Przemysław Wiśniewski

Indeks: s10436

Grupa: WIn II.3 - 1w

System Sklepu z Płytami

Spis treści:

1. Cel
2. Zakres
3. Wymagania funkcjonalne
4. Wymagania niefunkcjonalne
5. Ewolucja systemu
6. Ideowy diagram klas
7. Decyzje implementacyjne

7.1 Overlapping

7.2 Kompozycja

7.3 Asocjacje

7.4 Ekstensje

7.5 Asocjacja z atrybutem

7.6 Asocjacja kwalifikowana

1. **Cel:**

System ma za zadanie wsparcie sklepu zajmującego się sprzedażą płyt muzycznych.

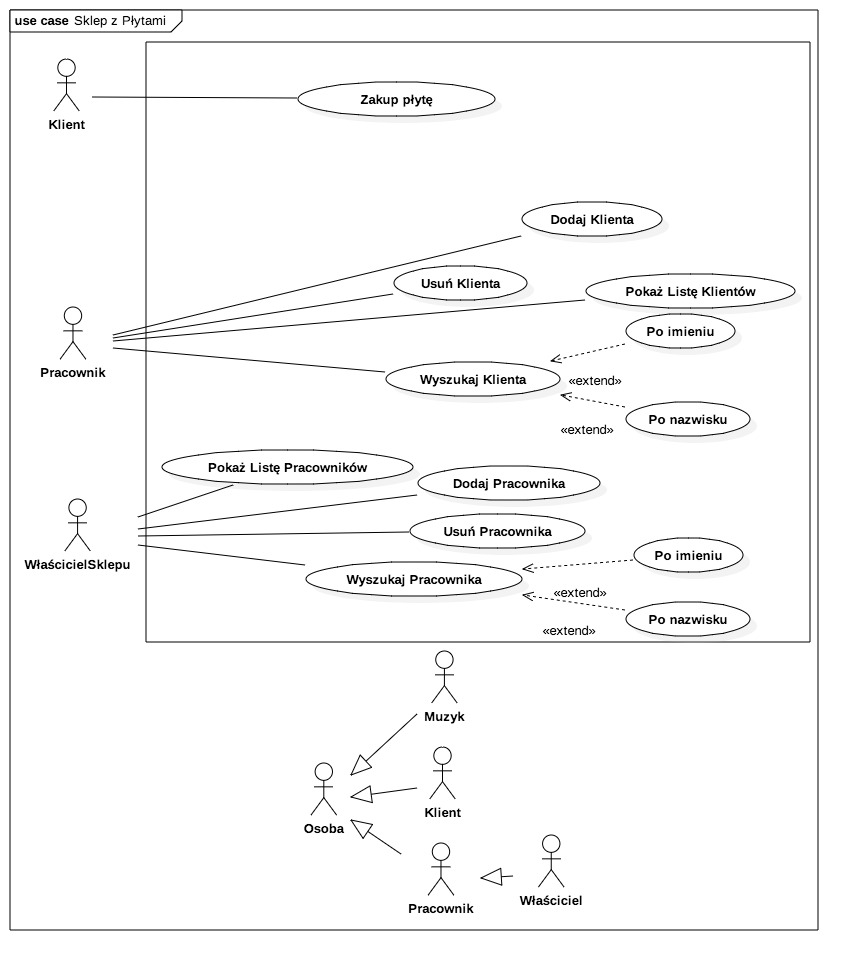
1. **Zakres:**

System będzie posiadać informacje o zakupach, płytach, klientach oraz pracownikach. Detale na temat płyt znacznie ułatwią sklepowi skatalogowanie swoich produktów, natomiast informacje o dotychczasowych zakupach, gwarantują śledzenie najwierniejszych klientów. System przetrzymuje informacje dotyczące klienta(czy jest stałym klientem, ile wydał do tej pory, kiedy został dodany do systemu, email) jak i pracowników sklepu(numer telefonu, kiedy został zatrudniony, pensja).

1. **Wymagania funkcjonalne:**

System przetrzymuje dane na temat płyt, osobach i organizacjach zaangażowanych w jej tworzenie, oraz transakcjach przeprowadzanych w sklepie. Każdy zakup posiada datę realizacji transakcji oraz łączny koszt danego zakupu. Klient i pracownik są traktowani jako osobne byty. Jeśli pracownik chce złożyć zamówienie, powinien użyć systemu jako klient. Każdy zakup przypisany jest dokładnie do jednego klienta, i jest obsługiwany przez jednego pracownika sklepu(sprzedawca w danym momencie). Dla pracownika chcemy przetrzymywać informacje o imieniu i nazwisku, numerze PESEL, numerze telefonu, dacie zatrudnienia oraz pensji. Dla klienta chcemy przetrzymywać informacje o imieniu i nazwisku, emailu, dacie dodania do systemu, sumie jaką klient wydał do tej pory w sklepie oraz czy suma ta kwalifikuje go do miana stałego klienta.

Dla płyt chcemy przetrzymywać informacje o twórcach płyty(jaki zespół, jacy muzycy, ile lat temu została wydana), wytwórnię która wydała płytę, włączając w to budynki należące do wytwórni z ich adresami i numerem kontaktowym. Każda płyta może należeć do kilku gatunków muzycznych jednocześnie. Dla muzyków chcemy zapamiętywać datę dołączenia rozpoczęcia i zakończenia pracy w zespole. Jeśli muzyk dalej jest w zespole, pracownik ma możliwość pozostawienia pustego miejsce w obiekcie klasy data zakończenia, co dam nam zamiast daty zakończenia napis „aktualnie”. W przypadku emaila zastosowane zostanie podobne rozwiązanie(klient nie ma konieczności podawania emaila, ale istnieje taka możliwość. Daje to szanse na wysyłanie informacji dokładnie dostosowanych do preferencji zakupowych klienta) Pozostawienie pustego pola podczas tworzenia obiektu Email dam nam informacje o braku przypisania emaila do danego klienta. Właściciel sklepu powinien mieć wgląd w całą listę pracowników razem z przypisanymi do nich transakcjami, włączając w to dodawanie i usuwanie pracowników. Pracownik może dodawać, usuwać, i wyszukiwać klientów w systemie.

****

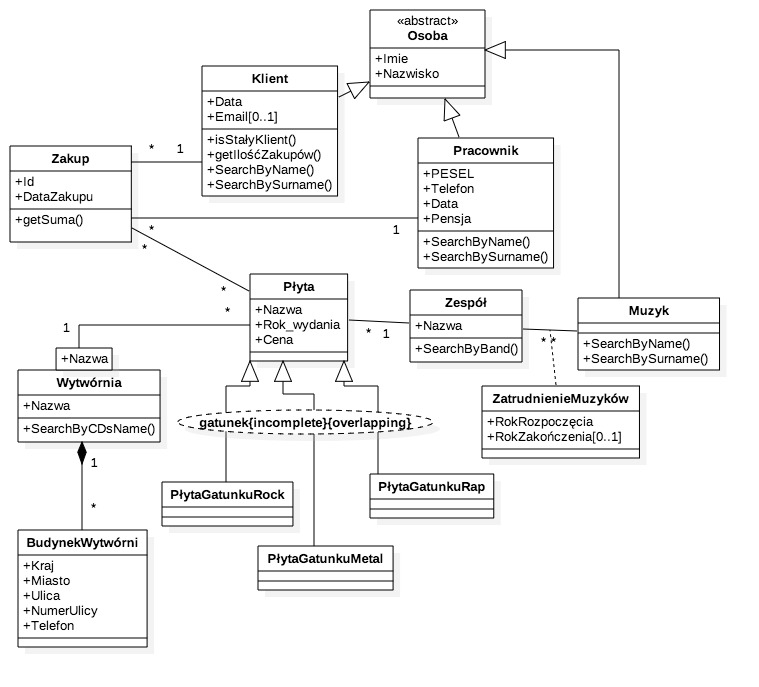
1. **Wymagania niefunkcjonalne:**

System do działania wymaga komputera PC oraz podłączenia do internetu.

1. **Ewolucja Systemu:**

System ma możliwości by w przyszłości został rozbudowany o obsługę zamówień składanych przez klientów drogą internetową.

1. **Ideowy Diagram Klas:**



1. **Decyzje implementacyjne:**
   1. **Overlapping**

Ten rodzaj dziedziczenia nie istnieje w języku java. Zastąpiłem go kompozycją.

* 1. **Kompozycja**

Do zaimplementowania kompozycji wykorzystam zwykłą asocjację jeden do wiele, gdzie część nie będzie mogła zostać utworzona bez istnienia całości.

**7.3 Asocjacje**

Asocjacje zaimplementuje za pomocą referencji oraz kolekcji, wykorzystywanej jako kontener przechowujący obiekty, tam gdzie zachodzi taka potrzeba.

* 1. **Ekstensje**

Ekstensje postanowiłem utworzyć w tych samych klasach, by nie mieć problemów z dostępem do atrybutów

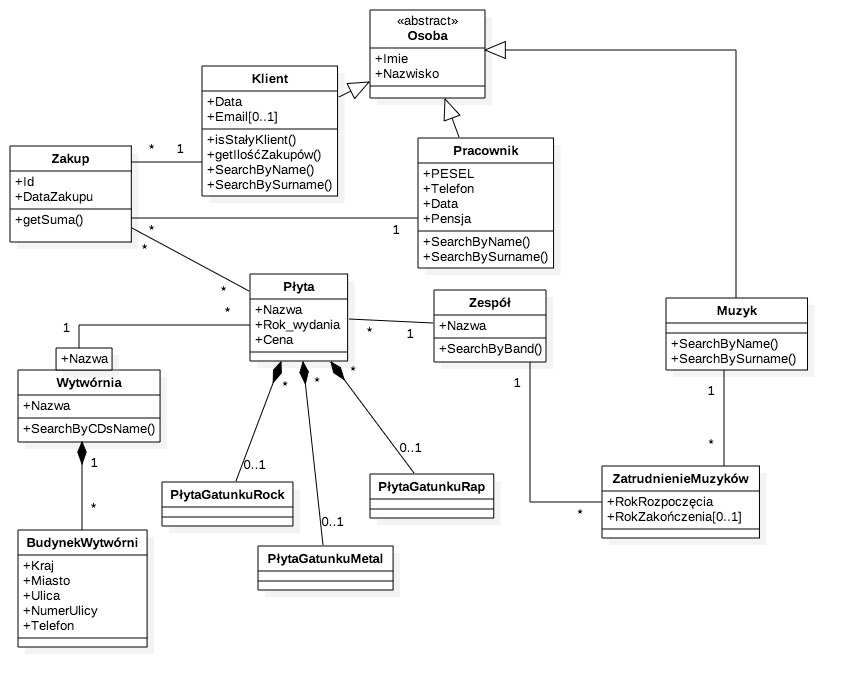
* 1. **Asocjacja z atrybutem**

Nie występuje w języku Java. Do implementacji atrybutu asocjacji wykorzystam dodatkową klasę, utworzoną specjalnie do tego celu.

* 1. **Asocjacja kwalifikowana**

Do implementacji asocjacji kwalifikowanej wykorzystam Treemapę oraz ArrayList

Po uwzględnieniu zmian diagram klas wygląda w następujący sposób:

**Ek**