Opis sytuacji:

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”

Zadanie:

* [#TYDZIEN2.1] Na bazie poznanych wzorców projektowych wybierz 3, które chciałbyś zastosować wraz z uzasadnieniem przy każdym z nich, dlaczego właśnie ten wzorzec projektowy będzie przydatny i jakie problemy ew. może rozwiązać. Uzasadnienie nie może być długie – przy każdym wzorcu postaraj się nie przekroczyć 5 punktów maksymalnie. Wystarczą mi 3 dobre powody.
* [#TYDZIEN2.2] Jeśli to możliwe, przy wybranym wzorcu projektowym, zaproponuj, których usług byś użył, by wzorzec ten zaimplementować. Krótko uzasadnij, dlaczego będzie to najbardziej efektywne podejście.

Pamiętaj: tu nie ma jednej dobrej odpowiedzi więc każdy pomysł może być cenny i adresować różny aspekt rozwiązania.

#TYDZIEN2.1

**Queue-Based Load Leveling**

- w przypadku wzrostu zapotrzebowania na zasoby da czas na skalowanie zarówno kolejek jak i serwisów je obsługujących

- zadania mogą być przetwarzane przez system asynchronicznie i równolegle

- oszczędność w przypadku niskiego obciążenia systemu

**Throttling**

- zapewni działanie system w przypadku wyjatkowych okresów w roku (wspomniany czarny piątek, lub np. gdy partnerzy będą chcieli importować/promować swoje produkty w podobnych okresach, konkurując o zasoby z użytkownikami)

- nie było mowy o bezpieczeństwie w wymaganiach, ale z racji że nie jest to test zamknięty zwracam również na to uwagę

- uzupełnienie Queue-Based Load Leveling

**Static Content Hosting**

**-** ze względu na charakterystykę platform e-commerce i wolumen statycznego kontentu

- minimalizacja kosztów

- klienci są z całej Europy, aby przyspieszyć czas ładowania podstron i kontentu można go umieścić w różnych lokalizacjach

#TYDZIEN2.2

**Queue-Based Load Leveling**

- Service Bus Queues – inny rodzaj kolejek jakie widzę to Storage Queues, a porównując te dwie usługi to pierwsza nadaje się do takiej implementacji

- Load Balancer – rozdzielenie ruchu pomiędzy dostępne serwisy

**Throttling**

- API Management services – wynik wyszukiwania w jaki sposób realizować throttling na platformie Azure… 😊

**Static Content Hosting**

- Storage Account – ze względu na możliwość bezpośredniego dostępu do treści ze storage oraz niski koszt usługi