1. Problem - problem uznawania przychodów

Nasza aplikacja będzie dotyczyć problemu związanego z finansami, znanego jako **"problem uznawania przychodów"**.

Uznawanie przychodów to powszechny problem w systemach biznesowych. Chodzi o to, kiedy można faktycznie zapisać otrzymane pieniądze w księgach i potraktować jako przychód firmy. Jeśli sprzedamy komuś filiżankę kawy, to sytuacja jest prosta: otrzymujemy kawę, bierzemy pieniądze i od razu możemy traktować otrzymane pieniądze jako przychód.

Jednak w przypadku wielu innych rzeczy sprawy się komplikują. Powiedzmy, że płacisz zaliczkę danej osoba, aby przez cały kolejny rok wykonywała dla Ciebie pewne czynności. Nawet jeśli zapłacisz jej dziś jakąś absurdalną opłatę, nie może być ona od razu zapisana w księgach jako przychód, ponieważ usługa ma być świadczona przez cały rok. Jednym z podejść może być uwzględnienie tylko jednej dwunastej tej opłaty na każdy miesiąc w roku, ponieważ możesz zrezygnować z umowy po miesiącu, kiedy zdasz sobie sprawę, że np. dana osoba nie jest w stanie wykonać powierzonych jej zadań.

Zasady uznawania przychodów są zróżnicowane i zmienne. Niektóre są ustalane przez przepisy, inne przez standardy zawodowe, a jeszcze inne przez politykę firmy. Śledzenie przychodów okazuje się być dość skomplikowanym problemem.

W rzeczywistości niewłaściwe uznawanie przychodów było przyczyną kilku dużych skandali korporacyjnych, takich jak Enron i WorldCom. Firmy te stosowały różne taktyki, aby fałszywie przedstawiać swoje kondycje finansowe, co prowadziło do poważnych konsekwencji prawnych i strat finansowych dla inwestorów. Dlatego dokładne i zgodne z przepisami uznawanie przychodów jest kluczowe dla utrzymania przejrzystości i zaufania na rynkach finansowych.

Dodatkowy fakt: Enron był firmą, która zainspirowała Evil Corp z serialu Mr. Robot.

2. Kontekst

Tworzymy aplikację REST API - System Uznawania Przychodów dla dużej korporacji ABC.

3. Wymagania funkcjonalne

Aby opracować system uznawania przychodów, musimy upewnić się, że odpowiednio rozpoznaje przychody dla różnych typów produktów. Powinniśmy również umożliwić użytkownikowi następują funkcje.

3.1. Zarządzanie klientami

Przypadki użycia:

1. dodaj klienta,
2. usuń klienta,
3. zaktualizuj dane o kliencie

Chcielibyśmy przechowywać informacje o klientach, którzy mogą być zarówno osobami fizycznymi, jak i firmami. Dla osób fizycznych musimy przechowywać imię, nazwisko, adres, email, numer telefonu i PESEL (Polski Numer Identyfikacyjny). Dla firm musimy przechowywać nazwę firmy, adres, email, numer telefonu i numer KRS (Numer Krajowego Rejestru Sądowego). Wszystkie wymienione dane są wymagane.

Potrzebujemy możliwości dodawania, aktualizacji i usuwania klientów. Dla klientów indywidualnych numer PESEL nie może być zmieniany po jego wprowadzeniu. Podobnie w przypadku firm, numer KRS nie może być zmieniany po jego wprowadzeniu.

Usuwając dane o kliencie indywidualnym, wykonujemy miękkie usunięcie. Oznacza to, że nadpisujemy dane w bazie, ale zachowujemy sam rekord w bazie danych. Dane o firmach nie mogą być usuwane.

3.2. Licencja na oprogramowanie

Nasz klient sprzedaje różne rodzaje dóbr i usług. Jednym z produktów są systemy oprogramowania. Każde oprogramowanie ma nazwę, opis, informacje o aktualnej wersji i kategorii (finanse, edukacja itp.). Każde oprogramowanie może być sprzedawane na zasadzie subskrypcji i/lub zakupione w postaci pojedynczej transakcji wraz z prawem do aktualizacji na dany okres.

3.2.1. Zniżki

Z produktami związane są zniżki. Zniżki mogą być stosowane do ceny zakupu oprogramowania lub do opłaty za pierwszy okres subskrypcji. Zniżki są wyrażane w procentach i mają określony przedział czasowy, w którym są aktywne.

Na przykład:
Nazwa Oferta Wartość Przedział czasowy
Black Friday Discount. Zniżka na subskrypcje 10%. 01-01 do 03-03

Zniżka może być tylko zawarto w momencie kiedy:

- aktywności kontraktu na oprogramowanie
- w momencie zakupu subskrypcji

3.2.2. Kontrakt na zakup oprogramowania

Przypadki użycia:
4. stworzenie umowy
5. wystawienie płatności za umowę

Dla określonego oprogramowania powinniśmy dokładnie wiedzieć, jaki jest koszt zakupu licencji na to oprogramowanie na rok.

Jeśli mamy klienta, który chce kupić nasze oprogramowanie na zasadzie jednorazowej opłaty:

Tworzenie umowy:

- Najpierw tworzymy umowę dla klienta. Umowa ma **datę rozpoczęcia i zakończenia**. Przedział czasowy powinien wynosić **co najmniej 3 dni i maksymalnie 30 dni**.
- Umowa **musi być opłacona przez klienta** we wspomnianym przedziale czasowym. W przeciwnym razie oferta jest już nieaktywna.
- Umowa ma również **cenę z nią związaną**. Cena obejmuje **wszystkie możliwe zniżki.**
- Umowa zawiera informacje o **aktualizacjach**, które klient może otrzymać dla zakupionego oprogramowania.
- Każda umowa zapewnia **co najmniej 1 rok aktualizacji** dla nabywcy produktu. Możemy oferować **dodatkowe wsparcie** w ramach umowy — **każdy dodatkowy rok dodaje dodatkowe 1000 PLN do kosztu umowy.** Możemy przedłużyć wsparcie tylko o 1, 2 lub 3 lata,
- Gdy **dostępnych jest wiele zniżek**, wybieramy **najwyższą.**
- Wszyscy **poprzedni klienci naszej firmy otrzymują 5% zniżki** dla powracających klientów. Poprzedni klient oznacza, że klient zakupił co najmniej jedną subskrypcję lub podpisał jedną umowę. Ta zniżka może być dodana do innych.
- Umowa jest związana z wybranym oprogramowaniem i konkretną wersją tego oprogramowania.
- Każda umowa **określa wersję oprogramowania** z nią związaną.
- Umowa **nie może być zmieniona po jej stworzeniu**; może być tylko usunięta.
- Tworzenie umowy **nie oznacza, że klient ją podpisał.**
- **Cena na umowie nie może być traktowana jako przychód.** Dopiero po pełnym uregulowaniu płatności możemy traktować wartość umowy jako przychód.
- Tworząc umowę, upewnij się, że klient **nie ma już aktywnej subskrypcji ani aktywnej umowy na ten produkt.**

Po przygotowaniu umowy dla klienta czekamy na uregulowanie płatności za umowę przez klienta. Klient musi dokonać pełnej płatności w przedziale czasowym określonym w umowie.

Płacenie za umowę:

- Klient może zapłacić za umowę. Płatności mogą być **dokonywane jako jednorazowa płatność lub w ratach.**
- Płatność powinna zawierać informacje o kliencie i kontrakcie jakiego dotyczy.
- **Całkowita wartość wszystkich płatności musi równać się kwocie podanej w umowie.**
- Po pełnym uregulowaniu płatności za umowę **zakładamy, że umowa została podpisana.**
- Po podpisaniu umowy **możemy traktować jej wartość jako przychód.**
- Nie możemy przyjąć płatności za umowę po dacie określonej w umowie. Jeśli klient się spóźnił - musimy przygotować dla niego nową ofertę. Poprzednie płatności zostaną zwrócone klientowi i nie mogą być traktowane jako przychód.

3.2.3. Subskrypcje - opcjonalne

Część dotycząca subskrypcji jest opcjonalna

Przypadki użycia:
6. zakup subskrypcji przez klienta
7. płacenie za subskrypcję

Nasza firma oferuje również subskrypcje na korzystanie z określonych usług (Software as a service).

- Każda subskrypcja jest związana z **pojedynczym klientem** (osobą fizyczną lub firmą).
- Każda subskrypcja jest związana z **pojedynczym serwisem.**.
- Każda oferta subskrypcji na oprogramowanie ma:
  - **Nazwę**
  - **Okres odnowienia** (miesięczny, roczny itp.)
  - **Cenę**, która musi być zapłacona na początku każdego okresu odnowienia
  - Okres odnowienia powinien wynosić **co najmniej 1 miesiąc do 2 lat maksymalnie.**

Zakup nowej subskrypcji:

- Rejestrujemy nową sprzedaż subskrypcji dla określonego klienta i określonego oprogramowania.
- Musimy uwzględnić niezbędne informacje o subskrypcji.
- Zakładamy, że rejestracja subskrypcji oznacza, że klient **już zapłacił za pierwszy okres odnowienia. Możemy traktować to jako nasz przychód.**
- Obliczając płatność, **pamiętaj o uwzględnieniu 5% zniżki dla naszych lojalnych klientów.** Ta zniżka jest stosowana do wszystkich płatności za odnowienia.
- Inne zniżki są stosowane tylko **jeśli są aktywne podczas rejestracji nowych subskrypcji.** Stosujemy najwyższą zniżkę do ceny za pierwszy okres odnowienia. **Zniżka jest stosowana tylko do pierwszej płatności.**
- Stosujemy dodatkową 5% zniżkę dla naszych lojalnych klientów.

Płacenie za subskrypcje:

- Klient powinien zapłacić za subskrypcję na początku nowego okresu odnowienia. Jeśli użytkownik nie zapłacił jeszcze za inne okresy odnowienia - anulujemy subskrypcję.
- System **nie powinien akceptować płatności**, jeśli klient już zapłacił za bieżący okres odnowienia. **Akceptujemy płatność za bieżący okres odnowienia tylko w jego ramach czasowych.**
- Jeśli kwota zapłacona przez klienta nie jest równa kwocie za okres odnowienia, zgłaszamy wyjątek.
- Stosujemy zniżkę dla lojalnych klientów przy płatności za odnowienie.

3.3. Obliczanie przychodu

Przypadki użycia:
7. Obliczanie przychodu
8. Obliczanie przewidywanego przychodu

Chcielibyśmy mieć możliwość obliczenia bieżącego przychodu firmy. Konkretnie, potrzebujemy możliwości obliczenia przychodu (mogą to być dodatkowe parametry przy wysyłaniu żądania o obliczenie przychodu):

- Dla całej firmy.
- Dla określonego produktu.
- Przeliczając przychód na określoną walutę. Możemy założyć, że wszystkie dane w bazie danych są przechowywane w PLN. Znajdź publiczną usługę, która zwraca kurs wymiany dla określonej waluty. Użyj tego kursu do przeliczenia przychodu na wymaganą walutę.

Chcielibyśmy móc obliczać zarówno nasz bieżący przychód, jak i przewidywany przychód. Różnica polega na tym, że bieżący przychód uwzględnia tylko pieniądze już otrzymane z płatności i umów.

W przypadku przewidywanego przychodu możemy obliczyć wartość na podstawie następujących założeń:

- Użytkownicy będą nadal płacić za swoje subskrypcje.
- Wszystkie umowy, które nie zostały jeszcze podpisane, zostaną ostatecznie podpisane.

Wymagania techniczne

- Pamiętaj o właściwym zaprojektowaniu architektury.
- Spróbuj stworzyć model domenowy, możesz ponownie użyć swojego modelu domenowego i jednocześnie użyć EF.
- Przygotuj zestaw testów jednostkowych do testowania logiki biznesowej.
- Użyj Entity Framework.
- Zaprojektuj odpowiednie punkty końcowe REST API.
- Upewnij się, że możemy testować API za pomocą Swagger.
- Stosuj wszystkie dobre praktyki i wzorce projektowe, które uważasz za przydatne.
- Możesz rozdzielić swoją aplikację, używając katalogów (przestrzeni nazw) lub oddzielnych projektów.
- Możesz użyć podejścia "pionowego" lub "poziomego" pod kątem podziału architektury.
- **Wszystkie punkty końcowe powinny być dostępne tylko dla zalogowanego pracownika. Musimy znać tylko login, hasło i rolę pracownika. Zakładamy, że tylko edycja i usuwanie klientów jest dostępne dla użytkowników z rolą admin. Reszta przypadków użycia jest dostępna dla standardowego użytkownika.**
- **Musisz zaimplementować tylko przypadki użycia wymienione w tekście.
- **Nawet jeśli nie zdążysz zaimplementować wszystkich funkcji - prześlij projekt.**