歡迎學習模組 2: 雲架構簡介。

模組概覽

章節

1. 什麼是雲架構？
2. Amazon Web Services (AWS) Well-Architected Framework
3. 在 AWS 上構建解決方案的最佳實踐
4. AWS 全球基礎設施

知識考核

本模組包括以下章節：

1. 什麼是雲架構？
2. Amazon Web Services (AWS) Well-Architected Framework
3. 在 AWS 上構建解決方案的最佳實踐
4. AWS 全球基礎設施

在本模組結束時，您需要完成一個知識考核，以測試您對本模組中涵蓋的關鍵概念的理解程度。

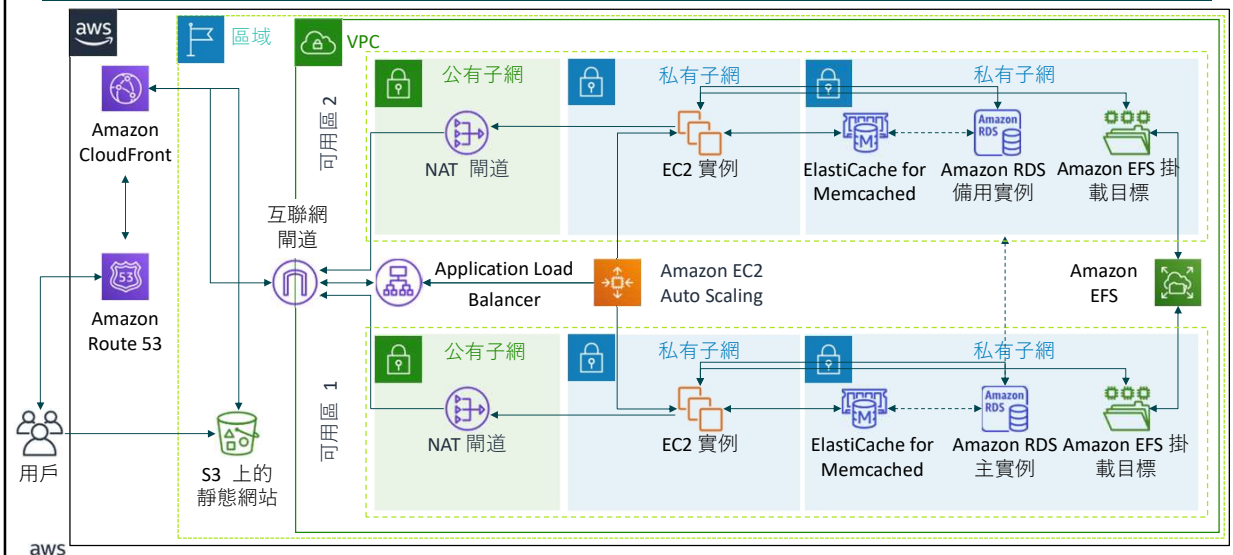
模組目標

學完本模組後，您應該能夠：

- 定義雲架構
- 描述如何使用 AWS Well-Architected Framework 來設計和評估架構
- 解釋在 AWS 上構建解決方案的最佳實踐
- 描述如何就 AWS 資源的放置位置作出明智的決策

學完本模組後，您應該能夠：
- 定義雲架構
- 描述如何使用 AWS Well-Architected Framework 來設計和評估架構
- 解釋在 AWS 上構建解決方案的最佳實踐
- 描述如何就 AWS 資源的放置位置作出明智的決策

一個大型架構



本課程結束時，您將瞭解此架構圖中的所有組件。您還能構建您自己的大型、穩健的解決方案架構，如同本示例中的一樣。在課程大部分模組的開頭，您都會重複看到本圖。圖中的新組件將隨著課程中的介紹揭示。

介紹第 1 節：什麼是雲架構？

架構需求

從 2000 年左右開始，Amazon 一直在努力讓自己的新購物網站具有高可用性和可擴展性。

要瞭解什麼是雲架構以及它為何重要，首先要舉例說明沒有雲架構時的軟體發展情況。

在 2000 年左右，Amazon 試圖創建一項電子商務服務，使協力廠商賣家能夠在 Amazon 電子商務引擎之上建立自己的線上購物網站。該公司一直在努力讓自己的新購物網站具有高可用性和可擴展性。

AWS 的起源

- 根據 AWS 首席執行官 Andy Jassy 的說法，當時 Amazon 電子商務工具是“一團糟”
  - 應用程式和架構的構建沒有經過正確的規劃
  - 很難將服務彼此分開
- 解決方案：Amazon 創建了一組記錄完備的 API，並作為公司的服務開發標準

在關於 AWS 起源的 TechCrunch 採訪中，AWS 首席執行官 (CEO) Andy Jassy 表示，一開始，Amazon 電子商務工具是“一團糟”。應用程式和架構的構建沒有經過正確的規劃。Jassy 還表示，“將各種服務分開，構建集中式開發平臺是一個巨大的挑戰。”

這個問題的解決方案是，創建一組記錄完備的 Application Programming Interface (API) 來組織開發環境。

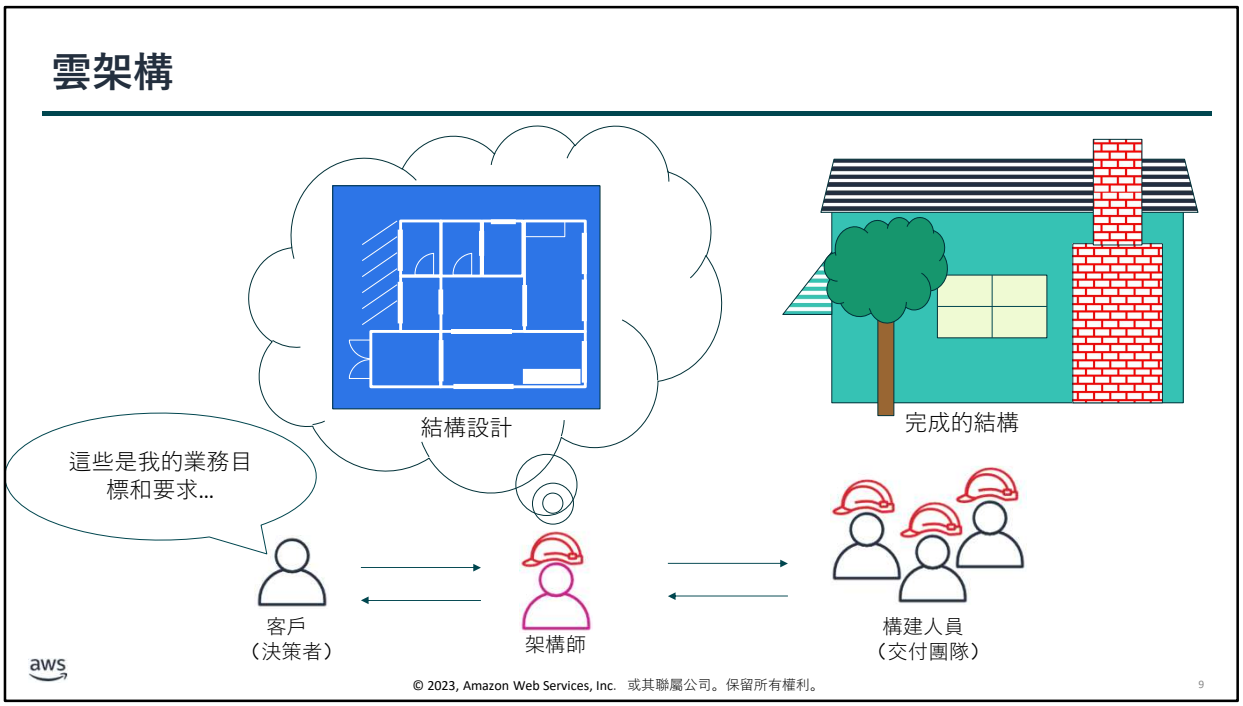
問題依然存在

- Amazon 仍然難以快速構建應用程式。
  - 資料庫、計算和存儲元件耗時 3 個月來構建。
  - 每個團隊構建自己的資源，沒有可擴展性或重用性方面的規劃。
- 解決方案：Amazon 構建了內部服務，以便在基礎設施上創建高度可用、可擴展並且可靠的架構。2006 年，Amazon 開始將這些服務作為 AWS 進行銷售。

但是，隨著公司的發展以及聘用了更多軟體工程師，Amazon 仍然難以快速構建應用程式：

- 本來整個專案預計需要 3 個月的時間，但是構建資料庫、計算和存儲元件卻花了 3 個月的時間。
- 每個團隊構建自己的資源，而沒有可擴展性或可重複使用性方面的規劃。

解決方案是構建內部服務，以便在 Amazon 基礎設施上創建高度可用、可擴展並且可靠的架構。2006 年，Amazon 開始將這些服務作為 AWS 進行銷售。



那麼，什麼是雲架構？雲架構是將雲特性應用於一個解決方案，使解決方案可以利用雲服務和功能來滿足企業技術需求和業務使用案例。解決方案類似於建築物的藍圖。

軟體系統需要架構師來管理其規模和複雜性。

雲架構師：
- 與決策者交流，以確定業務目標和需要改進的能力。
- 確保解決方案的技術交付成果與業務目標保持一致。
- 與實施解決方案的交付團隊合作，以確保技術功能適用。

擁有架構完善的系統會增加技術交付成果有助於實現業務目標的可能性。

第 1 節要點

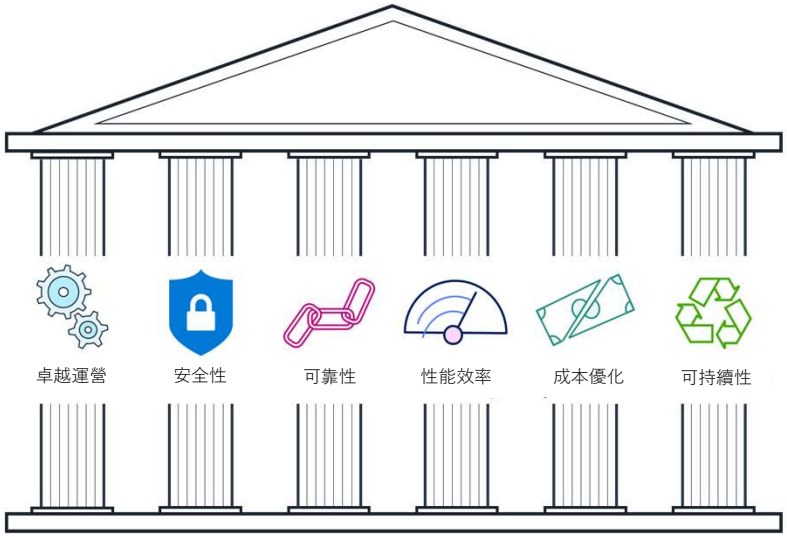
- 雲架構是將雲特性應用於一個解決方案，使解決方案可以利用雲服務和功能來滿足企業技術需求和業務使用案例
- 您可以使用 AWS 服務創建高度可用、可擴展且可靠的架構

本模組中這節內容的要點包括：

- 雲架構是將雲特性應用於一個解決方案，使解決方案可以利用雲服務和功能來滿足企業技術需求和業務使用案例
- 您可以使用 AWS 服務創建高度可用、可擴展且可靠的架構

介紹第 2 節：AWS Well-Architected Framework。

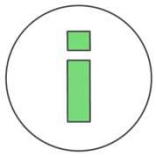
AWS Well-Architected Framework 的支柱



AWS Well-Architected Framework 是一份指南，提供了評估雲架構的一致方法和有助於實施設計方案的指導。它記錄了一系列基本問題和最佳實踐，可讓您瞭解某個特定架構是否高度符合雲最佳實踐。在審查了 AWS 上的數千個客戶架構後，AWS 開發了此框架。

AWS Well-Architected Framework 分為六大支柱：卓越運營、安全性、可靠性、性能效率、成本優化和可持續性。自 2015 年該框架推出以來，前五大支柱一直是該框架的組成部分。可持續性支柱是 2021 年新增的第六個支柱，旨在幫助企業瞭解如何最大限度地減少運行雲工作負載對環境的影響。

安全性支柱



身份機制

可跟蹤性

在所有層確保安全性

風險評估與緩解策略

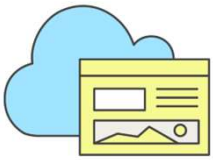
安全性支柱能夠保護資訊、系統和資產，同時通過風險評估和緩解策略實現商業價值。

只需採取一些措施，您的架構就會處於更好的安全狀態。這些措施包括採用強大的身份機制、實現可跟蹤性、在所有層確保安全性、自動應用安全性最佳實踐以及保護傳輸中的資料和靜態資料。

有關安全性最佳實踐的更多資訊，請參閱安全性支柱白皮書。

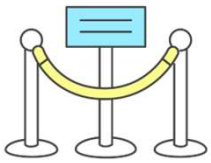
卓越運營支柱

- 能夠運行和監控各種系統
- 不斷改善支援流程和程式



已部署

已更新



已運行

卓越運營支柱能夠實現運行系統和深入瞭解其運營以實現商業價值。它還能實現持續改進支援流程和程式。

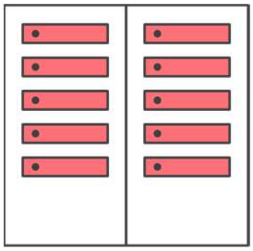
在設計運營工作負載時，您必須瞭解其如何部署、更新和運作。實施符合減少缺陷和快速安全修復的工程實踐。通過日誌記錄、儀器以及業務和技術指標啟用觀察，以便您可以深入瞭解架構內發生的情況。

在 AWS 中，您可以將整個工作負載（應用程式、基礎設施、策略、監管和操作）視為代碼。工作負載可以在代碼中定義，並使用代碼來更新。這意味著您可以將用於應用程式碼的設計規範應用到堆疊中的每個元素。

有關卓越運營最佳實踐的更多資訊，請參閱卓越運營支柱白皮書。

可靠性支柱

- 迅速從基礎設施或服務故障中恢復
- 動態獲取計算資源以滿足需求
- 減少中斷，例如：
  - 配置錯誤
  - 暫時性網路問題



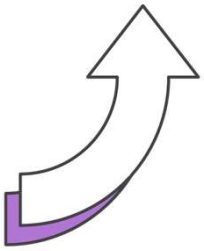
可靠性支柱能夠實現系統從基礎設施或服務中斷中恢復，以及動態獲取計算資源以滿足需求。它還能夠助力系統減少中斷（例如配置錯誤或暫時性網路問題）。

在傳統環境中確保可靠性可能會很困難。單點故障、缺乏自動化和缺乏彈性都會引起問題。通過應用可靠性支柱中概述的最佳實踐，您可以避免許多此類問題。在高可用性、容錯能力和整體冗餘方面正確設計架構對您和您的客戶都會有所説明。

有關可靠性最佳實踐的更多資訊，請參閱可靠性支柱白皮書。

性能效率支柱

- 選擇有效的資源並在需求變化時保持資源效率
- 普及先進技術
- 應用機械同感



在考慮性能時，您需要有效使用計算資源，從而最大限度地提高性能。隨著需求的變化，您還希望保持這種效率。

普及先進技術也同樣重要。在自己難以實施技術的情況下，請考慮使用供應商。通過為您實施技術，供應商利用所掌握的知識來處理複雜問題，讓您的團隊專注於附加價值更高的工作。

*機械同感*是在使用工具或系統時，瞭解其最佳運行方式。使用與您要實現的目標最為一致的技術方法。例如，在選擇資料庫或存儲方法時考慮資料訪問模式。

有關性能最佳實踐的更多資訊，請參閱性能效率支柱白皮書。

成本優化支柱

- 衡量效率
- 消除不必要的支出
- 考慮使用託管服務

成本優化是所有良好架構設計的長期要求。這一過程重複進行，應該在您的整個生產生命週期內完善和改進。瞭解當前的架構相對於目標的效率，可以消除不必要的費用。考慮使用託管服務，因為它們在雲規模下運行，因此可以降低交易處理成本或服務成本。

要瞭解更多資訊，請參閱成本優化支柱白皮書。

可持續性支柱

- 瞭解影響
- 制定可持續發展目標
- 最大限度地提高利用率
- 預測並採用更高效的新型軟硬體產品
- 降低雲工作負載對下游的影響

可持續性支柱涉及構建能最大限度提高效率和減少浪費的架構的能力。

雲中的可持續性是一項持續的工作，主要集中在工作負載所有元件的節能和增效上，方法是從所預置的資源中獲得最大效益，並最大限度地減少所需的資源總量。這項工作包括最初選擇高效的程式設計語言、採用現代演算法、使用高效的資料存儲技術、部署正確規模和高效的計算基礎設施，以及儘量減少對高功率終端使用者硬體的要求。

有關雲中可持續性設計原則的更多資訊，請參閱*可持續性支柱：AWS Well-Architected Framework* 白皮書，網址為 https://docs.aws.amazon.com/wellarchitected/latest/sustainability-pillar/sustainability-pillar.html。

AWS Well-Architected Tool

- 有助於您查看工作負載的狀態，並將其與最新的 AWS 架構最佳實踐進行對比
- 您可以根據需要，隨時訪問 AWS 架構師使用的知識和最佳實踐
- 提供行動計畫，並逐步指導如何為雲構建更好的工作負載
- 提供一致的流程，供您查看和測評自己的雲架構

如果您希望在設計架構完善的解決方案方面獲得幫助，可以使用 AWS Well-Architected Tool。AWS Well-Architected Tool 是一種自助式工具，借助此工具，您可以根據需要隨時訪問當前的 AWS 最佳實踐。這些最佳實踐旨在幫助您在 AWS 上構建安全、高性能、具有彈性和高效的應用程式基礎設施。

AWS Well-Architected Tool 有助於您查看工作負載的狀態，並將其與最新的 AWS 架構最佳實踐進行對比。它使您可以根據需要隨時訪問 AWS 架構師使用的知識和最佳實踐。

此工具在 AWS 管理主控台中提供。您可以定義工作負載，並回答卓越運營、安全性、可靠性、性能效率和成本優化領域中的一系列問題。之後，AWS Well-Architected Tool 會提供一個行動計畫，此計畫逐步指導您針對雲環境改進工作負載。

AWS Well-Architected Tool 提供一致的流程，供您查看和測評自己的雲架構。您可以使用該工具提供的結果來確定改進的後續步驟、推動架構決策，並將架構注意事項納入公司監管流程。

要瞭解有關 AWS Well-Architected Tool 的更多資訊，請參閱 AWS Well-Architected Tool 網站。

第 2 節要點

- AWS Well-Architected Framework 提供了一致的方法來評估雲架構，還提供了有助於實現設計方案的指南
- AWS Well-Architected Framework 分為六個支柱
- 每個支柱記錄了一系列基本問題，可讓您瞭解某個特定架構是否高度符合雲最佳實踐
- AWS Well-Architected Tool 有助於您查看工作負載的狀態，並將其與最新的 AWS 架構最佳實踐進行對比

本模組中這節內容的要點包括：

- AWS Well-Architected Framework 提供了一致的方法來評估雲架構，還提供了有助於實現設計方案的指南。
- AWS Well-Architected Framework 分為六個支柱。
- 每個支柱記錄了一系列基本問題，可讓您瞭解某個特定架構是否高度符合雲最佳實踐。
- AWS Well-Architected Tool 有助於您查看工作負載的狀態，並將其與最新的 AWS 架構最佳實踐進行對比。

介紹第 3 節：在 AWS 上構建解決方案的最佳實踐。

設計權衡

- 評估權衡，以便您可以選擇最佳方法
- 權衡的示例包括：
  - 在一致性、持久性和空間方面進行妥協以減少用時和延遲，從而提供更高的性能
  - 優先考慮新功能的上市速度而不是成本
- 基於經驗資料作出設計決策

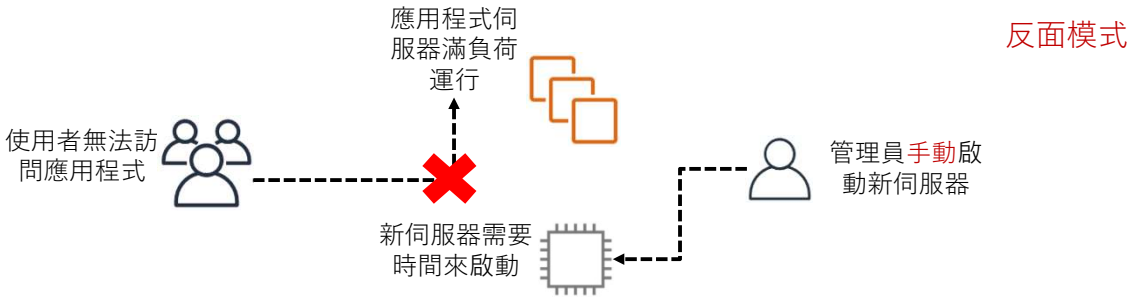
在設計解決方案時，請仔細考慮權衡，以便選擇最佳方法。例如，您可以在一致性、持久性和空間方面進行妥協以減少用時和延遲，從而提供更高的性能。或者，您可能會優先考慮上市速度而不是成本。

權衡可能會增加架構的成本和複雜性，因此您的設計決策應該基於經驗資料。例如，您可能需要執行負載測試以確保在性能方面獲得可衡量的優勢。或者，您可能需要執行基準測試，以實現隨著時間推移最具成本優勢的工作負載。在評估與性能相關的改進時，您還需要考慮架構設計選擇將如何影響客戶和工作負載效率。

在本節中，您將瞭解在 AWS 上設計解決方案的最佳實踐。您還將瞭解應避免的反面模式（或錯誤的解決方案設計）。

1. 啟用可擴展性 (1/2)

確保您的架構能夠應對需求的變化。



反面模式

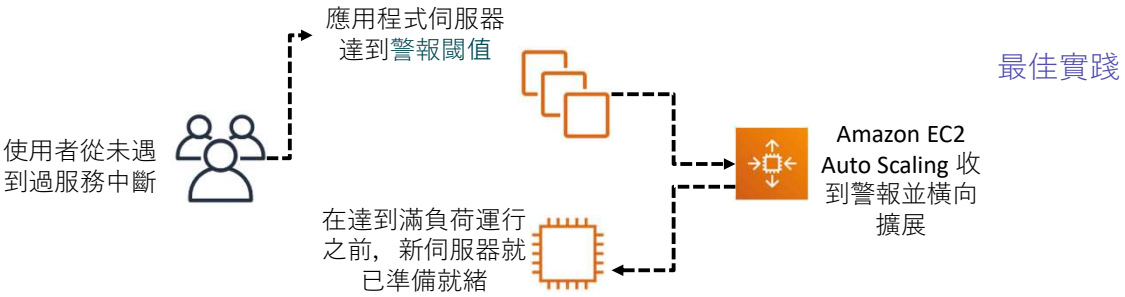
在 AWS 雲上運行工作負載時，您可以快速、主動地擴展基礎設施。確保在基礎設施的每個層實施可擴展性。

要瞭解可擴展性的重要性，請考慮一下這種反面模式，即以被動和手動的方式進行擴展。

在這種情況下，當應用程式伺服器滿負荷運行時，使用者將無法訪問應用程式。然後，管理員手動啟動一個或多個新實例來管理負載。遺憾的是，實例啟動後需要幾分鐘的時間才能使用。因此，使用者無法訪問應用程式的時間增加。

1. 啟用可擴展性 (2/2)

確保您的架構能夠應對需求的變化。



最佳實踐

通過啟用可擴展性，您可以改進設計，以預測對更多容量的需求，並提前提供這些容量。

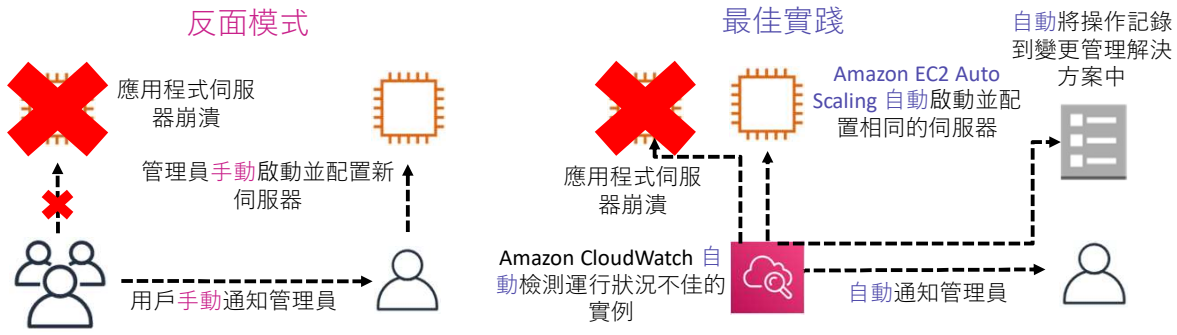
例如，您可以使用監控解決方案（如 Amazon CloudWatch）來檢測伺服器佇列的總負載是否達到指定的閾值。您可以將該閾值定義為 *CPU 利用率持續高於 60% 超過 5 分鐘*，或者與資源使用有關的任何內容。借助 CloudWatch，您還可以根據特定應用程式來設計自訂指標，從而觸發所需的資源擴展。

觸發警報後，Amazon EC2 Auto Scaling 會立即啟動新實例。在達到容量之前，該實例便已準備就緒，從而為用戶提供了無縫體驗。

理想情況下，您還應該將此系統設計為能夠在需求下降時*縮減容量*，這樣您就不會運行（和支付）不再需要的實例。

2. 讓您的環境實現自動化

在可能的情況下，實現資源預置、終止和配置的自動化。



AWS 幾乎在基礎設施的每一層都提供內置的監控和自動化工具。利用這些工具，確保您的基礎設施能夠快速應對變化。

您可以使用諸如 CloudWatch 和 Amazon EC2 Auto Scaling 等工具來檢測運行狀況不佳的資源，並自動啟動替換資源。當資源配置發生變化時，您還可以收到通知。

3. 將資源視為一次性資源

利用雲計算的動態預置特性。

反面模式

- 隨著時間的推移，不同的伺服器最終具有不同的配置
- 資源在不需要時依然運行
- 硬編碼的 IP 位址會妨礙靈活性
- 在正在使用的硬體上測試新的更新可能很困難或不方便

最佳實踐

- 自動部署具有相同配置的新資源
- 終止未使用的資源
- 自動切換到新的 IP 位址
- 測試新資源的更新，然後用更新的資源替換舊資源

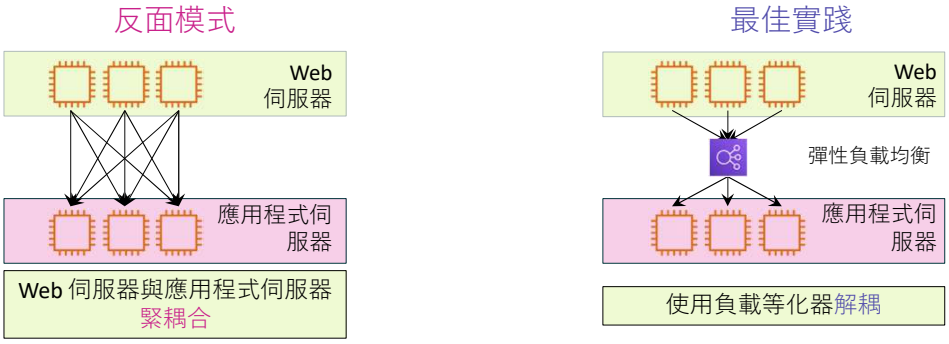
將資源視為一次性資源的最佳做法是指將基礎設施視為軟體而非硬體。

對於硬體，很容易購買比所需數量更多的特定元件，以便為使用高峰做好準備。這樣做既昂貴又缺乏靈活性 – 由於沉沒成本，升級難度更大。

相反，當您將資源視為一次性資源時，在實例或其他離散資源之間進行遷移就相當簡單。您可以快速應對容量需求的變化、升級應用程式並管理底層軟體。

4. 使用松耦合的元件

使用獨立的元件設計架構。



傳統的基礎設施由緊密集成的伺服器鏈組成，每個伺服器都有特定的用途。問題是，一旦其中一個元件或層出現故障，對系統造成的破壞可能是致命的。還會妨礙擴展。如果在一層添加或刪除伺服器，還必須連接每個連接層上的每個伺服器。

左側的示例顯示了緊耦合的 Web 伺服器和應用程式伺服器的集合。如果一台應用程式伺服器宕機，Web 伺服器在嘗試連接它時就會出錯。

通過松耦合，您可以使用託管解決方案作為系統各層之間的中介軟體。通過這種設計，中介軟體可以自動處理故障以及元件或層的擴展。

右側的示例顯示了在 Web 伺服器和應用伺服器之間路由請求的負載等化器（本例中為彈性負載均衡負載等化器）。如果一台應用程式伺服器宕機，負載等化器會自動開始將所有流量定向到兩台運行正常的伺服器。

用於解耦組件的兩個主要解決方案是**負載均衡器**和**訊息佇列**。

5. 設計服務，而不是伺服器

使用各種 AWS 服務。不要將基礎設施局限於伺服器。

反面模式

- 簡單的應用程式在持久性伺服器上運行
- 應用程式之間直接通信
- 靜態 Web 資產本機存放區在實例上
- 後端伺服器處理使用者身份驗證和使用者狀態存儲

最佳實踐

- 適當時，考慮使用容器或無伺服器解決方案
- 消息佇列處理應用程式之間的通信
- 靜態 Web 資產存儲在外部，例如存儲在 Amazon Simple Storage Service (Amazon S3) 上
- 使用者身份驗證和使用者狀態存儲均由託管的 AWS 服務進行處理

下一個最佳實踐是設計服務，而不是伺服器。雖然 Amazon Elastic Compute Cloud (Amazon EC2) 為設計和設置解決方案提供了極大的靈活性，但它並非始終是滿足每種需求的優先（或唯一的）解決方案。在某些情況下，容器或無伺服器解決方案可能更合適。因此，重要的是要考慮您的需求是什麼，以及哪種解決方案是合適的。

借助 AWS 無伺服器解決方案和託管服務，您無需預置、配置和管理整個 Amazon EC2 實例。

託管解決方案具有更低設定檔和更高性能，可以更低成本替代基於伺服器的解決方案。示例包括 AWS Lambda、Amazon Simple Queue Service (Amazon SQS)、Amazon DynamoDB、彈性負載均衡、Amazon Simple Email Service (Amazon SES) 和 Amazon Cognito。

6. 選擇正確的資料庫解決方案

將技術與工作負載相匹配，
而不是反之。

要考慮的事項：

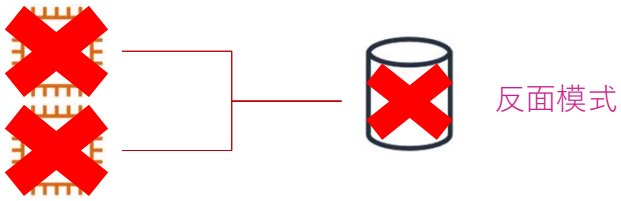
- 讀取和寫入需求
- 總存儲要求
- 典型的物件大小和這些物件的訪問性質
- 持久性要求
- 延遲要求
- 支持的最大併發用戶數
- 查詢性質
- 完整性控制所需的強度

請務必選擇正確的資料庫解決方案。在傳統的資料中心和本地部署環境中，可用硬體和許可證的限制會制約您對資料存儲解決方案的選擇。AWS 建議您根據對應用程式環境的需求選擇資料存儲。

7. 避免單點故障 (1/2)

假設一切都可能失敗。
然後，逆向設計。

盡可能地使用冗餘，以防止單點故障導致整個系統崩潰。



應用程式伺服器

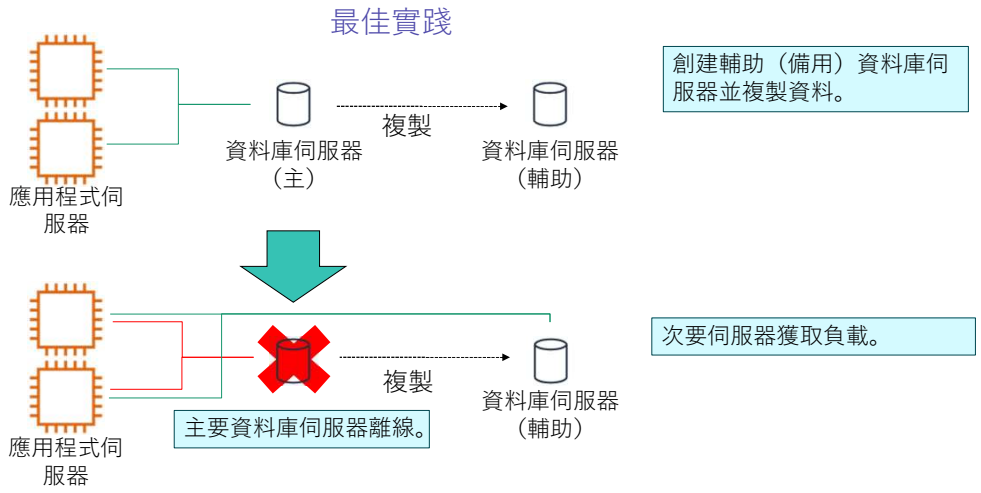
資料庫伺服器

盡可能消除架構中的單點故障。這並不意味著您必須總是複製每個元件。根據停機時間服務級別協定 (SLA)，您可以使用僅在需要時啟動元件的自動化解決方案。您還可以使用託管服務，在此服務中，AWS 會自動為您替換出現故障的底層硬體。

這個簡單的系統顯示了連接到單個資料庫伺服器的兩個應用程式伺服器。資料庫伺服器是單點故障，應避免使用。當它宕機時，應用伺服器也會宕機。

即使底層物理硬體發生故障、被移除或更換，應用程式伺服器也應繼續正常運行。

7. 避免單點故障 (2/2)



避免單點故障的常見方法是，創建輔助（備用）資料庫伺服器並複製資料。這樣，如果主要資料庫伺服器離線，次要伺服器就能承擔起負載。

在此示例中，當主要資料庫離線時，應用程式伺服器會自動將請求發送到備用資料庫。此示例也體現了最佳實踐 3：將資源視為一次性資源，並設計應用程式以支援硬體變化。

8. 優化成本

利用 AWS 的靈活性來提高成本效益。

要考慮的事項：

- 我的資源的規模和類型是否適合任務需要？
- 應該監控哪些指標？
- 如何確保關閉未使用的資源？
- 我需要多久使用一次該資源？
- 是否可以用託管服務替代任何伺服器？

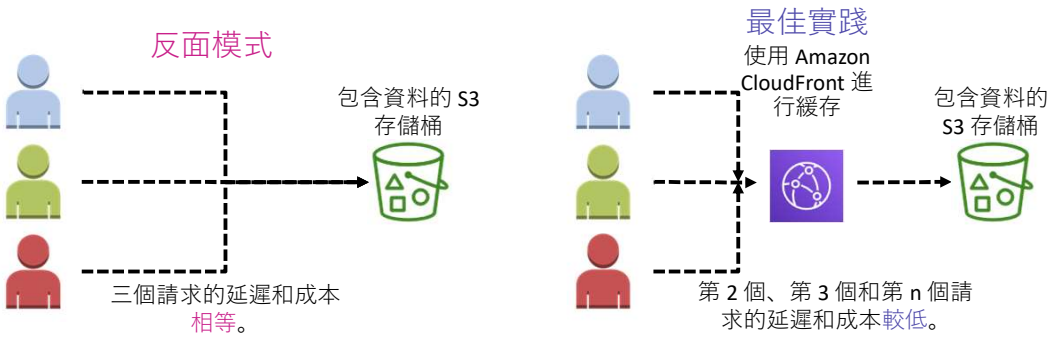
雲計算讓您可以用資本支出換取可變支出。*資本支出 (capex)* 是公司用於購置、升級和維護不動產、工業建築或設備等實體資產的資金。在這種模式下，無論資料中心的伺服器是否處於運行狀態，您都要為其付費。

相比之下，AWS 服務採用的是*可變支出*成本模式，這意味著您只需根據服務的使用量，為所需的單項服務付費。在每項服務中，您都可以優化成本。許多服務提供不同的定價套餐、模型或配置。

請記住，要在雲中複製本地部署中伺服器全天候運行的資料中心設置，費用可能會非常高。因此，從成本角度來看，構建基礎設施的最佳方法是僅預置所需的資源，並在不使用時停止服務。

9. 使用緩存

緩存最大限度地減少了冗餘數據檢索操作，優化了性能和成本。



*緩存*是一種技術，通過在請求者和永久存儲之間的中間位置臨時存儲資料，使未來的請求更快，並減少網路輸送量。

在反面模式示例中，沒有使用緩存服務。當任何人從 Amazon Simple Storage Service (Amazon S3) 的一個存儲桶中請求檔時，每次請求都需要相同的時間來完成，每次請求的成本也相同。

在最佳實踐模式示例中，基礎設施在 Amazon S3 之前使用 Amazon CloudFront 提供緩存。在這種情況下，初始請求會檢查 Amazon CloudFront 中的文件。如果找不到，CloudFront 會從 Amazon S3 請求檔。然後，CloudFront 將檔副本存儲在靠近使用者的邊緣網站，並將副本發送給提出請求的用戶。對檔的後續請求將從 CloudFront 中的（現在更接近的）邊緣網站而不是 Amazon S3 中檢索。

這樣可以減少延遲*和*成本，因為在第一次請求之後，無需再為從 Amazon S3 傳輸的文件付費。

10. 保護整個基礎設施

在基礎設施的每一層構建安全性。

要考慮的事項：

- 隔離基礎設施的各個部分
- 對傳輸中的資料和靜態資料進行加密
- 使用最低許可權原則精細地實施存取控制
- 使用多重身份驗證 (MFA)
- 使用託管服務
- 記錄資源的訪問情況
- 自動執行部署以保持一致的安全性

安全性不僅僅是通過基礎設施的外部邊界。這還包括確保各個環境及其元件相互安全。

例如，在 Amazon EC2 中，您可以創建安全性群組來確定實例上哪些埠可以發送和接收流量。安全性群組還可以確定流量的來源或去向。

您可以使用安全性群組來降低一個實例上的安全威脅擴散到環境中其他實例的可能性。您應該對其他服務採取類似的預防措施。整個課程都會討論實施這一最佳實踐的具體方法。

第 3 節要點

- 在設計解決方案時，評估權衡並根據經驗資料作出決策
- 在 AWS 上構建解決方案時遵循這些最佳實踐 –
  - 啟用可擴展性
  - 讓您的環境實現自動化
  - 將資源視為一次性資源
  - 使用松耦合的組件
  - 設計服務，而不是伺服器
  - 選擇正確的資料庫解決方案
  - 避免單點故障
  - 優化成本
  - 使用緩存
  - 保護整個基礎設施

本模組中這節內容的要點包括：

- 在設計解決方案時，評估權衡並根據經驗資料作出決策
- 在 AWS 上構建解決方案時遵循這些最佳實踐 –
  - 啟用可擴展性
  - 讓您的環境實現自動化
  - 將資源視為一次性資源
  - 使用松耦合的組件
  - 設計服務，而不是伺服器
  - 選擇正確的資料庫解決方案
  - 避免單點故障
  - 優化成本
  - 使用緩存
  - 保護整個基礎設施

介紹第 4 節：AWS 全球基礎設施。

AWS 區域

- AWS 區域是一個地理區域
- 每個 AWS 區域都由兩個或更多可用區組成
- 區域之間的通信使用 AWS 骨幹網路基礎設施
- 您可以實現和控制跨區域數據複制

示例：倫敦區域

AWS 雲基礎設施是圍繞區域構建的。AWS 在全球有 22 個區域。*AWS 區域*是一個實際的地理位置，擁有兩個或多個*可用區*。可用區又由一個或多個*數據中心*組成。

AWS 區域連接到多個網路服務提供商 (ISP)。區域還與一個專用全球骨幹網連接。與公共互聯網相比，專用全球骨幹網的費用更低，跨區域網路延遲更穩定。

2019 年 3 月 20 日之前推出的 AWS 區域預設*處於啟用狀態*。2019 年 3 月 20 日之後推出的區域（例如，亞太地區（香港）和中東（巴林））默認*處於禁用狀態*。您必須先啟用這些區域，然後才能使用它們。您可以使用 AWS 管理主控台來啟用或禁用區域。

某些區域的訪問受限。通過 AWS（**中國**）帳戶只能訪問北京區域和寧夏區域。要瞭解有關 AWS 中國的更多資訊，請參閱“AWS 中國”頁面。隔離的 **AWS GovCloud (US)** 區域專門面向美國政府機構和客戶，方便他們將敏感工作負載移至雲中，從而滿足其特定的法規和合規性要求。

為實現容錯能力和穩定性，區域之間相互隔離。一個區域中的資源不會自動複製到其他區域。在特定區域存儲資料時，資料不會複製到該區域之外。如果您的業務需要，您應自行負責在多個區域間複製資料。AWS 提供有關每個區域所在的國家/地區和省/市/自治區的資訊（如適用）。您有責任根據合規性和網路延遲要求，選擇存儲資料的區域。

AWS 產品和服務按區域提供，因此各個區域提供的服務可能不盡相同。有關按區域提供的 AWS 服務的清單，請參閱區域表。

有關 AWS 全球雲基礎設施的更多資訊，請參閱全球基礎設施網站。有關 AWS 全球基礎設施的最新互動式地圖，請參閱互動式 AWS 全球基礎設施地圖。

AWS 可用區

- 每個可用區都具有以下特點 –
  - 由一個或多個數據中心組成
  - 專為故障隔離而設計
  - 使用高速專用鏈接與區域中的其他可用區互連
- 對於某些服務，您可以選擇自己的可用區
- AWS 建議跨可用區進行複製，以獲得彈性

每個 AWS 區域都有兩個或多個相互隔離的位置，稱為*可用區*。每個可用區都包含一個或多個資料中心，某些可用區擁有多達六個資料中心。但是，每個資料中心只能屬於一個可用區。

每個可用區都被設計為獨立的故障區。這意味著可用區在典型的大都市區域內是物理隔絕的，並且位於風險較低的洪泛平原上（具體的洪泛區分類因區域而異）。除了具有分立的不斷電供應系統和現場備用發電設施外，它們還分別通過獨立公用事業公司的不同電網供電，以便進一步減少單點故障。可用區全部以冗餘的方式連接至多家第 1 層傳輸提供商。

可用區是您可以為某些服務（例如 Amazon EC2）制定的最精細級別的規範。

您負責選擇自己的系統所在的可用區。系統可以跨越多個可用區。您的系統應設計為，在災難發生時，能夠承受暫時或長期的可用區故障。將應用程式分佈在多個可用區內，可使應用程式在大多數故障情況下（包括自然災難或系統故障）保持彈性。

AWS Local Zones

- 使您能夠在距離終端使用者更近的特定地理位置，運行應用程式中對延遲敏感的部分和資源。
- 是 AWS 區域的擴展，您可以在靠近終端使用者的地理位置上使用 AWS 服務
- 讓您可以將 AWS 計算、存儲、資料庫和其他特定服務放在更靠近目前不存在 AWS 區域的大量人口聚居的位置，或者靠近行業和 IT 中心的位置
- 由 AWS 管理和支援
- 洛杉磯 (LA) AWS Local Zone 可應邀使用

AWS Local Zones 是 AWS 基礎設施部署的一種新形式，可將 AWS 計算、存儲、資料庫和其他特定服務放在更靠近目前不存在 AWS 區域的大量人口聚居的位置，或者靠近行業和 IT 中心的位置。借助 AWS Local Zones，您能夠在距離終端使用者更近的特定地理位置，運行應用程式中對延遲敏感的部分和資源。您可以使用 AWS Local Zones 為媒體和娛樂內容創建、即時遊戲、油藏模擬、Electronic Design Automation 和機器學習等使用案例提供個位數毫秒的延遲。

每個 AWS Local Zone 位置都是 AWS 區域的擴展。您可以在 AWS Local Zone 中運行對延遲敏感的應用程式，方法是在終端使用者附近使用 AWS 服務，如 Amazon EC2、Amazon Virtual Private Cloud (Amazon VPC)、Amazon Elastic Block Store (Amazon EBS)、Amazon FSx 和彈性負載均衡。AWS Local Zones 在本地工作負載和在 AWS 區域運行的工作負載之間提供高頻寬、安全的連接。因此，AWS Local Zones 使您能夠無縫連接回 AWS 中運行的其他工作負載，並通過相同的 API 和工具集連接到區域內的所有服務。

AWS Local Zones 由 AWS 管理和支援，並為您提供雲的所有彈性、可擴展性和安全性優勢。借助 AWS Local Zones，您可以使用一套一致的 AWS 服務，在更靠近終端使用者的地方構建和部署對延遲敏感的應用程式。您還可以擴大或縮小規模，而且只需為使用的資源付費。

目前，洛杉磯 AWS Local Zone 一般通過邀請即可使用，您可以期待更多 Local Zone 的出現。

要瞭解有關 AWS Local Zones 的更多資訊，請參閱 AWS Local Zones 頁面。

AWS 資料中心

- 資料中心是存放和處理資料的位置
- 一個資料中心通常包含上萬台伺服器
- 所有資料中心都是線上的，並為客戶服務
- AWS 自訂網路設備 –
  - 來自多個 ODM
  - 擁有自訂網路通訊協定堆疊

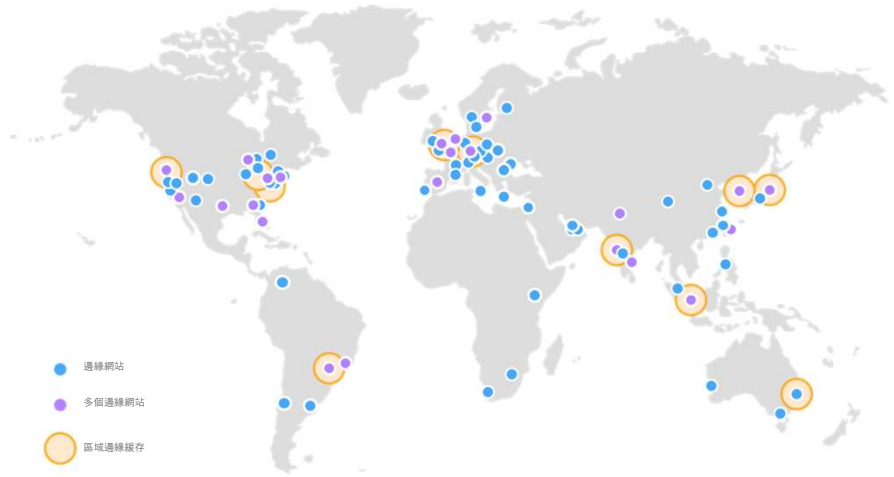
AWS 基礎設施的基礎是資料中心。您不指定用於部署資源的資料中心。數據中心則是實際資料的存放位置。Amazon 運行著具有高可用性的先進資料中心。儘管資料中心有時會發生一些故障，影響同一網站內實例的可用性，但是這種故障極少發生。如果您將所有實例都託管在受此類故障影響的同一個網站，一旦發生故障，您的所有實例都將不可用。

所有資料中心都是線上的，並為客戶服務。如果出現故障，自動進程會將客戶資料流程量從受影響的區域移除。核心應用程式以 N+1 的配置進行部署，故在資料中心出現故障時，有足夠的容量使流量能夠均衡載入至其餘的網站。

AWS 使用的自訂網路設備來自多個原始設備製造商 (ODM)。ODM 根據另外一家公司提供的規範來設計和製造產品。然後，這家公司在更換產品的品牌後進行銷售。

有關 AWS 資料中心的更多資訊，請參閱瞭解我們如何通過設計來保護 AWS 資料中心。

AWS 接入點



為了以較低的延遲向終端使用者交付內容，Amazon CloudFront 使用了包含 200 多個接入點的全球網路，這些接入點由邊緣網站和區域邊緣緩存組成。

邊緣網站位於北美、歐洲、亞洲、澳大利亞、南美洲、中東、非洲和中國。邊緣網站支持 Amazon Route 53 和 Amazon CloudFront 等 AWS 服務。

預設情況下，區域邊緣緩存與 Amazon CloudFront 一起使用。當您的內容訪問頻率不夠高，無法保留在邊緣網站時，可以使用區域邊緣緩存。區域邊緣緩存會保留這些內容，在使用者必須從原始伺服器獲取內容時提供一份替代方案。

有關 Amazon CloudFront 基礎設施的更多資訊，請參閱 **Amazon CloudFront 基礎設施**，網址為 https://aws.amazon.com/cloudfront/features/?whats-new-cloudfront.sort-by=item.additionalFields.postDateTime&whats-new-cloudfront.sort-order=desc#Amazon_CloudFront_Infrastructure

第 4 節要點

- AWS 全球基礎設施由區域、可用區和邊緣網站組成
- 您通常按照合規性要求或以減少延遲為原則來確定自己的首選區域
- 每個可用區都在物理上與其他可用區分開並具有冗餘電源、網路和連接
- 邊緣站點和區域邊緣緩存通過將內容緩存到離使用者更近的位置來提高性能

本模組中這節內容的要點包括：

- AWS 全球基礎設施由區域和可用區組成
- 您通常按照合規性要求或以減少延遲為原則來確定自己的首選區域
- 每個可用區都在物理上與其他可用區分開，並具有冗餘電源、網路和連接
- 邊緣站點和區域邊緣緩存通過將內容緩存到離使用者更近的位置來提高性能

Course capstone project

The capstone project provides you with the opportunity to apply the skills and knowledge you develop in the course to a real-world scenario. Details about the project are included in the Bridging to certification module.

現在，我們來回顧和總結一下本模組，然後進行知識考核。

模組總結

總的來說，在本模組中，您學習了如何：

- 定義雲架構
- 描述如何使用 AWS Well-Architected Framework 來設計和評估架構
- 解釋在 AWS 上構建解決方案的最佳實踐
- 描述如何就 AWS 資源的放置位置作出明智的決策

總的來說，在本模組中，您學習了如何：
- 定義雲架構
- 描述如何使用 AWS Well-Architected Framework 來設計和評估架構
- 解釋在 AWS 上構建解決方案的最佳實踐
- 描述如何就 AWS 資源的放置位置作出明智的決策

完成知識考核

現在該完成本模組的知識考核了。

其他資源

- AWS 全球基礎設施頁面
- 互動式 AWS 全球基礎設施地圖
- AWS Well-Architected Framework 白皮書
- 安全性支柱白皮書
- 卓越運營支柱白皮書
- 可靠性支柱白皮書
- 性能效率支柱白皮書
- 成本優化支柱白皮書
- 可持續性支柱白皮書

如果您想進一步瞭解本模組中涵蓋的主題，以下額外資源可能會對您有所幫助：

- AWS 全球基礎設施頁面
- 互動式 AWS 全球基礎設施地圖
- AWS Well-Architected Framework 白皮書
- 安全性支柱白皮書
- 卓越運營支柱白皮書
- 可靠性支柱白皮書
- 性能效率支柱白皮書
- 成本優化支柱白皮書
- 可持續性支柱白皮書

感謝您完成本模組的學習。