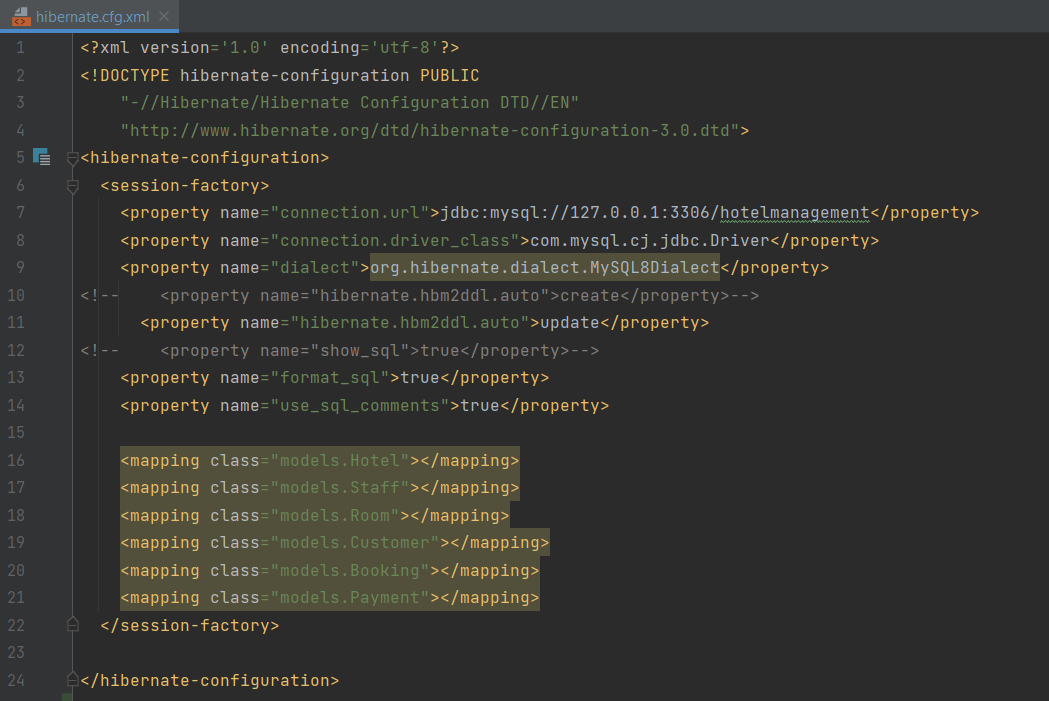
**Rapport**

1. **hibernate.cfg.xml:**



Ten plik konfiguracyjny Hibernate definiuje, jak twoja aplikacja będzie łączyć się i pracować z bazą danych MySQL za pomocą Hibernate ORM. Zapewnia podstawowe informacje i konfiguracje dla sesji Hibernate, pomagając Hibernate zrozumieć, jak zarządzać encjami i komunikować się z bazą danych.

1. **Diagram:**

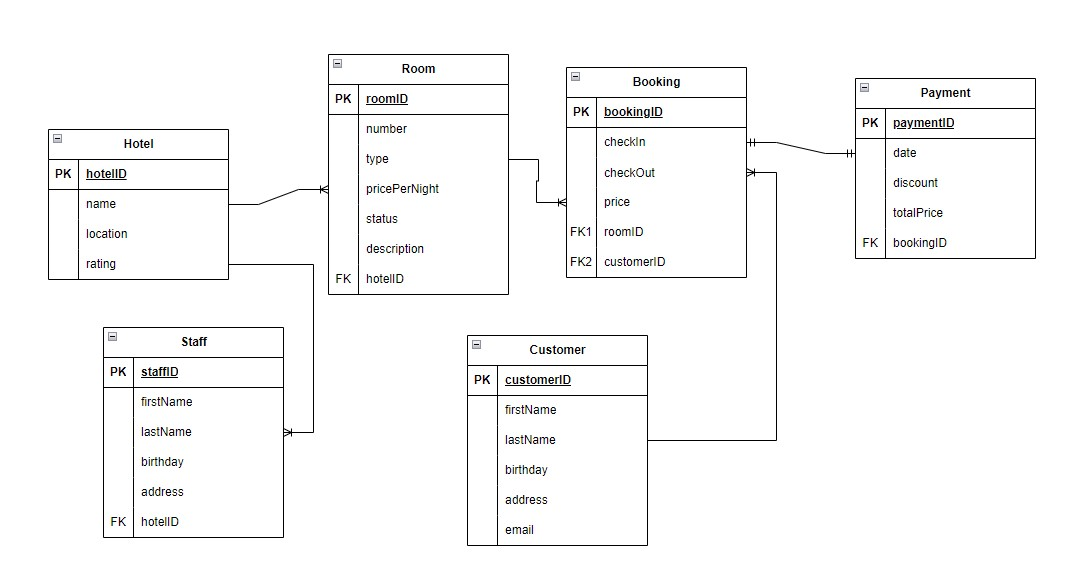


Diagram projektu jest opisany za pomocą relacyjnej bazy danych, zawiera ona kilka tabel reprezentujących różne encje w systemie zarządzania hotelem. Oto krótkie wyjaśnienie diagramu projektu:

- Tabela "Hotel": reprezentuje hotele w systemie. Każdy hotel ma wiele pracowników (staff) i wiele pokoi (room).

- Tabela "Staff": reprezentuje pracowników hotelu. Każdy pracownik jest związany z jednym hotelem.

- Tabela "Room" reprezentuje pokoje w hotelach. Każdy pokój może być zarezerwowany wielokrotnie, więc istnieje relacja wiele do wielu między tabelą "Room" a tabelą "Booking".

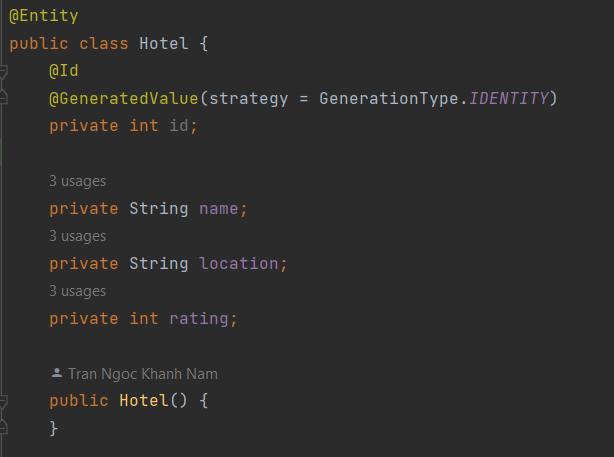
- Tabela "Customer" reprezentuje klientów hotelu. Każdy klient może dokonać wielu rezerwacji (booking).

- Tabela "Booking" reprezentuje rezerwacje dokonane przez klientów. Każda rezerwacja jest związana z jednym pokojem (room) i jednym klientem (customer). Ponadto, każda rezerwacja ma tylko jedno płatność (payment).

- Tabela "Payment" reprezentuje płatności dokonane w ramach rezerwacji. Każda płatność jest związana z jedną rezerwacją (booking).

1. **Modelu:**

*- Pola klasy Hotel:*



int id: Jest to pole przechowujące identyfikator (klucz główny) hotelu.

String name: Przechowuje nazwę hotelu.

String location: Przechowuje lokalizację hotelu.

int rating: Przechowuje ocenę hotelu.

*- Pola klasy Room:*

int roomId: Pole przechowujące identyfikator (klucz główny) pokoju.

String number: Przechowuje numer pokoju.

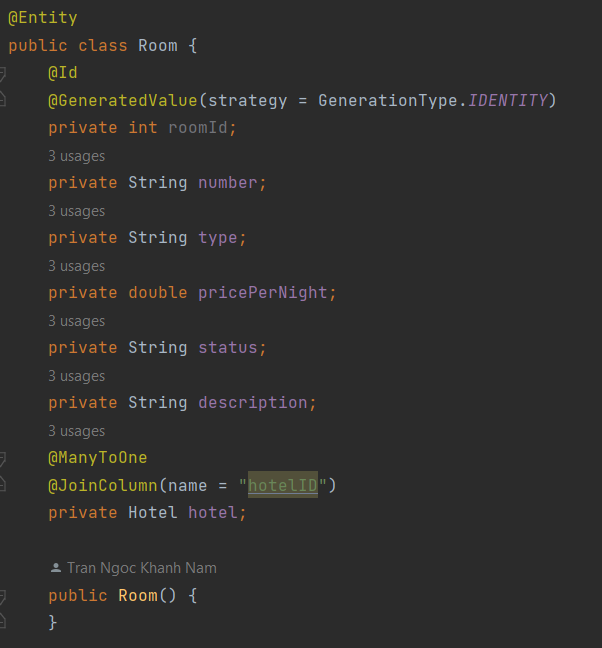
String type: Określa typ pokoju (np. "single", "double", "triple", "quad").

double pricePerNight: Przechowuje cenę za nocleg w pokoju.

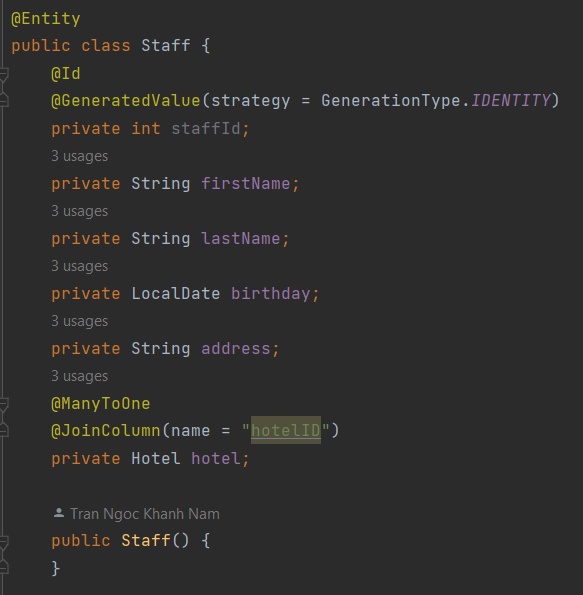
String status: Informacja o statusie pokoju (np. "available", "booked").

String description: Opis pokoju.

Hotel hotel: Oznacza to, że wiele pokoi może być przypisanych do jednego hotelu.



*- Pola klasy Staff:*



int staffId: Jest to pole przechowujące identyfikator (klucz główny) pracownika.

String firstName: Przechowuje imię pracownika.

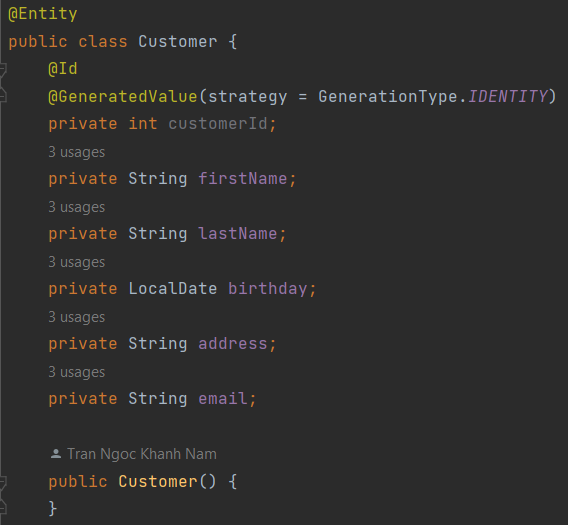
String lastName: Przechowuje nazwisko pracownika.

LocalDate birthday: Przechowuje datę urodzenia pracownika.

String address: Przechowuje adres pracownika.

Hotel hotel: Oznacza to, że wiele pracowników może być przypisanych do jednego hotelu.

*- Pola klasy Customer:*



int customerId: Jest to pole przechowujące identyfikator (klucz główny) klienta.

String firstName: Przechowuje imię klienta.

String lastName: Przechowuje nazwisko klienta.

LocalDate birthday: Przechowuje datę urodzenia klienta.

String address: Przechowuje adres klienta.

String email: Przechowuje adres e-mail klienta.

*- Pola klasy Booking:*

Long bookingId: Jest to pole przechowujące identyfikator (klucz główny) rezerwacji.

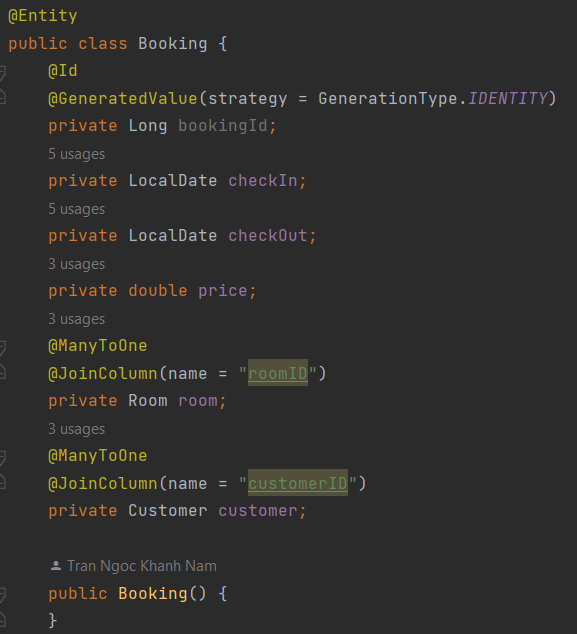
LocalDate checkIn: Przechowuje datę zameldowania (check-in) klienta.

LocalDate checkOut: Przechowuje datę wymeldowania (check-out) klienta.

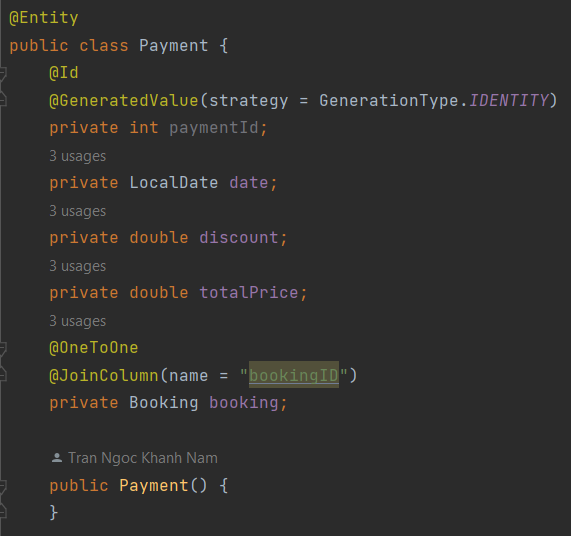
double price: Przechowuje cenę rezerwacji.

Room room: Jest kluczem obcym łączącym rezerwację z pokojem.

Customer customer: Jest kluczem obcym łączącym rezerwację z klientem.



*- Pola klasy Payment:*



int paymentId: Jest to pole przechowujące identyfikator (klucz główny) płatności.

LocalDate date: Przechowuje datę płatności.

double discount: Przechowuje wartość rabatu (jeśli jest).

double totalPrice: Przechowuje łączną cenę płatności.

Booking booking: Jest kluczem obcym łączącym płatność z rezerwacją.

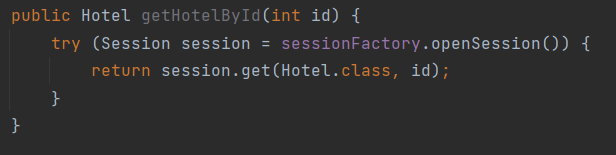
1. **DAO (data access object):**

*- Klasa HotelDAO:*

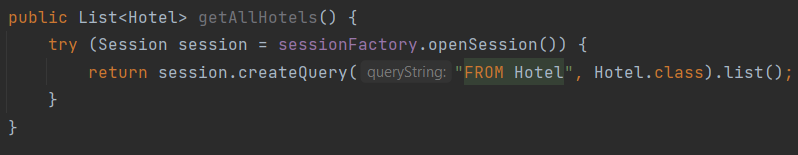
addHotel: Metoda ta służy do dodawania nowego hotelu.



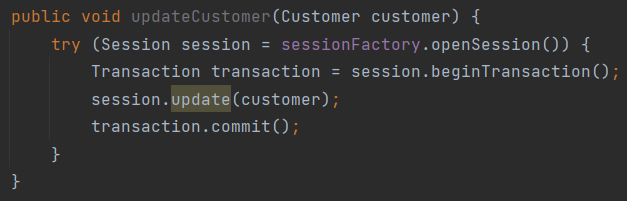
getHotelById: Ta metoda służy do pobierania hotelu z bazy danych na podstawie jego ID.



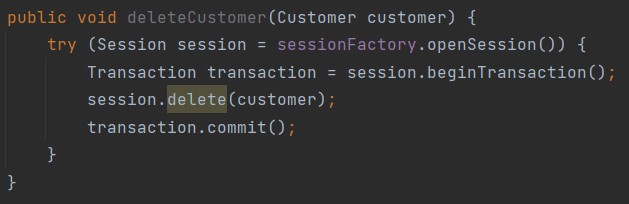
getAllHotels: Metoda ta zwraca listę wszystkich hoteli znajdujących się w bazie danych.



updateHotel: Metoda ta służy do aktualizacji istniejącego rekordu hotelu.



deleteHotel: Metoda ta służy do usuwania hotelu z bazy danych.



Takie wszystkie pozostałe klasy DAO posiadają funkcje podobne do klasy HotelDAO, z tym że każda z tych klas ma także swoje własne funkcje specyficzne dla niej:

*- Klasa RoomDAO:*

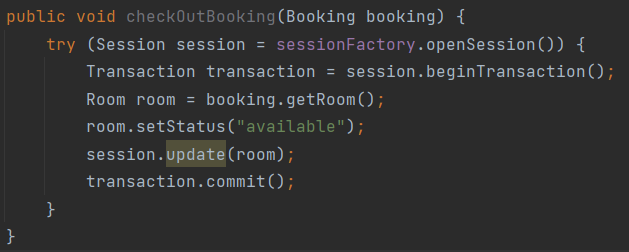
isValidRoomType: Służy do sprawdzenia, czy typ pokoju jest prawidłowy. Sprawdza, czy wartość roomType nie jest nullem i czy jest jednym z czterech prawidłowych typów pokoju: "single", "double", "triple" lub "quad".

getAvailableRooms: Służy do pobrania listy dostępnych pokoi (zapytanie HQL).



*- Klasa BookingDAO:*  
 isRoomAvailable: Sprawdza, czy dany pokój jest dostępny (status "available").

checkOutBooking: Używana do procesu "checkout" dla danego booking. Status pokoju przypisanego do danego booking jest zmieniany na "available".



*- Klasa PaymentDAO:*

paidBeforeCheckin: Sprawdza, czy płatność (payment) została dokonana przed datą zameldowania (check-in) związana z rezerwacją (booking).

