてのAPIにJSON脅威保護ポリシーを適用して、潜在的な脅威ベクトルを検出する
****
>> 通常、API が特定の消費者 (既知の消費者/顧客) 向けに設計および開発されている場合は、トラフィックがその消費者/顧客からのみ発生するように、同じものを IP ホワイトリストに登録します。

>> ただし、このシナリオでは、API が一般に公開されており、非常に多くのモバイル アプリケーションや Web アプリケーションで使用されているため、すべての悪意のある行為者を特定してブラックリストに登録することは不可能です。
>> したがって、JSON 脅威保護ポリシーは、そのような悪意のある行為者による不正な JSON ペイロードを防ぐための最善の機会です。

最新問題: **9**
展示品を参照してください。



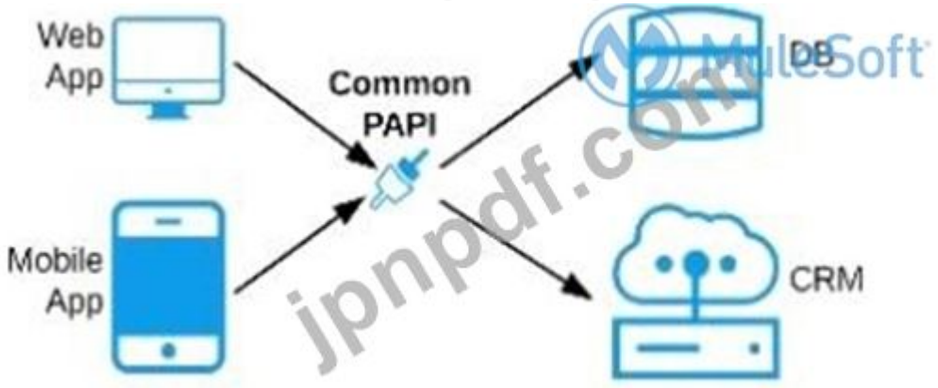
組織は、すべての CloudHub デプロイメントに対して 1 つの特定の CloudHub (AWS) リージョンを使用します。
組織の Mule アプリケーションがそのリージョンの CloudHub にデプロイされている場合、CloudHub ワーカーはどのようにしてアベイラビリティーゾーン (AZ) に割り当てられますか?
A. 特定の環境に属するワーカーは、そのリージョン内の同じ AZ に割り当てられます。
B. AZはMuleアプリケーションのデプロイメント構成の一部として選択されます
C. ワーカーは、そのリージョン内の利用可能なAZにランダムに分散されます。
D. Mule アプリケーションに対して AZ がランダムに選択され、Mule アプリケーションのすべての CloudHub ワーカーがその 1 つの AZ に割り当てられます。
Answer: D (メーッセージを残す**)**
ワーカーは、そのリージョン内の利用可能な AZ 全体にランダムに分散されます。
****
>> 現在、どの AWS リージョンを選択するかは制御できますが、どのアベイラビリティーゾーン (AZ) をどのワーカーに割り当てるかを決定するための構成やデプロイメント オプションを使用する制御はまったくありません。

>> 環境やアプリケーションに基づいてワーカーに AZ を割り当てることに関しても、プラットフォームには固定または暗黙のルールはありません。
>> これらは完全にランダムに割り当てられます。ただし、Cloudhub では、すべてのワーカーが同じアプリケーションに対して同じ AZ に割り当てられないように、ワーカーを複数の AZ に割り当てることで HA が確実に実現されます。

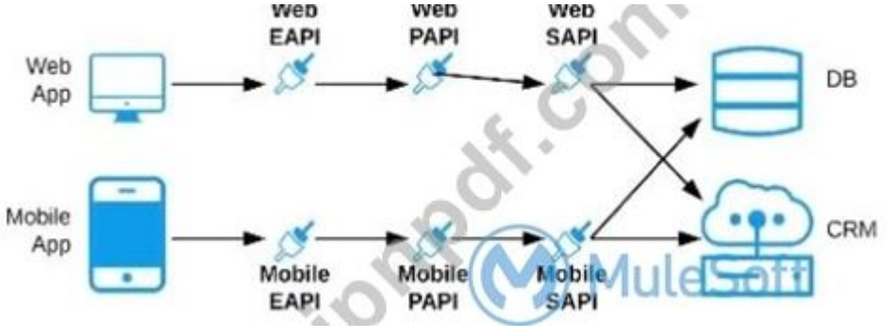
最新問題: **10**
What are the major benefits of MuleSoft proposed IT Operating Model?
A. 1. Decrease the IT delivery gap
2. Meet various business demands without increasing the IT capacity
3. Focus on creation of reusable assets first. Upon finishing creation of all the possible assets then inform the LOBs in the organization to start using them
B. 1. Decrease the IT delivery gap
2. Meet various business demands by increasing the IT capacity and forming various IT departments
3. Make consumption of assets at the rate of production
C. 1. ITデリバリーギャップの縮小
2. IT能力を増強することなく、さまざまなビジネスニーズに対応
3. 生産量に応じて資産を消費する
Answer: C (メーッセージを残す**)**
1. ITデリバリーギャップの縮小
2. IT能力を増強することなく、さまざまなビジネスニーズに対応
3. 生産速度に応じて資産を消費する。
****

最新問題: **11**
展示を参照してください。組織は、モバイル アプリと Web アプリケーションの両方から顧客データにアクセスできるようにする必要があります。これらのアプリケーションは、共通フィールドと特定の固有フィールドの両方にアクセスする必要があります。
データは部分的にデータベースで利用可能であり、部分的にサードパーティの CRM システムで利用可能です。
これらの設計要件に最適な API を作成するには、どのような API を作成する必要がありますか?

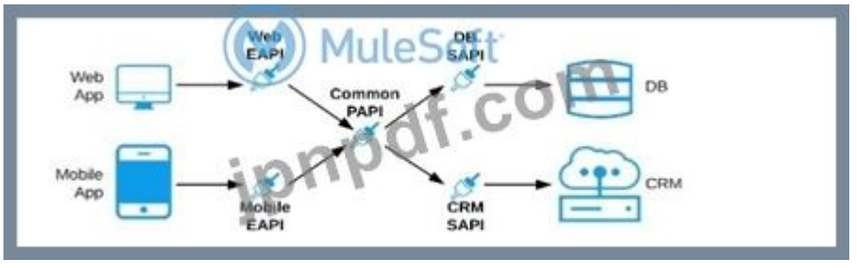
A) A Process API that contains the data required by both the web and mobile apps, allowing these applications to invoke it directly and access the data they need thereby providing the flexibility to add more fields in the future without needing API changes



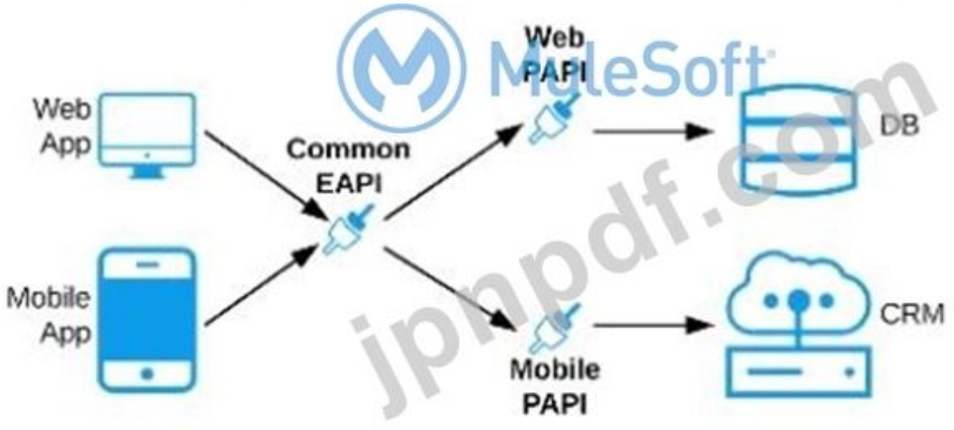
B) One set of APIs (Experience API, Process API, and System API) for the web app, and another set for the mobile app



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

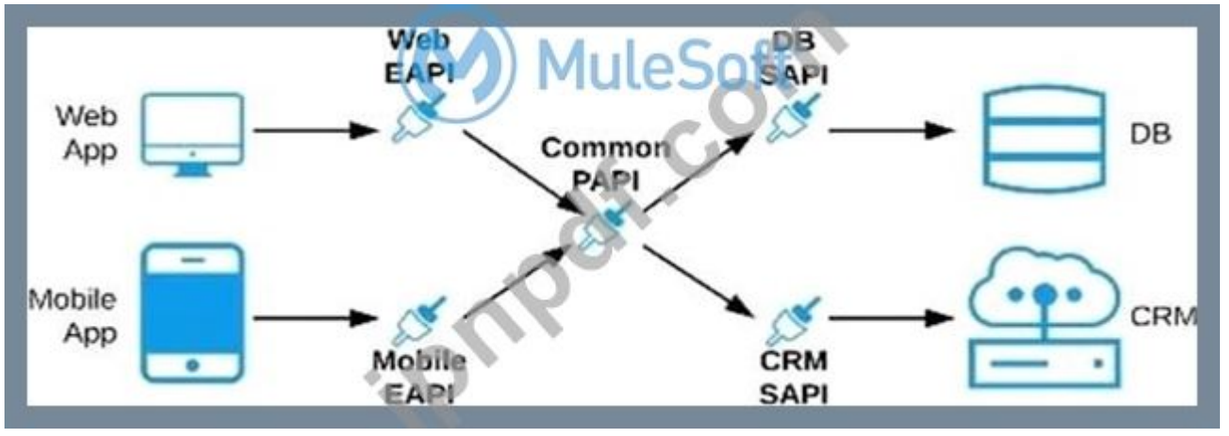


D) A common Experience API used by both the web and mobile apps, but separate Process APIs for the web and mobile apps that interact with the database and the CRM System

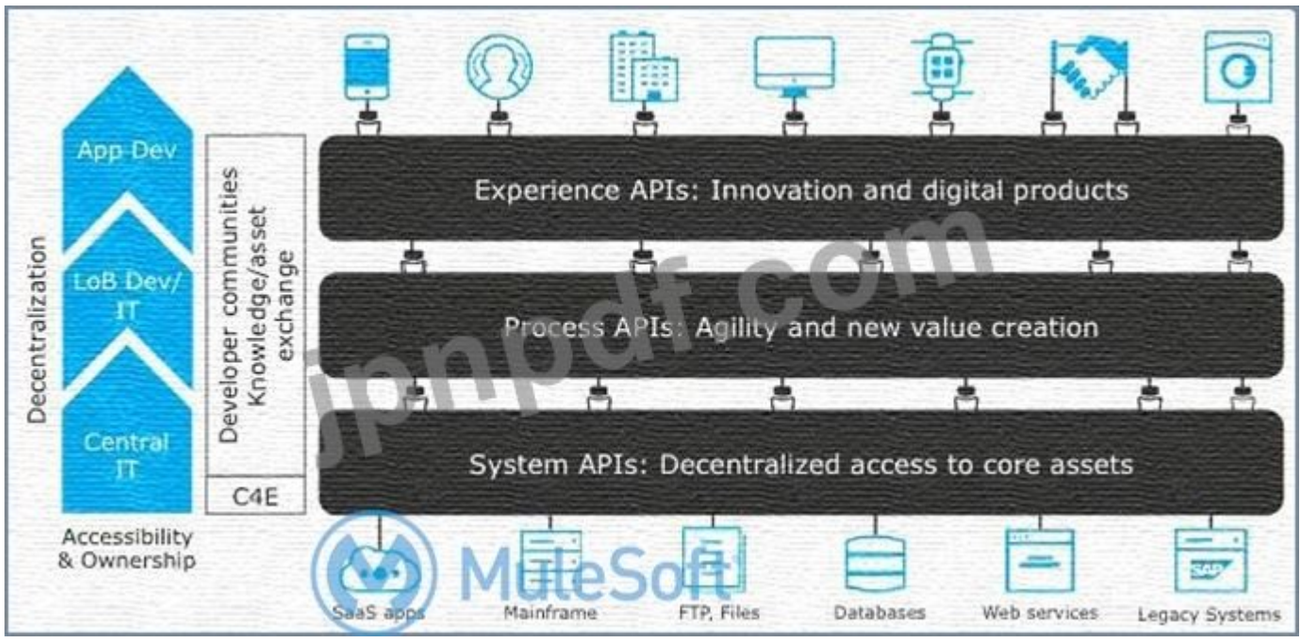


A. Option A

B. Option B
C. Option C
D. Option D
Answer: C (メーッセージを残す)
正解: モバイルとウェブアプリ用に別々のエクスペリエンス API がありますが、データベースと CRM システム用に作成された別々のシステム API を呼び出す共通のプロセス API があります。
****
MuleSoft の API 主導の接続性について:
>> エクスペリエンス API は、各消費者のニーズとエクスペリエンスに応じて構築する必要があります。
>> プロセス API には、ビジネス機能を実現するためのすべてのオーケストレーション ロジックが含まれている必要があります。
>> 各バックエンド システムのデータをロック解除するには、システム API を構築する必要があります。
参照 :



最新問題: **12**
API 主導の接続性のどのレイヤーが、主要なシステム、レガシー システム、データ ソースなどのロックを解除し、機能を公開することに重点を置いていますか?
A. エクスペリエンスレイヤー
B. プロセス層
C. システム層
Answer: C (メーッセージを残す)
システム層



API 主導の接続アプローチで使用される API は、次の 3 つのカテゴリに分類されます。
システム API - これらは通常、レコードのコア システムにアクセスし、ユーザーを基盤システムの複雑さや変更から隔離する手段を提供します。一度構築されると、多くのユーザーは基盤システムを学習することなくデータにアクセスでき、複数のプロジェクトでこれらの API を再利用できます。
プロセス API - これらの API は、単一のシステム内またはシステム間で (データ サイロを解体して) データと対話してデータを形成します。これらの API は、データの元となるソース システムや、そのデータが配信されるターゲット チャネルに依存せずにここで作成されます。
エクスペリエンス API - エクスペリエンス API は、各チャネルに個別のポイントツーポイント統合を設定するのではなく、共通のデータ ソースからデータを再構成して、対象ユーザーが最も簡単にデータを利用できるようにするための手段です。エクスペリエンス API は通常、API ファーストの設計原則に基づいて作成され、特定のユーザー エクスペリエンスを念頭に置いて API が設計されます。

最新問題: **13**
レート制限 API ポリシーの適用を API の RAML 定義に正確に反映するにはどうすればよいでしょうか?
A. レート制限ポリシーの動作の説明を追加してリソース定義を改良することにより
B. 残りのリクエストクエリパラメータに説明、タイプ、例を追加してリクエスト定義を改良する
C. すぐに使用可能なAnypoint Platformのレート制限強制セキュリティスキームを説明、タイプ、例とともに追加して、応答定義を改良します。
D. 説明、タイプ、例を含むx-ratelimit-*レスポンスヘッダーを追加してレスポンス定義を改良する
Answer: D (メーッセージを残す**)**
正解: 説明、タイプ、例を含むx-ratelimit-*レスポンスヘッダーを追加してレスポンス定義を改良する
****

参考文献:
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

最新問題: **14**
プロセス API の実装を変更する必要があります。
この変更が API クライアントに与える影響を最小限に抑える有効なアプローチは何ですか?
A. プロセスAPIの実装に必要な変更を実装し、可能な限りプロセスAPIのRAML定義が変更されないようにします。
B. Process API の変更を新しい API 実装に実装し、古い API 実装が HTTP ステータス コード 301 - Moved Permanently を返すようにして、API クライアントに新しい API 実装を呼び出す必要があることを通知します。
C. 現在のプロセスAPIのRAML定義を更新し、更新されたRAML定義へのリンクを送信してAPIクライアント開発者に通知します。
D. APIコンシューマーが新しいプロセスAPIまたはAPIバージョンへの移行の準備ができていることを確認するまで、変更を延期します。
Answer: B (メーッセージを残す**)**

最新問題: **15**
すべてのデータ処理を特定の管轄区域（米国や EU など）内で実行することを要求する法的規制に対処する場合の API 実装について正しいのは何ですか?
A. オブジェクト ストアは米国東部地域にのみデプロイされたサービスに依存しているため、使用を避ける必要があります。
B. Anypoint MQではなく、Active MQなどの管轄地域の外部メッセージングシステムを使用する必要があります。
C. これらは、Anypoint Platform コントロールプレーンによって管理される Anypoint Platform ランタイムプレーンにデプロイされ、両方のプレーンが同じ管轄内にある必要があります。
D. 転送中も保存中もすべてのデータが暗号化されていることを確認する必要があります。
Answer: C (メーッセージを残す**)**
正解: これらは、Anypoint Platform コントロール プレーンによって管理される Anypoint Platform ランタイム プレーンにデプロイする必要があり、両方のプレーンは同じ管轄区域内にある必要があります。
****

>> 法的規制に従い、すべてのデータ処理は特定の管轄区域内で実行されます。つまり、米国のデータは米国内に保存され、外部に流出してはいけません。同様に、EU のデータは EU 内に保存され、外部に流出してはいけません。
>> したがって、転送中および保存中のデータを暗号化するだけでは、規則に準拠するのに役立ちません。データが外部に漏れないようにする必要もあります。
>> ここで取り上げているデータは、Anypoint MQ に公開されるメッセージだけではありません。実行中のアプリ、トランザクション状態、アプリケーション ログ、イベント、メトリック情報、その他のメタデータも含まれます。したがって、Anypoint MQ をローカルでホストされている ActiveMQ に置き換えるだけでは役に立ちません。
>> ここで話しているデータは、オブジェクト ストアに保存されているキーと値のペアだけではありません。公開されたメッセージ、実行中のアプリ、トランザクションの状態、アプリケーション ログ、イベント、メトリック情報、その他のメタデータも含まれます。したがって、オブジェクト ストアの使用を避けるだけでは役に立ちません。
>> 与えられた選択肢の中で唯一残された、そして正しい選択肢は、管轄区域内にあるランタイム プレーンとコントロール プレーンにアプリケーションをデプロイすることです。

最新問題**: 16**
次の順序のうち正しいものはどれですか?
A. APIクライアントはAPIを呼び出すロジックを実装します >> APIコンシューマはAPIへのアクセスを要求します >> API実装はリクエストをAPIにルーティングします
B. API コンシューマが API へのアクセスを要求 >> API クライアントが API を呼び出すロジックを実装 >> API がリクエストをルーティング >> API 実装
C. API コンシューマーは API を呼び出すロジックを実装します >> API クライアントは API へのアクセスを要求します >> API 実装は要求を API にルーティングします
D. API クライアントは API を呼び出すロジックを実装します >> API コンシューマーは API へのアクセスを要求します >> API は要求をルーティングします >> API 実装
Answer: B (メーッセージを残す**)**
APIコンシューマがAPIへのアクセスを要求 >> APIクライアントがAPIを呼び出すロジックを実装 >> APIがリクエストをルーティング >> API実装
****
>> API コンシューマーは、API を呼び出すロジックを実装しません。単なるロールです。したがって、 "API コンシューマーは API を呼び出すロジックを実装します」というオプションは無効です。
>> API 実装はリクエストをルーティングしません。これは、ターゲット システムの機能が明らかになるロジックの最終部分です。したがって、リクエストは他のエンティティによって API 実装にルーティングされる必要があります。したがって、 "API 実装はリクエストを API にルーティングします」というオプションは無効です。
>> 選択肢の1つは正しいですが、順序が間違っています。順序は "APIクライアントがAPIを呼び出すロジックを実装する >> APIコンシューマがAPIへのアクセスを要求 >> APIがリクエストをルーティングする」となっています。

>> API 実装"。ここでは、オプション内のステートメントは有効ですが、シーケンスが間違っています。
>> 正しいオプションとシーケンスは、API コンシューマーが最初に Anypoint Exchange 上の API へのアクセスを要求し、クライアント資格情報を取得するものです。次に、API クライアントは、API コンシューマーによって要求されたアクセス クライアント資格情報を使用して API を呼び出すロジックを記述し、その要求は API マネージャーによって管理される API を介して API 実装にルーティングされます。

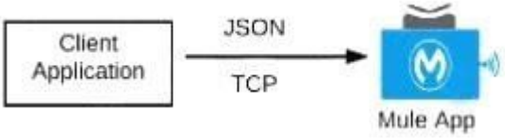
最新問題: **17**
ある組織では、さまざまな API レイヤーを使用してモバイル クライアントをバックエンド システムに統合する API 主導のアーキテクチャを作成しました。バックエンド システムは、いくつかの特殊なコンポーネントで構成されており、REST API を介してアクセスできます。プロセス API とエクスペリエンス API は、バックエンド データ モデルとは異なる同じ境界コンテキスト モデルを共有します。バックエンド システムから消費されるデータの処理を支援するために、このアーキテクチャに追加するのに最適な標準モデル、境界コンテキスト モデル、または破損防止レイヤーは何ですか。
A. 各レイヤーに境界コンテキストモデルを作成し、境界コンテキストが重なる場合はそれらを重ね合わせ、API 開発者が上流と下流のデータモデルの違いを認識できるようにします。
B. バックエンド モデルと API 主導のモデルを組み合わせた標準モデルを作成し、データ モデルを簡素化および統合し、データ変換を最小限に抑えます。
C. システム層の境界付きコンテキスト モデルを作成してバックエンド データ モデルと密接に一致させ、破損防止層を追加して、異なる境界付きコンテキストがシステム層とプロセス層全体で連携できるようにします。
D. すべての API に破損防止レイヤーを作成し、すべてのデータ モデルが互いに一致するように変換を実行し、データが API 間で簡単に移動できるようにして、標準モデルを構築する複雑さとオーバーヘッドを回避します。
Answer: C (メーッセージを残す)
システム層の境界コンテキスト モデルを作成してバックエンド データ モデルと密接に一致させ、破損防止層を追加して、さまざまな境界コンテキストがシステム層とプロセス層全体で連携できるようにします。
****
>> 組織はすでに努力を重ね、エクスペリエンス API とプロセス API 用の境界付きコンテキスト モデルを作成しているため、ここでは標準モデルは選択肢になりません。
>> エクスペリエンス API とプロセス API は同じ境界コンテキスト モデルを共有するため、すべての API に対する破損防止レイヤーは不要かつ無効です。現在、アプローチを選択する必要があるのは、システム レイヤー API だけです。

>> したがって、プロセス層とシステム層の間に破損防止層を配置するとうまく機能します。また、このアプローチを高速化するために、システム API はバックエンド システム データ モデルを模倣できます。

最新問題: **18**
ダウンタイムが繰り返し発生することが知られている Order API を呼び出す必要がある API 実装が設計されています。
このため、Order API が利用できない場合は、フォールバック API が呼び出されます。
フォールバック API の呼び出しを設計する際に、どのようなアプローチが最高の回復力を提供しますか?
A. Order APIが利用できない場合は、HTTP 307 Temporary Redirectステータスコードを介してクライアントリクエストをフォールバックAPIにリダイレクトします。
B. HTTP リクエスター コンポーネントに、Order API を呼び出すオプションを設定して、Order API から HTTP 4xx または 5xx 応答ステータス コードが返されるたびにフォールバック API を呼び出すようにします。
C. Anypoint Exchangeで適切な既存のフォールバックAPIを検索し、注文APIに加えてこのフォールバックAPIへの呼び出しを実装します。
D. API マネージャーで注文 API の別のエントリを作成し、プライマリ注文 API が利用できない場合にこの API をフォールバック API として呼び出します。
Answer: A (メーッセージを残す**)**

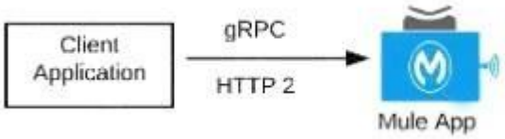
最新問題: **19**
どの Mule アプリケーションで、その Mule アプリケーションによって公開されるエンドポイントに Anypoint Platform によって API ポリシーを適用できますか?
A) HTTP/1.x 経由でリクエストを受け入れる Mule アプリケーション

B) TCP経由でJSONリクエストを受け入れるが、応答を返す必要がないMuleアプリケーション



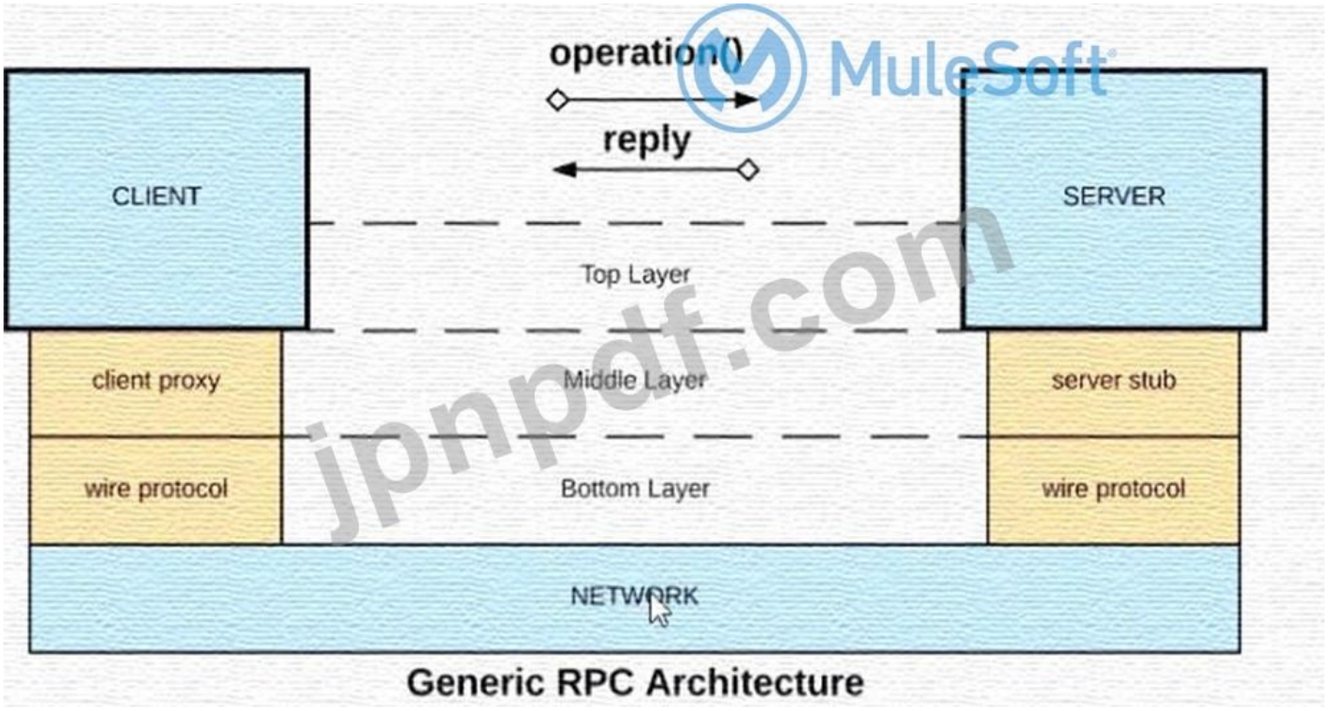
C) WebSocket経由でJSONリクエストを受け入れるMuteアプリケーション

D) HTTP/2 経由で gRPC リクエストを受け入れる Mule アプリケーション

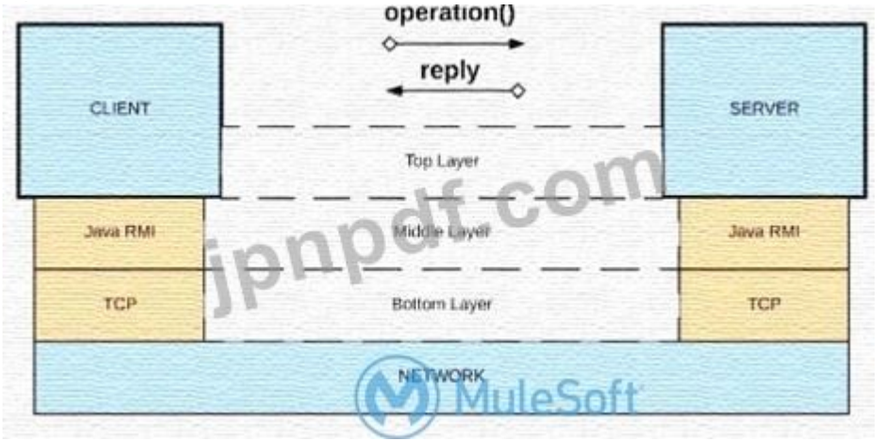


A. オプションC
B. オプションD
C. オプションA
D. オプションB
Answer: C (メーッセージを残す**)**

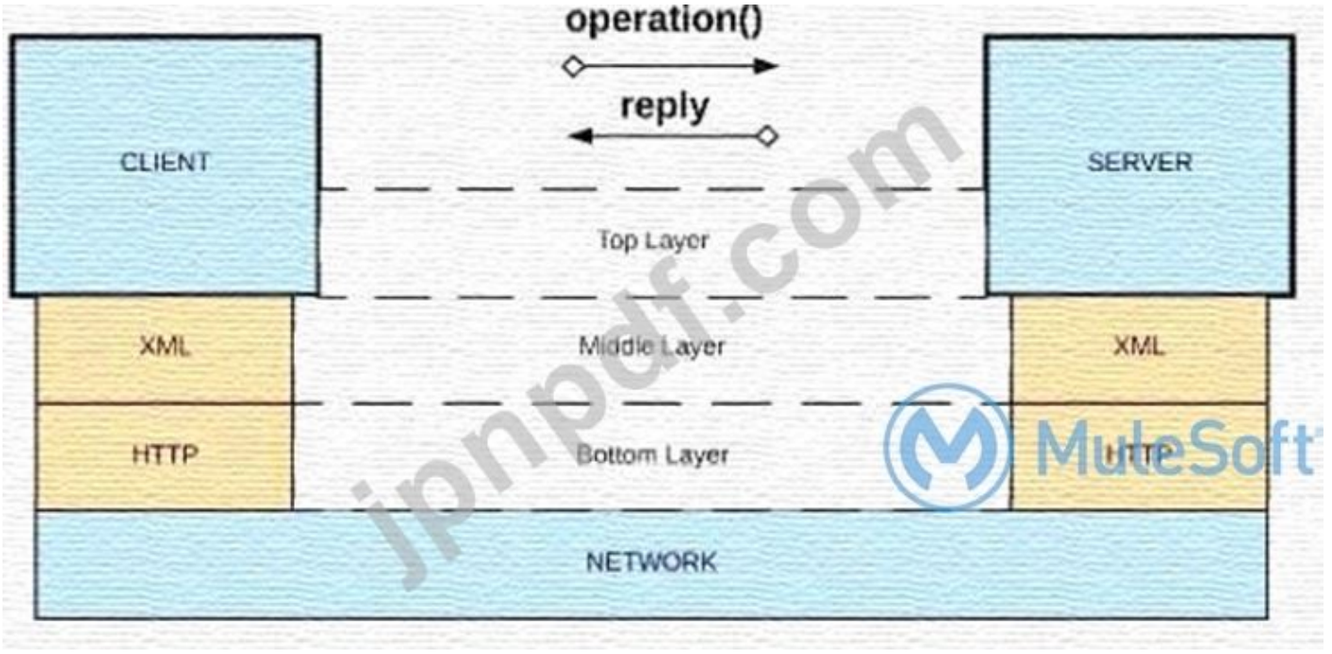
最新問題**: 20**
展示品を参照してください。



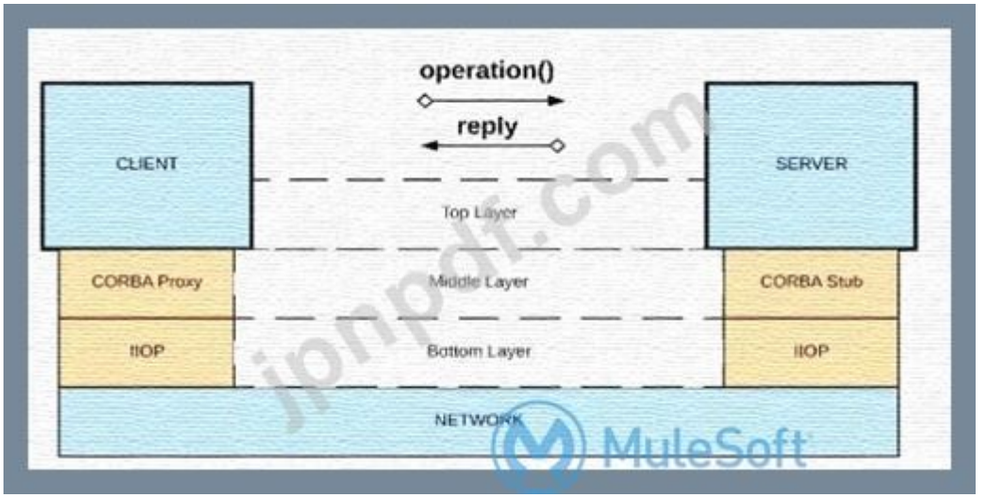
API 主導の接続性とアプリケーション ネットワークの意味で有効な API とは何でしょうか?
A)TCP 経由の Java RMI



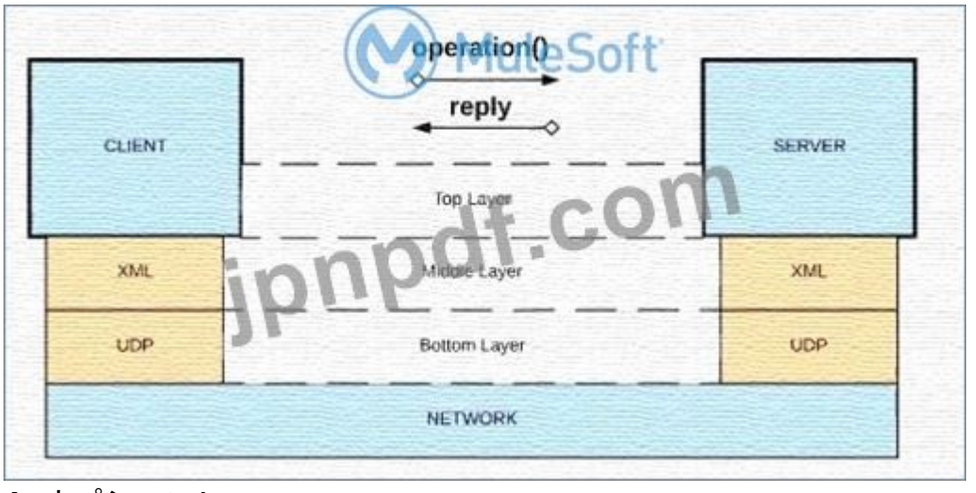
B)TCP 経由の Java RMI



C)HOP 経由の CORBA



D)UDP経由のXML

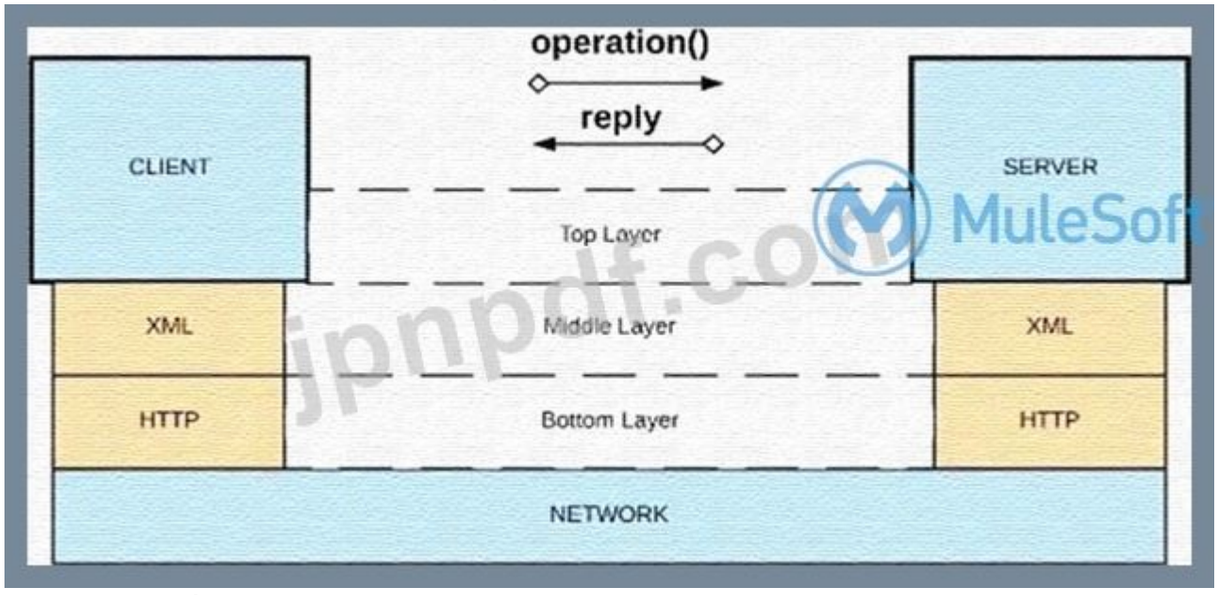


A. オプションA

B. オプションB

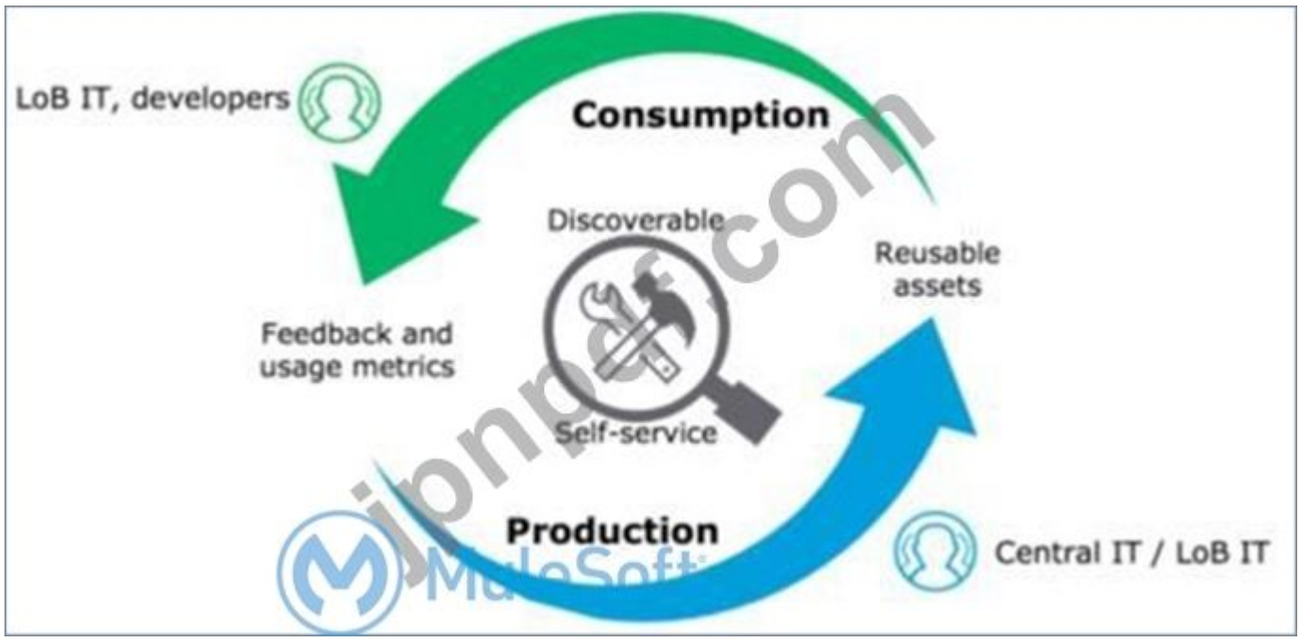
C. オプションC

D. オプションD
Answer: D (メーッセージを残す**)**
HTTP 経由の XML
**********>> API 主導の接続性とアプリケーション ネットワークでは、最も効果的な API とその上にネットワークを構築するために、
HTTP ベースのプロトコル上に API を配置することが求められています。
>> HTTPベースのAPIにより、プラットフォームはさまざまなポリシーを適用して多くのNFRに対応できます。
>> HTTP ベースの API を使用すると、HTTP ベースの w3c ルールに準拠した多くの標準的で効果的な実装パターンを実装することもできます。



フォームの下部
フォームの先頭

最新問題**: 21**
以下のどれを組み合わせて使用すると、IT 運用モデルが効果的になりますか?
A. 再利用可能な資産を作成し、作成した資産を組織全体でマーケティングし、資産が消費されているかどうかを確認するために定期的に LOB レビューを実施します。
B. 再利用可能なアセットを作成し、LOB チームがセルフサービスで API を参照できるように検出可能にし、アクティブなフィードバックと使用状況の指標を取得します。
C. 再利用可能なアセットを作成し、LOB チームがセルフサービスで API を参照できるように、それらを検出可能にします。
Answer: (解答を表示する**)**
再利用可能なアセットを作成し、それらを検出可能にして、LOB チームがセルフサービスで API を参照できるようにし、アクティブなフィードバックと使用状況メトリックを取得します。
****
図、矢印 説明は自動的に生成されます



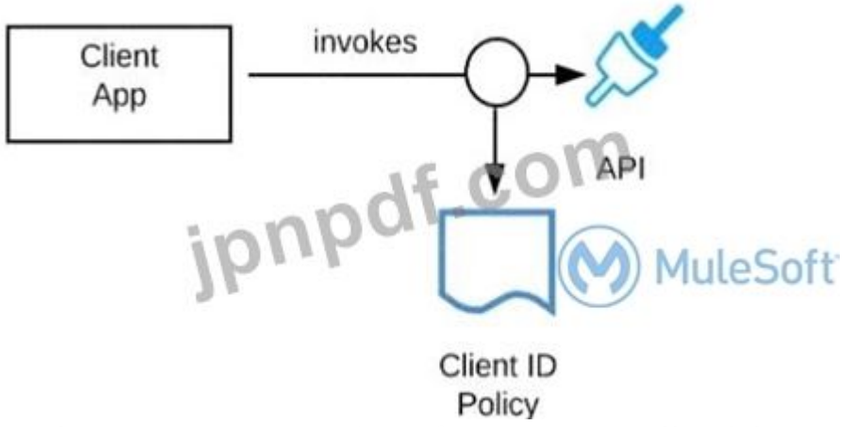
最新問題**: 22**
Mule アプリケーションは HTTPS エンドポイントを公開し、CloudHub Shared Worker Cloud にデプロイされます。その Mule アプリケーションへのすべてのトラフィックは AWS VPC 内に留まる必要があります。
Mule アプリケーションへの API 呼び出しはどの TCP ポートに送信する必要がありますか?
A. 443
B. 8081
C. 8091
D. 8092
Answer: D (メーッセージを残す**)**
説明
https://help.mulesoft.com/s/question/0D52T00004mXXULSA4/multiple-http-listerners-on-cloudhub-one-with-p

最新問題**: 23**
レート制限 API ポリシーの適用を API の RAML 定義に正確に反映するにはどうすればよいでしょうか?
A. レート制限ポリシーの動作の説明を追加してリソース定義を改良することにより
B. 残りのリクエストクエリパラメータに説明、タイプ、例を追加してリクエスト定義を改良する
C. すぐに使用できるAnypoint Platformのレート制限強制セキュリティスキームを説明、タイプ、例とともに追加して、レスポンス定義を改良します。
D. 説明、タイプ、例を含むx-ratelimit-*レスポンスヘッダーを追加してレスポンス定義を改良する
Answer: D (メーッセージを残す**)**
説明、タイプ、例を含むx-ratelimit-*レスポンスヘッダーを追加してレスポンス定義を改良する
****
参考文献:
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

最新問題: **24**
トラフィックは API プロキシを介して API 実装にルーティングされます。API プロキシは API Manager によって管理され、API 実装は Runtime Manager を使用して CloudHub VPC にデプロイされます。この API には API ポリシーが適用されています。このデプロイ シナリオでは、どの時点で API ポリシーが着信 API クライアント リクエストに適用されますか?
A. APIプロキシで
B. API実装時
C. APIプロキシとAPI実装の両方で
D. MuleSoft がホストするロードバランサー
Answer: A (メーッセージを残す**)**
APIプロキシで
****
>> API ポリシーは、Mule プラットフォームの 2 か所で適用できます。
>> 1 - API 実装が実行されているのと同じ Mule ランタイム内の埋め込みポリシーの適用として。
>> 2 - API 実装が実行されている Mule ランタイムの前にある API プロキシ上。
>> 質問のデプロイメント シナリオには API プロキシが関係しているため、ポリシーは API プロキシで適用されます。

最新問題: **25**
展示品を参照してください。



開発者は、クライアント ID 適用ポリシーによって管理されるステージング環境にデプロイされた API を呼び出すクライアント アプリケーションを構築しています。
API を正常に呼び出すには何が必要ですか?
A. STAGING 環境で API を所有する Anypoint Platform アカウントのクライアント ID とシークレット
B. Anypoint Platform アカウントのステージング環境のクライアント ID とシークレット
C. STAGING環境のAPIインスタンスのAnypoint Exchangeから取得したクライアントIDとシークレット
D. Anypoint Platform から取得した有効な OAuth トークンとそれに関連付けられたクライアント ID およびシークレット
Answer: C (メーッセージを残す**)**

STAGING環境のAPIインスタンス用にAnypoint Exchangeから取得したクライアントIDとシークレット
****
>> APIにアクセスするためにAnypoint Platformアカウントまたは個々の環境のクライアントIDとシークレットを使用することはできません。
>> 問題の API に適用されるポリシーの種類は 「クライアント ID 強制ポリシー」であるため、OAuth トークンベースのアクセスは機能しません。
API にアクセスする正しい方法は、作業する特定の環境の API インスタンスに対して Anypoint Exchange から取得したクライアント ID とシークレットを使用することです。
参考文献:
API マネージャーでの API インスタンス契約の管理
https://docs.mulesoft.com/api-manager/1.x/request-access-to-api-task
https://docs.mulesoft.com/exchange/to-request-access
https://docs.mulesoft.com/api-manager/2.x/policy-mule3-client-id-based-policies

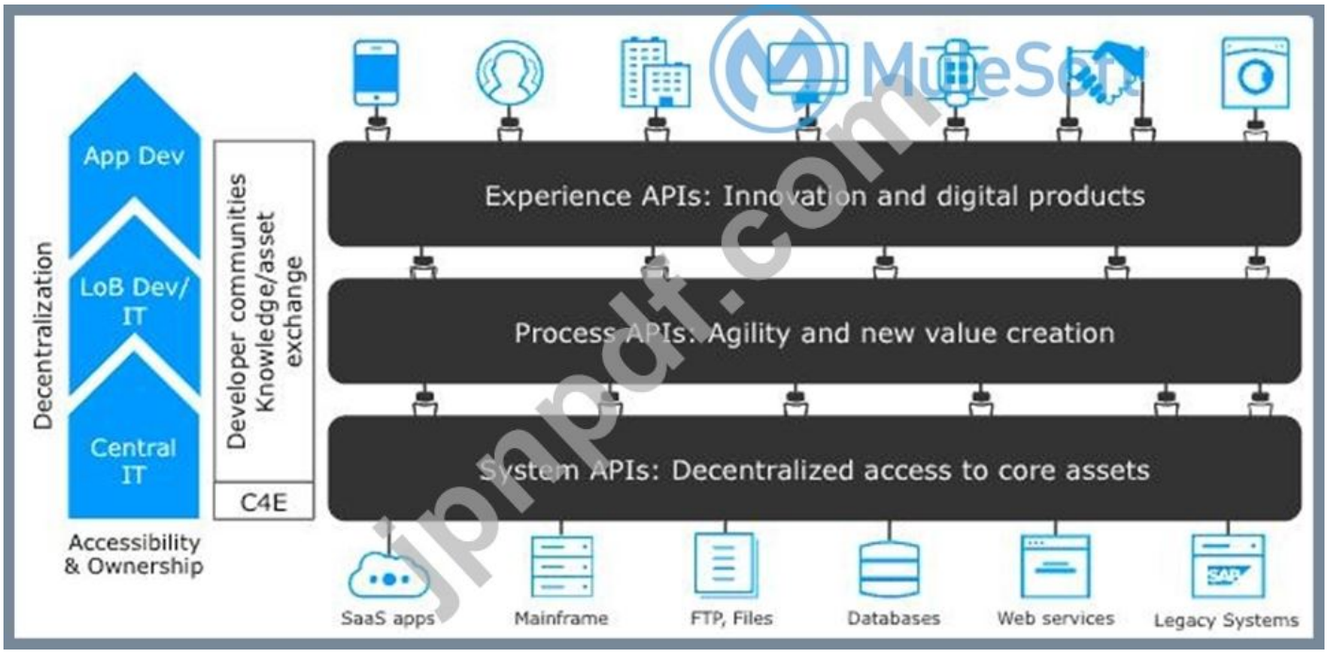
最新問題**: 26**
システム API は、スケーラビリティの課題があるバックエンド システムからデータを取得するように設計されています。
バックエンド システムを最も効果的に保護できる API ポリシーは何ですか?
A. IP ホワイトリスト
B. SLAベースのレート制限
C. OAuth 2 トークンの強制
D. クライアントIDの強制
Answer: (解答を表示する**)**
説明/参照: https://dzone.com/articles/how-to-secure-apis

最新問題**: 27**
A REST API is being designed to implement a Mule application.
What standard interface definition language can be used to define REST APIs?
A. YAML
B. OpenAPI Specification (OAS)
C. AsyncAPI Specification
D. Web Service Definition Language(WSDL)
Answer: B (メーッセージを残す**)**

最新問題**: 28**
Anypoint Platform で API ポリシーが定義される場所と、それが API インスタンスにどのように適用されるかについて正しいのはどれですか?
A. API ポリシーは、Mule ランタイムへの API デプロイメントの一部として Runtime Manager で定義され、特定の API インスタンスにのみ適用されます。

B. API ポリシーは、特定の API インスタンスに対して API Manager で定義され、特定の API インスタンスにのみ適用されます。
C. APIポリシーはAPI Managerで定義され、すべてのAPIインスタンスに自動的に適用されます。
D. APIポリシーはAPI Managerで定義され、指定された環境内のすべてのAPIインスタンスに適用されます。
Answer: (解答を表示する**)**
API ポリシーは、特定の API インスタンスに対して API Manager で定義され、特定の API インスタンスにのみ適用されます。
****
>> API 仕様が準備され、Exchange に公開されたら、API マネージャーにアクセスして、各 API の API インスタンスを登録する必要があります。
>> API マネージャーは、ポリシーを適用して NFR に対処するなど、API の側面の管理が行われる場所です。
>> 同じ API に対して複数のインスタンスを作成し、目的に応じて異なる方法で管理できます。
>> 1 つのインスタンスに API ポリシーのセットを適用し、同じ API の別のインスタンスに別の目的で異なるポリシーのセットを適用することができます。
>> これらの API とそのインスタンスは環境ごとに定義されます。したがって、各環境で個別に管理する必要があります。
>> プラットフォーム機能を使用して上位環境に昇格するときに、API インスタンスの同じ構成 (SLA、ポリシーなど) が昇格されることを保証できます。ただし、これはオプションのみです。必要に応じて、環境ごとに変更することもできます。
>> ランタイム マネージャーは、API 実装とその Mule ランタイムを管理する場所ですが、API 自体を管理する場所ではありません。API ポリシーは Mule ランタイムで実行されますが、ランタイム マネージャーで API ポリシーを適用することはできません。環境内の厳選されたインスタンスに対してのみ、API マネージャー経由でこれを行う必要があります。
したがって、これらの事実に基づくと、与えられた選択肢の正しい記述は、「API ポリシーは特定の API インスタンスに対して API Manager で定義され、特定の API インスタンスにのみ適用されます」です。

最新問題**: 29**
API 主導の接続性の定義に最も適したものは次のどれですか?
A. API 主導の接続性は、単なるアーキテクチャやテクノロジーではなく、組織内での効率的な IT 提供のために人材とプロセスを編成する方法でもあります。
B. API主導の接続性は、エクスペリエンス、プロセス、システム層をカバーする3層アーキテクチャです。
C. API主導の接続性は、エクスペリエンス、プロセス、システム層ベースのAPIを実装することを可能にする技術です。
Answer: A (メーッセージを残す**)**
正解: API 主導の接続性は、単なるアーキテクチャやテクノロジーではなく、組織内で効率的な IT 提供のために人材とプロセスを編成する方法でもあります。
****
参照：



最新問題: **30**
API 主導の接続のどのレイヤーにビジネス ロジック オーケストレーションが存在します?
A. システム層
B. エクスペリエンスレイヤー
C. プロセス層
Answer: C (メーッセージを残す**)**
プロセスレイヤー
****

>> エクスペリエンス レイヤーは、エンド ユーザー エクスペリエンスの強化を目的としています。このレイヤーは、さまざまな API クライアント/消費者のニーズを満たすためのものです。
>> システム層は、本質的にモジュール化されたAPI専用であり、バックエンドシステムのさまざまな個別の機能を実装/公開します。
>> プロセス レイヤーは、1 つまたは複数のシステム レイヤー モジュラー API を呼び出して、単純または複雑なビジネス オーケストレーション ロジックが記述される場所です。したがって、プロセス レイヤーが正しい答えです。

最新問題: **31**
システム API を構築する際のベストプラクティスは何ですか?
A. API実装とバックエンドシステムの相互作用に関するすべての技術的詳細をAPIクライアントに公開します。
B. RAML定義のような簡単に使用できるアセットを使用してAPIをドキュメント化する
C. すべてのAPIリソースとメソッドをモデル化して、バックエンドシステムの操作を厳密に模倣します。
D. 各バックエンドシステムのエンタープライズデータモデル（標準データモデル）を構築し、システムAPIに適用する
Answer: B (メーッセージを残す**)**

最新問題: **32**
アプリケーションは、在庫の一貫性を保つために、常に 1 つのプロセスのみを実行して在庫を更新します。このプロセスの実行には 200 ミリ秒 (0.2 秒) かかります。したがって、アプリケーションのスケーラビリティしきい値は 1 秒あたり 5 件のリクエストです。
水平スケーリングを適用して Mule ワーカーの数を増やすと、アプリケーションにどのような影響がありますか?
A. アプリケーションのスケーラビリティしきい値は、水平スケーリングに関係なく、1秒あたり5リクエストです。
B. プロセス実行時間の合計は 100 ミリ秒 (0.1 秒) になりました
C. アプリケーションのスケーラビリティしきい値は、1秒あたり10リクエストになりました。
D. すでに実行中のアプリケーションには水平スケーリングを適用できません
Answer: A (メーッセージを残す**)**
アプリケーションはデータの一貫性を維持するために一度に 1 つのプロセスのみを処理するように設計されているため、水平スケーリングによって処理制限が増加しない理由は次のとおりです。
* 単一プロセス制約:
* アプリケーションは一貫性を重視した設計のため、一度に 1 つのトランザクションを処理するように制限されています。つまり、水平スケーリング (ワーカーの追加) を行っても、この制限を超えて処理速度が上がることはありません。
* 実行時間:
* 各リクエストには 200 ミリ秒かかるため、1 秒あたり 5 件のリクエストが最大処理しきい値となります。ワーカーの数を増やしても、この単一プロセスの制限は回避されません。
* 正解 A)の説明：
* この制約はアプリケーションの設計に固有のものであるため、スケーラビリティは 1 秒あたり 5 件のリクエストのままです。
* 誤ったオプションの説明:
* オプション B は実行時間の変更を示唆しますが、これは水平スケーリングの影響を受けません。
* オプション C ではスループットが 2 倍になることを想定していますが、アプリケーションがシングルスレッドであるため、これは不可能です。
* オプション D は、水平スケーリングを適用できないことを示唆していますが、これは誤りです。ただし、このコンテキストではスケーリングによってスループットは向上しません。

参考資料: Mule アプリケーションのスケーリングと同時実行性の詳細については、アプリケーションのパフォーマンスとスケーリングの制限に関する MuleSoft のドキュメントを参照してください。

最新問題: **33**
ある組織では、今日の引用をキャッシュする Quote of the Day API を実装しています。
どのようなシナリオで、オブジェクト ストア コネクタを介して CloudHub オブジェクト ストアを使用して、キャッシュの状態を永続化できますか?
A. CloudHub への API 実装のデプロイメントが 1 つあり、顧客がホストする Mule ランタイムへの別のデプロイメントがあり、キャッシュ状態を共有する必要がある場合。
B. キャッシュ状態を共有する必要のある 3 つの CloudHub ワーカーに API 実装の CloudHub デプロイメントが 1 つある場合。
C. キャッシュ状態を共有する必要がある 3 つの別々の CloudHub リージョンに API 実装の 3 つの CloudHub デプロイメントがある場合。
D. キャッシュ状態を共有する必要がある同じ CloudHub リージョンに、2 つの Anypoint Platform ビジネス グループによる API 実装の 2 つの CloudHub デプロイメントがある場合。
Answer: A (メーッセージを残す**)**

最新問題: **34**
共有ロードバランサで CloudHub を使用する場合、Anypoint Platform ではなく API 実装 (Mule アプリケーション) によって排他的に管理されるものは何ですか?
A. API実装に割り当てられたDNSエントリの数
B. API実装がHTTPSエンドポイントを公開するために使用するSSL証明書
C. 各HTTPリクエストを特定のCloudHubワーカーに割り当てる
D. ログエントリをランタイムマネージャーで表示できるようにするログ設定
Answer: (解答を表示する**)**

最新問題: **35**
ある組織が、OrderStatus システム API の新しい実装を CloudHub の複数のワーカーにデプロイしています。この API は組織のオンプレミスの注文管理システムの前面に配置されており、API 実装によって IPsec トンネル経由でアクセスされます。
通常、OrderStatus システム API のサービス停止を引き起こさないエラーの種類は何ですか?
A. API 実装の初期展開中に API Manager が長時間停止しました
B. 関連するAWSデータセンターへの大規模なネットワーク障害によりAWSリージョンがオフラインになる
C. 組織のオンプレミス データ センターのネットワーク障害のため、注文管理システムにアクセスできません。
D. CloudHub ワーカーがメモリ不足例外で失敗する
Answer: C (メーッセージを残す**)**

最新問題: **36**
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: (解答を表示する**)**
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

最新問題**: 37**
What API policy would LEAST likely be applied to a Process API?
A. Rate limiting
B. Client ID enforcement
C. Custom circuit breaker
D. JSON threat protection
Answer: C (メッセージを残す**)**

最新問題**: 38**
A European company has customers all across Europe, and the IT department is migrating from an older platform to MuleSoft. The main requirements are that the new platform should allow redeployments with zero downtime and deployment of applications to multiple runtime versions, provide security and speed, and utilize Anypoint MQ as the message service.
Which runtime plane should the company select based on the requirements without additional network configuration?
A. Anypoint Runtime Fabric on Self-Managed Kubernetes for the runtime plane
B. MuleSoft-hosted runtime plane (CloudHub)
C. Runtime Fabric on VMs / Bare Metal for the runtime plane
D. Customer-hosted runtime plane
Answer: B (メッセージを残す**)**

最新問題**: 39**
API では、小さなメッセージ ペイロードに対してクライアント リクエスト (TPS) のレートが高くなります。クライアント アプリケーションの種類に基づいて、API に使用制限を課すにはどうすればよいでしょうか。
A. SLAベースのレート制限ポリシーを使用し、クライアントアプリケーションをそのタイプに基づいて一致するSLA層に割り当てます。
B. クライアントアプリケーションの種類ごとにリクエスト数を制限するスパイク制御ポリシーを使用する

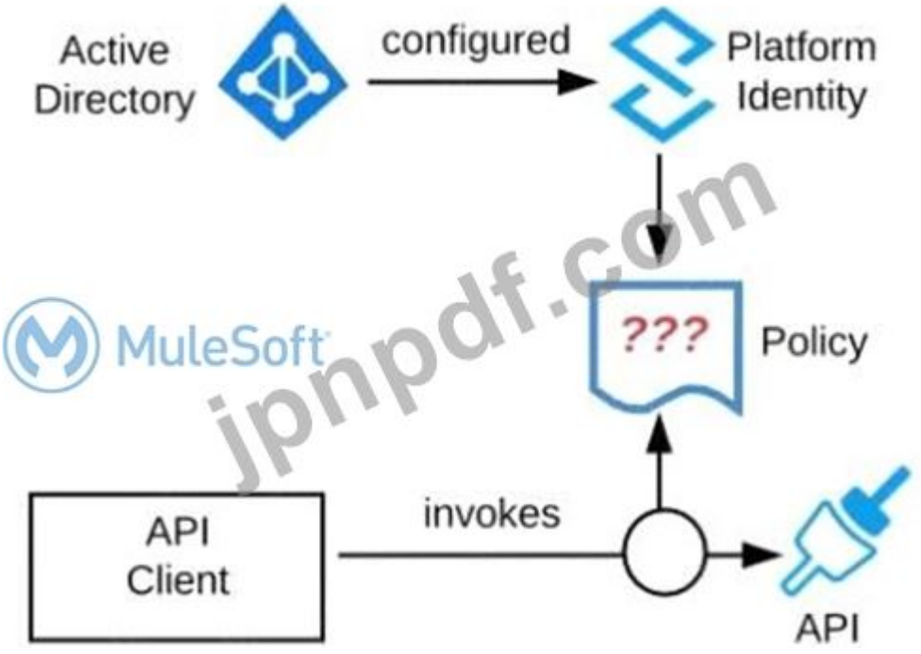
C. クライアント アプリケーションの種類によって設定された、クロス オリジン リソース共有 (CORS) ポリシーを使用して、クライアント アプリケーション間のリソース共有を制限します。
D. レート制限ポリシーとクライアントID強制ポリシーを使用します。それぞれクライアントアプリケーションの種類によって設定されます。
Answer: A (メーッセージを残す**)**
SLA ベースのレート制限ポリシーを使用し、クライアント アプリケーションをそのタイプに基づいて一致する SLA 層に割り当てます。
****
>> クライアントの種類に基づいてAPIに制限が課されるときはいつでも、SLA階層が作用します

最新問題**: 40**
ある組織では、今日の引用をキャッシュする Quote of the Day API を実装しています。
What scenario can use the GoudHub Object Store via the Object Store connector to persist the cache's state?
A. When there are three CloudHub deployments of the API implementation to three separate CloudHub regions that must share the cache state
B. When there is one CloudHub deployment of the API implementation to three CloudHub workers that must share the cache state
C. When there are two CloudHub deployments of the API implementation by two Anypoint Platform business groups to the same CloudHub region that must share the cache state
D. When there is one deployment of the API implementation to CloudHub and anottV deployment to a customer-hosted Mule runtime that must share the cache state
Answer: D (メーッセージを残す**)**

最新問題**: 41**
プロセス API の実装を変更する必要があります。
この変更が API クライアントに与える影響を最小限に抑える有効なアプローチは何ですか?
A. 現在のプロセス API の RAML 定義を更新し、更新された RAML 定義へのリンクを送信して API クライアント開発者に通知します。
B. APIコンシューマーが新しいプロセスAPIまたはAPIバージョンへの移行の準備ができていることを確認するまで、変更を延期します。
C. プロセスAPIの実装に必要な変更を実装し、可能な限りプロセスAPIのRAML定義が変更されないようにします。
D. Process API の変更を新しい API 実装に実装し、古い API 実装が HTTP ステータス コード 301 - Moved Permanently を返すようにして、API クライアントに新しい API 実装を呼び出す必要があることを通知します。
Answer: C (メーッセージを残す**)**
可能な限り、プロセス API の RAML 定義が変更されないように、プロセス API 実装に必要な変更を実装します。
****

質問の主な要件は次のとおりです。
>> APIクライアントに対するこの変更の影響を最小限に抑えるアプローチ
上記に基づき:
>> 変更によってクライアント側で何か必須のことが要求される場合、RAML 定義を更新すると API クライアントに影響する可能性があります。したがって、本当に必要なとき以外は、これを行わないようにしてください。
>> 変更を完全に異なる API として実装し、3xx ステータス コードでクライアントをリダイレクトすることは、設計を混乱させ、API クライアントに大きな影響を与えます。
>> 組織と IT 部門は、すべての API 利用者が新しいプロセス API または API バージョンに移行する準備ができていることを認めるまで、必要な変更を単純に延期することはできません。これは非現実的であり、不可能です。
変更を常に処理する最善の方法は、API 実装に必要な変更を実装して、可能な限り API の RAML 定義が変更されないようにすることです。

最新問題: **42**
展示を参照してください。組織は Mule スタンドアロン ランタイムを実行しており、Active Directory を Anypoint Platform 外部 ID プロバイダーとして構成しています。組織には他のシステム コンポーネントのための予算がありません。



特定の内部ユーザー グループへのアクセスを最も効果的に制限するには、組織内のすべての API インスタンスにどのようなポリシーを適用する必要がありますか?
A. 基本認証 - LDAP ポリシーを適用します。内部 Active Directory がユーザー認証用の LDAP ソースとして構成されます。
B. クライアントID強制ポリシーを適用します。特定のユーザーグループは、特定のクライアント資格情報を使用するようにクライアントアプリケーションを構成します。

C. IPホワイトリストポリシーを適用します。特定のユーザーのワークステーションのみがホワイトリストに追加されます。
D. OAuth 2.0アクセストークン強制ポリシーを適用します。内部Active DirectoryがOAuthサーバーとして構成されます。
Answer: A (メーッセージを残す**)**
基本認証 - LDAP ポリシーを適用します。内部 Active Directory がユーザー認証用の LDAP ソースとして構成されます。
****
>> IP ホワイトリストはこの目的には適していません。さらに、ユーザーのワークステーションは、ネットワーク内で必ずしも静的 IP を持つとは限りません。
>> OAuth 2.0 の適用には、組織のシステム コンポーネントに含まれていないクライアント プロバイダーが必要です。
>> すべてのユーザーが個別のクライアント資格情報を作成し、使用方法に合わせて構成できるようにするのは効果的なアプローチではありません。
効果的な方法は、基本認証 - LDAP ポリシーを適用することです。内部 Active Directory は、ユーザーを認証するための LDAP ソースとして構成されます。

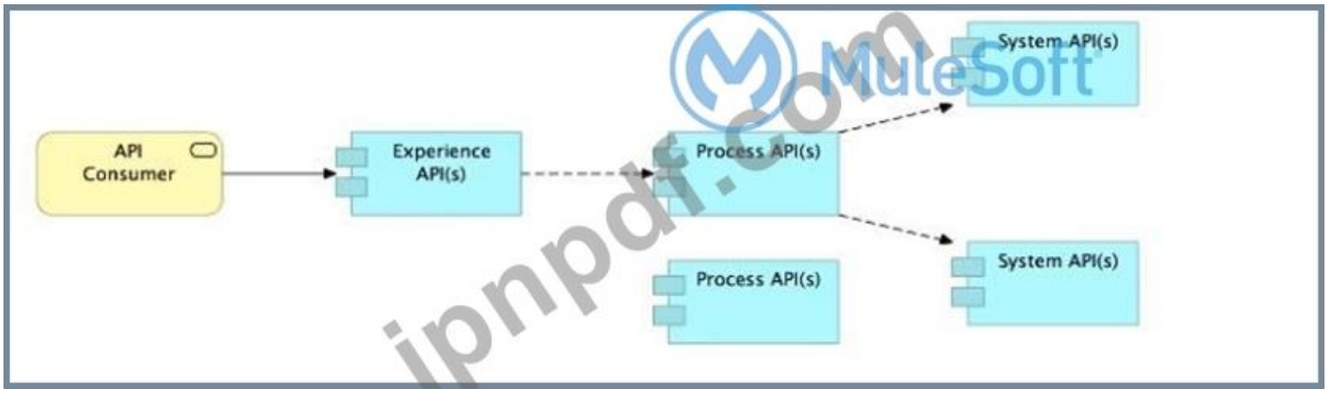
最新問題**: 43**
Anypoint Platform が提供する API 呼び出しメトリクスは何を提供しますか?
A. ビジネス ユーザーと直接共有できる API からの ROI メトリック
B. 特定の脅威しきい値を超える可能性のある将来のポリシー違反を積極的に特定する
C. 過去の API 呼び出しに関するデータ。さまざまな API の異常や使用パターンの特定に役立ちます。
D. 再利用レベルに基づくアプリケーションネットワークの有効性の測定
Answer: D (メーッセージを残す**)**

最新問題**: 44**
Anypoint Exchange では、API プロデューサーによって、承認されたセマンティック バージョン管理プラクティスに従って API がバージョン 3.1.1 から 3.2.0 に更新され、その変更は API のパブリック ポータルを通じて通知されました。
新しいバージョンでは API エンドポイントは変更されません。
API クライアントの開発者はこの変更にどのように対応すべきでしょうか?
A. 既存の機能の変更を理解するには、API プロデューサーに連絡する必要があります。
B. 更新はプロジェクトリスクとして識別され、このAPIを使用する機能の完全な回帰テストを実行する必要があります。
C. APIプロデューサーは、古いバージョンを新しいバージョンと並行して実行するように要求される必要があります。
D. APIクライアントコードは、新しい機能を利用する必要がある場合にのみ変更する必要があります。
Answer: (解答を表示する**)**

最新問題**: 45**

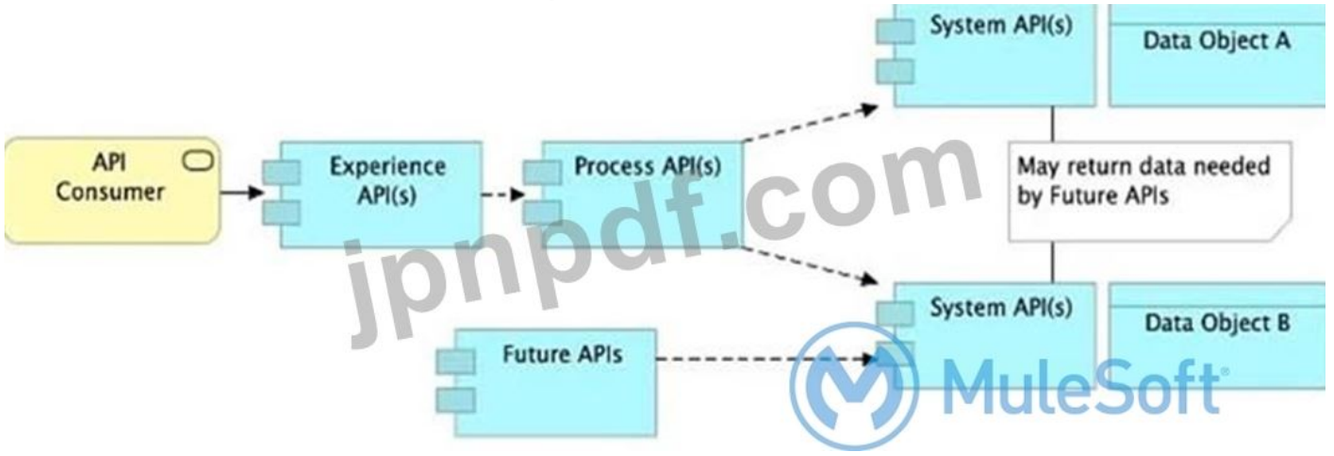
What Mule application deployment scenario requires using Anypoint Platform Private Cloud Edition or Anypoint Platform for Pivotal Cloud Foundry?
A. すべてのAPIがプライベートであり、パブリッククラウドに公開されないことが求められる場合
B. 規制要件により、メタデータを含むすべてのデータ項目のオンプレミス処理が義務付けられている場合
C. アプリケーションネットワーク内のすべてのバックエンドシステムが組織のイントラネットに展開されている場合
D. 複数のデータセンターにわたってすべてのアプリケーションの高可用性を実現する必要がある場合
Answer: C (メーッセージを残す**)**

最新問題: **46**
展示品を参照してください。
エンドツーエンドのビジネス プロセスをエクスペリエンス、プロセス、システム API のコラボレーションに分解する最適な方法は何ですか?
A) エンドユーザー アプリケーションのカスタマイズをエクスペリエンス API レベルではなくプロセス API レベルで処理する B) 特定のプロセス API またはエクスペリエンス API で現在必要とされていないデータをシステム API が返すことを許可する C) 3 つのレイヤー (エクスペリエンス API、プロセス API、システム API) ごとに 1 つの API を作成することにより、常に階層化アプローチを使用する D) プロセス API を使用して複数のシステム API への呼び出しを調整し、他のプロセス API への呼び出しは調整しない



A. オプションA
B. オプションB
C. オプションC
D. オプションD
Answer: (解答を表示する**)**
特定されたプロセス API またはエクスペリエンス API で現在必要とされていないデータをシステム API が返すことを許可します。
****
>> エンドユーザー アプリケーションのすべてのカスタマイズは、「エクスペリエンス API」でのみ処理する必要があります。プロセス API では処理できません。
>> 階層型アプローチを使用する必要がありますが、3 つのレイヤーごとに 1 つの API を常に作成する必要はありません。

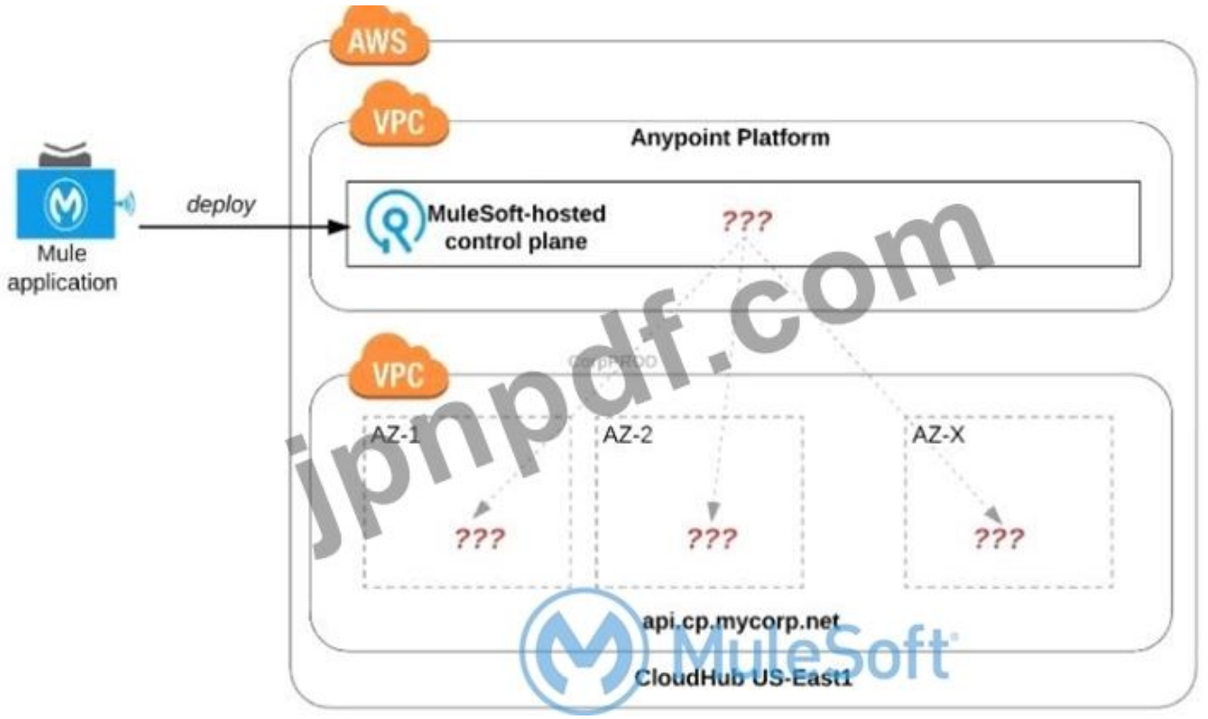
エクスペリエンス API は 1 つかもしれませんが、プロセス API とシステム API は複数になることがよくあります。システム API は、エンド システムの前に構築される最小のモジュール API であるため、常に複数になります。
>> プロセス API は、システム API だけでなく他のプロセス API も呼び出すことができます。API 主導の接続には、プロセス API が他のプロセス API を呼び出すべきではないという反設計パターンはありません。
したがって、API 主導の接続原則に従って意味をなす、指定されたオプション セットにおける正しい答えは、特定されたプロセス API またはエクスペリエンス API で現在必要とされていないデータをシステム API が返すようにすることです。
この方法により、将来のプロセス API はシステム API からそのデータを利用できるようになり、システム層 API に何度も触れる必要がなくなります。



最新問題: **47**
API 実装、API クライアント、API コンシューマーはどのように組み合わされて API を呼び出して処理するのでしょうか?
A. APIクライアントはAPIコンシューマーを作成し、APIからのAPI呼び出しを受信してAPI実装用に処理します。
B. ApIクライアントはAPIコンシューマーを作成し、API呼び出しをAPI実装によって処理されるようにAPIに送信します。
C. ApIコンシューマはAPIクライアントを作成し、API呼び出しをAPI実装によって処理されるようにAPIに送信します。
D. APIコンシューマはAPI実装を作成し、APIからのAPI呼び出しを受信してAPIクライアント用に処理します。

Answer: (解答を表示する**)**

最新問題: **48**
展示品を参照してください。



組織は、すべての CloudHub デプロイメントに対して 1 つの特定の CloudHub (AWS) リージョンを使用します。
組織の Mule アプリケーションがそのリージョンの CloudHub にデプロイされている場合、CloudHub ワーカーはどのようにしてアベイラビリティーゾーン (AZ) に割り当てられますか?
A. 特定の環境に属するワーカーは、そのリージョン内の同じ AZ に割り当てられます。
B. AZはMuleアプリケーションのデプロイメント構成の一部として選択されます
C. ワーカーは、そのリージョン内の利用可能なAZにランダムに分散されます。
D. Mule アプリケーションに対して AZ がランダムに選択され、Mule アプリケーションのすべての CloudHub ワーカーがその 1 つの AZ に割り当てられます。
Answer: (解答を表示する**)**
正解: ワーカーは、そのリージョン内の利用可能な AZ 全体にランダムに分散されます。
****
>> 現在、どの AWS リージョンを選択するかは制御できますが、どのアベイラビリティーゾーン (AZ) をどのワーカーに割り当てるかを決定するための構成やデプロイメント オプションを使用する制御はまったくありません。
>> 環境やアプリケーションに基づいてワーカーに AZ を割り当てることに関しても、プラットフォームには固定または暗黙のルールはありません。
>> これらは完全にランダムに割り当てられます。ただし、Cloudhub では、すべてのワーカーが同じアプリケーションに対して同じ AZ に割り当てられないように、ワーカーを複数の AZ に割り当てることで HA が確実に実現されます。

参照：

最新問題: **49**

Anypoint Platform が提供する 4 つの重要なプラットフォーム機能は何ですか?

A. API のバージョン管理、API ランタイムの実行とホスティング、API の呼び出し、API コンシューマーのエンゲージメント

B. API の設計と開発、API ランタイムの実行とホスティング、API のバージョン管理、API の廃止

C. API の設計と開発、API ランタイムの実行とホスティング、API の運用と管理、API コンシューマーのエンゲージメント

D. API の設計と開発、API の廃止、API のバージョン管理、API 消費者エンゲージメント

Answer: C (メーッセージを残す**)**

正解: API の設計と開発、API ランタイムの実行とホスティング、API の運用と管理、API コンシューマーエンゲージメント

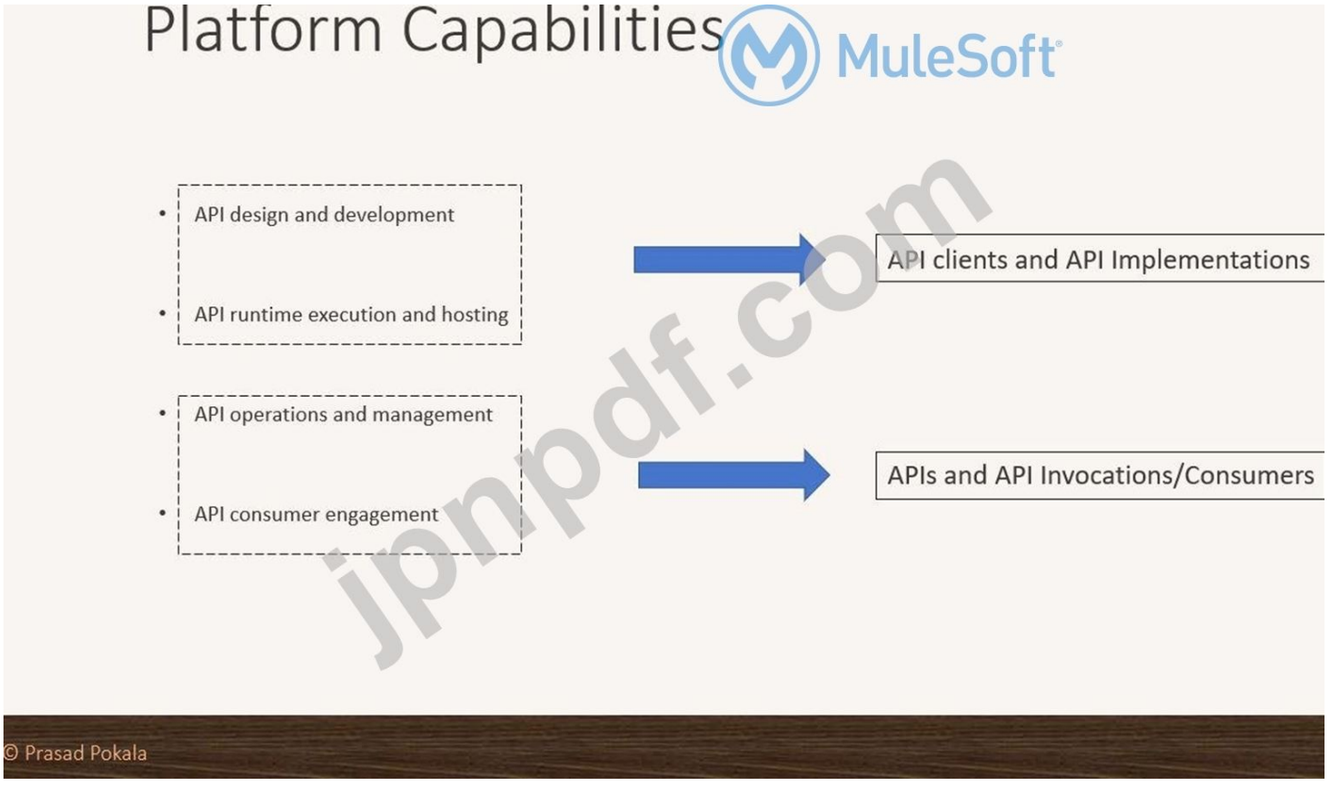
****

>> API 設計と開発 - Anypoint Studio、Anypoint Design Center、Anypoint Connectors

>> API ランタイム実行とホスティング - Mule ランタイム、CloudHub、ランタイム サービス

>> API 運用と管理 - Anypoint API Manager、Anypoint Exchange

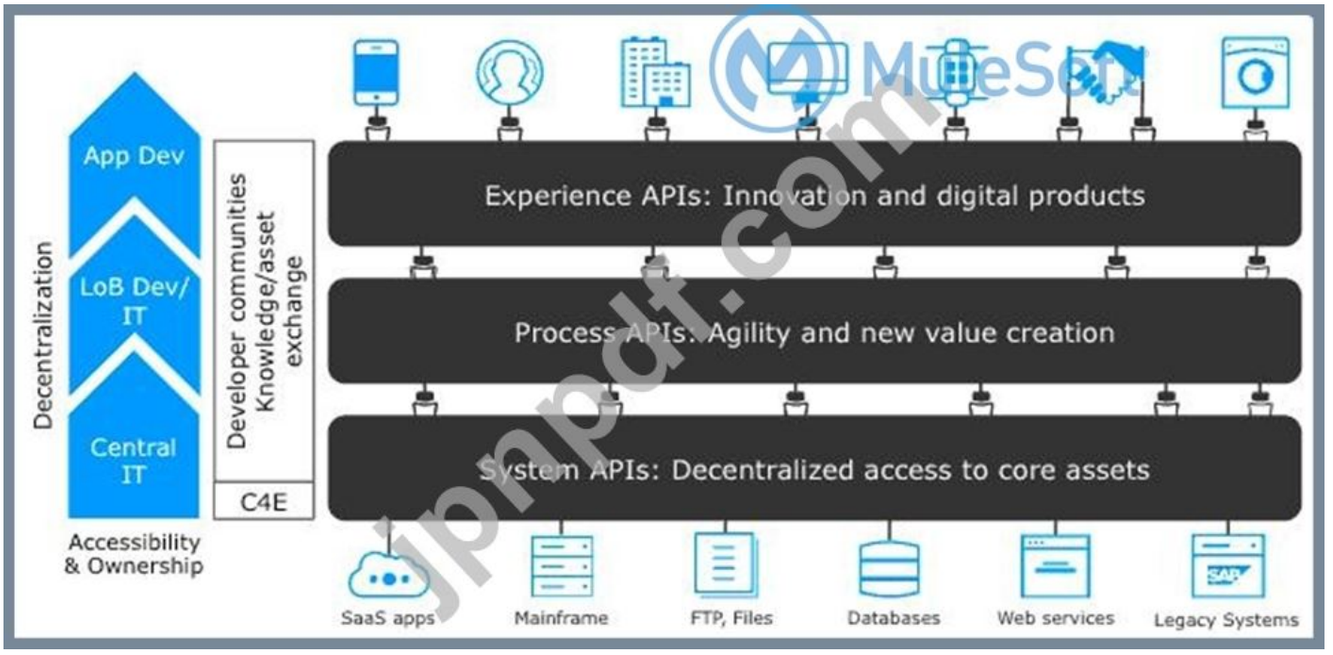
>> API コンシューマー管理 - API 契約、パブリック ポータル、Anypoint Exchange、API ノートブック



最新問題: **50**
多数の REST API を実装する Mule アプリケーションは、組織外部からアクセスできない独自のサブネットにデプロイされます。
外部のビジネス パートナーはこれらの API にアクセスする必要がありますが、これらの API は、パートナー専用の別のサブネット (パートナー サブネット) からのみ呼び出すことができます。このサブネットはパブリック インターネットからアクセス可能であり、外部のパートナーがアクセスできます。
Anypoint Platform および Mule ランタイムはすでにパートナーサブネットにデプロイされています。これらの Mule ランタイムはすでに API にアクセスできます。
現在 API を使用している他のアプリケーションへの影響を最小限に抑えながら、これらの要件に準拠するための最もリソース効率の高いソリューションは何ですか?
A. パートナーによる利用のために各APIに追加のエンドポイントを追加します。
B. APIごとにAPIプロキシMuleアプリケーションを実装 または生成）し、APIプロキシをMuleランタイムにデプロイします。
C. API を Mule アプリケーションとして複製し、Mule ランタイムにデプロイします。
D. Mule ランタイムを実行している同じサーバーに API 実装を再デプロイします。
Answer: D (メーッセージを残す**)**

最新問題: **51**
API 主導の接続性のどのレイヤーが、主要なシステム、レガシー システム、データ ソースなどのロックを解除し、機能を公開することに重点を置いていますか?
A. エクスペリエンスレイヤー
B. プロセス層

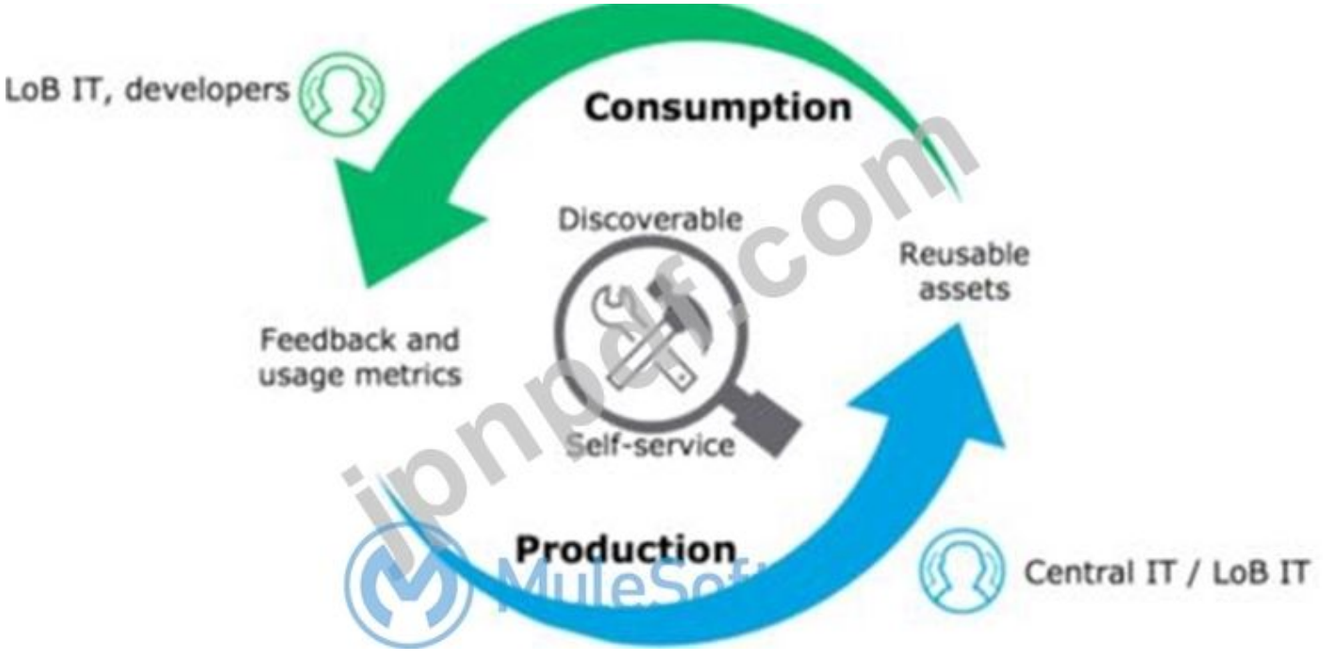
C. システム層
Answer: (解答を表示する**)**
正解: システム層



API 主導の接続アプローチで使用される API は、次の 3 つのカテゴリに分類されます。
システム API - これらは通常、レコードのコア システムにアクセスし、ユーザーを基盤システムの複雑さや変更から隔離する手段を提供します。一度構築されると、多くのユーザーは基盤システムを学習することなくデータにアクセスでき、複数のプロジェクトでこれらの API を再利用できます。
プロセス API - これらの API は、単一のシステム内またはシステム間で (データ サイロを解体して) データと対話してデータを形成します。これらの API は、データの元となるソース システムや、そのデータが配信されるターゲット チャネルに依存せずにここで作成されます。
エクスペリエンス API - エクスペリエンス API は、各チャネルに個別のポイントツーポイント統合を設定するのではなく、共通のデータ ソースからデータを再構成して、対象ユーザーが最も簡単にデータを利用できるようにするための手段です。エクスペリエンス API は通常、API ファーストの設計原則に基づいて作成され、特定のユーザー エクスペリエンスを念頭に置いて API が設計されます。

最新問題**: 52**
以下のどれを組み合わせて使用すると、IT 運用モデルが効果的になりますか?
A. 再利用可能な資産を作成し、作成した資産を組織全体でマーケティングし、資産が消費されているかどうかを確認するために定期的に LOB レビューを実施します。
B. 再利用可能なアセットを作成し、LOB チームがセルフサービスで API を参照できるように検出可能にし、アクティブなフィードバックと使用状況の指標を取得します。
C. 再利用可能なアセットを作成し、LOB チームがセルフサービスで API を参照できるように、それらを検出可能にします。
Answer: (解答を表示する**)**
正解: 再利用可能なアセットを作成し、それらを検出可能にして LOB チームがセルフサービスで API を参照できるようにし、アクティブなフィードバックと使用状況メトリックを取得します。

****



最新問題: **53**
ある組織では、今日の引用をキャッシュする Quote of the Day API を実装しています。
A. オブジェクト ストア コネクタを介して GoudHub オブジェクト ストアを使用してキャッシュの状態を永続化できるシナリオは何ですか?
B. CloudHubへのAPI実装のデプロイメントが1つあり、顧客がホストするMuleランタイムへのottVデプロイメントが1つあり、キャッシュ状態を共有する必要がある場合
C. API実装のCloudHubデプロイメントが1つあり、キャッシュ状態を共有する必要がある3つのCloudHubワーカーがある場合
D. 2つのAnypoint Platformビジネスグループによる API実装の2つのCloudHubデプロイメントが同じCloudHubリージョンにあり、キャッシュ状態を共有する必要がある場合
E. API実装のCloudHubデプロイメントが3つあり、キャッシュ状態を共有する必要がある3つの別々のCloudHubリージョンにある場合
Answer: D (メーッセージを残す**)**

最新問題: **54**
ある組織では、今日の引用をキャッシュする Quote of the Day API を実装しています。
A. CloudHubへのAPI実装のデプロイメントが1つあり、顧客がホストするMuleランタイムへのottVデプロイメントが1つあり、キャッシュ状態を共有する必要がある場合
B. API実装のCloudHubデプロイメントが3つあり、キャッシュ状態を共有する必要がある3つの別々のCloudHubリージョンにある場合
C. オブジェクト ストア コネクタを介して GoudHub オブジェクト ストアを使用してキャッシュの状態を永続化できるシナリオは何ですか?
D. API実装のCloudHubデプロイメントが1つあり、キャッシュ状態を共有する必要がある3つのCloudHubワーカーがある場合

E. 2つのAnypoint Platformビジネスグループによる API実装の2つのCloudHubデプロイメントが同じCloudHubリージョンにあり、キャッシュ状態を共有する必要がある場合
Answer: A (メーッセージを残す**)**

最新問題**: 55**
CloudHub 専用ロードバランサーを使用する必要がある条件は何ですか?
A. 同じ Mule アプリケーションの別々のデプロイメント間でクロスリージョン負荷分散が必要な場合
B. 顧客がホストする Mule ランタイムにデプロイされた API 実装にカスタム DNS 名が必要な場合
C. 複数の CloudHub ワーカー間での API 呼び出しを負荷分散する必要がある場合
D. API実装とAPIクライアント間でサーバー側負荷分散TLS相互認証が必要な場合
Answer: D (メーッセージを残す**)**
正解: API実装とAPIクライアント間でサーバー側負荷分散TLS相互認証が必要な場合
****
事実/メモリのヒント: CloudHub 専用ロードバランサには多くの利点がありますが、検討する際に心に留めておくべき重要な点が 2 つあります。
>> CloudHub にデプロイされたアプリにカスタム DNS 名を持つ URL エンドポイントを設定する
>> HTTPS と双方向 (相互) 認証の両方に対してカスタム証明書を構成します。
これに関して提供されているオプションについて
>> 私たち
DLB を使用して、同じ Mule アプリケーションの個別のデプロイメント間でクロスリージョン負荷分散を実行することはできません。
>> 複数の DLB URL が同じ Mule アプリを指すようにマッピング ルールを設定できます。ただし、その逆 (複数の Mule アプリが同じ DLB URL を持つ) は不可能です。
>> DLB は Cloudhub にデプロイされた Mule アプリのカスタム DNS 名の設定に役立ちますが、顧客がホストする Mule ランタイムにデプロイされたアプリには当てはまりません。
>> DLB を使用して複数の CloudHub ワーカー間で API 呼び出しの負荷を分散できることは事実ですが、必須ではありません。SLB (共有ロード バランサ) を使用しても同じこと (負荷分散) を実現できます。これを実現するために必ずしも DLB が必要というわけではありません。
したがって、シナリオに適合し、DLB を使用する必要がある唯一の適切なオプションは、API 実装と API クライアント間で TLS 相互認証が必要な場合です。

最新問題**: 56**
チームは Experience API 仕様の強化を計画しており、API 主導の接続設計原則に従っています。
API を強化する動機は何ですか?
A. 主要なAPIコンシューマーは、特定の種類のエンドポイントをCenter for Enablement標準からコンシューマーシステム標準に変更することを望んでいます。
B. 基盤となるシステムAPIが更新され、頻繁に使用されるリソースのより詳細なデータが提供されるようになりました。
C. 開発環境とステージング環境のAPIインスタンスにIP許可リストポリシーが追加されています
D. APIに含まれるいくつかの種類のデータに影響を与える標準データモデルが採用されています。

Answer: D (メーッセージを残す)
API 主導の設計では、Experience API が強化され、エンドユーザー アプリケーションへのデータの配信方法が改善されます。
Experience API を強化する主な理由の 1 つは、標準データ モデルなどの新しいデータ標準が採用される場合です。その理由は次のとおりです。
* 標準データモデル (CDM):
* CDM を採用すると、組織全体でデータ表現が標準化され、API の一貫性が向上し、さまざまなサービスやアプリケーションで使いやすくなります。
* Experience API を更新すると、この標準化された形式でデータが提供されるようになり、相互運用性と再利用性が向上します。
* 正解（D）の説明：
* CDM は API が提供するデータの構造とタイプに影響を与えます。また、この更新は、アプリケーションの主要なインタラクション ポイントである Experience API に直接関係します。
* 誤ったオプションの説明:
* オプション A では、消費者固有の標準に適応することになり、API 主導の設計原則に反します。
* オプション B にはシステム API の変更が含まれますが、データ形式の調整が必要な場合を除き、エクスペリエンス API への変更は直接的には必須ではありません。
* オプション C (IP 許可リスト) は、API 設計ではなくセキュリティに関連し、API の機能強化を促すものではありません。
参考資料 API 主導のアーキテクチャでの標準データ モデルの使用の詳細については、MuleSoft のデータ標準化に関するガイドラインと Experience API のベスト プラクティスを参照してください。

最新問題: **57**
プロセス API の実装を変更する必要があります。
この変更が API クライアントに与える影響を最小限に抑える有効なアプローチは何ですか?
A. Process API の変更を新しい API 実装に実装し、古い API 実装が HTTP ステータス コード 301 - Moved Permanently を返すようにして、API クライアントに新しい API 実装を呼び出す必要があることを通知します。
B. APIコンシューマーが新しいプロセスAPIまたはAPIバージョンへの移行の準備ができていることを確認するまで、変更を延期します。
C. プロセスAPIの実装に必要な変更を実装し、可能な限りプロセスAPIのRAML定義が変更されないようにします。
D. 現在のプロセス API の RAML 定義を更新し、更新された RAML 定義へのリンクを送信して API クライアント開発者に通知します。
Answer: C (メーッセージを残す)

最新問題: **58**
どの Mule アプリケーション展開シナリオで、Anypoint Platform Private Cloud Edition または Anypoint Platform for Pivotal Cloud Foundry を使用する必要がありますか?

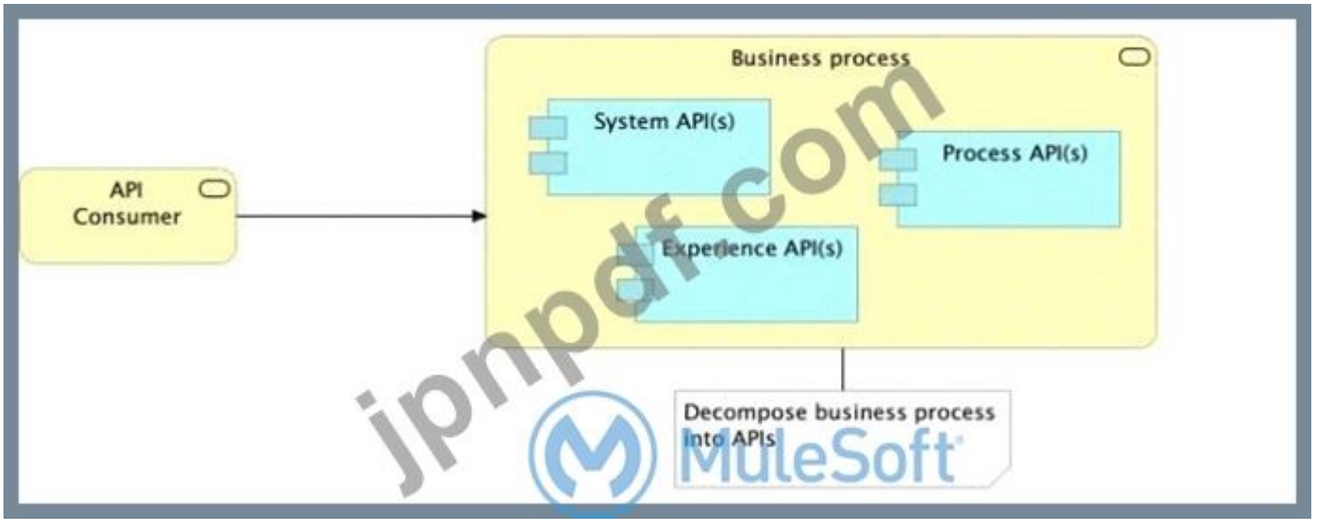
A. 複数のデータセンターにわたってすべてのアプリケーションの高可用性を実現する必要がある場合
B. すべてのAPIがプライベートであり、パブリッククラウドに公開されないことが求められる場合
C. 規制要件により、メタデータを含むすべてのデータ項目のオンプレミス処理が義務付けられている場合
D. アプリケーションネットワーク内のすべてのバックエンドシステムが組織のイントラネットに展開されている場合
Answer: C (メーッセージを残す**)**
正解: 規制要件により、メタデータを含むすべてのデータ項目のオンプレミス処理が義務付けられている場合。
****
以下の場合、Anypoint Platform PCE または PCF を使用する必要はありません。したがって、これらのオプションは無効です。
>> CloudHub を使用すると、複数のデータセンターにわたってすべてのアプリケーションの高可用性を実現できます。
>> Anypoint VPN と CloudHub からのトンネリングを使用して、組織のイントラネットに展開されているアプリケーション ネットワーク内のすべてのバックエンド システムに接続できます。
>> Anypoint VPC とファイアウォール ルールを使用して、すべての API をプライベートにし、パブリック クラウドに公開しないようにすることができます。
指定されたオプションの中で、Anypoint Platform PCE/PCF の使用が求められる唯一の有効な理由は、規制要件により、メタデータを含むすべてのデータ項目のオンプレミス処理が義務付けられている場合です。

最新問題**: 59**
Anypoint Platform REST API、Anypoint CLI、Mule Maven プラグインなどのツールを使用して Anypoint Platform とのやり取りを自動化することについて正しいのはどれですか?
A. Anypoint Platform API と Anypoint CLI へのアクセスは、Anypoint Platform のロールと権限を通じて個別に制御できるため、特定のユーザーは Anypoint CLI にアクセスでき、他のユーザーはプラットフォーム API にアクセスできます。
B. デフォルトでは、Anypoint CLI と Mule Maven プラグインは Mule ランタイムに含まれていないため、デプロイされた Mule アプリケーションでは使用できません。
C. API ポリシーを Anypoint Platform API に適用して、特定の LOB のみが特定の機能にアクセスできるようにすることができます。
D. Anypoint Platform API は CloudHub とのやり取りのみを自動化できますが、顧客がホストする Mule ランタイムへの展開には Mule Maven プラグインが必要です。
Answer: B (メーッセージを残す**)**

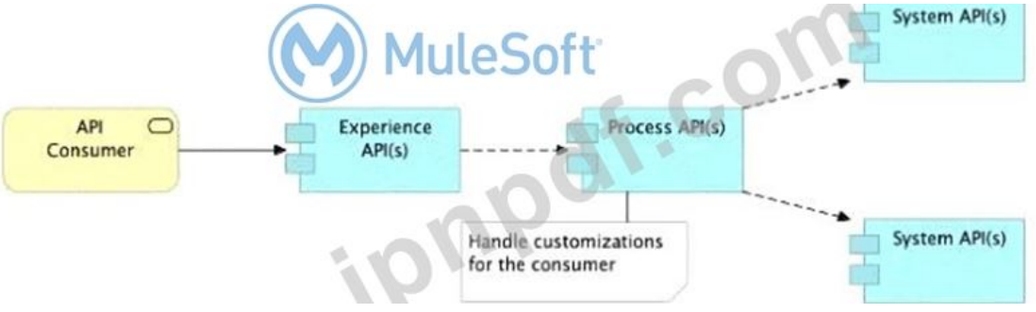
最新問題**: 60**
API 主導の接続性と呼ばれるフレームワーク内で作成された API の機能で通常含まれないものは何ですか?
A. 基盤となるバックエンド システムの上に追加の耐障害性レイヤーを提供し、それによってクライアントをこれらのシステムの長期にわたる障害から保護します。

B. バックエンド システムからデータがどのように抽出されるかを意識せずに、基礎となる資産を消費することで、ユーザー インターフェイス レベルでのイノベーションを可能にします。
C. バックエンド システムからデータを再利用可能かつ消費可能な方法でロック解除することで、基盤となるバックエンド システムへの依存を軽減します。
D. さまざまなソースからデータを作成し、オーケストレーション ロジックと組み合わせて、より高いレベルの価値を生み出すことができます。
Answer: A (メーッセージを残す**)**
説明
https://dzone.com/articles/api-led-connectivity-with-mule

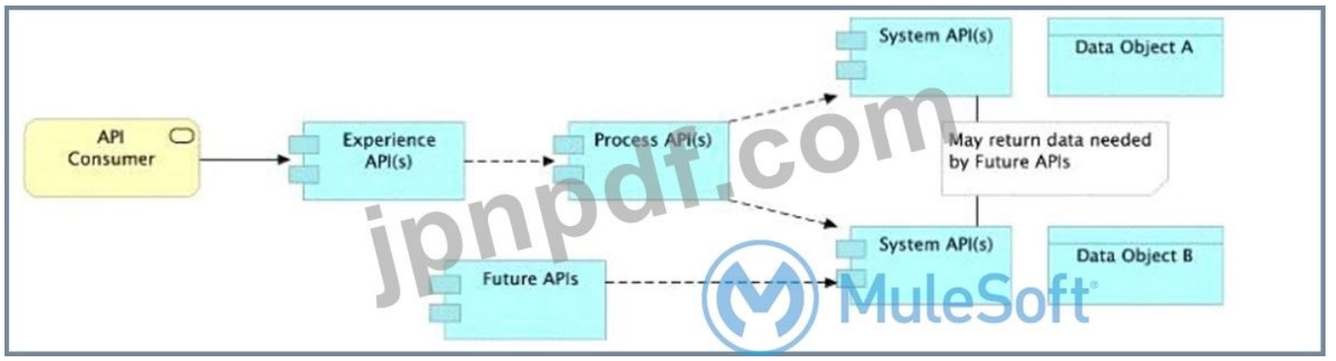
最新問題**: 61**
展示品を参照してください。



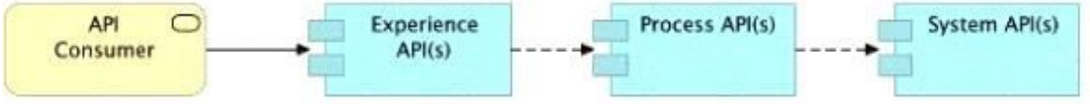
エンドツーエンドのビジネス プロセスをエクスペリエンス、プロセス、システム API のコラボレーションに分解する最適な方法は何ですか?
A) エンドユーザー アプリケーションのカスタマイズをエクスペリエンス API レベルではなくプロセス API レベルで処理する



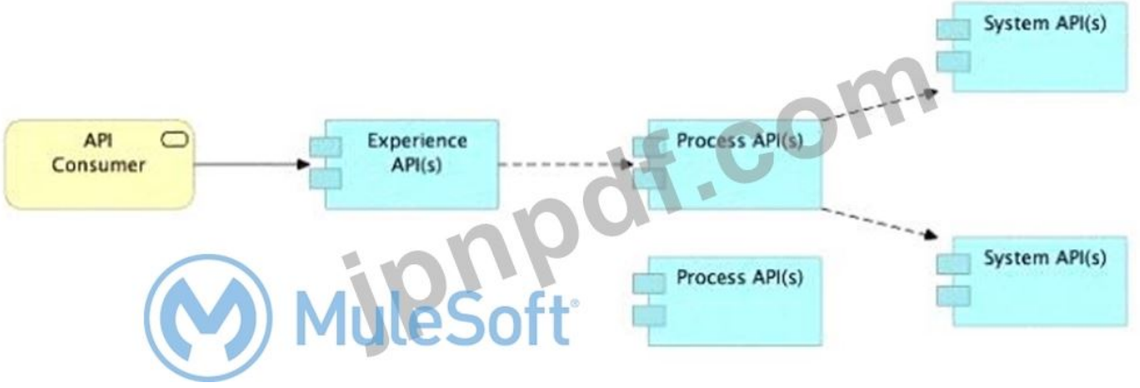
B) システムAPIが、特定されたプロセスAPIまたはエクスペリエンスAPIで現在必要とされていないデータを返すことを許可する



C) 3 つのレイヤー (エクスペリエンス、プロセス、システム API) ごとに 1 つの API を作成し、常に階層化アプローチを使用します。

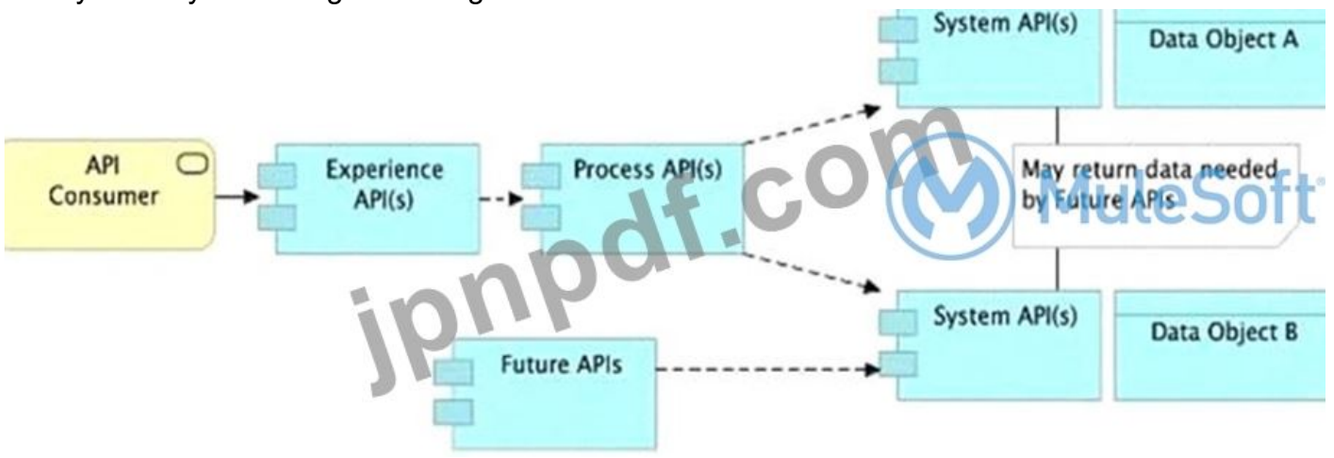


D)Use a Process API to orchestrate calls to multiple System APIs, but NOT to other Process APIs



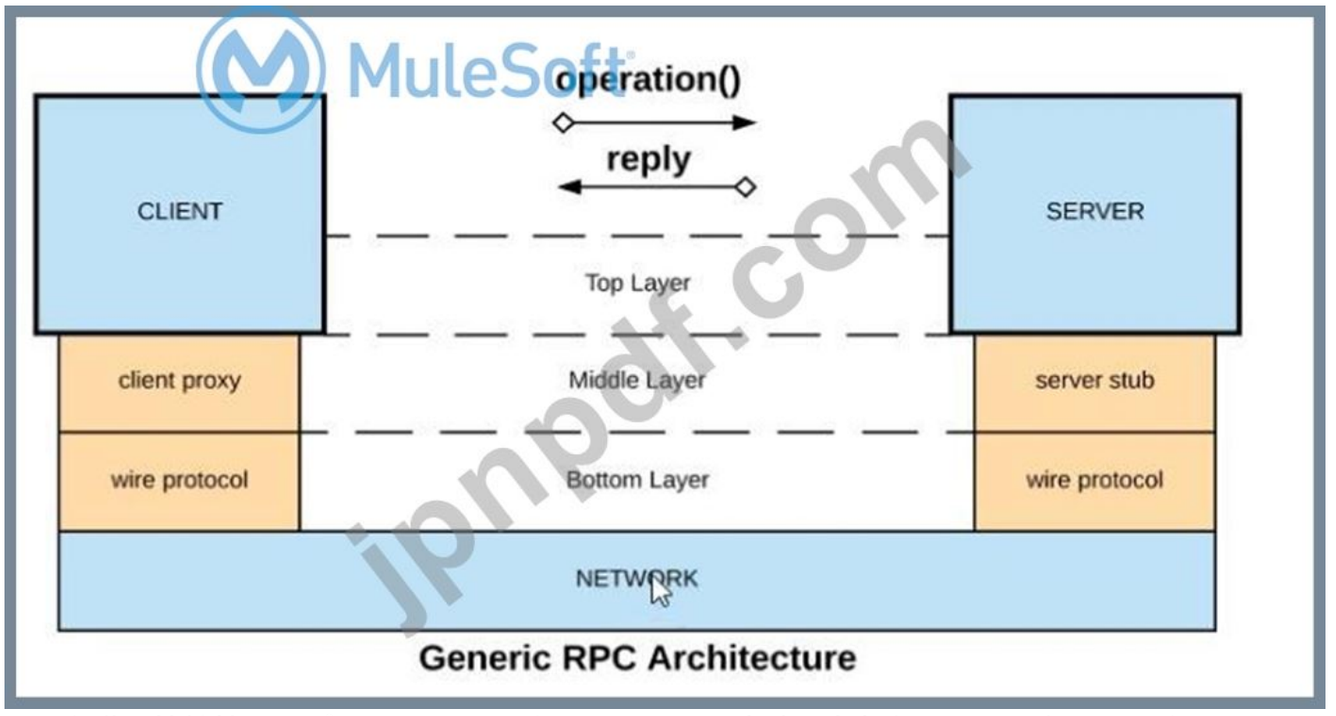
A. Option A
B. Option B
C. Option C
D. Option D
Answer: (解答を表示する**)**
Allow System APIs to return data that is NOT currently required by the identified Process or Experience APIs.
****
>> All customizations for the end-user application should be handled in "Experience API" only. Not in Process API
>> We should use tiered approach but NOT always by creating exactly one API for each of the 3 layers. Experience APIs might be one but Process APIs and System APIs are often more than one. System APIs for sure will be more than one all the time as they are the smallest modular APIs built in front of end systems.
>> Process APIs can call System APIs as well as other Process APIs. There is no such anti-design pattern in API-Led connectivity saying Process APIs should not call other Process APIs.
So, the right answer in the given set of options that makes sense as per API-Led connectivity principles is to allow System APIs to return data that is NOT currently required by the identified Process or Experience APIs.

This way, some future Process APIs can make use of that data from System APIs and we need NOT touch the System layer APIs again and again.



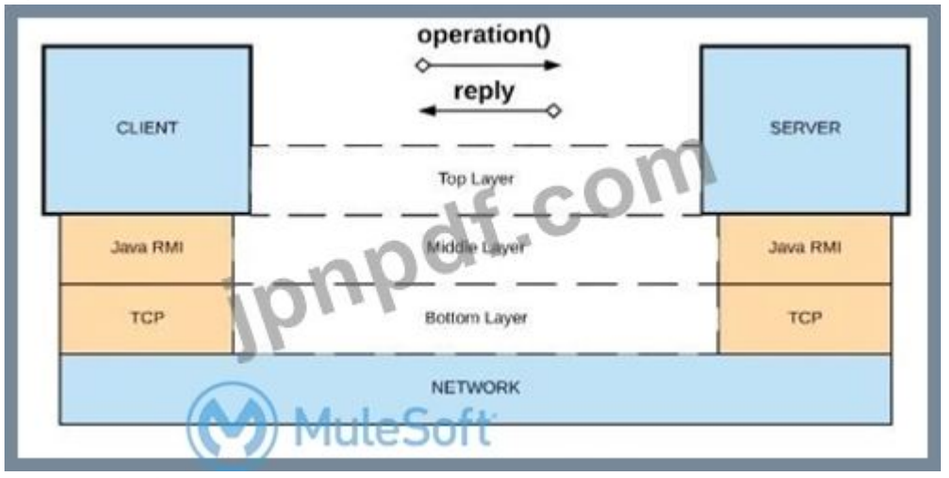
最新問題: **62**
What should be ensured before sharing an API through a public Anypoint Exchange portal?
A. The visibility level of the API instances of that API that need to be publicly accessible should be set to public visibility
B. The users needing access to the API should be added to the appropriate role in Anypoint Platform
C. The API should be functional with at least an initial implementation deployed and accessible for users to interact with
D. The API should be secured using one of the supported authentication/authorization mechanisms to ensure that data is not compromised
Answer: (解答を表示する**)**
Explanation
https://docs.mulesoft.com/exchange/to-share-api-asset-to-portal

最新問題: **63**
展示品を参照してください。

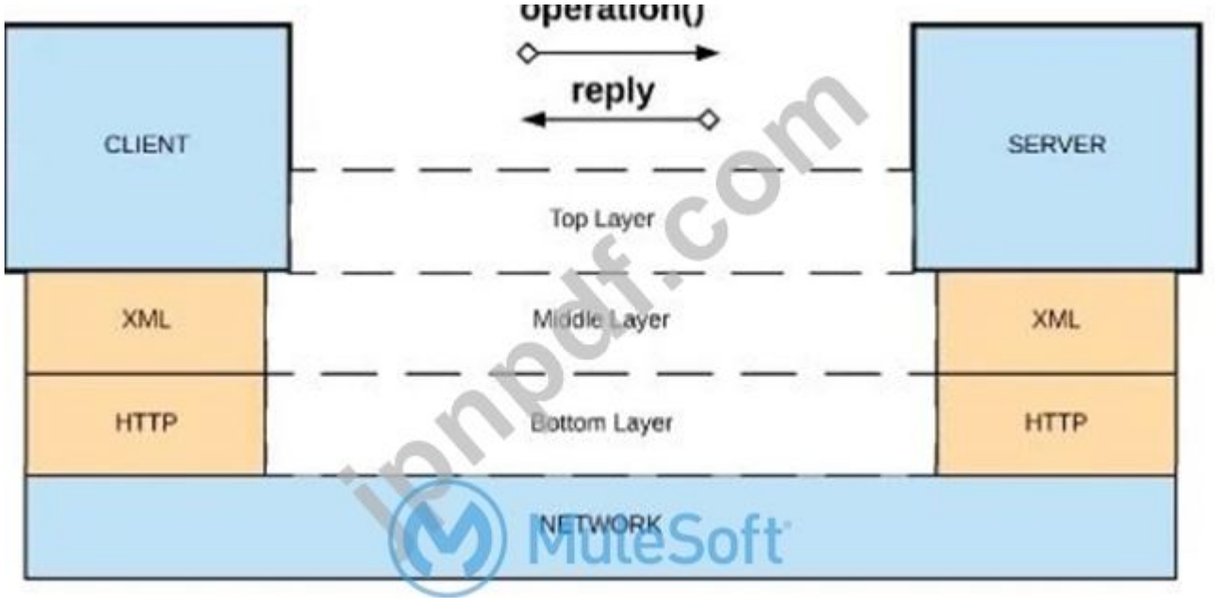


API 主導の接続性とアプリケーション ネットワークの意味で有効な API とは何でしょうか?

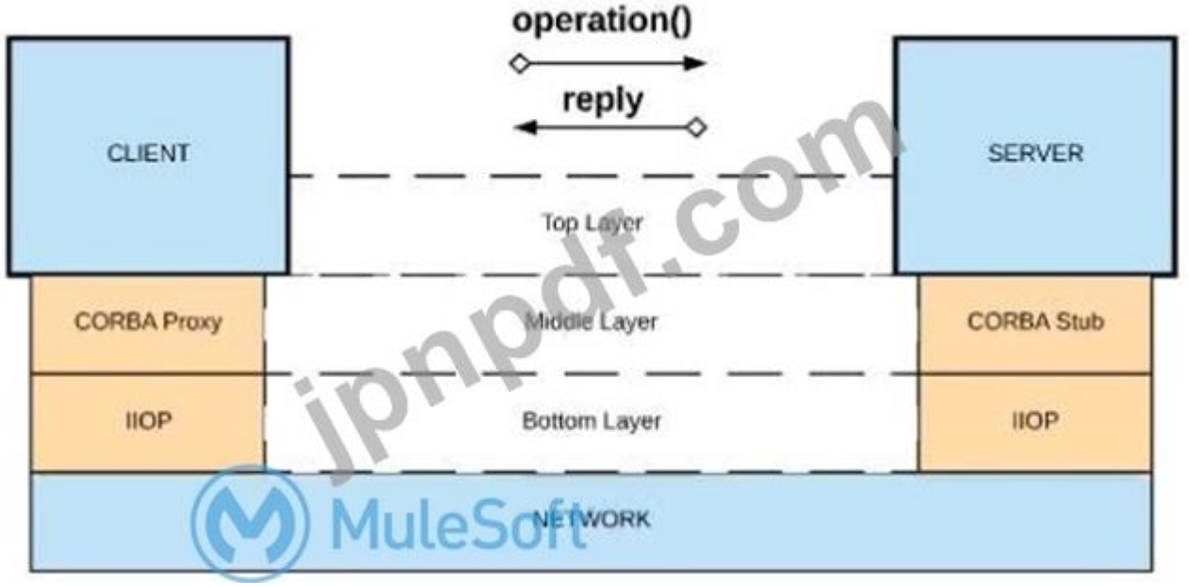
A) TCP 経由の Java RMI



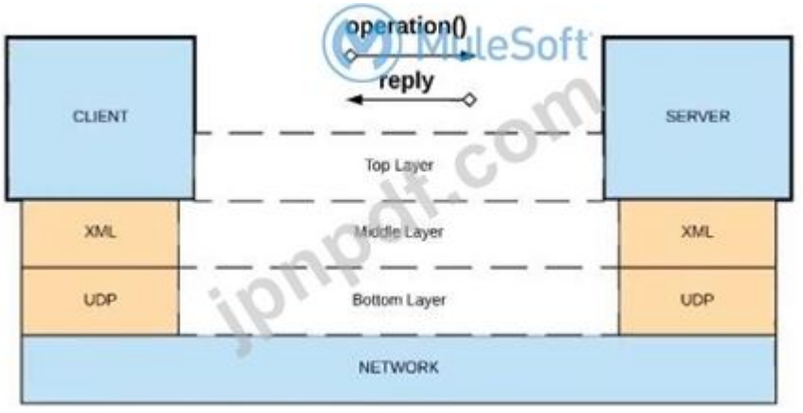
B) TCP 経由の Java RMI



C) HOP 経由の CORBA



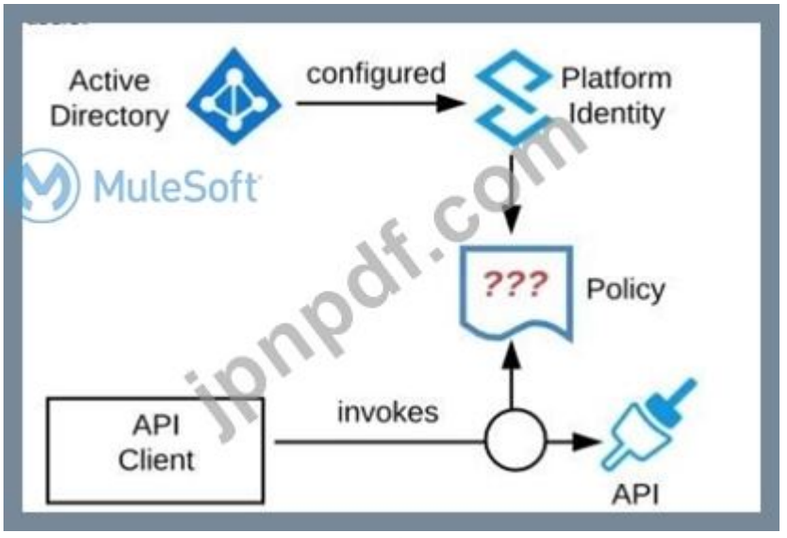
D) UDP経由のXML



A. オプションD
B. オプションB
C. オプションC

D. オプションA
Answer: B (メッセージを残す**)**

最新問題**: 64**
展示を参照してください。組織は Mule スタンドアロン ランタイムを実行しており、Active Directory を Anypoint Platform 外部 ID プロバイダーとして構成しています。組織には他のシステム コンポーネントのための予算がありません。



特定の内部ユーザー グループへのアクセスを最も効果的に制限するには、組織内のすべての API インスタンスにどのようなポリシーを適用する必要がありますか?
A. OAuth 2.0アクセストークン強制ポリシーを適用します。内部Active DirectoryがOAuthサーバーとして構成されます。
B. クライアントID強制ポリシーを適用します。特定のユーザーグループは、特定のクライアント資格情報を使用するようにクライアントアプリケーションを構成します。
C. 基本認証 - LDAP ポリシーを適用します。内部 Active Directory がユーザー認証用の LDAP ソースとして構成されます。
D. IPホワイトリストポリシーを適用します。特定のユーザーのワークステーションのみがホワイトリストに追加されます。
Answer: C (メッセージを残す**)**

最新問題**: 65**
API では、小さなメッセージ ペイロードに対してクライアント リクエスト (TPS) のレートが高くなります。クライアント アプリケーションの種類に基づいて、API に使用制限を課すにはどうすればよいでしょうか。
A. クライアントアプリケーションの種類ごとにリクエスト数を制限するスパイク制御ポリシーを使用する
B. クライアント アプリケーションの種類によって設定された、クロス オリジン リソース共有 (CORS) ポリシーを使用して、クライアント アプリケーション間のリソース共有を制限します。
C. SLAベースのレート制限ポリシーを使用し、クライアントアプリケーションをそのタイプに基づいて一致するSLA層に割り当てます。

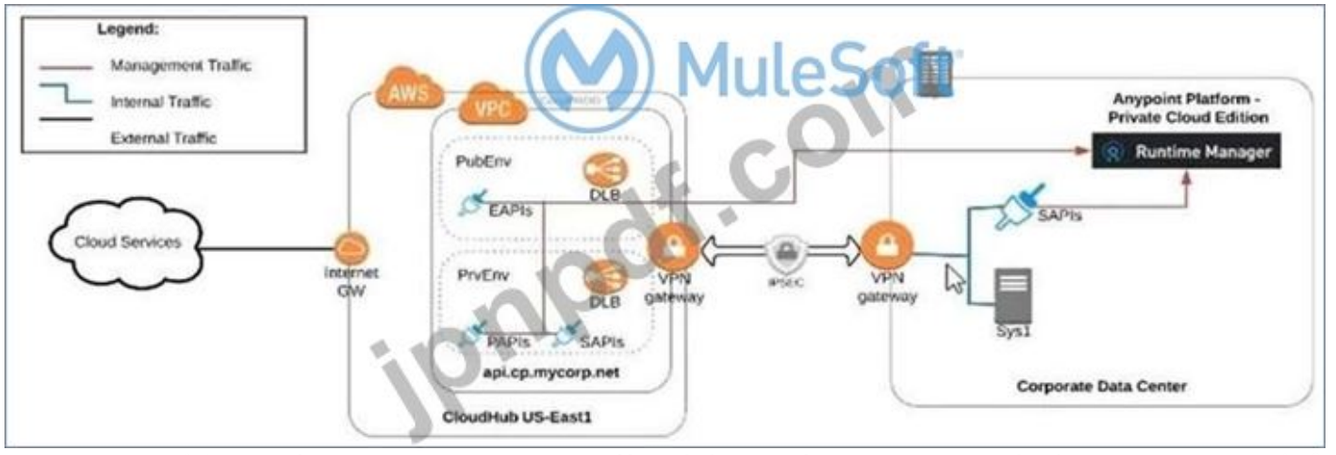
D. レート制限ポリシーとクライアントID強制ポリシーを使用します。それぞれクライアントアプリケーションの種類によって設定されます。
Answer: C (メーッセージを残す**)**

最新問題**: 66**
API 主導の接続性と呼ばれるフレームワーク内で作成された API の機能で通常含まれないものは何ですか?
A. 基盤となるバックエンド システムの上に追加の耐障害性レイヤーを提供し、それによってクライアントをこれらのシステムの長期にわたる障害から保護します。
B. バックエンド システムからデータがどのように抽出されるかを意識せずに、基礎となる資産を消費することで、ユーザー インターフェイス レベルでのイノベーションを可能にします。
C. バックエンド システムからデータを再利用可能かつ消費可能な方法でロック解除することで、基盤となるバックエンド システムへの依存を軽減します。
D. さまざまなソースからデータを作成し、オーケストレーション ロジックと組み合わせて、より高いレベルの価値を生み出すことができます。
Answer: (解答を表示する**)**
これらは、基盤となるバックエンド システムの上に追加の回復力レイヤーを提供し、それによってクライアントをこれらのシステムの長期的な障害から保護します。
****
API主導の接続では、
>> エクスペリエンス API - バックエンド システムからデータがどのように抽出されるかを意識せずに、基盤となるアセットを消費することで、ユーザー インターフェイス レベルでのイノベーションを可能にします。
>> プロセスAPI - さまざまなソースからデータを作成し、オーケストレーションロジックと組み合わせて、より高いレベルの価値を生み出します
>> システム API - バックエンド システムからデータを再利用可能かつ消費可能な方法でロック解除できるようにすることで、基盤となるバックエンド システムへの依存を軽減します。
However, they NEVER promise that they provide an additional layer of resilience on top of the underlying backend system, thereby insulating clients from extended failure of these systems.
https://dzone.com/articles/api-led-connectivity-with-mule

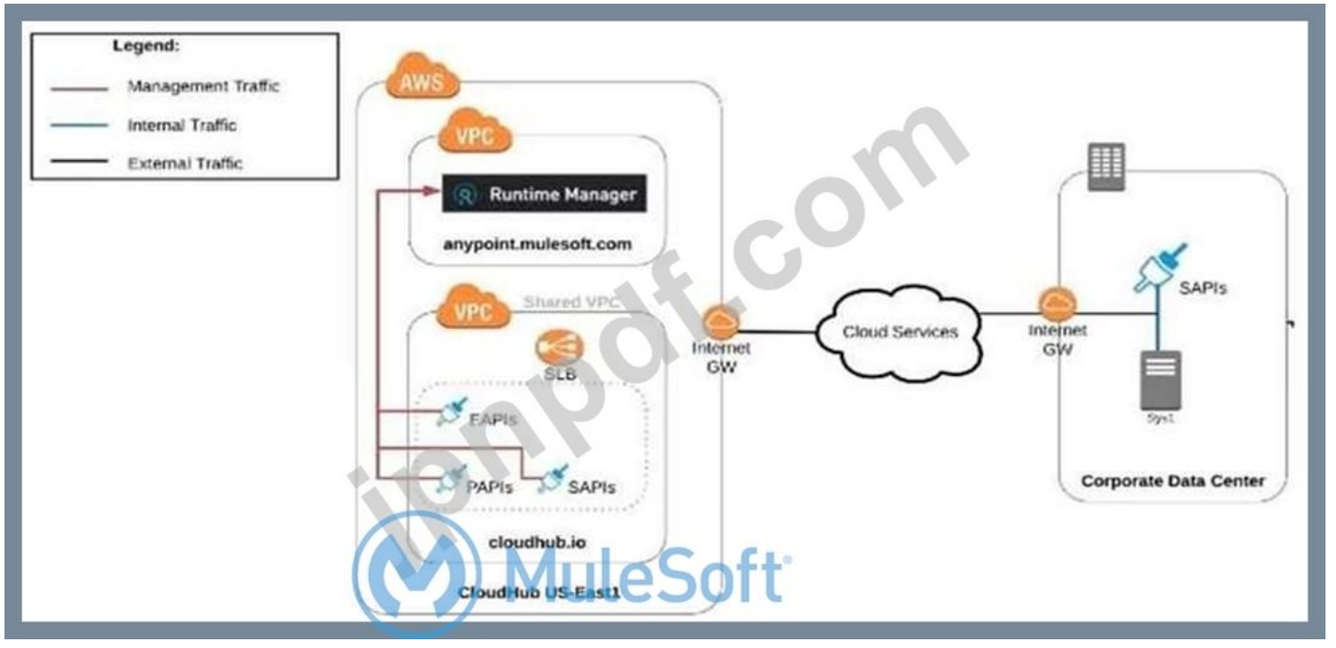
最新問題**: 67**
パブリック Anypoint Exchange ポータルを通じて API を共有する前に確認すべきことは何ですか?
A. 公開アクセスが必要なAPIインスタンスの可視性レベルは、パブリック可視性に設定する必要があります。
B. APIへのアクセスが必要なユーザーは、Anypoint Platformの適切なロールに追加する必要があります。
C. APIは少なくとも初期実装が展開され、ユーザーが操作できるようにアクセス可能で機能する必要があります。
D. データが侵害されないように、サポートされている認証/承認メカニズムのいずれかを使用して API を保護する必要があります。
Answer: A (メーッセージを残す**)**

パブリックにアクセスできるようにする必要がある API の API インスタンスの可視性レベルは、パブリック可視性に設定する必要があります。
****

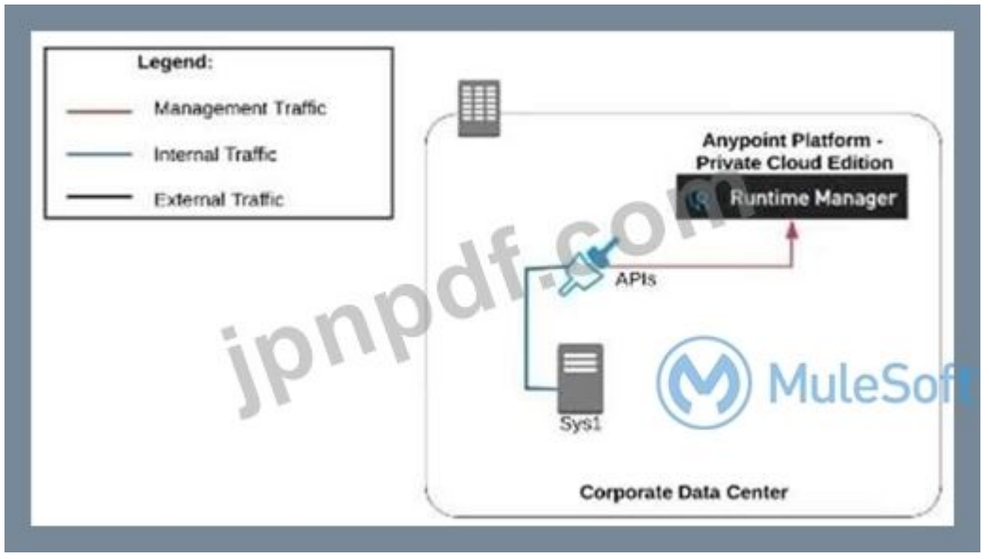
最新問題: **68**
組織では、さまざまなクラウドベースの SaaS システムと複数のオンプレミス システムを使用しています。オンプレミス システムは組織のアプリケーション ネットワークの重要な部分であり、組織のイントラネット内からのみアクセスできます。
クラウドベースの SaaS システムとオンプレミス システムの両方との統合をサポートするために Anypoint Platform を構成して使用する最適な方法は何ですか?
A. Anypoint Platform Private Cloud Edition コントロール pl によって管理される Anypoint VPC で CloudHub でデプロイされた Mule ランタイムを使用する



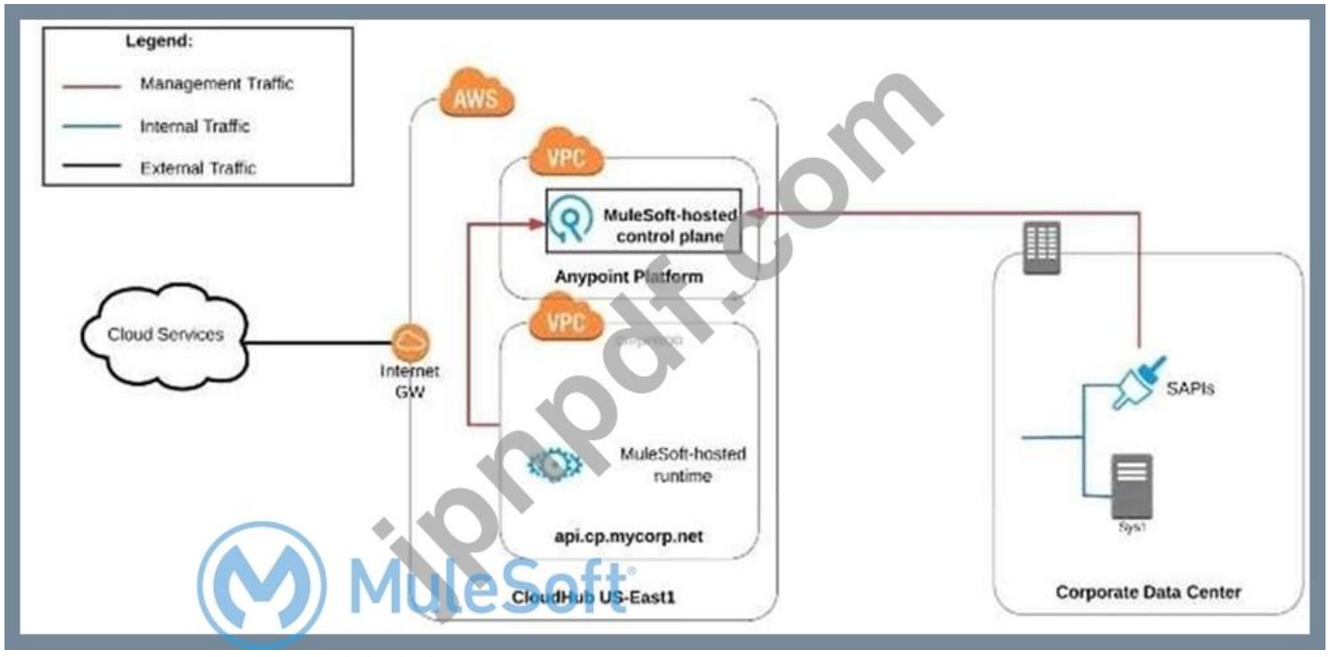
B. MuleSoftがホストするAnypoint Platformで管理される共有ワーカークラウドでCloudHubにデプロイされたMuleランタイムを使用する



C. Anypoint Platform Private Cloud Edition コントロール プレーンによって管理され、外部ネットワーク アクセスなしで完全に分離された Mule ランタイムのオンプレミス インストールを使用します。



D. MuleSoft プラットフォーム コントロール プレーンによって管理される、CloudHub でデプロイされたオンプレミス Mule ランタイムと手動でプロビジョニングされたオンプレミス Mule ランタイムの組み合わせを使用します。

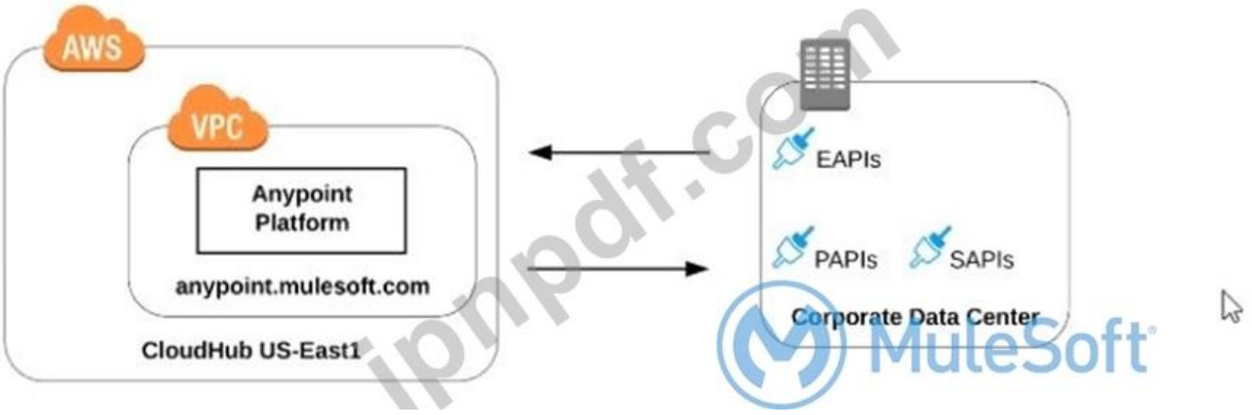


Answer: B (メーッセージを残す**)**

説明/参照:

最新問題**: 69**

Refer to the exhibit.



what is true when using customer-hosted Mule runtimes with the MuleSoft-hosted Anypoint Platform control plane (hybrid deployment)?

A. Anypoint Runtime Manager initiates a network connection to a Mule runtime in order to deploy Mule applications

B. The MuleSoft-hosted Shared Load Balancer can be used to load balance API invocations to the Mule runtimes

C. API implementations can run successfully in customer-hosted Mule runtimes, even when they are unable to communicate with the control plane

D. Anypoint Runtime Manager automatically ensures HA in the control plane by creating a new Mule runtime instance in case of a node failure

Answer: C (メーッセージを残す)

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.

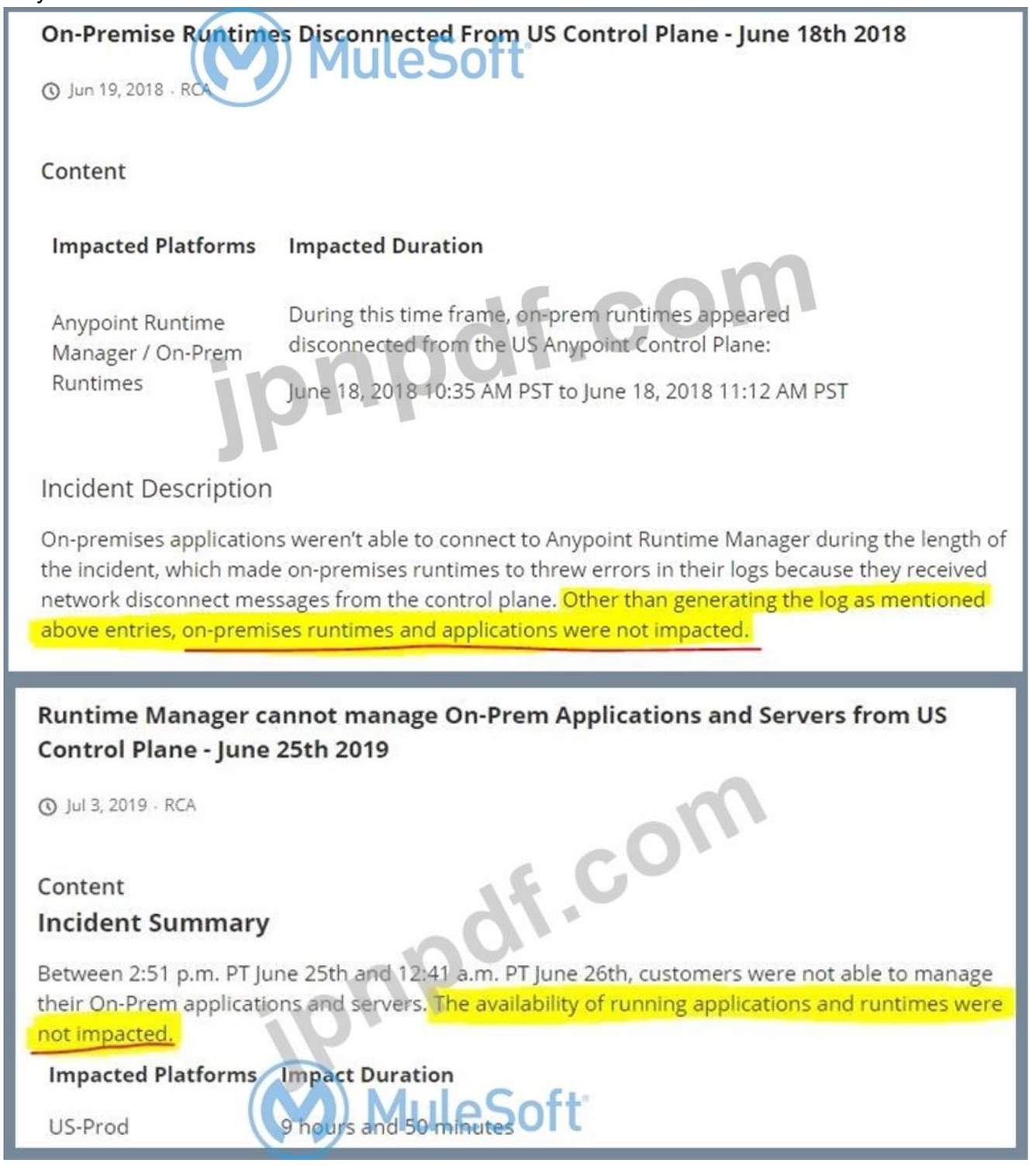
References:

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.

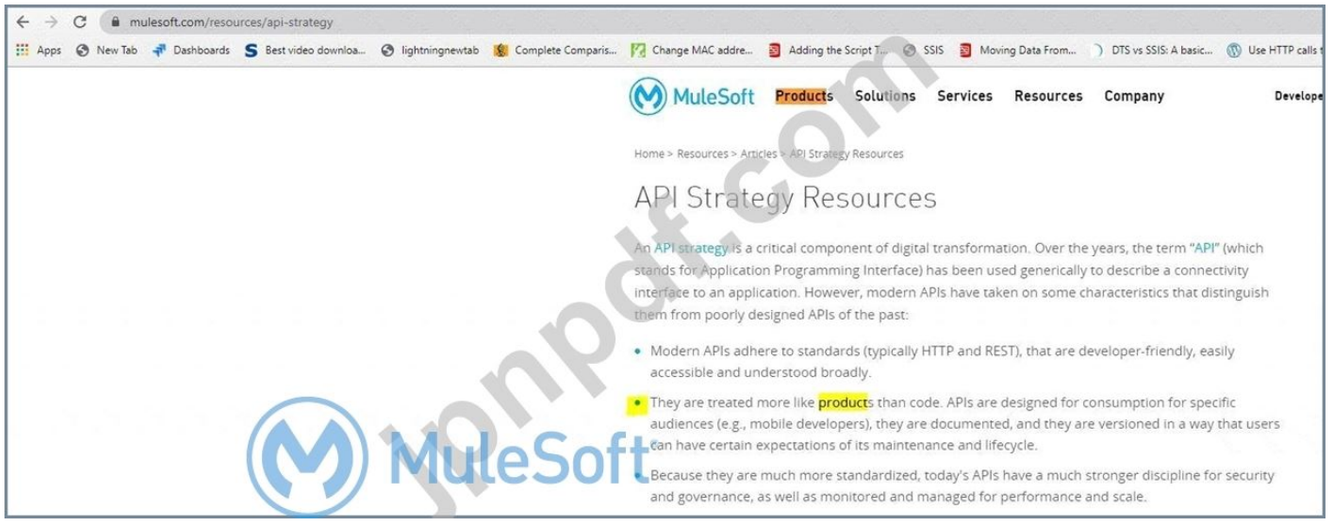
Impacted Platforms	Impact Duration
US-Prod	9 hours and 50 minutes

最新問題: **70**
Anypoint Platform が提供する API 呼び出しメトリクスは何を提供しますか?
A. 特定の脅威しきい値を超える可能性のある将来のポリシー違反を積極的に特定する
B. ビジネス ユーザーと直接共有できる API からの ROI メトリック
C. 再利用レベルに基づくアプリケーションネットワークの有効性の測定
D. 過去の API 呼び出しに関するデータ。さまざまな API の異常や使用パターンの特定に役立ちます。
Answer: D (メーッセージを残す**)**

最新問題: **71**
Anypoint Platform でクライアント管理に外部 ID プロバイダーを使用する場合の主な要件は何ですか?
A. Anypoint Platform にサインインするにはシングルサインオンが必要です
B. アプリケーションネットワークには、アイデンティティプロバイダと対話するシステムAPIが含まれている必要があります。
C. Anypoint Platform によって管理される OAuth 2.0 で保護された API を呼び出すには、API クライアントは同じアイデンティティ プロバイダによって発行されたアクセス トークンを送信する必要があります。
D. Anypoint Platform によって管理される API は、SAML 2.0 ポリシーによって保護される必要があります。
Answer: C (メーッセージを残す**)**
https://www.folkstalk.com/2019/11/mulesoft-integration-and-platform.html 説明:
Anypoint Platformによって管理されるOAuth 2.0で保護されたAPIを呼び出すには、APIクライアントは同じアイデンティティプロバイダによって発行されたアクセストークンを送信する必要があります。

****
>> クライアント管理には外部のIDプロバイダを使用しているため、Anypoint Platformにサインインするためにシングルサインオンは必要ありません。
>> クライアント管理に外部のアイデンティティプロバイダを使用しているため、Anypoint Platform によって管理されるすべての API を SAML 2.0 ポリシーで保護する必要はありません。
>> クライアント管理に外部 ID プロバイダーを使用しているため、アプリケーション ネットワークに ID プロバイダーと対話するシステム API を含める必要があるというのは正しくありません。指定されたオプションで正しいのは、 Anypoint Platform によって管理される OAuth 2.0 で保護された API を呼び出すには、API クライアントは同じ ID プロバイダーによって発行されたアクセス トークンを送信する必要があります」のみです。参照:
https://docs.mulesoft.com/api-manager/2.x/external-oauth-2.0-token-validation-policy
https://blogs.mulesoft.com/dev/api-dev/api-security-ways-to-authenticate-and-authorize/

最新問題**: 72**
組織は、最新の API (MuleSoft の定義による) を使用して再利用可能な IT 資産の消費を重視する IT 運用モデルに移行するという戦略的決定を下します。
この新しい IT 運用モデルに関連して、各最新 API を最もよく表すものは何ですか?
A. 最新のAPIにはそれぞれ独自のソフトウェア開発ライフサイクルがあり、ドキュメント作成や自動化の必要性が軽減されます。
B. 各モデム API は製品のように扱われ、特定の対象ユーザー (たとえば、モバイル アプリ開発者) 向けに設計される必要があります。
C. 各モダンAPIは簡単に使用できる必要があるため、SAMLやJWTなどの複雑な認証メカニズムは避けるべきです。
D. 最新のAPIはRESTとHTTPベースでなければならない
Answer: (解答を表示する**)**
正解:
1. 各最新APIは製品のように扱われ、特定の対象ユーザー（モバイルアプリ開発者など）向けに設計される必要がある
****



フォームの下部
フォームの先頭

最新問題: **73**
スケジュールされた間隔で API とエンドポイントを監視し、テストの合格または不合格に関するレポートを受信し、API とエンドポイントのパフォーマンスに関する統計を表示するコンポーネントはどれですか?
A. API 分析
B. Anypoint Monitoring ダッシュボード
C. APT 機能監視
D. Anypoint Runtime Manager アラート
Answer: C (メーッセージを残す**)**
* API機能監視の理解:
* API 機能モニタリングは、MuleSoft の Anypoint プラットフォーム内の機能であり、ユーザーはスケジュールされた間隔で機能テストを実行して、API とエンドポイントの健全性とパフォーマンスを監視できます。
* テスト呼び出しを実行し、応答が目的の条件を満たしているかどうかを評価することで、API が期待どおりに機能しているかどうかを確認します。これは、エンドポイントの可用性をテストしたり、応答内の特定のデータを確認したり、時間の経過に伴う API パフォーマンスを測定したりする場合などに特に役立ちます。
* コンポーネントの機能:
* スケジュールされた間隔: 機能監視では、監視要件に応じて、毎分、毎時間、または毎日などの定期的な間隔でテストを実行するように構成できます。
* テストの合格/不合格ステータスのレポート: 各テストの実行後、API 機能モニタリングは、API がテスト条件に合格したか不合格になったかを報告します。
* パフォーマンス統計: 平均応答時間、成功率、エラー率などの指標が表示され、API の健全性とパフォーマンスに関する洞察が得られます。
* オプションの評価:
* オプション A (API 分析): API 分析では、API の使用状況とメトリックに関する分析情報が提供されますが、合格/不合格ステータスやエンドポイントのヘルス チェックのスケジュールされたテストは含まれません。

* オプション B (Anypoint モニタリング ダッシュボード): これらのダッシュボードには API メトリクスが表示されますが、API エンドポイントを積極的にテストしたり、スケジュールに基づいて合格/不合格のレポートを提供したりすることはありません。
* オプション C (正解): API 機能モニタリングは、スケジュールされたテスト実行で API とエンドポイントの健全性を監視し、パフォーマンスに関する統計を表示するように設計されているため、説明に適合します。
* オプション D (Anypoint Runtime Manager アラート): Runtime Manager アラートは、アプリケーション ステータスの問題をユーザーに通知しますが、スケジュールされた間隔でエンドポイントを積極的にテストすることはありません。
* 結論：
* オプション C (API 機能モニタリング) は、API 機能をテストし、エンドポイントの状態を監視し、パフォーマンス統計をリアルタイムで表示するために必要なツールを提供するため、正解です。
Anypoint Platform でこれらのテストをセットアップおよび構成する方法の詳細については、API 機能モニタリングに関する MuleSoft のドキュメントを参照してください。

最新問題: **74**
コード中心の API ドキュメント環境では、API コンシューマーが、代表的なシナリオの一部として 1 つ以上の API の呼び出しを示す API クライアント ソース コードを調査および実行できるようにする必要があります。
Anypoint Platform を使用して、このようなコード中心の API ドキュメント環境を提供する最も効果的な方法は何ですか?
A. 関連する各APIのモックサービスを有効にし、Anypoint Exchangeエントリを介して公開します。
B. APIがAnypoint ExchangeエントリとAPIコンソールを通じて適切に文書化されていることを確認し、これらのページをすべてのAPIコンシューマーと共有します。
C. APIノートブックを作成し、関連するAnypoint Exchangeエントリに含めます。
D. Anypoint Exchangeエントリを介して関連するAPIを検出可能にする
Answer: C (メーッセージを残す**)**
APIノートブックを作成し、関連するAnypoint Exchangeエントリに含める
****
>> APIノートブックは、コード中心のAPIドキュメントを提供できるAnypoint Platformの1つです。

最新問題: **75**
What CANNOT be effectively enforced using an API policy in Anypoint Platform?
A. Logging HTTP requests and responses
B. Guarding against Denial of Service attacks
C. Maintaining tamper-proof credentials between APIs
D. Backend system overloading
Answer: A (メーッセージを残す**)**

最新問題: **76**

ある組織では、Center for Enablement (C4E) を設立したいと考えています。IT ディレクターは、IT シニア マネージャーとの一連の会議をスケジュールします。
最初の会議の議題には何を含めるべきでしょうか?
A. C4Eの目的、ミッションステートメント、指針、
B. MuleSoft を通じて特定されたユースケースに基づいて API 収益化オプションを探索する
C. ログ記録、監査、例外処理、キャッシュ、ポリシーによるセキュリティ、ポリシーによるレート制限/スロットリングに関する共通サービスのベストプラクティスのウォークスルー
D. MuleSoft 統合部門の運用モデルを指定します
Answer: A (メーッセージを残す**)**
イネーブルメント センター (C4E) を設立するための最初の会議では、チームの基本的なビジョン、目標、および指針を定めることが不可欠です。これが重要である理由は次のとおりです。
* 明確なビジョンとミッション:
* 最初にミッション ステートメントと目標を定義することで、組織内の整合性が確保され、API 主導の開発と統合の実践をサポートする C4E の役割が明確になります。
* 指導原則:
* ガイドラインを確立することで、C4E はプロジェクト間で一貫した実践と戦略を維持できるようになります。これは意思決定のフレームワークとして機能し、IT リーダーと関係者の間で共通の理解を促進します。
* 正解 Ⓐの説明 :
* C4E の目的と使命を優先することで、組織は強固な基盤を構築し、技術標準、プロセス、運用モデルに焦点を当てた後続の会議への道を開きます。
* 誤ったオプションの説明:
* オプション B (API 収益化) とオプション C (共通サービスのベスト プラクティス) は、後で議論するのに適した特定のトピックです。
* オプション D (運用モデルの指定) は重要なステップですが、通常は C4E の目的とビジョンの確立後に行われます。
参考資料 C4E の目的と基本的な設定の詳細については、C4E の確立と、そのような取り組みに推奨される役割とミッション ステートメントに関する MuleSoft のドキュメントを参照してください。

最新問題**: 77**
組織は、最新の API (MuleSoft の定義による) を使用して再利用可能な IT 資産の消費を重視する IT 運用モデルに移行するという戦略的決定を下します。
この新しい IT 運用モデルに関連して、各最新 API を最もよく表すものは何ですか?

A. 各モデム API は製品のように扱われ、特定の対象ユーザー (たとえば、モバイル アプリ開発者) 向けに設計される必要があります。
B. 最新のAPIにはそれぞれ独自のソフトウェア開発ライフサイクルがあり、ドキュメント作成や自動化の必要性が軽減されます。
C. 各モダンAPIは簡単に使用できる必要があるため、SAMLやJWTなどの複雑な認証メカニズムは避けるべきです。
D. 最新のAPIはRESTとHTTPベースでなければならない
Answer: D (メーッセージを残す**)**

最新問題**: 78**
Anypoint Platform REST API、Anypoint CU、Mule Maven プラグインなどのツールを使用して Anypoint Platform とのやり取りを自動化することについて正しいのは何ですか?
A. Anypoint Platform API と Anypoint CU へのアクセスは、Anypoint Platform のロールと権限を通じて個別に制御できるため、特定のユーザーは Anypoint CLI にアクセスでき、他のユーザーはプラットフォーム API にアクセスできます。
B. APIポリシーをAnypoint Platform APIに適用して、特定のLOBのみが特定の機能にアクセスできるようにすることができます。
C. Anypoint Platform API は CloudHub とのやり取りのみを自動化できますが、顧客がホストする Mule ランタイムへの展開には Mule Maven プラグインが必要です。
D. デフォルトでは、Anypoint CLI と Mule Maven プラグインは Mule ランタイムに含まれていないため、デプロイされた Mule アプリケーションでは使用できません。
Answer: (解答を表示する**)**

最新問題**: 79**
Anypoint Platform 組織は、ID 管理とクライアント管理のために外部 ID プロバイダー (IdP) を使用して設定されています。Anypoint Platform API に対してコマンドを実行するには、Anypoint CLI にどのような資格情報またはトークンを提供する必要がありますか?
A. ID管理のためにIdPが提供する資格情報
B. クライアント管理のためにIdPによって提供される資格情報
C. クライアント管理用に IdP から提供された資格情報を使用して生成された OAuth 2.0 トークン
D. ID管理のためにIdPによって提供された資格情報を使用して生成されたOAuth 2.0トークン
Answer: A (メーッセージを残す**)**
正解: ID管理のためにIdPが提供する資格情報
****
参照：
>> Anypoint CLI 経由で認証するためのクライアント/ID プロバイダーからの OAuth 2.0 トークンはサポートされていません。使用可能なトークンは、https://anypoint.mulesoft.com/accounts/login の Anypoint 組織/環境クライアント ID とシークレットを使用してのみ生成される ベアラー トークン」のみです。クライアント プロバイダーのクライアント資格情報ではありません。したがって、OAuth 2.0 は使用できません。さらに、

トークンは主に API マネージャーの目的のためであり、ユーザーに関連付けられていません。この Mulesoft Knowledge 記事に従って、ほとんどの API (Cloudhub など) を呼び出すために使用することはできません。
>> Anypoint CLI で許可されているもう 1 つのオプションは、クライアント資格情報を使用することです。クライアント プロバイダーのクライアント資格情報を使用することは可能ですが、クライアント管理で接続されたアプリケーションを設定する必要がありますが、質問で説明されているシナリオではそのような詳細は提供されていません。
>> 残された唯一の選択肢は、IDプロバイダーからのユーザー認証情報を使用することです

最新問題**: 80**
REST API 実装のバージョン 3.0.1 では、ISO 8601 hh:mm:ss 形式を使用して PST 時間で時間値を表します。API 実装を変更して、代わりに ISO 8601 hh:mm:ss 形式を使用して CEST 時間で時間値を表す必要があります。semver.org のセマンティック バージョン管理仕様に従う場合、更新された API 実装にどのバージョンを割り当てる必要がありますか?
A. 3.0.2
B. 4.0.0
C. 3.1.0
D. 3.0.1
Answer: B (メーッセージを残す**)**
正解: 4.0.0
****
semver.org のセマンティック バージョン管理仕様によると:
バージョン番号 MAJOR.MINOR.PATCH が指定されている場合、次の値を増分します。
- 互換性のない API 変更を行った場合のメジャー バージョン。
- 下位互換性のある方法で機能を追加する場合のマイナー バージョン。
- 下位互換性のあるバグ修正を行う場合の PATCH バージョン。
質問で示されたシナリオによると、API 実装の動作は完全に変更されています。時間の形式は引き続き hh:mm:ss として維持され、形式に関するスキーマは変更されていませんが、時代が完全に変わるため、この変更後、API は異なる機能を開始します。
例: 変更前は、時間は PST を表す 09:00:00 と表示されていました。変更後は、中央ヨーロッパ夏時間が太平洋標準時間より 9 時間進んでいるため、同じ時間は 18:00:00 になります。
>> これにより、API 応答で時間を処理する方法に応じて、API クライアントで不確実な動作が発生する可能性があります。すべての API クライアントに、API 機能が変更され、CEST 形式で返されることを通知する必要があります。したがって、これはメジャー変更と見なされ、この新しい変更の API バージョンは 4.0.0 になります。

最新問題**: 81**
システム API には、リクエストごとに 100 ミリ秒の SLA が保証されています。システム API は、プライマリ環境と災害復旧 (DR) 環境に展開され、各環境では DNS 名が異なります。アップストリーム プロセス API がシステム API を呼び出します。このプロセス API の主な目的は、クライアントのリクエストに最短時間で応

答することです。システム API はどのような順序で呼び出す必要がありますか。また、プロセス API からのリクエストの応答時間を短縮するには、どのような変更を加える必要がありますか。
A. プライマリ環境にデプロイされたシステムAPIとDR環境にデプロイされたシステムAPIを並行して呼び出し、最初の応答のみを使用します。
B. タイムアウトが設定されたスキャッターギャザーを使用して、プライマリ環境にデプロイされたシステムAPIとDR環境にデプロイされたシステムAPIを並行して呼び出し、応答をマージします。
C. プライマリ環境にデプロイされたシステムAPIを呼び出し、失敗した場合はDR環境にデプロイされたシステムAPIを呼び出す
D. プライマリ環境にデプロイされたシステムAPIのみを呼び出し、断続的な障害を回避するためにタイムアウトと再試行ロジックを追加します。
Answer: A (メーッセージを残す**)**
正解: プライマリ環境にデプロイされたシステム API と DR 環境にデプロイされたシステム API を並行して呼び出し、最初の応答のみを使用します。
****

>> 与えられたシナリオにおける API 要件は、可能な限り最短時間で応答することです。
>> 最初にプライマリ環境で API を試し、次に DR 環境で API にフォールバックすることを提案するオプションでは、応答は成功しますが、最短時間は得られません。したがって、これは特定の要件に対する実装の正しい選択ではありません。
>> プライマリ環境でのみ API を呼び出し、タイムアウトと再試行を追加することを提案する別のオプションも、再試行時に応答が成功する可能性がありますが、最短時間ではありません。したがって、これも特定の要件に対する実装の正しい選択ではありません。
>> スキャッター ギャザーを使用してプライマリ環境の API と DR 環境の API を並行して呼び出すことを提案するもう 1 つのオプションは、マージされた結果を返すため間違った API 応答になり、さらに、スキャッター ギャザーは並列で処理を行いますが、そのスコープは内部のすべてのルートを終了したときにのみ完了します。したがって、これもまた、特定の要件に対する実装の正しい選択ではありません。正しい選択は、プライマリ環境の API と DR 環境の API を並行して呼び出し、それらのいずれかから受信した最初の応答のみを使用することです。

最新問題**: 82**
CloudHub 専用ロードバランサーを使用する必要がある条件は何ですか?
A. 同じ Mule アプリケーションの別々のデプロイメント間でクロスリージョン負荷分散が必要な場合
B. 顧客がホストする Mule ランタイムにデプロイされた API 実装にカスタム DNS 名が必要な場合
C. 複数の CloudHub ワーカー間での API 呼び出しを負荷分散する必要がある場合
D. API実装とAPIクライアント間でサーバー側負荷分散TLS相互認証が必要な場合
Answer: (解答を表示する**)**
API実装とAPIクライアント間でサーバー側負荷分散TLS相互認証が必要な場合
****

事実/メモリのヒント: CloudHub 専用ロードバランサには多くの利点がありますが、検討する際に心に留めておくべき重要な点が 2 つあります。
>> CloudHub にデプロイされたアプリにカスタム DNS 名を持つ URL エンドポイントを設定する

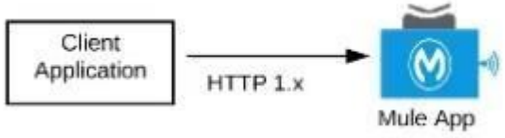
>> HTTPS と双方向 (相互) 認証の両方に対してカスタム証明書を構成します。
この質問に対して提供されているオプションは次のとおりです。
>> DLB を使用して、同じ Mule アプリケーションの個別のデプロイメント間でリージョン間の負荷分散を実行することはできません。
>> 複数の DLB URL が同じ Mule アプリを指すようにマッピング ルールを設定できます。ただし、その逆 (複数の Mule アプリが同じ DLB URL を持つ) は不可能です。
>> DLB は Cloudhub にデプロイされた Mule アプリのカスタム DNS 名の設定に役立ちますが、顧客がホストする Mule ランタイムにデプロイされたアプリには当てはまりません。
>> DLB を使用して複数の CloudHub ワーカー間で API 呼び出しの負荷を分散できることは事実ですが、必須ではありません。SLB (共有ロード バランサ) を使用しても同じこと (負荷分散) を実現できます。これを実現するために必ずしも DLB が必要というわけではありません。
したがって、シナリオに適合し、DLB を使用する必要がある唯一の適切なオプションは、API 実装と API クライアント間で TLS 相互認証が必要な場合です。

最新問題**: 83**
たとえば、CRM-Z と呼ばれるレガシー CRM システムがあり、以下の機能を提供しているとします。
1. 顧客の創造
2. 既存顧客の詳細を修正する
3. 顧客の詳細を取得する
4. 顧客を停止する
A. さまざまな操作/リソースとしてすべての機能がラップされた、customerManagement という名前のシステム API を実装します。
B. createCustomer、amendCustomer、retrieveCustomer、suspendCustomer という異なるシステム API を実装します。これらはモジュール化されており、関心の分離が可能です。
C.
createCustomerInCRMZ、amendCustomerInCRMZ、retrieveCustomerFromCRMZ、suspendCustomerInCRMZ という異なるシステム API を実装します。これらはモジュール化されており、関心の分離が可能です。
Answer: B (メーッセージを残す**)**
モジュール化されており、関心の分離があるため、createCustomer、amendCustomer、retrieveCustomer、suspendCustomer という異なるシステム API を実装します。
****
>> 単一の API と異なる動詞 + リソースの組み合わせを持つことはごく普通のことです。ただし、これはエクスペリエンス API またはプロセス API には適していますが、システム API には最適なアーキテクチャ スタイルではありません。したがって、customerManagement API を 1 つだけ使用するオプションは、ここでは最適な選択ではありません。
>> createCustomerInCRMZ 形式の API を使用するオプションは、モジュール化とメンテナンスの軽減という点で次に近い選択肢ですが、API の命名はレガシー システムと直接結びついています。より適切なアプ

ローチは、バックエンド システム名を抽象化して API に名前を付けることです。これにより、シームレスな置き換えが可能になります。
/いつでもバックエンド システムを移行できます。したがって、これも正しい選択ではありません。
>> createCustomer、amendCustomer、retrieveCustomer、suspendCustomer は正しいアプローチであり、他のオプションと比較して最適です。これらは両方ともモジュール化されており、同時に名前がバックエンド システムから分離されており、システム API に必要なすべての要件をカバーしています。

最新問題: **84**
Anypoint Platform API からの応答ですぐにわかる、一般的な C4E の成功を測定する主要業績評価指標 (KPI) は何ですか?
A. 過去 24 時間に報告された生産停止インシデントの数
B. CI/CD ツールを使用してデプロイされた API 実装と比較して、手動でデプロイされた API 実装の割合
C. Anypoint Exchangeに公開されているRAMLまたはOAS形式のAPI仕様の数
D. パブリックにアクセス可能なHTTPエンドポイントを持ち、Anypoint Platformによって管理されているAPI実装の数
Answer: D (メーッセージを残す**)**

最新問題: **85**
どのMuleアプリケーションにAPIポリシーを適用できるか
Anypoint Platform からその Mule アプリケーションによって公開されるエンドポイントに接続しますか?
A) HTTP/1.x 経由でリクエストを受け入れる Mule アプリケーション



B) TCP 経由で JSON リクエストを受け入れるが、応答を提供する必要がない Mule アプリケーション C) WebSocket 経由で JSON リクエストを受け入れる Mute アプリケーション D) HTTP/2 経由で gRPC リクエストを受け入れる Mule アプリケーション
A. オプションA
B. オプションB
C. オプションC
D. オプションD
Answer: A (メーッセージを残す**)**
オプションA
****
>> Anypoint API Manager と API ポリシーは、すべてのタイプの HTTP/1.x API に適用できます。
>> WebSocket API、HTTP/2 API、gRPC APIには適用されません

最新問題: **86**
たとえば、CRM-Z と呼ばれるレガシー CRM システムがあり、以下の機能を提供しているとします。
1. 顧客の創造

2. 既存顧客の詳細を修正する
3. 顧客の詳細を取得する
4. 顧客を停止する
A. さまざまな操作/リソースとしてすべての機能がラップされた、customerManagement という名前のシステム API を実装します。
B. createCustomer、amendCustomer、retrieveCustomer、suspendCustomer という異なるシステム API を実装します。これらはモジュール化されており、関心の分離が可能です。
C.
createCustomerInCRMZ、amendCustomerInCRMZ、retrieveCustomerFromCRMZ、suspendCustomerInCRMZ という異なるシステム API を実装します。これらはモジュール化されており、関心の分離が可能です。
Answer: (解答を表示する**)**
createCustomer、amendCustomer、retrieveCustomer という異なるシステム API を実装します。
モジュール化されており、関心の分離があるため、suspendCustomerを使用します。
****
>> 単一の API と異なる動詞 + リソースの組み合わせを持つことはごく普通のことです。ただし、これはエクスペリエンス API またはプロセス API には適していますが、システム API には最適なアーキテクチャ スタイルではありません。したがって、customerManagement API を 1 つだけ使用するオプションは、ここでは最適な選択ではありません。
>> createCustomerInCRMZ 形式の API を使用するオプションは、モジュール化とメンテナンスの軽減という点で次に近い選択肢ですが、API の命名はレガシー システムと直接結びついています。より適切なアプローチは、バックエンド システム名を抽象化して API に名前を付けることです。これにより、いつでも任意のバックエンド システムをシームレスに置き換えたり移行したりできます。したがって、これも正しい選択ではありません。
>> createCustomer、amendCustomer、retrieveCustomer、suspendCustomer は正しいアプローチであり、他のオプションと比較して最適です。これらは両方ともモジュール化されており、同時に名前がバックエンド システムから分離されており、システム API に必要なすべての要件をカバーしています。

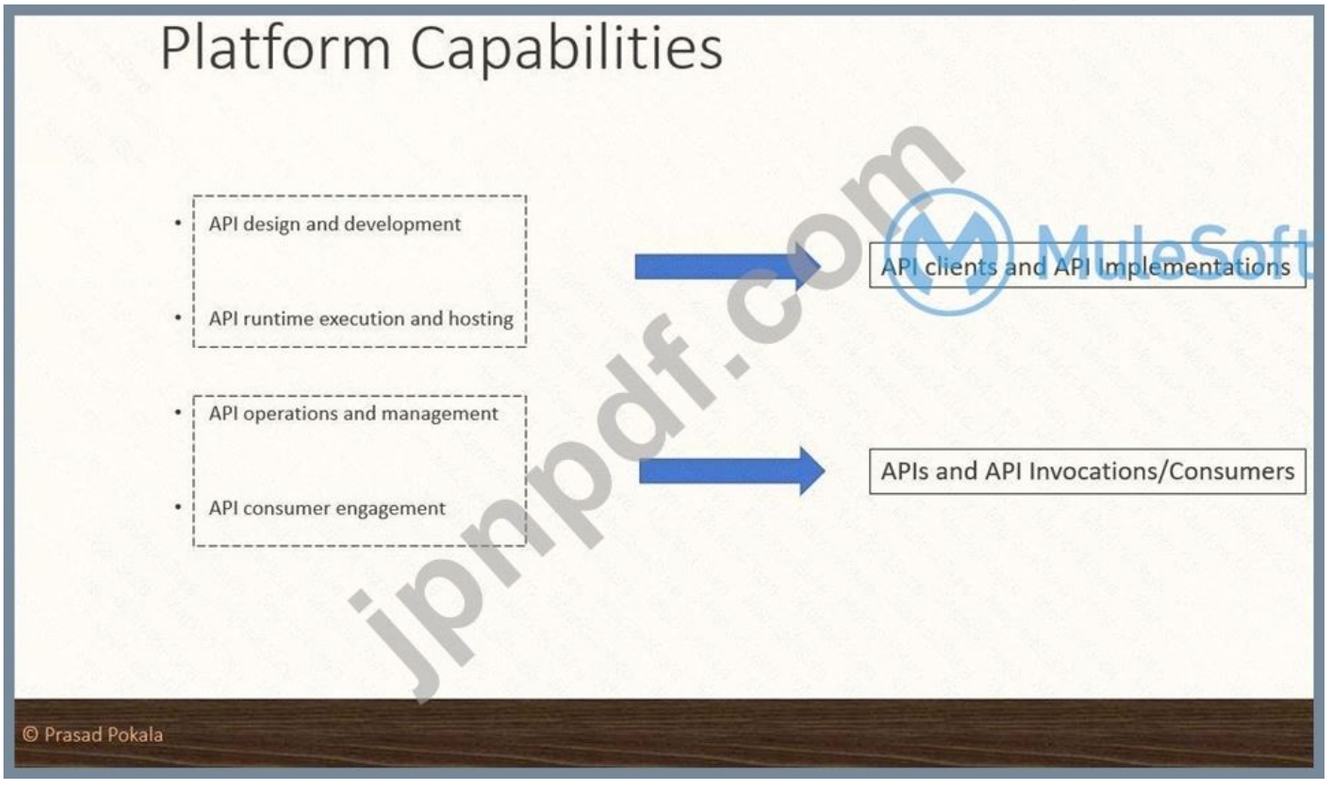
最新問題**: 87**
What is a typical result of using a fine-grained rather than a coarse-grained API deployment model to implement a given business process?
A. A decrease in the number of connections within the application network supporting the business process
B. A higher number of discoverable API-related assets in the application network
C. A better response time for the end user as a result of the APIs being smaller in scope and complexity
D. An overall tower usage of resources because each fine-grained API consumes less resources
Answer: B (メーッセージを残す**)**
正解: アプリケーション ネットワーク内で検出可能な API 関連アセットの数が増える。
****
>> 粗粒度のアプローチと比較した場合、細粒度のアプローチでは応答時間が速くなりません。
>> 実際、粒度の粗い API モデルを持つネットワークでは、粒度の細かい API モデルを持つネットワークよりも応答時間が短くなります。その理由は次のとおりです。

きめ細かなアプローチ:
1. 粗粒度に比べてAPIの数が多くなる
2. したがって、ビジネス プロセスの機能性を実現するには、さらにオーケストレーションを行う必要があります。
3. つまり、大量の API 呼び出しが必要になります。そのため、より多くの接続を確立する必要があります。したがって、大量の機能が組み込まれた API が少ない粗粒度のアプローチと比較すると、ホップ、ネットワーク I/O、統合ポイントの数が増えるのは明らかです。
4. そのため、これらすべての追加ホップと追加のレイテンシにより、細粒度のアプローチでは、粗粒度のアプローチに比べて応答時間が少し長くなります。
5. レイテンシと接続が追加されるだけでなく、API の数が増えるため、きめ細かいアプローチで使用されるリソースも増えます。
そのため、きめ細かな API は、ネットワーク内で再利用可能な資産をより多く公開し、検出できるようにするのに適しています。ただし、ネットワーク ホップと応答時間に関して多少の妥協をしながら、統合ポイント、接続、リソースを管理するために、より多くのメンテナンスが必要になります。

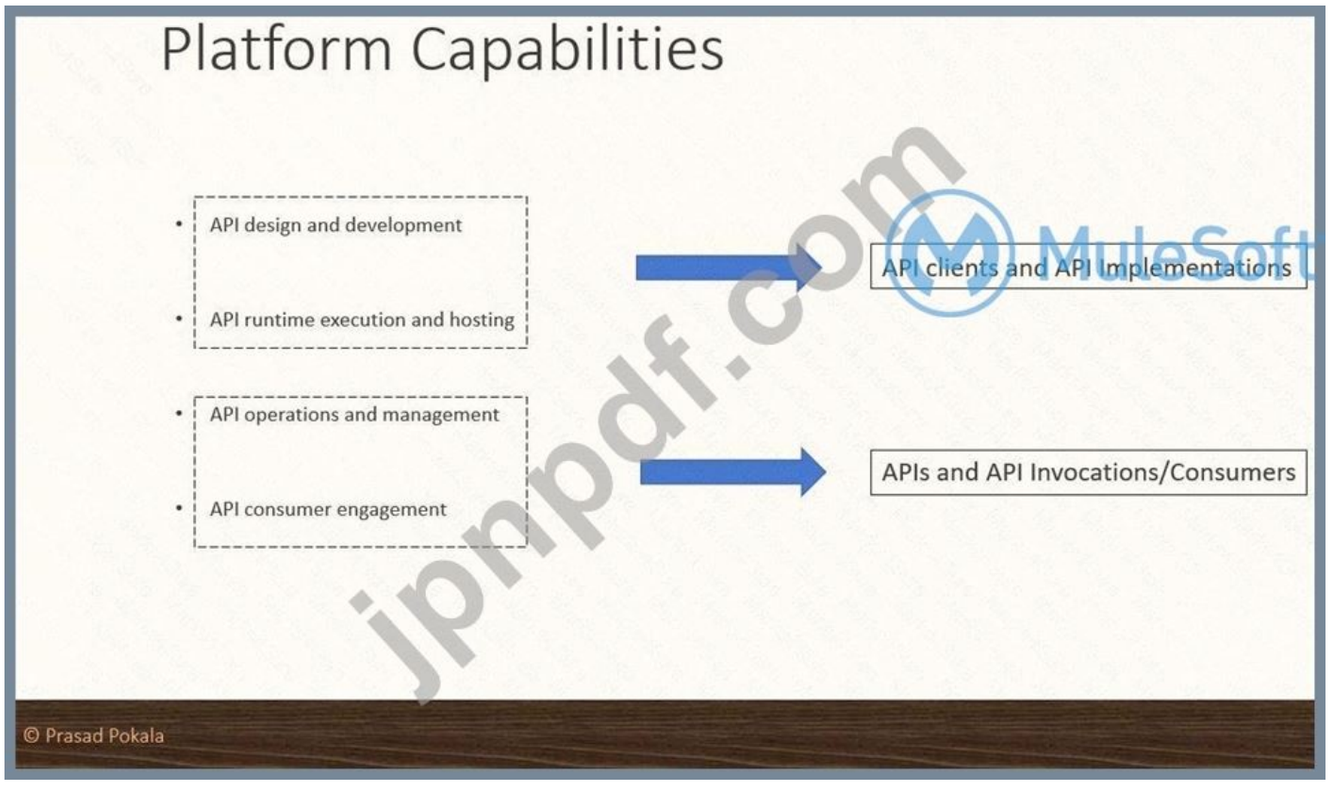
最新問題**: 88**
API 実装の準備が整い、API が API Manager に登録されたら、Anypoint Exchange 上の API へのアクセスを誰がリクエストすればよいでしょうか?
A. なし
B. 両方
C. API クライアント
D. API コンシューマー
Answer: D (メーッセージを残す**)**
正解: API コンシューマー
****
>> APIクライアントは、APIコンシューマのクライアント資格情報を使用するコードまたはプログラムですが、アクセスを取得するためにAnypoint Exchangeと直接やり取りすることはありません。
>> API コンシューマーは登録して API へのアクセスをリクエストする必要があり、API クライアントはそれらのクライアント資格情報を使用して API にアクセスする必要があります。つまり、API コンシューマーは Anypoint Exchange から API へのアクセスをリクエストする必要がある人です。

最新問題**: 89**
以下にリストされている Anypoint Platform 機能のうち、API および API 呼び出し/コンシューマーのカテゴリに該当するものはどれですか?
2つ選択してください。
A. API 操作と管理
B. API ランタイムの実行とホスティング
C. API 消費者エンゲージメント
D. API 設計と開発
Answer: (解答を表示する**)**

正解: API の設計と開発、API ランタイムの実行とホスティング
****
>> API 設計と開発 - Anypoint Studio、Anypoint Design Center、Anypoint Connectors
>> API ランタイム実行とホスティング - Mule ランタイム、CloudHub、ランタイム サービス
>> API 運用と管理 - Anypoint API Manager、Anypoint Exchange
>> API コンシューマー管理 - API 契約、パブリック ポータル、Anypoint Exchange、API ノートブック



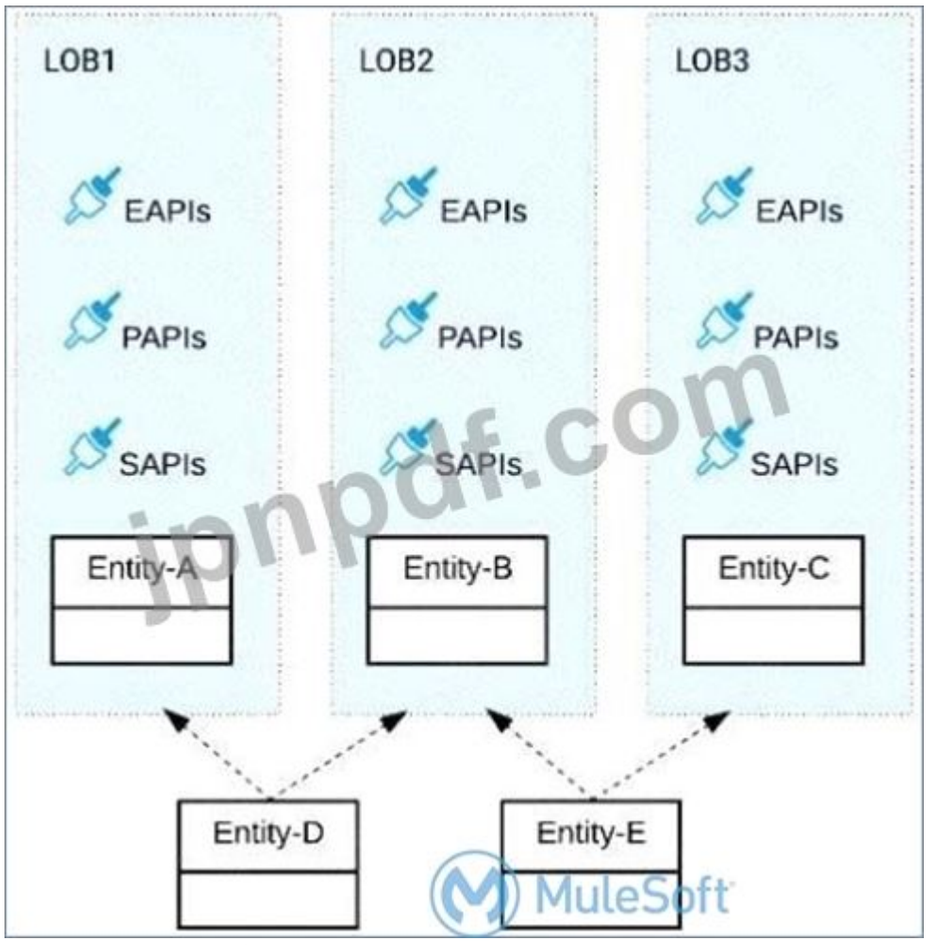
Explanation:
正解: API 運用と管理、API コンシューマーエンゲージメント
****
>> API 設計と開発 - Anypoint Studio、Anypoint Design Center、Anypoint Connectors
>> API ランタイム実行とホスティング - Mule ランタイム、CloudHub、ランタイム サービス
>> API 運用と管理 - Anypoint API Manager、Anypoint Exchange
>> API コンシューマー管理 - API 契約、パブリック ポータル、Anypoint Exchange、API ノートブック



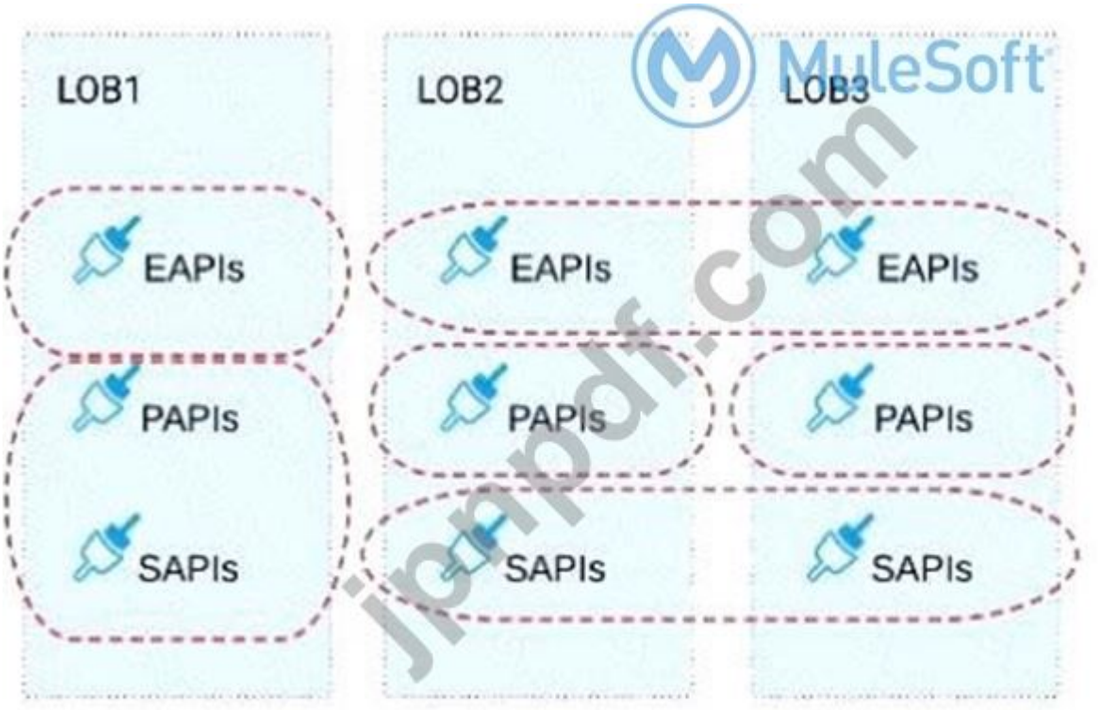
フォームの下部
フォームの先頭

最新問題: **90**
一部の HTTP リクエストに対する応答は、リクエストで使用される HTTP 動詞に応じてキャッシュできます。
HTTP 仕様によれば、どの HTTP 動詞に対してこれを実行しても安全ですか?
A. PUT、POST、DELETE
B. GET、HEAD、POST
C. GET、PUT、オプション
D. GET、オプション、HEAD
Answer: (解答を表示する**)**
GET、オプション、ヘッド
http://restcookbook.com/HTTP%20Methods/べき等性/

最新問題: **91**
展示品を参照してください。

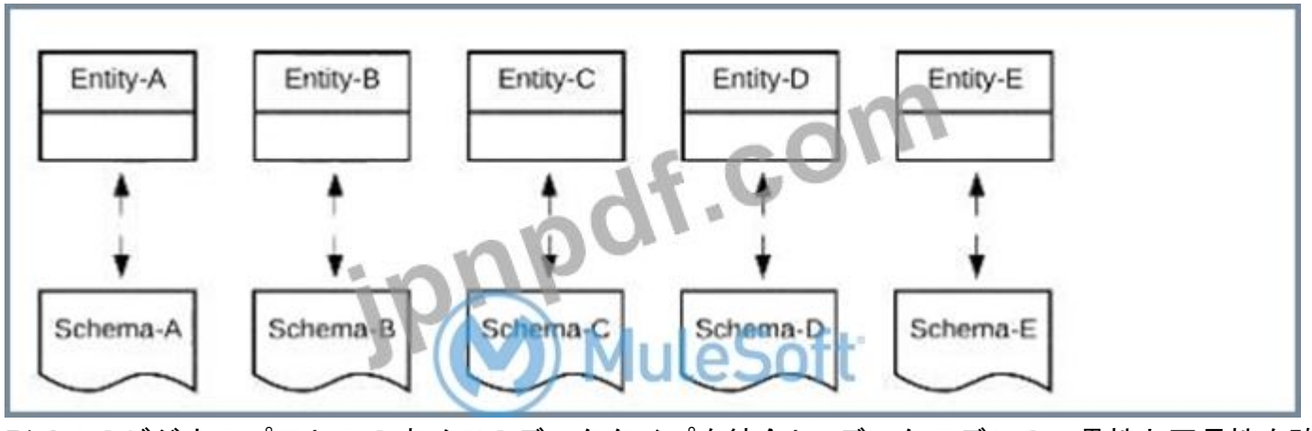


3 つのビジネス プロセスを実装する必要があり、実装では複数の異なる SaaS アプリケーションと通信する必要があります。
これらのプロセスは、個別の (サイロ化された) LOB によって所有され、主に互いに独立していますが、いくつかのビジネス エンティティを共有しています。各 LOB には 1 つの開発チームと独自の予算があります。
この組織のコンテキストでは、データ モデルの冗長性を最小限に抑えてこれらのビジネス プロセスを実装する API の API データ モデルを選択する最も効果的な方法は何ですか。
A) ビジネスプロセスの一貫した部分と関連するビジネスエンティティの定義に一致する複数の境界付きコンテキストデータモデルを構築する

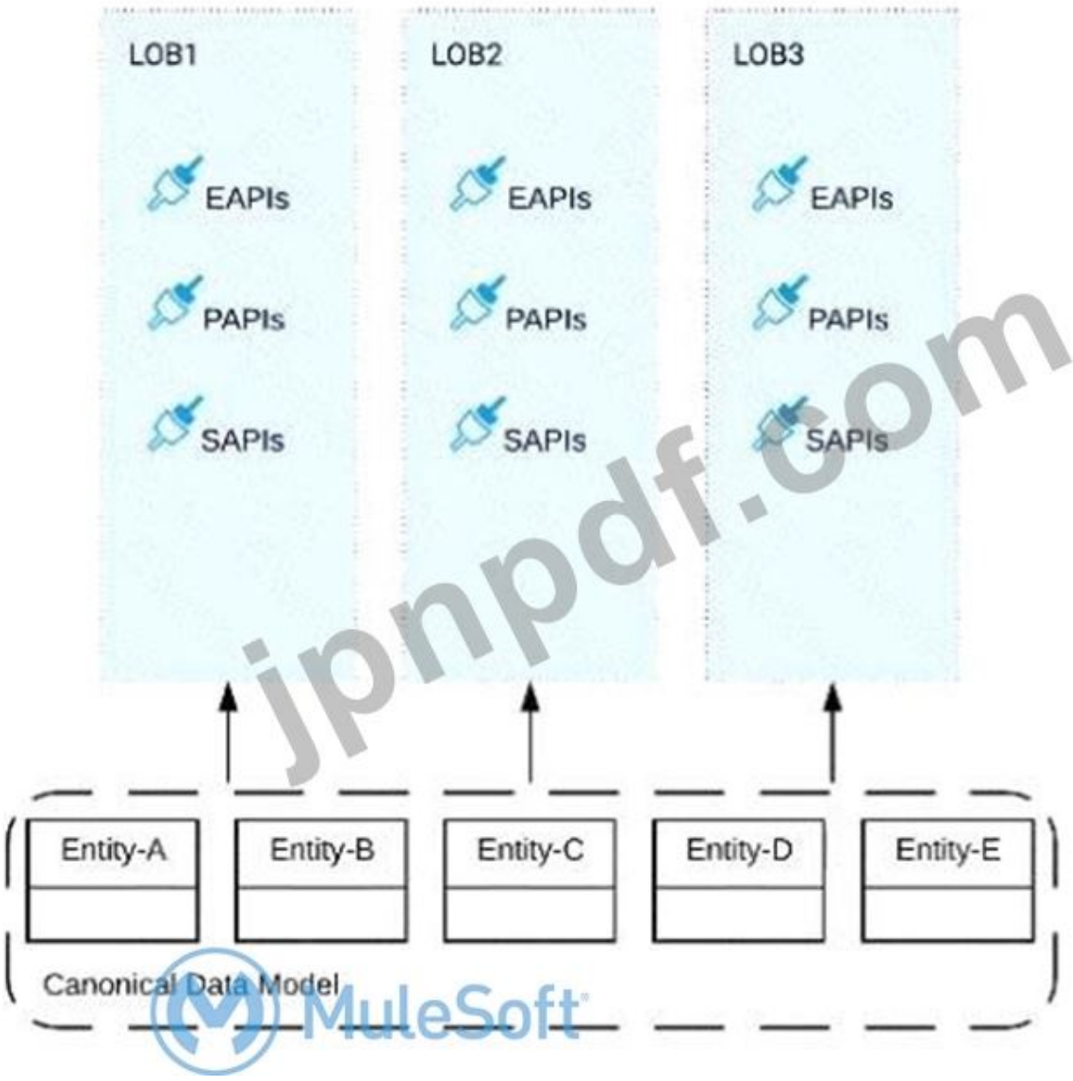


B) 確立されたマイクロサービスとアジャイルAPI中心のプラクティスに従うために、各APIごとに異なるデータモデルを構築する

C) 組織全体で一貫性と再利用性を高めるために、XMLスキーマを使用してすべてのAPIデータモデルを構築する



D) 3つのビジネスプロセスのすべてのデータタイプを統合し、データモデルの一貫性と冗長性を確保した、集中型の標準データモデル（エンタープライズデータモデル）を構築する



A. オプションA
B. オプションC
C. オプションD
D. オプションB
Answer: A (メーッセージを残す**)**

最新問題: **92**

一部の HTTP リクエストに対する応答は、リクエストで使用される HTTP 動詞に応じてキャッシュできます。
HTTP 仕様によれば、どの HTTP 動詞に対してこれを実行しても安全ですか?
A. PUT、POST、DELETE
B. GET、HEAD、POST
C. GET、PUT、オプション
D. GET、オプション、HEAD
Answer: D (メーッセージを残す)
説明/参照: http://restcookbook.com/HTTP%20Methods/idempotency/

最新問題**: 93**
質問10: スキップ
API 実装は、要求元の API クライアントに 3 つの X-RateLimit-* HTTP 応答ヘッダーを返します。これらの応答ヘッダーは API クライアントにどのような種類の情報を示しますか?
A. スロットリングの結果生じるエラーコード
B. 次のリクエストで送信される相関ID
C. HTTPレスポンスサイズ
D. API実装によって許可された残りの容量
Answer: D (メーッセージを残す)
API 実装によって許可される残りの容量。
****
>>
参考:https://docs.mulesoft.com/api-manager/2.x/rate-limiting-and-throttling-sla-based-policies#response-he

最新問題**: 94**
システム API を構築する際のベストプラクティスは何ですか?
A. RAML定義のような簡単に使用できるアセットを使用してAPIをドキュメント化する
B. すべてのAPIリソースとメソッドをモデル化して、バックエンドシステムの操作を厳密に模倣します。
C. 各バックエンドシステムのエンタープライズデータモデル（標準データモデル）を構築し、システムAPIに適用する

D. API実装とバックエンドシステムの相互作用に関するすべての技術的詳細をAPIクライアントに公開します。
Answer: B (メーッセージを残す**)**
正解: すべての API リソースとメソッドをモデル化して、バックエンド システムの操作を厳密に模倣します。
****
>> API のデータ モデルを選択する際に、固定された明確なベスト プラクティスはありません。それらは完全にコンテキストに依存し、多くの要因に依存します。これらの要因に基づいて、企業はエンタープライズ標準データ モデルまたは境界コンテキスト モデルなどを使用する必要があるかどうかを選択できます。
>> API 実装の技術的な詳細を API クライアントに公開しないでください。API クライアントに公開されるのは、API インターフェース/RAML のみです。
>> API の RAML 定義は可能な限り詳細にし、ドキュメントの大部分を反映させる必要があるのは事実です。ただし、それだけでは、API を最もよくドキュメント化された API と呼ぶには不十分です。開発者にとって使いやすい API とリポジトリを作成するには、Anypoint Exchange の API ノートブックなどに関するドキュメントがさらに多く必要です。
>> システム API を作成する際のベスト プラクティスは、バックエンド システムの操作と機能を厳密に反映するようにリソースとメソッドをモデル化して API インターフェイスを作成することです。

最新問題**: 95**
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 (メーッセージを残す**)**
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.
>> これらのテストは、実際のトランスポートを有効にした状態でアプリケーション全体を実行します。そのため、これらのテストを実行すると外部システムが影響を受けます。
したがって、これらのテストは、Mule アプリケーションがコンパイルされパッケージ化された直後には実行されません。
参考までに... ユニット テストは、Mule アプリケーションがコンパイルされパッケージ化された直後に実行されるテストです。

最新問題**: 96**
API 主導の接続性のどのレイヤーが、主要なシステム、レガシー システム、データ ソースなどのロックを解除し、機能を公開することに重点を置いていますか?

A. エクスペリエンスレイヤー
B. プロセス層
C. システム層
Answer: C (メーッセージを残す**)**
システム層

API 主導の接続アプローチで使用される API は、次の 3 つのカテゴリに分類されます。
システム API - これらは通常、レコードのコア システムにアクセスし、ユーザーを基盤システムの複雑さや変更から隔離する手段を提供します。一度構築されると、多くのユーザーは基盤システムを学習することなくデータにアクセスでき、複数のプロジェクトでこれらの API を再利用できます。
プロセス API - これらの API は、単一のシステム内またはシステム間で (データ サイロを解体して) データと対話してデータを形成します。これらの API は、データの元となるソース システムや、そのデータが配信されるターゲット チャネルに依存せずにここで作成されます。
エクスペリエンス API - エクスペリエンス API は、各チャネルに個別のポイントツーポイント統合を設定するのではなく、共通のデータ ソースからデータを再構成して、対象ユーザーが最も簡単に使用できるようにするための手段です。エクスペリエンス API は通常、API ファーストの設計原則に基づいて作成され、特定のユーザー エクスペリエンスを念頭に置いて API が設計されます。

最新問題**: 97**
Mule アプリケーションは HTTPS エンドポイントを公開し、静的 IP アドレスを使用しない 3 つの CloudHub ワーカーにデプロイされます。Mule アプリケーションは、短期間に大量のクライアント要求が発生することを想定しています。大量のクライアント要求に対応するために使用すべき、最もコスト効率の高いインフラストラクチャ コンポーネントは何ですか?
A. 顧客がホストするロードバランサ
B. CloudHub共有ロードバランサー
C. APIプロキシ
D. ランタイム マネージャーの自動スケーリング
Answer: (解答を表示する**)**
正解: CloudHub 共有ロードバランサー
****
この質問のシナリオは以下のように分割できます。

>> CloudHub ワーカーは 3 つあります (つまり、大量のリクエストを処理するのに十分な数のワーカーがすでにあります)
>> ワーカーは静的 IP アドレスを使用していません (したがって、静的 IP なしでは顧客の負荷分散ソリューションを使用することはできません)
>> ワーカー間でクライアント要求の負荷を分散するための最もコスト効率の高いコンポーネントを探しています。
シナリオに記載されている上記の詳細に基づきます。
>> 実行時の自動スケーリングは、追加コストがかかるため、まったく費用対効果が高くありません。ほとんどの場合、すでに 3 つのワーカーが実行されており、これは適切な数です。
>> 顧客がホストするロード バランサは、コスト効率が最も良くないため (保守とライセンスにカスタム ロード バランサが必要)、選択できません。また、Mule アプリには静的 IP アドレスがないため、カスタム ロード バランシングの使用が制限されます。
>> API プロキシは、大量の処理や負荷分散に関して役割を果たさないため、ここでは無関係です。
したがって、最もコスト効率が高く、シナリオの目的に合う唯一の適切なオプションは、CloudHub 共有ロードバランサを使用することです。

最新問題: **98**
An API implementation is being designed that must invoke an Order API, which is known to repeatedly experience downtime.
このため、Order API が利用できない場合は、フォールバック API が呼び出されます。
フォールバック API の呼び出しを設計する際に、どのようなアプローチが最高の回復力を提供しますか?
A. Order APIが利用できない場合は、HTTP 307 Temporary Redirectステータスコードを介してクライアントリクエストをフォールバックAPIにリダイレクトします。
B. HTTP 4xx または 5xx 応答ステータス コードが Order API から返されるたびに、Order API を呼び出す HTTP リクエスター コンポーネントに、フォールバック API を呼び出すオプションを設定します。
C. Anypoint Exchangeで適切な既存のフォールバックAPIを検索し、注文APIに加えてこのフォールバックAPIへの呼び出しを実装します。
D. API マネージャーで Order API の別のエントリを作成し、プライマリ Order API が利用できない場合にこの API をフォールバック API として呼び出します。
Answer: C (メーッセージを残す**)**

最新問題: **99**
一部の HTTP リクエストに対する応答は、リクエストで使用される HTTP 動詞に応じてキャッシュできます。
HTTP 仕様によれば、どの HTTP 動詞に対してこれを実行しても安全ですか?
A. PUT、POST、DELETE
B. GET、HEAD、POST
C. GET、PUT、オプション
D. GET、オプション、HEAD

Answer: A (メーッセージを残す)

最新問題: **100**
ある企業では、CloudHub にデプロイされた Mule アプリケーションを非本番環境と本番環境の間で分離する必要があります。これは、非本番環境にデプロイされた Mule アプリケーションが、顧客がホストする非本番環境で実行されているバックエンド システムにのみアクセスできるようにするためであり、本番環境にデプロイされた Mule アプリケーションが、顧客がホストする本番環境で実行されているバックエンド システムにのみアクセスできるようにするためです。MuleSoft では、Mule アプリケーションとバックエンド システム間のこのような環境ごとの分離をサポートするために、Mule アプリケーションの変更、環境の構成、またはインフラストラクチャの変更をどのように推奨していますか?
A. 本番環境の Anypoint Platform にデプロイされた Mule アプリケーションのプロパティを変更して、本番環境以外の Mule アプリケーションからのアクセスを防止します。
B. それぞれの顧客ホスト環境内のインフラストラクチャでファイアウォールルールを設定し、対応するAnypoint Platform環境のIPアドレスのみが対応するバックエンドシステムと通信できるようにします。
C. 異なるAnypoint Platformビジネスグループに非本番環境と本番環境を作成する
D. 非本番環境と本番環境用に別々の Anypoint VPC を作成し、対応する顧客ホスト環境のバックエンド システムへの接続を設定します。
Answer: D (メーッセージを残す)
非本番環境と本番環境用に個別の Anypoint VPC を作成し、対応する顧客ホスト環境のバックエンド システムへの接続を構成します。
****
>> 異なるビジネス グループを作成しても、非本番環境と本番環境の顧客ホスト環境へのアクセスには影響しません。プロセス ネットワーク制限が設定されていない限り、両方のビジネス グループからアクセスすることになります。
>> Mule アプリケーション実装を環境に合わせて変更または結合する必要があります。実際、プロパティにバインドして環境と結合したアプリケーションを実装することは絶対に避けてください。エンドポイント URL などの基本的なものだけをプロパティにバンドルし、環境レベルのアクセス制限はバンドルしないでください。
>> CloudHub 上の IP アドレスは、特別な静的アドレスが割り当てられない限り動的です。そのため、顧客がホストするインフラストラクチャでファイアウォール ルールを設定することはできません。さらに、静的 IP アドレスが割り当てられている場合でも、CloudHub 上で実行されるアプリケーションは数百に上る可能性があり、それらすべてにルールを設定するのは大変な作業で、メンテナンスが不可能であり、間違いなく良い習慣になります。
>> Mulesoft (実際はどのクラウド プロバイダーでも) が推奨するベスト プラクティスは、Anypoint VPC を Prod と Non-Prod に分離し、これらの Anypoint VPC に対して、それぞれの Prod および Non-Prod の顧客ホスト環境ネットワークへの VPC ピアリングまたは VPN トンネリングを実行することです。

最新問題: **101**

下流 API の応答時間と需要を改善するという目標を達成するために、サーキット ブレーカー戦略が計画されています。
* サーキットオープン: 3分間に1分あたり10件以上のエラー
* 回路半開: 1分あたり1回のエラー
* 回路が閉じている: 5分間に1分あたり1件未満のエラー
エンジニアリング チームからのいくつかの提案のうち、どのオプションがこの目標を満たすでしょうか?
A. サーキットブレーカーを実装し、必要な設定のポリシーテンプレート式を含むカスタムポリシーを作成します。
B. 回線オープン/クローズ構成のAnypoint Monitoringアラートを作成し、回線ハーフオープン構成の再試行戦略を実装します。
C. APIインスタンスにサーキットブレーカーポリシーを追加し、必要な設定を構成します。
D. Muleアプリケーションで戦略を実装し、YAML構成で設定を提供します。
Answer: C (メーッセージを残す)
* サーキットブレーカーポリシーの理解:
* サーキット ブレーカーは、障害を検出し、アプリケーションが失敗した操作を継続的に実行しようとするのを防ぐために使用される設計パターンです。この場合、応答時間を改善し、ダウンストリーム API の需要を減らすのに役立ちます。
* 指定された構成には、時間の経過に伴うエラー率に基づいて回路を開く、半開にする、閉じる条件が含まれます。
* 回路オープン: 3 分連続で 1 分あたり 10 件を超えるエラーが発生した場合にトリガーされます。
* 回路半開: 1 分間に 1 つのエラーが発生すると、回路は半開状態に移行します。
* 回路が閉じる: エラー率が 5 分間に 1 分あたり 1 エラー未満になると、回路が閉じます。
* オプションの評価:
* オプション A: テンプレート式を使用してカスタム ポリシーを作成することもできますが、カスタム開発が必要になります。Anypoint Platform にはすでに Circuit Breaker ポリシーが用意されているため、このソリューションは効率が低く、複雑になります。
* オプション B: Anypoint Monitoring アラートは API の監視に使用できますが、サーキット ブレーカー機能は提供されません。また、半開状態の再試行戦略を実装するだけでは、必要なサーキット ブレーカーの動作を実現するには不十分です。
* オプション C (正解): Anypoint Platform の API インスタンスに Circuit Breaker ポリシーを追加すると、回路遮断条件を直接設定できます。このアプローチでは、組み込みの Circuit Breaker ポリシーを使用し、要件に合わせてエラーしきい値や時間間隔などのパラメータを設定できます。このソリューションは効率的で信頼性が高く、Anypoint のすぐに使用できる機能を活用します。
* オプション D: YAML 構成を使用して Mule アプリケーション内で戦略を実装すると、複雑になり、管理しにくくなる可能性があります。また、このシナリオに適した Anypoint Platform の組み込み Circuit Breaker ポリシーは活用されません。
* 結論：

* オプション C は、Anypoint Platform の Circuit Breaker ポリシーを活用するため、正しい選択です。このソリューションでは、しきい値と時間間隔を指定どおりに設定できるため、Anypoint の管理ポリシー機能を活用しながら、応答時間を改善し、ダウンストリーム API に対する需要を削減できます。
詳細な構成ガイダンスについては、API Manager での Circuit Breaker ポリシーの実装に関する MuleSoft のドキュメントを参照してください。

最新問題: **102**
共有ロードバランサで CloudHub を使用する場合、Anypoint Platform ではなく API 実装 (Mule アプリケーション) によって排他的に管理されるものは何ですか?
A. 各HTTPリクエストを特定のCloudHubワーカーに割り当てる
B. ログエントリをランタイムマネージャーで表示できるようにするログ設定
C. API実装がHTTPSエンドポイントを公開するために使用するSSL証明書
D. API実装に割り当てられたDNSエントリの数
Answer: (解答を表示する**)**

最新問題: **103**
Anypoint Platform の ID 管理とクライアント管理について正しい記述はどれですか?
A. 外部アイデンティティプロバイダが設定されている場合、アイデンティティプロバイダによって発行されたSAML 2.0ベアラトークンはAnypoint Platform Web APIの呼び出しには使用できません。
B. 外部クライアントプロバイダが設定されている場合、Anypoint Platform 組織レベルで設定する必要があり、個々のビジネスグループや環境に割り当てることはできません。
C. Anypoint Platformは、1つの外部IDプロバイダの設定をサポートします。
D. クライアント管理とID管理の両方にIDプロバイダが必要です
Answer: C (メーッセージを残す**)**
Anypoint Platform を使用すると、組織は ID およびアクセス管理 (IAM) 用に 1 つの外部 ID プロバイダー (IdP) を統合し、SSO と集中型ユーザー認証をサポートできます。
* アイデンティティプロバイダーの制限:
* Anypoint Platform は、組織に対して単一の IdP を構成することをサポートしており、これを使用して、Anypoint 組織内のビジネス グループと環境全体のすべてのユーザーを認証できます。
* 正解 C)の説明：
* 1 つの IdP を構成すると、MuleSoft のアーキテクチャに沿った集中化された安全な ID 管理が保証されます。
* 誤ったオプションの説明:
* オプション A は不正解です。外部 IdP からの SAML 2.0 ベアラー トークンは、Anypoint Platform API の呼び出しに実際に使用できます。
* クライアント プロバイダーは特定のビジネス グループと環境に割り当てることができるため、オプション B は不正解です。
* オプション D は不正解です。ID 管理のみに厳密に IdP が必要であり、クライアント管理には必要ないためです。

参考資料: ID 管理オプションの詳細については、Anypoint Platform の IAM 機能に関する MuleSoft のドキュメントを参照してください。

最新問題: **104**
組織は、既知のパートナーのみが組織の API を呼び出せるようにしたいと考えています。このセキュリティ目標を達成するために、組織は API Manager でクライアント ID 強制ポリシーを適用し、登録されたパートナー アプリケーションのみが組織の API を呼び出せるようにしたいと考えています。MuleSoft は、どのようなタイプの API 実装で、アプリケーションの JVM にポリシーを直接埋め込むのではなく、API プロキシを追加してクライアント ID 強制ポリシーを適用することを推奨していますか?
A. APIkit を使用した Mule 3 アプリケーション
B. カスタム Java コードで変更された Mule 3 または Mule 4 アプリケーション
C. API仕様を備えたMule 4アプリケーション
D. 非 Mule アプリケーション
Answer: D (メーッセージを残す**)**
非Muleアプリケーション
****
>> Mule ランタイム上で実行されるすべてのタイプの Mule アプリケーション (Mule 3/Mule 4/APIkit 付き/カスタム Java コード付きなど) は、埋め込みポリシーの適用をサポートしています。
>> 埋め込みポリシーの適用ができない、またはサポートされておらず、API プロキシが必要な唯一のオプションは、非 Mule アプリケーション用です。
したがって、非 Mule アプリケーションが正しい答えです。

最新問題: **105**
システム API の API データ モデルは、バックエンド システムのデータ モデルを最小限に改良して、対応するバックエンド システムによって公開されるデータ モデルを適切に模倣できる場合とはどのような場合ですか。
A. 対応するバックエンドシステムが近い将来に置き換えられることが予想される場合
B. システムAPIを対応するデータモデルを持つ境界付きコンテキストに割り当てることができる場合
C. バックエンドシステムから限定的に分離した実用的なアプローチが適切であると判断された場合
D. 組織全体で広く使用されている既存のエンタープライズデータモデルがある場合
Answer: (解答を表示する**)**

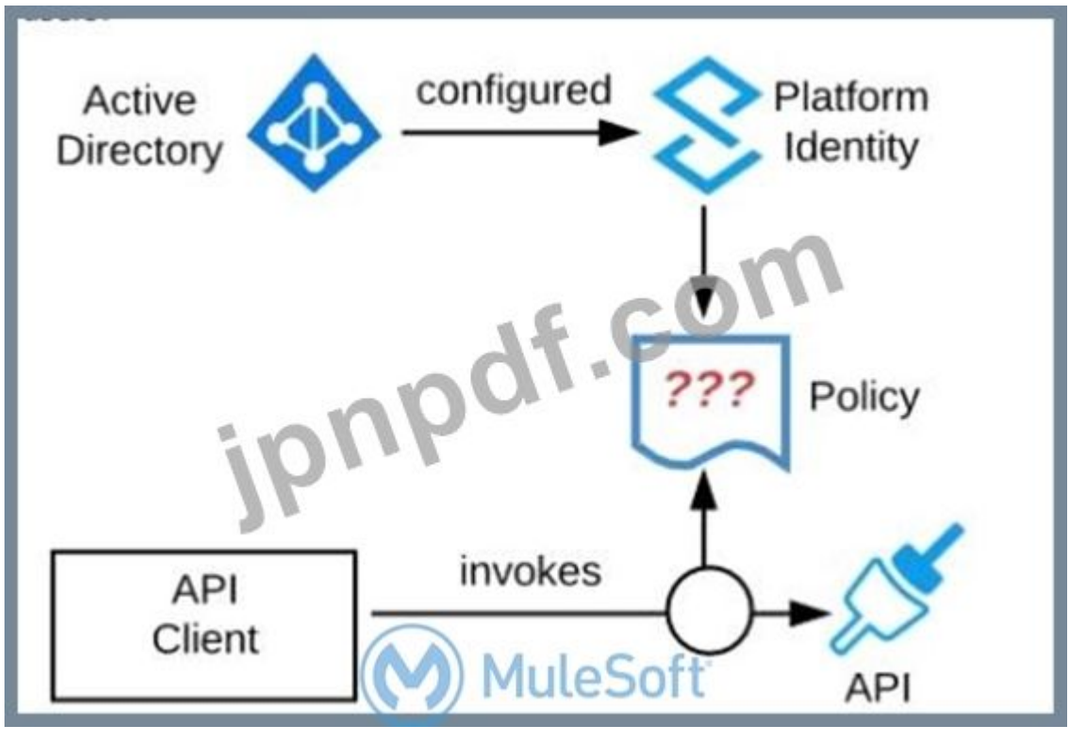
最新問題: **106**
運用チームは、アプリケーション ネットワークの監視を設定するために必要な作業を分析しています。カスタム スクリプトを作成したり、追加の分析ソフトウェアやツールをインストールしたりすることなく、どの API 呼び出しメトリックを使用してトラブルを特定および予測できるかを検討しています。
障害を直接特定して予測するという目標を満たすことができるのは、どのタイプのメトリックですか?
A. 1日あたりのAPIポリシー違反の数と種類
B. 再利用レベルに基づくアプリケーションネットワークの有効性
C. アプリケーションネットワーク全体での過去のAPI呼び出しの数と種類

D. 各APT呼び出しからのROI
Answer: (解答を表示する**)**
アプリケーション ネットワークを監視し、カスタム スクリプトを使用せずに問題を予測するには、ポリシー違反メトリックが重要です。ポリシー違反メトリックは、API の使用が定義されたポリシーに準拠していないインスタンスを追跡することで、潜在的な問題に関する洞察を提供します。このアプローチが適している理由は次のとおりです。
* 予測監視:
* API ポリシー違反 (レート制限やスパイク制御のヒットなど) を追跡すると、トラフィックの急増や誤用が示唆される可能性があり、対処しないとスロットリングやサービスの低下につながる可能性があります。
* これらの違反を監視することで、チームは制限を積極的に調整したり、API 処理を最適化したりして、実際の障害を防ぐことができます。
* カスタムスクリプトは不要:
* ポリシー違反メトリックは MuleSoft の Anypoint Monitoring 内で利用できるため、このデータを収集して解釈するためにカスタム ソリューションや外部ツールを実装する必要はありません。
* 誤ったオプションの説明:
* オプション B (再利用に基づく有効性) では、障害を直接予測することはできません。
* オプション C (過去の呼び出し回数) では、使用状況の履歴データが提供されますが、本質的には問題が特定されるわけではありません。
* オプション D (API 呼び出しからの ROI) はビジネス メトリックであり、障害予測に関する技術的な洞察は提供されません。
参考資料: プロアクティブな監視にポリシー違反メトリックを活用する方法の詳細については、Anypoint Monitoring に関する MuleSoft のドキュメントを参照してください。

最新問題**: 107**
顧客は、MuleSoft アプリケーションを CloudHub 1.0 でホストしたいと考えています。これらのアプリケーションは、ドメイン https://api.acmecorp.com で利用できる必要があります。
acme-dib-prod という専用ロードバランサー (DLB) を作成した後、構成を完了するために顧客がさらに実行する必要があるアクションは何ですか?
A. api.acmecorp.com の TLS 証明書を使用して DLB を構成し、api の A レコードを作成します。
acmecorp.com を DLB に関連付けられたパブリック IP アドレスに接続します。
B. api.acmecorp.com の TLS 証明書を使用して DLB を構成し、api から CNAME レコードを作成します。
acmecorp.com から acme-dib-prod.|lb.anypointdns.net へ

C. acme-dib-prod.Jb.anypointdns.net の TLS 証明書を使用して DLB を設定し、api.acmecorp:com から acme-dlb-prod.lb.anypointdns.net への CNAME レコードを作成します。
D. aplacmecorp.com の TLS 証明書を使用して DLB を構成し、api.aomecorp.com から acme-dib-prod.ei.cloubhub.io への CNAME レコードを作成します。
Answer: B (メーッセージを残す**)**
専用ロードバランサ (DLB) を使用して CloudHub 1.0 でホストされている MuleSoft アプリケーションのカスタム ドメインを設定する場合は、次の手順に従います。
* TLS 証明書の設定: カスタム ドメイン api.acmecorp.com をカバーする TLS 証明書を使用して DLB (acme-dib-prod) を構成します。この証明書により、HTTPS トラフィックを DLB 経由で安全に Mule アプリケーションに送信できるようになります。
* CNAMEを使用したDNS設定:
* api.acmecorp.com を DLB ホスト名 acme-dib-prod.lb にポイントする CNAME レコードを作成します。.net です。
* CNAME レコードにより、カスタム ドメインは MuleSoft の Anypoint Platform によって提供される DLB に解決できるようになります。この CNAME マッピングにより、すべてのトラフィックが適切な DLB に送信され、処理と負荷分散が行われます。
* オプション B が正しい理由:
* CNAME レコードは、DLB 用に Anypoint Platform によって管理されるエンドポイントである acme-dib-prod.lb.anypointdns.net への必要なエイリアスを提供します。
* オプション B では、DLB の内部ホスト名ではなく、api.acmecorp.com 専用の TLS 証明書を使用して DLB を構成する必要があることも正しく識別されます。
* 誤ったオプションの説明:
* DLB の内部ホスト名の TLS 証明書を使用して DLB を構成するか、A レコードを使用することを提案するオプションは、このシナリオには適していません。MuleSoft CloudHub 1.0 DLB は CNAME レコードを使用して柔軟でスケーラブルなドメイン管理を提供しますが、これらのロード バランサーでは直接 IP (A レコード) はサポートされていません。
参考資料: CloudHub 1.0 でのカスタム ドメインと DLB の構成の詳細については、DLB のセットアップと DNS 構成に関する MuleSoft のドキュメントを参照してください。

最新問題**: 108**
展示を参照してください。組織は Mule スタンドアロン ランタイムを実行しており、Active Directory を Anypoint Platform 外部 ID プロバイダーとして構成しています。組織には他のシステム コンポーネントのための予算がありません。



特定の内部ユーザー グループへのアクセスを最も効果的に制限するには、組織内のすべての API インスタンスにどのようなポリシーを適用する必要がありますか?

A. 基本認証 - LDAP ポリシーを適用します。内部 Active Directory がユーザー認証用の LDAP ソースとして構成されます。

B. クライアントID強制ポリシーを適用します。特定のユーザーグループは、特定のクライアント資格情報を使用するようにクライアントアプリケーションを構成します。

C. IPホワイトリストポリシーを適用します。特定のユーザーのワークステーションのみがホワイトリストに追加されます。

D. Apply an OAuth 2.0 access token enforcement policy; the internal Active Directory will be configured as the OAuth server

Answer: (解答を表示する**)**

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.

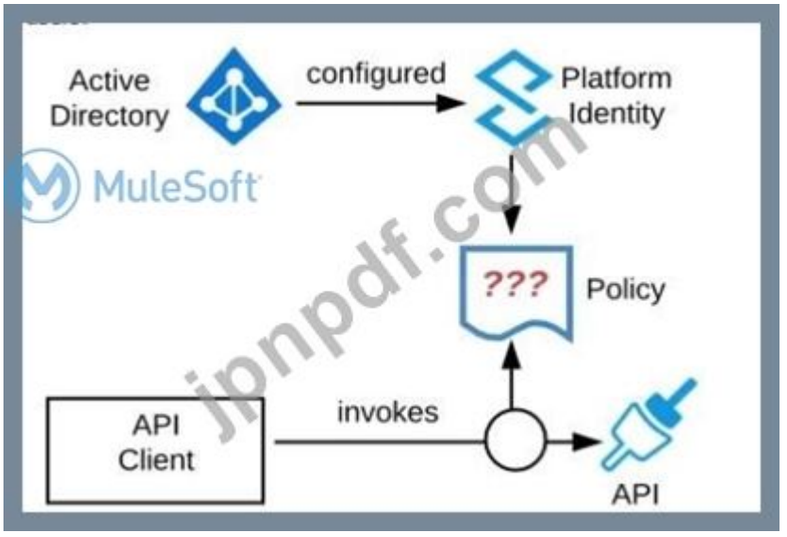
>> OAuth 2.0 enforcement requires a client provider which isn't in the organizations system components.

>> 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.

最新問題**: 109**

展示を参照してください。組織は Mule スタンドアロン ランタイムを実行しており、Active Directory を Anypoint Platform 外部 ID プロバイダーとして構成しています。組織には他のシステム コンポーネントのための予算がありません。



特定の内部ユーザー グループへのアクセスを最も効果的に制限するには、組織内のすべての API インスタンスにどのようなポリシーを適用する必要がありますか?

A. 基本認証 - LDAP ポリシーを適用します。内部 Active Directory がユーザー認証用の LDAP ソースとして構成されます。

B. OAuth 2.0アクセストークン強制ポリシーを適用します。内部Active DirectoryがOAuthサーバーとして設定されます。

C. IPホワイトリストポリシーを適用します。特定のユーザーのワークステーションのみがホワイトリストに追加されます。

D. クライアントID強制ポリシーを適用します。特定のユーザーグループは、特定のクライアント資格情報を使用するようにクライアントアプリケーションを構成します。

Answer: D (メーッセージを残す**)**

最新問題**: 110**

ある組織が、OrderStatus システム API の新しい実装を CloudHub の複数のワーカーにデプロイしています。この API は組織のオンプレミスの注文管理システムの前面に配置されており、API 実装によって IPsec トンネル経由でアクセスされます。

通常、OrderStatus システム API のサービス停止を引き起こさないエラーの種類は何ですか?

A. CloudHub ワーカーがメモリ不足例外で失敗する

B. API 実装の初期展開中に API Manager が長時間停止しました

C. 関連するAWSデータセンターへの大規模なネットワーク障害によりAWSリージョンがオフラインになる

D. 組織のオンプレミス データ センターのネットワーク障害のため、注文管理システムにアクセスできません。

Answer: (解答を表示する**)**

正解: CloudHub ワーカーがメモリ不足例外で失敗します。

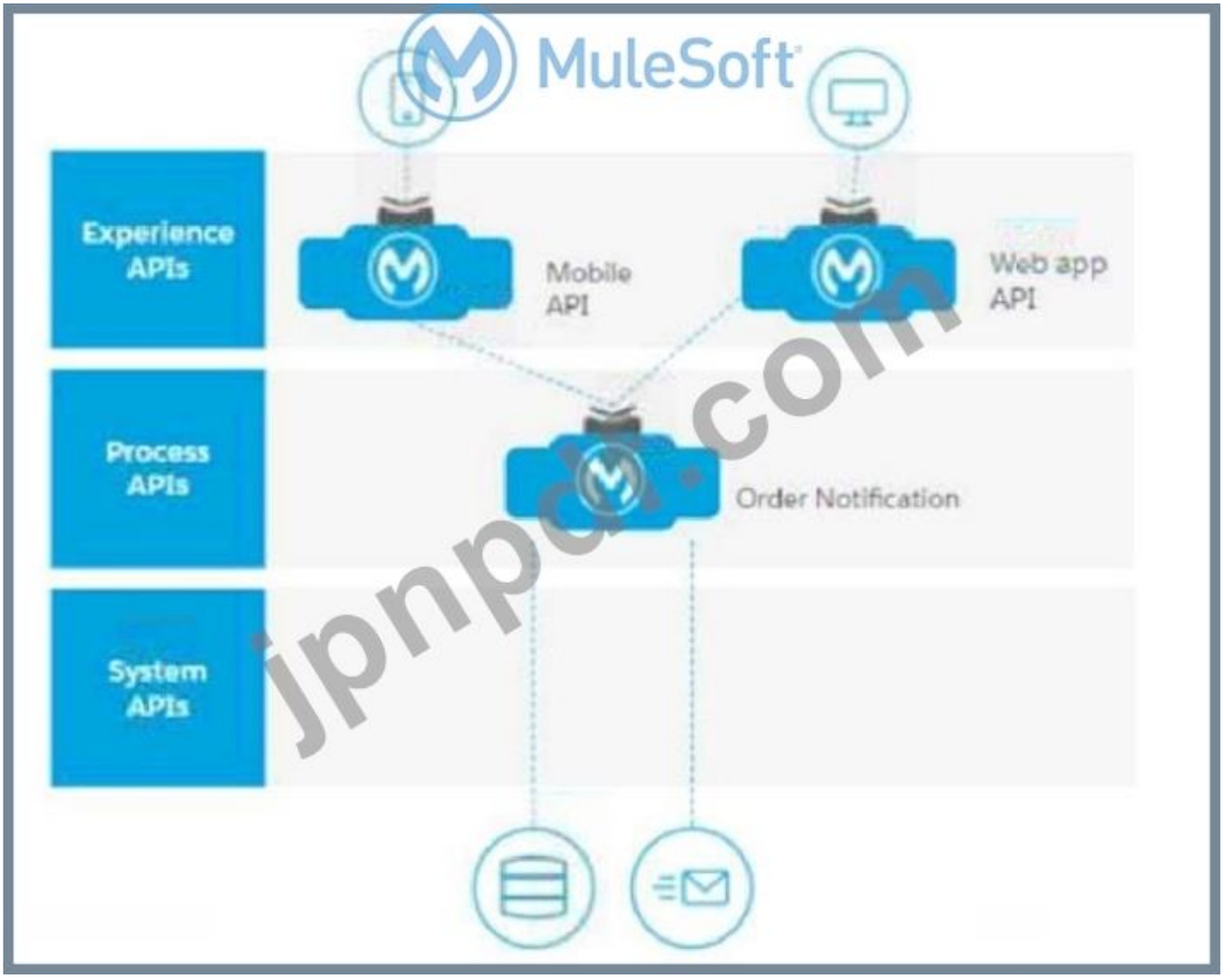
****

>> AWS リージョン自体がダウンすると、Mule アプリに割り当てられているワーカーの数は関係なく、そのリージョン内のすべてのワーカーがダウンするため、確実に停止が発生します。これは完全なダウンタイムと停止です。
>> API 実装の初期展開中に API マネージャーが長時間停止すると、当然のことながら、API 自動検出が失敗したり、API ポリシー テンプレートとポリシーがアプリケーションの起動時にダウンロードされずに埋め込まれなかったりするなど、アプリケーションの適切な起動自体に問題が発生します。問題が発生する理由は多数あります。
>> オンプレミスのネットワークが停止すると、当然ながら注文管理システムにアクセスできなくなります。また、アプリに何人の作業者が割り当てられていても、全員が失敗し、確実に停止が発生します。
サービス停止につながらない唯一のオプションは、クラウドハブ ワーカーがメモリ不足例外で失敗した場合です。ワーカーが失敗してダウンした場合でも、リクエストを処理して API を稼働状態に保つ他のワーカーがまだ存在します。したがって、これが正しい答えです。

最新問題**: 111**
Mule アプリケーションは HTTPS エンドポイントを公開し、CloudHub Shared Worker Cloud にデプロイされます。その Mule アプリケーションへのすべてのトラフィックは AWS VPC 内に留まる必要があります。
Mule アプリケーションへの API 呼び出しはどの TCP ポートに送信する必要がありますか?
A. 443
B. 8081
C. 8091
D. 8082
Answer: D (メーッセージを残す**)**
正解: 8082
****
>> 8091 ポートと 8092 ポートは、それぞれ HTTP アプリと HTTPS アプリをローカル VPC に対してプライベートに保つ場合に使用します。
>> 上記の 2 つのポートは、共有 AWS VPC/共有ワーカー クラウド用ではありません。
>> 8081は、Shared LBを介してHTTPエンドポイントアプリをインターネットに公開するときに使用されます。
>> 8082 は、Shared LB を介して HTTPS エンドポイント アプリをインターネットに公開するときに使用されます。したがって、この HTTPS ベースのアプリを呼び出すときは、API 呼び出しをポート 8082 に送信する必要があります。
参考文献:
https://docs.mulesoft.com/runtime-manager/cloudhub-networking-guide
https://help.mulesoft.com/s/article/Configure-Cloudhub-Application-to-Send-a-HTTPS-Request-Directly-to-Another-Cloudhub-Application
https://help.mulesoft.com/s/question/0D52T00004mXXULSA4/multiple-http-listerners-on-cloudhub-one-with-port-9090

最新問題**: 112**

電子商取引会社の事業部門 (LoB) では、顧客のモバイル アプリケーションまたは社内の Web アプリケーションを通じて新しい注文が処理されるたびに、電子メールで自動通知を送信するプロセスを要求しています。将来的には、テキスト メッセージやプッシュ通知など、複数の通知チャネルが追加される可能性があります。
上記のシナリオで最も効果的な API 主導の接続アプローチは何ですか?
A. Web アプリケーション用とモバイル アプリケーション用に 1 つの Experience API を作成します。
プロセス API を作成して、データベースから電子メール テンプレートを調整および取得します。
Anypoint Connector for Email を使用して電子メールを送信するシステム API を作成します。
Web アプリケーション用とモバイル アプリケーション用に 1 つの Experience API を作成します。
プロセス API を作成して、データベースから電子メール テンプレートを調整および取得します。
Anypoint Connector for Email を使用して電子メールを送信するシステム API を作成します。
B. Web アプリケーション用とモバイル アプリケーション用に 1 つの Experience API を作成し、オーケストレーションを行う Process API を作成し、データベースから電子メール テンプレートを取得し、Anypoint Connector for Email を使用して電子メールを送信します。



C. Web アプリケーションとモバイル アプリケーションの両方に Experience API を作成します。
プロセス API を作成して、電子メール テンプレートをオーケストレーションし、データベースから取得し、Anypoint Connector for Email を使用して電子メールを送信します。

\
D. Web アプリケーションとモバイル アプリケーションの両方に Experience API を作成します。
(2 つのデータベースから電子メール テンプレートを調整および取得するための 3 つのプロセス API を作成します。
Anypoint Connector for Email を使用して電子メールを送信するシステム API を作成します。
Answer: (解答を表示する**)**
このシナリオでは、API 主導の接続原則を満たし、将来のスケーラビリティをサポートするための最善のアプローチは次のとおりです。
* エクスペリエンスAPI:
* Web アプリケーションとモバイル アプリケーション用に個別のエクスペリエンス API を作成します。これにより、各アプリケーションは最適化されたインターフェイスを持つことができ、さまざまなニーズや、リクエスト/レスポンス構造やセキュリティ構成の潜在的な違いをサポートできます。
* プロセスAPI:
* 単一のプロセス API を使用して、データベースから電子メール テンプレートを取得し、電子メール コンテンツを準備するなど、ワークフローを調整できます。このロジックをプロセス レイヤーに集中させることで、将来的にさまざまな通知チャネルに再利用して簡単に適応できるようになります。
* システムAPI:
* 電子メールの送信専用に設計されたシステム API (Anypoint Connector for Email を使用) は、ビジネス ロジックから電子メール送信機能を抽象化します。このアプローチにより、電子メール送信機能が再利用可能かつスケーラブルになり、後で他の通知チャネル (SMS やプッシュ通知など) が追加されても簡単に拡張または変更できます。
オプション A が正しい理由:
* この構造は、エクスペリエンス、プロセス、システム レイヤー間で懸念事項を分離することで、API 主導の接続原則と一致しています。これにより、将来の通知チャネルに柔軟性が提供され、各レイヤーの責任が分離されるため、保守と拡張が容易になります。
誤ったオプションの説明:
* オプション B には、電子メールを送信するための個別のシステム API がないため、システム API でバックエンド機能を分離するという原則に反します。
* オプション C にも同様に専用のシステム API がないため、柔軟性と再利用性が低下します。
* オプション D では、データベース取得用に複数のプロセス API を作成することが提案されていますが、これにより不必要な複雑さが増し、API 主導の設計で通常従われる単一オーケストレーションの原則に準拠しなくなります。
参考資料 API 主導の接続性と各 API レイヤーの役割に関する詳細なガイダンスについては、MuleSoft の API 主導のアーキテクチャと設計のベスト プラクティスに関するドキュメントを参照してください。

最新問題**: 113**
特定のビジネス プロセスを実装するために、粗粒度ではなく細粒度の API デプロイメント モデルを使用した場合の一般的な結果は何ですか。
A. ビジネスプロセスをサポートするアプリケーションネットワーク内の接続数の減少

B. アプリケーションネットワーク内で検出可能なAPI関連アセットの数の増加
C. API の範囲と複雑さが小さくなるため、エンドユーザーへの応答時間が短縮されます。
D. 細粒度APIごとに消費するリソースが少なくなるため、タワー全体のリソース使用量が削減されます。
Answer: B (メーッセージを残す**)**
アプリケーション ネットワーク内で検出可能な API 関連アセットの数が増えます。
****

>> 粗粒度のアプローチと比較した場合、細粒度のアプローチでは応答時間が速くなりません。
>> 実際、粒度の粗い API モデルを持つネットワークでは、粒度の細かい API モデルを持つネットワークよりも応答時間が短くなります。その理由は次のとおりです。
きめ細かなアプローチ:
1. 粗粒度に比べてAPIの数が多くなる
2. したがって、ビジネス プロセスの機能性を実現するには、さらにオーケストレーションを行う必要があります。
3. つまり、大量の API 呼び出しが必要になります。そのため、より多くの接続を確立する必要があります。したがって、大量の機能が組み込まれた API が少ない粗粒度のアプローチと比較すると、ホップ、ネットワーク I/O、統合ポイントの数が増えるのは明らかです。
4. そのため、これらすべての追加ホップと追加のレイテンシにより、細粒度のアプローチでは、粗粒度のアプローチに比べて応答時間が少し長くなります。
5. レイテンシと接続が追加されるだけでなく、API の数が増えるため、きめ細かいアプローチで使用されるリソースも増えます。
そのため、きめ細かな API は、ネットワーク内で再利用可能な資産をより多く公開し、検出できるようにするのに適しています。ただし、ネットワーク ホップと応答時間に関して多少の妥協をしながら、統合ポイント、接続、リソースを管理するために、より多くのメンテナンスが必要になります。

最新問題**: 114**
正しいか間違いか。たとえより多くの人的労力とリソースが必要になったとしても、設計および開発中の API がセルフサービス可能であることを常に確認する必要があります。
A. 偽
B. 真
Answer: B (メーッセージを残す**)**
正解: TRUE
****

>> MuleSoft が提案する IT 運用モデルによれば、API を設計し、それらが検出可能かつセルフサービス可能であることを確認することは非常に重要であり、API とそのアプリケーション ネットワークの成功を決定します。

最新問題**: 115**
Anypoint Platform で API ポリシーが定義される場所と、それが API インスタンスにどのように適用されるかについて正しいのはどれですか?
A. APIポリシーはAPI Managerで定義され、すべてのAPIインスタンスに自動的に適用されます。

B. API ポリシーは、Mule ランタイムへの API デプロイメントの一部として Runtime Manager で定義され、特定の API インスタンスにのみ適用されます。
C. APIポリシーはAPI Managerで定義され、指定された環境内のすべてのAPIインスタンスに適用されます。
D. API ポリシーは、特定の API インスタンスに対して API Manager で定義され、特定の API インスタンスにのみ適用されます。
Answer: B (メーッセージを残す**)**

最新問題**: 116**
イノベーションとクロックスピードを向上させるために MuleSoft が組織に推奨する IT 運用モデルの主な変更点は何ですか?
A. アセットの生産と同様に消費も促進します。これにより、開発者は他のプロジェクトからアセットを発見して再利用できるようになり、標準化が促進されます。
B. マスターデータ管理 (MDM) システムを使用して資産を公開します。これによりプロジェクトが標準化され、開発者は他のプロジェクトから資産をすばやく発見して再利用できるようになります。
C. 再利用可能な API に SOA を実装して、消費よりも生産に重点を置きます。これにより、XML および WSDL 形式が標準化され、意思決定が迅速化されます。
D. 毎日多くの小さな決定を下す、無駄のない機敏な組織を構築します。これにより意思決定が迅速化され、各事業部門がプロジェクトの所有権を取得できるようになります。
Answer: A (メーッセージを残す**)**
正解: アセットの生産と同様に消費も促進します。これにより、開発者は他のプロジェクトからアセットを発見して再利用できるようになり、標準化が促進されます。
****
>> MuleSoft が推奨し普及させた新しい IT 運用モデルの主なモットーは、API 主導の接続性と呼ばれる API 戦略を通じて、提供方法を実稼働モデルから実稼働 + 消費モデルに変更することです。
>> 構築された資産は、LOB や組織全体で再利用できるように、検出可能でセルフサービス可能である必要があります。
>> MuleSoft の IT 運用モデルでは、SDLC モデル (Agile/Lean など) や MDM についてはまったく触れられていません。したがって、これらを提案するオプションは無効です。
参考文献:
https://blogs.mulesoft.com/biz/connectivity/what-is-a-center-for-enablement-c4e/
https://www.mulesoft.com/resources/api/secret-to-managing-it-projects

最新問題**: 117**
API クライアントは、既存の API 実装から 1 つのメソッドを呼び出します。API 実装は後で更新されます。API 実装にどのような変更を加えると、API クライアントの呼び出しロジックも更新する必要がありますか?
A. APIクライアントによって呼び出されたメソッドのレスポンスのデータ型が変更された場合
B. APIクライアントが使用するリソースに新しいメソッドが追加されたとき
C. APIクライアントによって呼び出されたメソッドに新しい必須フィールドが追加されたとき
D. APIクライアントによって呼び出されたメソッドに子メソッドが追加された場合

Answer: C (メーッセージを残す)
正解: APIクライアントによって呼び出されたメソッドに新しい必須フィールドが追加されたとき
****
>> 一般的に、API 契約が破綻した場合、API クライアントのロジックを更新する必要があります。
>> API に新しいメソッドまたは子メソッドが追加されても、API クライアントは既存のメソッドを引き続き使用できるため、動作が中断されることはありません。したがって、これら 2 つのオプションは無効です。
>> 残っているのは、 応答のデータ型が変更された場合」と 新しい必須フィールドが追加された場合」の 2 つです。
>> 応答のデータ型を変更すると、API 契約が破棄されます。ただし、質問は 呼び出し」ロジックに関するものであり、応答処理ロジックに関するものではありません。API クライアントは引き続き API を正常に呼び出して応答を受け取ることができますが、応答の一部のフィールドのデータ型は異なります。
>> 新しい必須フィールドを追加すると、API の呼び出し契約が破棄されます。新しい必須フィールドを追加すると、API 契約により、API クライアント/API コンシューマーと API プロバイダーの間で締結された RAML または API 仕様の合意が破棄されます。そのため、API クライアントの呼び出しロジックも更新する必要があります。

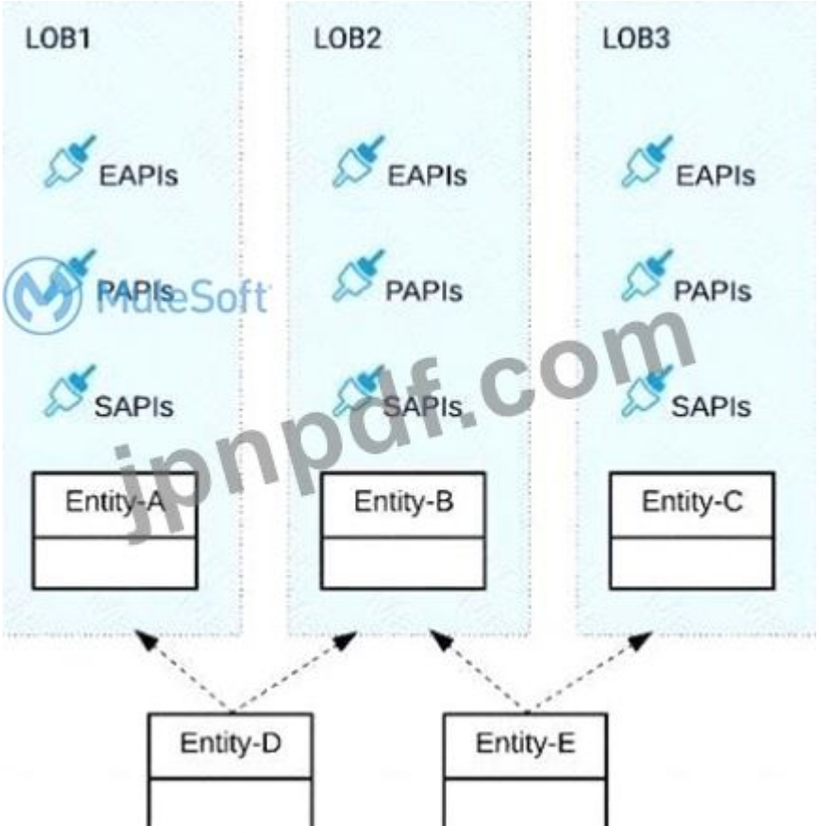
最新問題**: 118**
プラットフォーム アーキテクトは、クライアントに属するすべてのポリシーの表示など、さまざまなタスクを実行する従来のモノリシック SOAP ベースの Web サービスを継承します。このサービスは、生命保険管理システムと損害保険管理システムの 2 つのバックエンド システムに接続し、各システム内で保険ポリシー情報を照会して結果を集約し、ユーザー インターフェイス (UI) に SOAP ベースの応答を表示します。
アーキテクトは、API 主導の規則に従うためにモノリシック Web サービスを分割したいと考えています。
サービスのどの部分をプロセス層に配置する必要がありますか?
A. 管理システムからの保険契約情報を統合する
B. SOAPベースの応答をUIに表示する
C. 各バックエンド管理システムへの接続を認証および維持する
D. 管理システムからのデータのクエリ
Answer: A (メーッセージを残す)
API 主導の接続アプローチでは、各レイヤー (システム、プロセス、エクスペリエンス) には明確な目的があります。
* システム API: これらの API はバックエンド システムに直接接続し、標準化された方法でデータを公開およびロック解除します。
* プロセス API: さまざまなシステム間でデータをオーケストレーションおよび処理し、必要に応じて情報を組み合わせます。
* エクスペリエンス API: 特定のユーザー インターフェイスまたはアプリケーション向けに設計されており、多くの場合、各コンシューマー アプリケーションのニーズに合わせてデータ形式を変換します。
オプション A が正しい理由:

* プロセス API は、複数のシステムからのデータを組み合わせるように設計されており、生命保険システムと損害保険システムの両方からポリシー情報を集約する機能と一致しています。この集約ロジックは、理想的にはプロセス層に存在し、データ取得とデータ オーケストレーションを分離します。
* この機能をプロセス レイヤーに移動すると、再利用性とモジュール性が実現され、必要に応じて他のエクスペリエンス API やサービスでも結合されたポリシー データを活用できるようになります。
誤ったオプションの説明:
* オプション B (SOAP ベースの応答の提示) は、エクスペリエンス レイヤーによって管理されます。このレイヤーは、特定のインターフェイスに合わせてデータ形式を調整するためです。
* オプション C (バックエンド接続の認証と維持) は通常、バックエンドの統合とセキュリティ処理が行われるシステム層内で処理されます。
* オプション D (データのクエリ) は、バックエンド システムに直接アクセスし、追加の処理なしで生データを公開するシステム API の機能です。
参考資料 API 主導のアーキテクチャと各レイヤーの役割の詳細については、MuleSoft の API 主導の接続性と API レイヤーに関するドキュメントを参照してください。

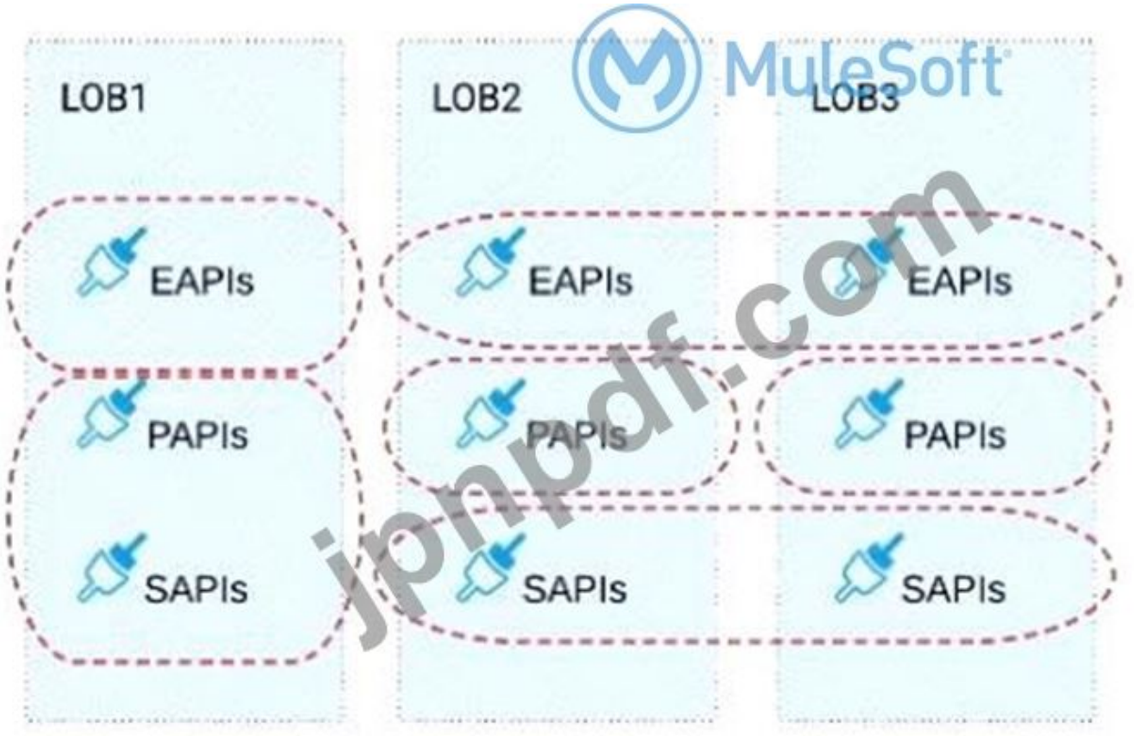
最新問題**: 119**
API クライアントは、既存の API 実装から 1 つのメソッドを呼び出します。API 実装は後で更新されます。API 実装にどのような変更を加えると、API クライアントの呼び出しロジックも更新する必要がありますか?
A. APIクライアントによって呼び出されたメソッドのレスポンスのデータ型が変更された場合
B. APIクライアントが使用するリソースに新しいメソッドが追加されたとき
C. APIクライアントによって呼び出されたメソッドに新しい必須フィールドが追加されたとき
D. APIクライアントによって呼び出されたメソッドに子メソッドが追加された場合
Answer: C (メーッセージを残す**)**
APIクライアントによって呼び出されたメソッドに新しい必須フィールドが追加された場合
****
>> 一般的に、API 契約が破綻した場合、API クライアントのロジックを更新する必要があります。
>> API に新しいメソッドまたは子メソッドが追加されても、API クライアントは既存のメソッドを引き続き使用できるため、動作が中断されることはありません。したがって、これら 2 つのオプションは無効です。
>> 残っているのは、 応答のデータ型が変更された場合」と 新しい必須フィールドが追加された場合」の 2 つです。
>> 応答のデータ型を変更すると、API 契約が破棄されます。ただし、質問は 呼び出し」ロジックに関するものであり、応答処理ロジックに関するものではありません。API クライアントは引き続き API を正常に呼び出して応答を受け取ることができますが、応答の一部のフィールドのデータ型は異なります。
>> 新しい必須フィールドを追加すると、API の呼び出し契約が破棄されます。新しい必須フィールドを追加すると、API 契約により、API クライアント/API コンシューマーと API プロバイダーの間で締結された RAML または API 仕様の合意が破棄されます。そのため、API クライアントの呼び出しロジックも更新する必要があります。

最新問題: **120**
CloudHub 専用ロードバランサーを使用する必要がある条件は何ですか?
A. API実装とAPIクライアント間でサーバー側負荷分散TLS相互認証が必要な場合
B. 同じ Mule アプリケーションの別々のデプロイメント間でリージョン間の負荷分散が必要な場合
C. 複数の CloudHub ワーカー間での API 呼び出しを負荷分散する必要がある場合
D. 顧客がホストする Mule ランタイムにデプロイされた API 実装にカスタム DNS 名が必要な場合
Answer: (解答を表示する**)**

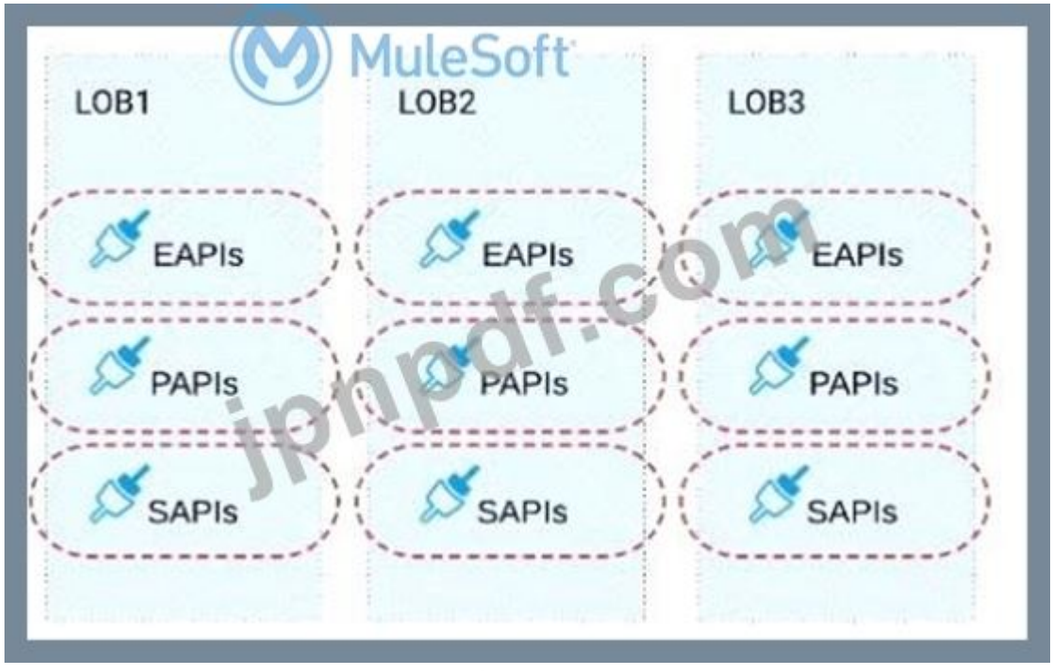
最新問題: **121**
展示品を参照してください。



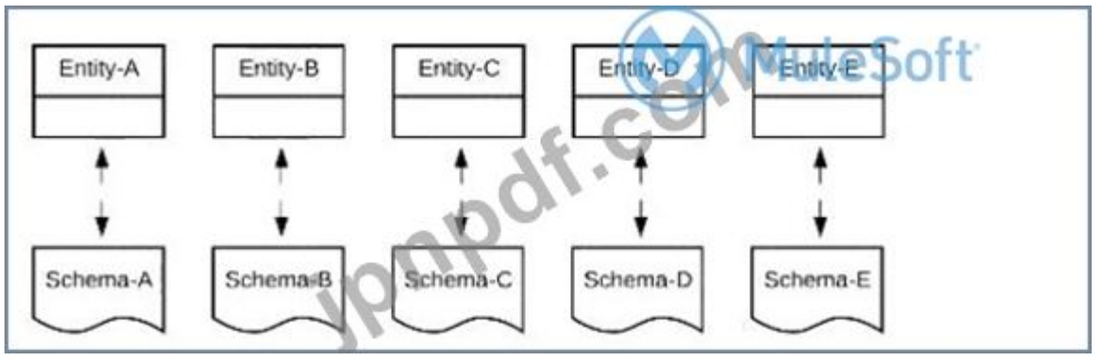
3 つのビジネス プロセスを実装する必要があり、実装では複数の異なる SaaS アプリケーションと通信する必要があります。
これらのプロセスは、個別の (サイロ化された) LOB によって所有され、主に互いに独立していますが、いくつかのビジネス エンティティを共有しています。各 LOB には 1 つの開発チームと独自の予算があります。
この組織のコンテキストでは、データ モデルの冗長性を最小限に抑えてこれらのビジネス プロセスを実装する API の API データ モデルを選択する最も効果的な方法は何ですか。
A) ビジネスプロセスの一貫した部分と関連するビジネスエンティティの定義に一致する複数の境界付きコンテキストデータモデルを構築する



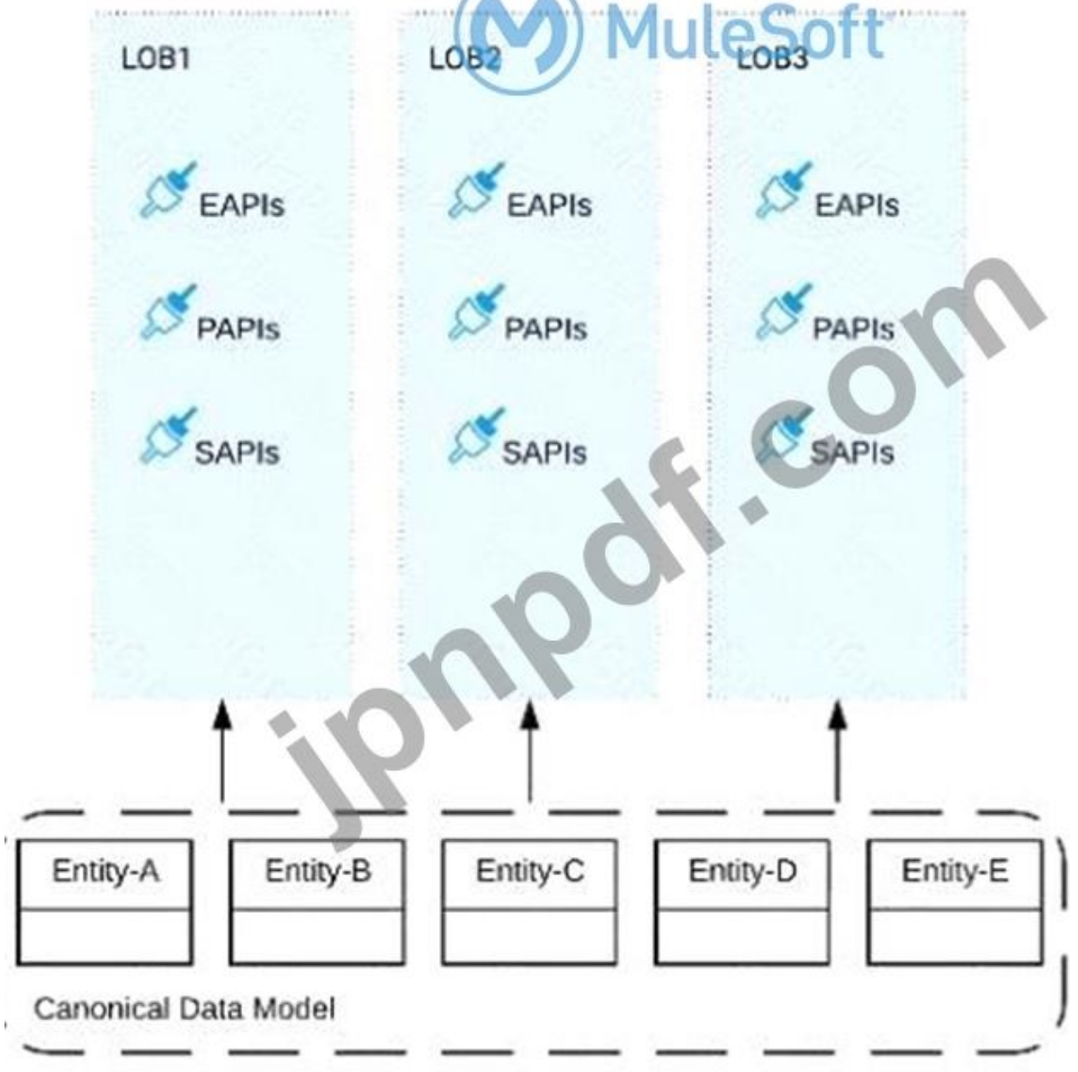
B)確立されたマイクロサービスとアジャイルAPI中心のプラクティスに従うために、各APIごとに異なるデータモデルを構築する



C) 組織全体で一貫性と再利用性を高めるために、XMLスキーマを使用してすべてのAPIデータモデルを構築する



D) 3 つのビジネス プロセスのすべてのデータ タイプを統合し、データ モデルの一貫性と冗長性を確保した 1 つの集中型標準データ モデル (エンタープライズ データ モデル) を構築します。



A. オプションA
B. オプションB
C. オプションC
D. オプションD
Answer: A (メーッセージを残す**)**

ビジネス プロセスの一貫した部分と関連するビジネス エンティティの定義に一致する複数の境界付きコンテキスト データ モデルを構築します。
****
>> XML スキーマ/アジャイル API 中心のプラクティスを使用して API データ モデルを構築するオプションは、質問で示されているシナリオとは無関係です。したがって、これら 2 つは無効です。
>> チームと LOB がサイロで作業し、それぞれが異なるイニシアチブ、予算などを持っているため、EDM (エンタープライズ データ モデル) の構築は実行可能ではなく、このシナリオには適していません。EDM の構築には、チーム全体の間での徹底的な調整が必要ですが、このシナリオでは明らかに不可能と思われます。
したがって、このシナリオに最適なのは、ビジネス プロセスの一貫した部分と関連するビジネス エンティティの定義に一致する複数の境界付きコンテキスト データ モデルを構築することです。