

Projekt PROO

Zespół:

Mikołaj Bańkowski 310408

Rafał Zan 311214

Hotelooo app

1.Cel projektu:

a) Przeznaczenie:

Na projekt składa się aplikacja z interfejsem użytkownika pozwalająca na rezerwację pokoju hotelowego w danym mieście oraz hotelu.

b) Sposób działania i jego efekty:

Klient dostanie możliwość wyszukania wszystkich hoteli w danej miejscowości (jeśli czas pozwoli będzie możliwość wyszukiwania także według innych kryteriów, takich jak nazwa hotelu, udogodnienia itp.). Wybrania z listy interesującego go obiektu a następnie zarezerwowania na swoje (na chwilę obecną) podstawowe dane osobowe danego pokoju.

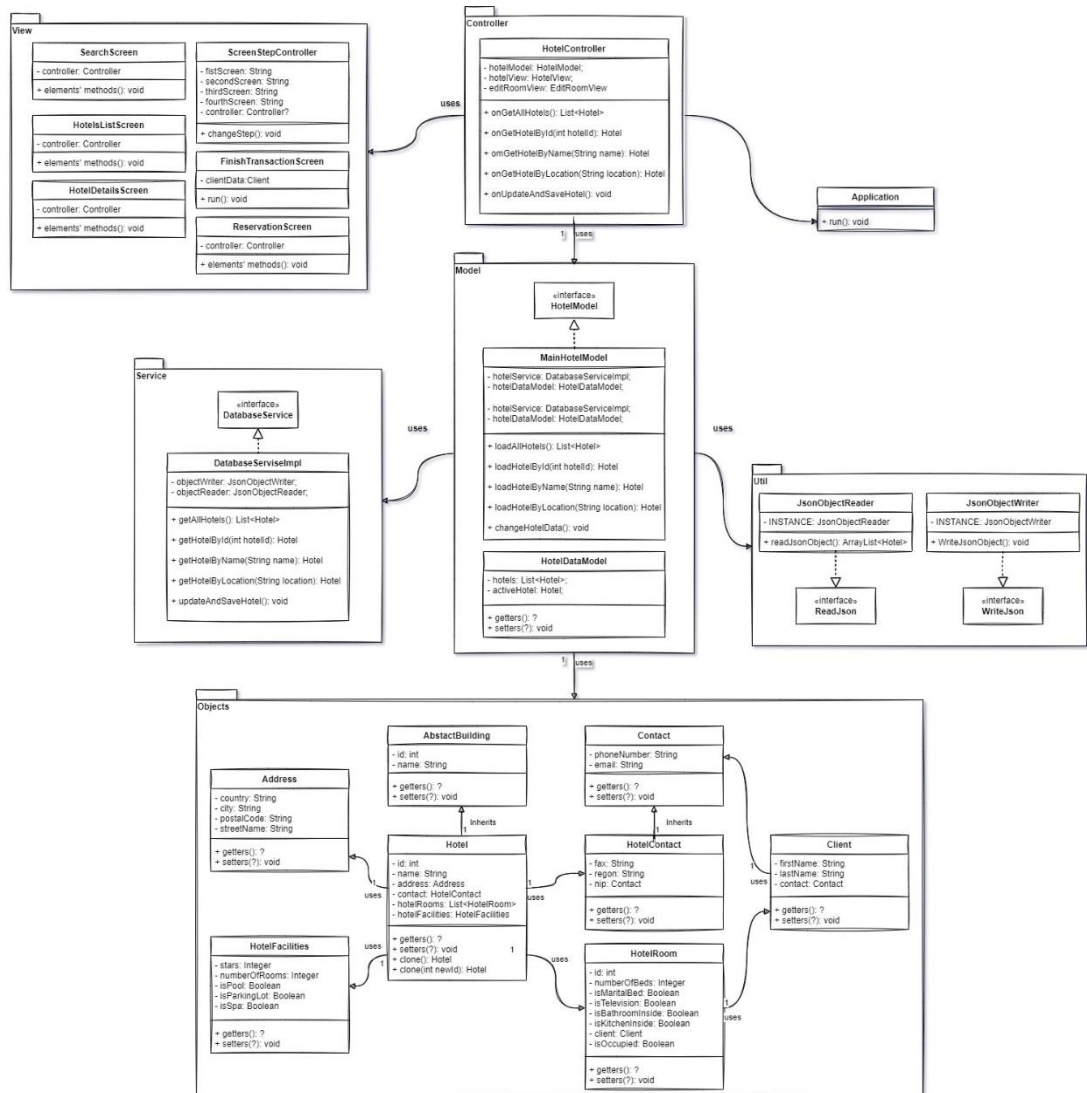
c) Potrzebne pliki:

Jedynym potrzebnym plikiem jaki będzie wymagany do działania aplikacji to baza danych wszystkich dostępnych hoteli wraz ze zdjęciami tychże hoteli.

2.Propozycja logo:



3.Diagram UML architektury klas:



4.Makiety ekranów interfejsu użytkownika:

Ekran wyszukiwania:

The wireframe shows a light blue rectangular area representing the search screen. In the center, the word "Hoteleooo" is written in a large, bold, yellow font. Below the title, there is a horizontal search bar. The search bar consists of a white input field on the left, containing a series of dots (.....), and a gray button on the right with the text "szukaj" in black.

Lista dostępnych obiektów:

The wireframe shows a light blue rectangular area representing the search results screen. In the top left corner, the word "Hoteleooo" is written in a large, bold, yellow font. In the top right corner, there is a gray button with the text "Powrót do wyszukiwarki" in black. In the center, there is a table with a light gray header and a white body. The header contains the text "Wyniki wyszukiwarki". The body contains a list of five items, each starting with "Nazwa hotelu " followed by a number from 1 to 5.

Wyniki wyszukiwarki
Nazwa hotelu 1
Nazwa hotelu 2
Nazwa hotelu 3
Nazwa hotelu 4
Nazwa hotelu 5

Dane wybranego hotelu:

Hotelo		Powrót do wyszukiwarki
Zdjęcie hotelu	Nazwa hotelu	Zarezerwuj
	Lokalizacja	
	Cena za dobę	
	Liczba pokoi	
	Udogodnienia(np. Wi-Fi itd.)	
	Recenzja 5/5	
	Krótki opis	

Ekran rezerwacji:

Hotelo		Powrót do wyszukiwarki = Anuluj rezerwacje
Rezerwacja		
Imię i nazwisko rezerwującego		
Numer telefonu		
Liczba osób		
Zameldowanie		
Wymeldowanie		
Cena do zapłaty xxx PLN(liczba dni * cena za dobę)		Potwierdź i zapłać

Ekran finalizacji procesu rezerwacji pokoju:

Hotelooo

Podsumowanie			
Imię i nazwisko rezerwującego			
Numer telefonu			
Liczba osób			
Zameldowanie			
Wymeldowanie			
Cena			
Anuluj	Do anulowania transakcji pozostało	0:30	Zapłać

5.Opis zależności i zastosowań poszczególnych modułów:

Moduł Service:

- znajdują się w nim klasy, których zadanie będzie polegało na wystawieniu podstawowych operacji na bazie dla reszty programu w tym przypadku modułu model, o którym zaraz. Moduł ten będzie w sobie zawierał klasy DatabaseService wystawiające gotowe, odczytane z bazy obiekty, za pomocą spersonalizowanych getterów.

Moduł Util:

- moduł ten będzie zawierać klasy ObjectReader oraz ObjectWriter, których zadanie będzie polegało na odczycie i zapisie obiektów do pliku. Będzie on zawierał także odpowiednie interfejsy do opisanych w poprzednim zdaniu klas.

- *Jeśli czas pozwoli w module tym pojawią się także klasy odpowiedzialne za walidację wprowadzonych i wyświetlanych danych, oraz error handlers.

Moduł Model:

- moduł ten odpowiedzialny będzie za całą logikę biznesową aplikacji. Zawierać będzie on klasy odpowiedzialne za wystawianie danych, do kontrolera, najważniejszymi klasami będą HotelModel, MainHotelModel oraz interfejsy. Będzie tu także ekran wątku przypominającego o dokończeniu rezerwacji hotelu.

Moduł Objects:

- zadaniem tego podmodułu będzie przechowywanie wszystkich obiektów z których korzysta aplikacja, czyli m.in. Hotel, Client, Address, Contact itp.

Moduł View:

- moduł ten odpowiedzialny będzie za przechowywanie całej logiki ekranów (czyli interfejsu użytkownika), wyświetlanie, aktualizacja ekranów, mechanizm przełączania między nimi.

Moduł Controller:

- moduł ten będzie szyną mediacyjną pomiędzy backendem a frontendem czyli modułami View oraz Model. Będzie on najprawdopodobniej zawierać góra 1,2 klasy odpowiedzialne właśnie za ten fakt.

*Ostateczna architektura oraz forma projektu może ulec zmianie w zależności od ilości dostępnego czasu (oraz ilości kawy w szafkach kuchennych deweloperów oraz wytrzymałości ich mięśni sercowych)