Opis sytuacji:

Cześć Drogi Architekcie!

Pewnie doskonale pamiętasz o produkcie czyli Twoim e-commerce’owym rozwiązaniu.

Już miałeś okazję projektować architektury z nim związane, czas by pójść o krok dalej i rozważyć jak wykorzystać tą całą wiedzę o danych, by dobrze zaprojektować model ich składowania.

Przypomnę Ci o Twojej roli i firmie.

Jesteś architektem w firmie, która tworzy systemy w branży e-commerce dla innych firm, które sprzedają w modelu business-to-consumer w całej Europie. Systemy budujesz w oparciu o Microsoft Azure i strategia budowania systemów Cloud Native została przyjęta i jest w pełni akceptowana przez Twoich klientów

Jako architekt możesz zdecydować zarówno o architekturze systemu jak i wzorcach, które wykorzystasz. Klienci oczekują dobrze zaprojektowanego systemu, nie wnikają w Twoje decyzje.

System, który projektujesz, będzie odwiedzany przez klientów końcowych, a więc możesz się spodziewać, że:

1. W wybranych godzinach będzie zarówno dużo odwiedzin strony jak i dużo zamówień
2. System będzie miał bardzo nieprzewidywalną liczbę zamówień – zdarzą się okresy, że z systemu nikt nie będzie korzystał, ale też zdarzy się promocja typu „Black Friday”
3. Do systemu importują swoje towary również partnerzy firmy, którzy w różnych okresach roku promują wybrane produkty
4. System musi być jak najbardziej odporny na sytuacje awaryjne – jego podstawowa funkcjonalność, czyli sprzedaż i prezentacja ofert dla klientów powinna być dostępna „zawsze”

A więc mam dla Ciebie wyzwanie!

Zadanie:

8.1 Przeanalizuj swoją architekturę aplikacji i zdecyduj, czy i w którym miejscu umieściłbyś wszystkie poznane typy składowania danych.

Chciałbym być na poważnie przemyślał, jak i kiedy warto użyć Table Storage, dla jakich sytuacji dobry będzie Azure Search, gdzie przyda się Cosmos DB i jak zrobić analitykę dzięki wszystkim opcjom, które poznałeś. Rozważ DataWarehouse, HDInsights czy DataCatalog. Jeśli coś gdzieś nie pasuje, napisz o tym wprost. Decyzja musi być z głową i przemyślana.

Zwróć uwagę na cenę, każdej z wybieranych usług. Może się okazać, że może zrobić coś lepiej, taniej, szybciej za pomocą innej usługi – czekam na takie propozycje.

Dobór usług, z perspektywy kosztów to ważny element w pracy architekta dlatego musisz nabrać tutaj wprawy.

8.2 Czas na praktykę!

Powołaj najmniejszą z możliwych bazę Cosmos DB. Następnie spróbuj wykorzystać tzw. Change Feed a więc możliwość wyzwalania funkcji w reakcji na zmieniające się dane.

Więcej o Change Feed możesz zobaczyć tutaj:

<https://azure.microsoft.com/pl-pl/resources/videos/azure-cosmosdb-change-feed/>

A przykład zbudowania takiej funkcji zobaczysz w ramach tego linku: <https://docs.microsoft.com/en-us/azure/cosmos-db/change-feed>

POWODZENIA! To już 8 tydzień naszego Kursu!

#TYDZIEN8.1

**Table Storage**

* Dane sesji zalogowanego użytkownika – chociaż w tym wypadku ze względu na gwarantowaną wydajność lepiej nada się Cosmos DB Table API. Jednak jest to droższa usługa, więc musimy też o tym pamiętać.
* Dane w formacie key/value które później planujemy analizować natomiast nie zależy nam na szczególnie na zapewnionej przez dostawcę wydajności
* Wydaje mi się, ze nie będzie to najlepszy pomysł w przypadku gdy nasz sklep jest międzynarodowy, bo wtedy potrzebowalibyśmy wielu tego typu storage w różnych regionach, co może być uciążliwe ze względu na utrzymanie, zarządzanie, czy pobieranie danych z wielu źródeł do hurtowni

**Azure Search**

* użyłbym Azure Search w wyszukiwarce sklepu ze względu na podpowiedzi jakie może sugerować klientom ta usługa
* nigdy nie miałem okazji używać Azure Search poza własnymi testami, ale może się okazać drogą usługą, no chyba że zakładamy że nie będziemy często przebudowywać naszych indeksów, to wtedy pewnie na samo odpytywanie Azure Search nie będzie potrzebne wiele unitów
* nie widzę żadnej alternatywy dla tej usługi

**Cosmos DB**

* Pierwsze kryterium jakie biorę pod uwagę to czy jestem na tyle duży, że Cosmos DB (a właściwie koszty które za tym idą są dla mnie) – sprawdzam czy rzeczywiście takie istotne dla mnie są dystrybucja globalna, możliwości skalowania horyzontalnego, gwarancja poziomu latencji, HA, SLA
* Jeżeli jesteśmy dużym sklepem (skala enterprise) to na pewno jest to kompleksowa usługa, powinna się dobrze sprawdzić razem z Azure Search gdy np. mamy bardzo bogaty katalog produktów (coś a’la Ebay), możemy przechowywać dane dot. sesji użytkownika,
* na pewno możemy rozważyć tę opcję jeżeli nasz sklep to sklep międzynarodowy i chcemy składować w niej dane które logicznie będzie

**SQL DB**

* dla sklepu internetowego może to być dobre rozwiązanie do implementacji mniejszej hurtowni danych, oszczędzimy na kosztach względem np. SQL DW i możemy skalować ją w dół w momencie gdy nie jest używana
* bardzo dobrze nadaje się do postawienia na niej Analysis Services

**SQL DW**

* rozważam głównie w zależności od tego ile danych mam ogólnie i ile z nich zamierzam przetwarzać w celach późniejszej analizy
* ważnym czynnikiem będzie też oczekiwany czas dostępności danych, bo gdy np. potrzebna będzie analiza near-real time (albo real-time) może to być jedyna usługa która w krótkim czasie pozwoli na wykonanie niezbędnych obliczeń
* bardzo dobrze nadaje się do postawienia na niej Analysis Services

**HDInsights**

* może się przydać gdy np będziemy chcieli zbierać i analizować dane z clickstreamu i przyda się jakaś duża siła obliczeniowa zanim dane trafią do późniejszej analizy

#TYDZIEN8.2

Próbowałem stworzyć funkcje na 2 sposoby (C#):

- „Add Azure Function” z poziomu Cosmos DB

- „New Function” z poziomu Function App

I tylko udało mi się to w drugim przypadku, moja funkcja wygląda tak:

#r "Microsoft.Azure.DocumentDB.Core"

using System;

using System.Collections.Generic;

using Microsoft.Azure.Documents;

public static void Run(IReadOnlyList<Document> input, ILogger log)

{

if (input != null && input.Count > 0)

{

log.LogInformation("Documents modified " + input.Count);

log.LogInformation("First document Id " + input[0].Id);

}

}

Analogicznie Tworząc funkcję z poziomu „Add Azure Function” różnica była tylko w wygenerowanej referencji do biblioteki

#r "Microsoft.Azure.Documents.Client"

Błąd wskazywał na to, że nie może jej znaleźć, przez co funkcja nie mogła się poprawnie skompilować.

Biblioteka "Microsoft.Azure.Documents.Client" jest użyta w przykładach w dokumentacji zarówno dla wersji 1.x jak i 2.x. Ktoś może napotkał i rozwiązał / zrozumiał ten problem?