Aplikacje Baz Danych (APBD) - projekt nieobowiązkowy na ocenę 5

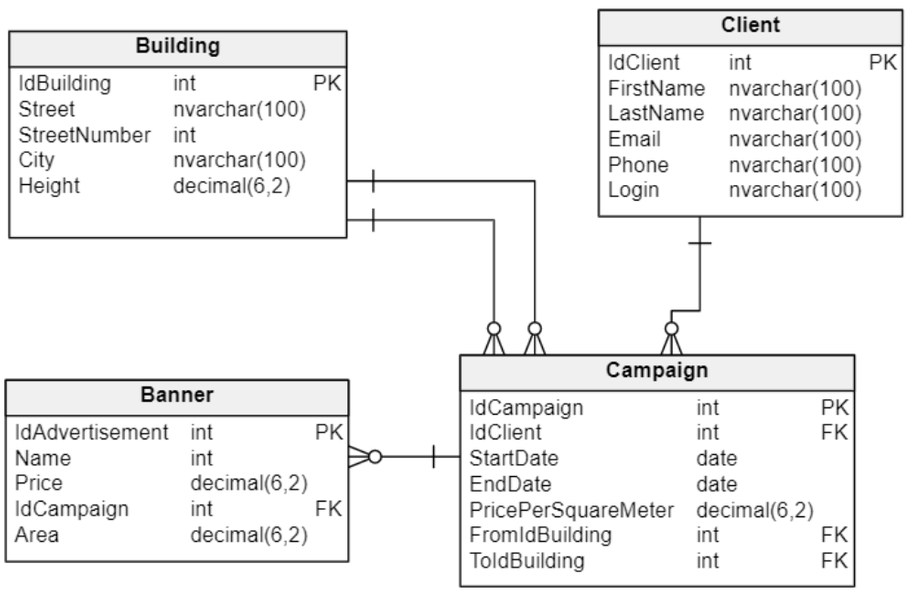
6 czerwca 2020

1 Projekt

Niniejszy projekt jest nieobowiązkowy. Wymagany jest wyłącznie w przypadku osób, które chciałyby uzyskać ocenę 5 (bardzo dobra).

2 Opis

Przygotowujemy aplikację, która służy wykupowania przestrzeni reklamowej i tworzenia kampanii reklamowej. Aplikacja przechowuje informacje na temat budynków, kampanii reklamowych i klientów. Poniższy diagram reprezentuje bazę danych.



3 Wymagania funkcjonalne

- Stwórz aplikację typu API REST o nazwie AdvertApi.
- Stwórz bazę danych wykorzystując EF i podejście CodeFirst lub DatabaseFirst. Tutaj każdy może samemu wybrać podejście.
- Pamiętaj o poprawnym nazewnictwie wszystkich zmiennych, migracji, metod itd.
- Pamiętaj o wydzieleniu osobnej warstwy komunikacji z bazą danych (osobna warstwa DAL lub serwis wstrzykiwany do kontrolera).
- Pamiętaj o poprawnej walidacji modeli i obsłudze błędów.
- W poprawny sposób wykorzystuj metody protokołu HTTP.
- Przygotuj odpowiednie modele DTO (request, response).
- Przygotuj końcówki opisane w poniższych wymaganiach.
- Dodaj automatycznie generowaną dokumentację API z pomocą Swagger'a
- Przygotuj osobny projekt z testami jednostkowymi i integracyjnymi dla przygotowanych końcówek.

4 Końcówka do rejestracji użytkowników

Końcówka do rejestracji nowych użytkowników. Pamiętaj o zaimplementowaniu w poprawny sposób mechanizmu zabezpieczania hasła (hash i solenie). Po poprawnej rejestracji zwróć dane na na temat utworzonego użytkownika i kod 201.

Przykład żądania:

```
POST /api/clients HTTP/1.1
Host: localhost:54965
Content-Type: application/json

{
"FirstName": "John",
"LastName": "Kowalski",
"Email": "kowalski@wp.pl",
"Phone": "454-232-222"
"Login": "Jan125",
"Password": "asd124"
}
```

5 Końcówka do odświeżania access token'u

Przygotuj końcówkę, która przyjmie refresh token klienta i na tej podstawie wyda nowy access token i refresh token.

6 Końcówka do logowania

Przygotuj końcówkę, która pozwoli się zalogować użytkownikowi i uzyskać access token i refresh token.

Przykład żądania:

```
POST /api/clients/login HTTP/1.1
Host: localhost:54965
Content-Type: application/json

{
"Login": "asd1245",
"Password": "AlaMaKota"
}
```

7 Lista kampanii

Przygotuj końcówkę dostępną tylko dla zalogowanych użytkowników, która pozwoli zwrócić listę kampanii wraz z danymi klienta i reklamami związanymi z kampanią. Kampanie powinny być posortowane w kolejności malejącej po dacie rozpoczęcia kampanii.

8 Tworzenie nowej kampanii

Końcówka pozwalająca na tworzenie nowych kampanii zawiera dodatkową logikę biznesową. Poniżej zaprezentowane jest przykładowe żądanie zawierające wszystkie niezbędne informacje wysyłane przez klienta.

8.1 Scenariusz przypadku użycia

- Aktorzy: Klient API
- Warunki początkowe: W bazie danych przechowujemy informacje o budynkach z danego miasta. Klient wykonujący żądania został wcześniej zarejestrowany. Jego dane znajdują się w bazie danych.
- Warunki końcowe: Informacje o nowej kampanii zostają zapisane w bazie danych.

Główny scenariusz:

1. Klient przesyła wszystkie wymagane dane. Przykład poniżej.
2. API sprawdza czy klient wybrał dwa budynki na których będzie rozwieszony baner reklamowy. Budynki muszą znajdować się na tej samej ulicy. W przeciwnym wypadku zwracamy kod 400.
3. Następnie obliczamy koszt reklamy. Banery reklamowe mają zostać rozwieszone między dwoma budynkami.
4. Każda kampania reklamowa dla klienta obejmuje dwa banery. Banery mają kształt prostokąta. Jako firma chcemy zminimalizować koszt wydruku baneru, więc staramy się wyliczyć zestaw dwóch banerów, które pokrywają przestrzeń między dwoma budynkami i obejmują jak najmniejszą przestrzeń (chcemy zminimalizować zużycie materiału). Nasza końcówka powinna postarać się znaleźć taki układ banerów, aby ich powierzchnia byłą jak najmniejsza i równocześnie, aby oba banery pokrywały wymaganą przestrzeń.
5. Następnie dodajemy dane do bazy danych.
6. Zwracamy nowo utworzony obiekt reprezentujący kampanię wraz z danymi na temat dwóch reklam. Całość powinna być opakowana w kod zwrotny 201.

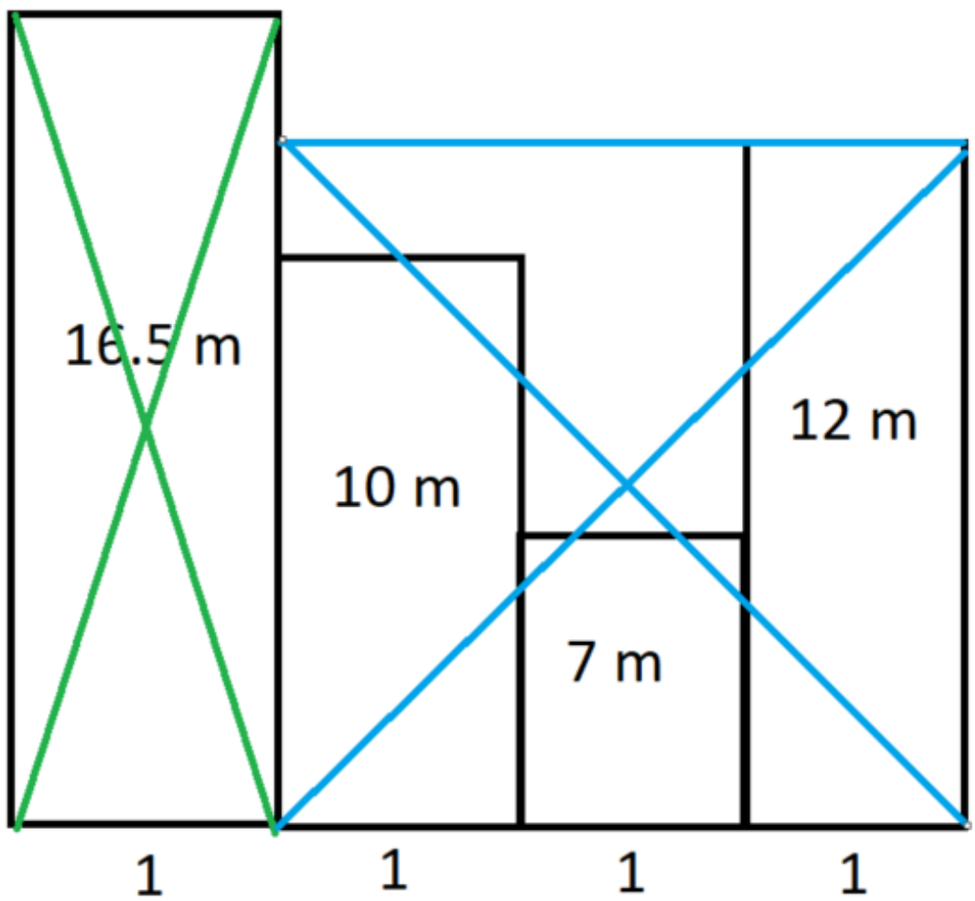
8.2 Przykład graficzny

1. Klient wysyła żądanie w postaci.

```
{
"IdClient": 1,
"StartDate": "2020-1-1",
```

```
"EndDate": "2020-3-1",
"PricePerSquareMeter": 35,
"FromIdBuilding": 1,
"ToIdBuilding": 4
}
```


2. Następnie aplikacja serwerowa waliduje żądanie. Następnie odpowiedni serwis oblicza powierzchnię dwóch reklam. Załóżmy, że budynki z id 1 i 4 leżą na tej samej ulicy. Załóżmy, że ich wysokość jest podobno do poniższego obrazka. Dla uproszczenia zakładamy, że szerokość wszystkich budynków jest taka sama - 1.



Na powyższym obrazku widzimy 4 budynki umiejscowione obok siebie. Każdy z nich ma określoną wysokość. Zielone i niebieskie linie określają dwa baneru, które wyliczyliśmy. Pamiętajmy, że chcemy przygotować zawsze 2 banery i muszą one pokrywać całe budynki. Mogę "wystawać"poza budynek. Chcemy, żeby równocześnie ich powierzchnia była jak najmniejsza. W tym wypadku:

- Baner 1 = 16.5 * 1 = 16.5 m$\hat{2}$
- Baner 2 = 12 * 3 = 36 m$\hat{2}$

Cenę końcową wyliczamy w następujący sposób:

- TotalPrice = (Baner1 + Baner2) * PricePerSquareMeter= (16.5 + 36) * 35 = 1837.5

9 Wymagania niefunkcjonalne

- Postaraj się, aby serwis umożliwiający obliczanie najbardziej optymalnej powierzchni przestrzeni reklamowej (minimalizowane powierzchni) miał możliwie jak najmniejszą złożoność obliczeniową i pamięciową.

10 Warunki zaliczenia

Staramy się zaimplementować jak największą liczbę wymagań. Nawet jeśli nie udało się zrealizować danej funkcjonalności w pełni - warto aby umieścić taki kod.