

Przetwarzanie danych w chmurach obliczeniowych

Dokumentacja Projektu

Adam Łaba

21 grudnia 2022

Tematyka projektu

Tematyką projektu jest platforma służąca do ocenia albumów muzycznych, z wbudowaną funkcją rekomendacji albumów na podstawie ocen innych użytkowników. Serwis wykorzystuje do działania grafową bazę danych neo4j.

Linki do projektu

Repozytorium z kodem: <https://github.com/unlvy/album-recommender>

Aplikacja: <https://album-recommender.onrender.com/>

Funkcjonalności serwisu

Serwis pozwala użytkownikowi na:

- rejestrację,
- logowanie,
- przeglądanie i ocenianie albumów,
- dodanie nowego albumu,
- przeglądanie i polubienie gatunków muzycznych,
- dodanie nowego gatunku,
- przeglądanie i polubienie wykonawców,
- dodanie nowego wykonawcy

Aplikacja klienta

Interfejs użytkownika jest tworzony przez aplikację napisaną w Angularze, dodatkowo wykorzystującą bibliotekę gotowych komponentów Angular Material UI.

Aplikacja wita odwiedzających użytkowników ekranem logowania. W górnej części ekranu znajduje się pasek służący do nawigacji, na którym na ten moment znajduje się jedynie przycisk przenoszący do ekranu logowania:

Album recommender sign in

Sign in

[Not an user? Register](#)

Udostępnia także możliwość rejestracji nowego użytkownika:

Album recommender sign in

Register

[Already an user? Sign in](#)

Po pomyślnym zalogowaniu użytkownik jest przenoszony na widok z albumami - może tutaj wyświetlić informacje o wszystkich dostępnych w bazie albumach, ich ocenach, a także może oceniać albumy i dodać nowy album do bazy. Dodatkowo, na pasku nawigacji pojawiło się kilka przycisków dostępnych jedynie dla zalogowanego użytkownika: albumy, gatunki, wykonawcy, a także przycisk służący do wylogowania:

Invaders Must Die

Prodigy, 2009

46:01, 11 tracks
Genres: Electronic, Big beat

Average user rating: 3
Your rating: 3 [Change](#)

The Fat of the Land

Prodigy, 1997

56:21, 10 tracks
Genres: Big beat, Electronic

Average user rating: 3.5
Your rating: Rate

Add new album

Name*

Artist*

Genres*

Year*

Number of tracks*

Album length
h* m* s*

Image URL*

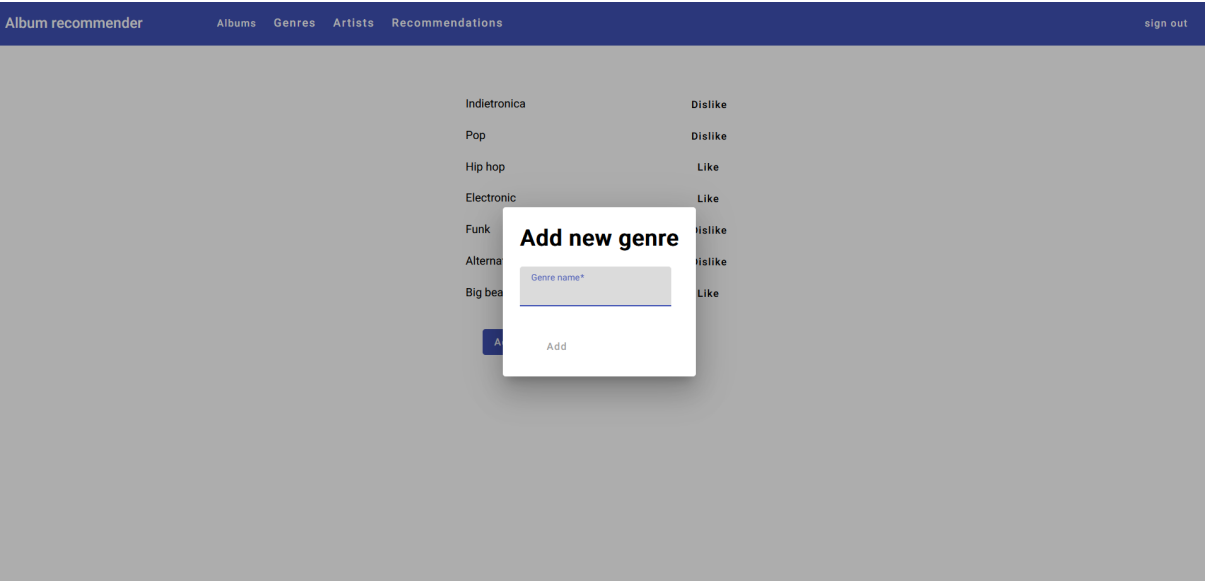
Add

Add new album

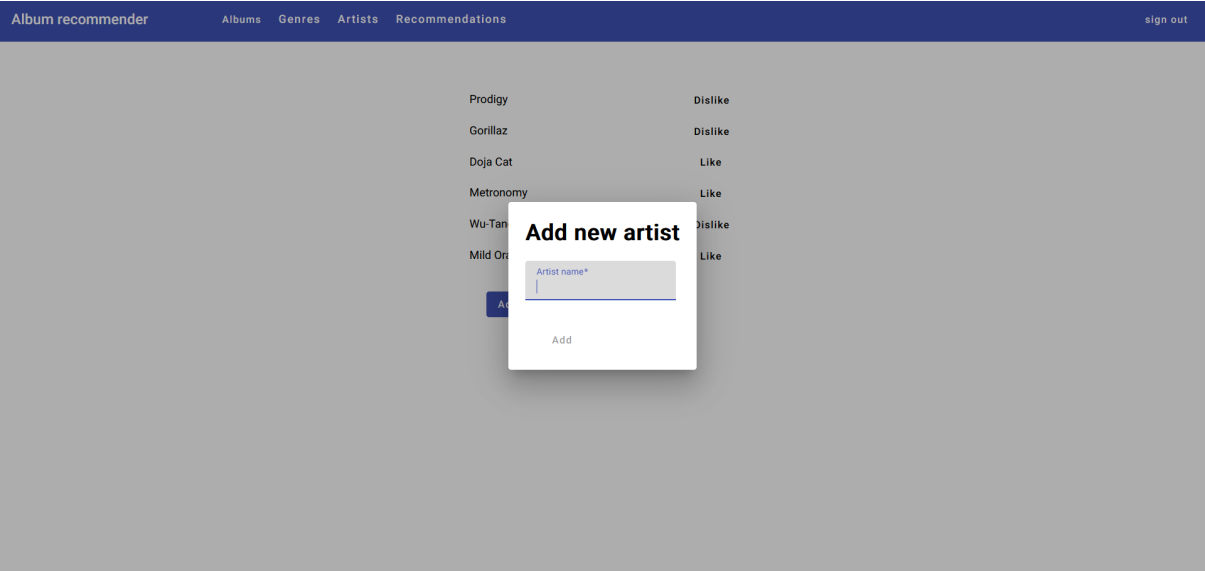
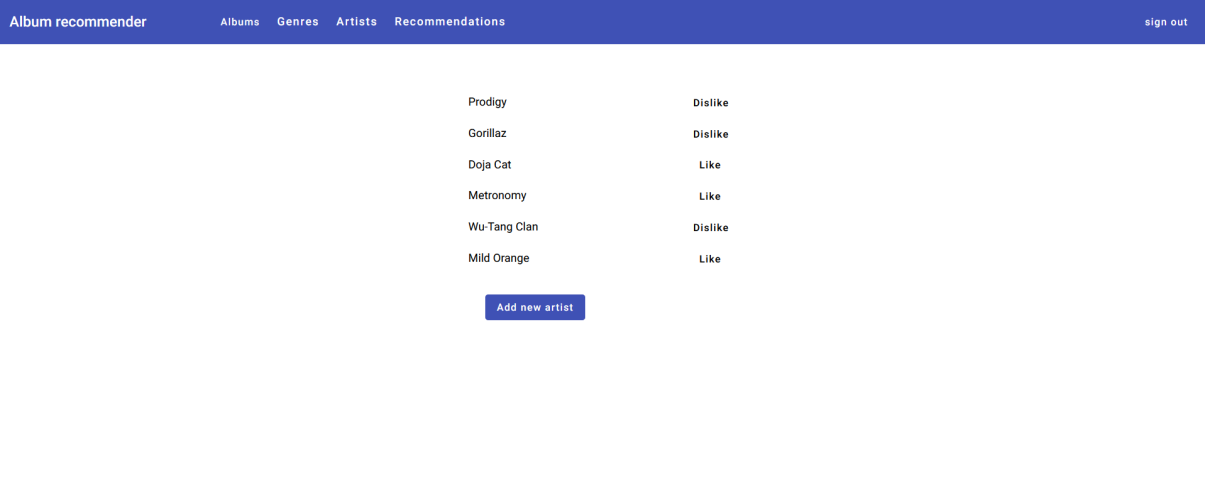
Widok zawierający listę gatunków pozwala na polubienie/odlubienie konkretnego gatunku, a także dodanie nowego:

Prodigy	Dislike
Gorillaz	Dislike
Doja Cat	Like
Metronomy	Like
Wu-Tang Clan	Dislike
Mild Orange	Like

Add new artist




Widok zawierający listę wykonawców pozwala na polubienie/odlubienie konkretnego wykonawcy, a także dodanie nowego:



Ostatnim widok prezentuje użytkownikowi polecane dla niego albumy:

[Album recommender](#) [Albums](#) [Genres](#) [Artists](#) [Recommendations](#) [sign out](#)



Wu-Tang Forever
Wu-Tang Clan, 1997

44:58, 11 tracks
Genres: Hip hop, Hip hop, Hip hop
Recommendation score: 0.2

Aplikacja serwera

Aplikacja serwera została napisana w języku TypeScript z wykorzystaniem biblioteki Express. Serwer wystawia REST API, za pomocą którego można wyświetlać i modyfikować węzły i relacje istniejące w bazie danych.

Serwer korzysta z kilku kontrolerów, każdy z nich obsługuje konkretną podkategorię zapytań. Endpointy serwera to:

Użytkownicy:

- `POST /auth/login` - logowanie
- `POST /auth/register` - rejestracja

Albumy:

- `GET /albums` - lista wszystkich albumów i ich średnich ocen
- `GET /albums/:userID` - oceny albumów wystawione przez konkretnego użytkownika
- `POST /albums/rate` - ocenianie albumu
- `POST /albums/change-rating` - zmiana oceny albumu
- `POST /albums/new` - dodanie nowego albumu

Gatunki:

- `GET /genres` - lista wszystkich gatunków
- `GET /genres/:userID` - gatunki polubione przez danego użytkownika
- `POST /genres/like` - polubienie gatunku
- `POST /genres/dislike` - odlubienie gatunku
- `POST /genres/new` - dodanie nowego gatunku

Wykonawcy:

- `GET /artists` - lista wszystkich wykonawców
- `GET /artists/:userID` - lista wykonawców polubionych przez konkretnego użytkownika

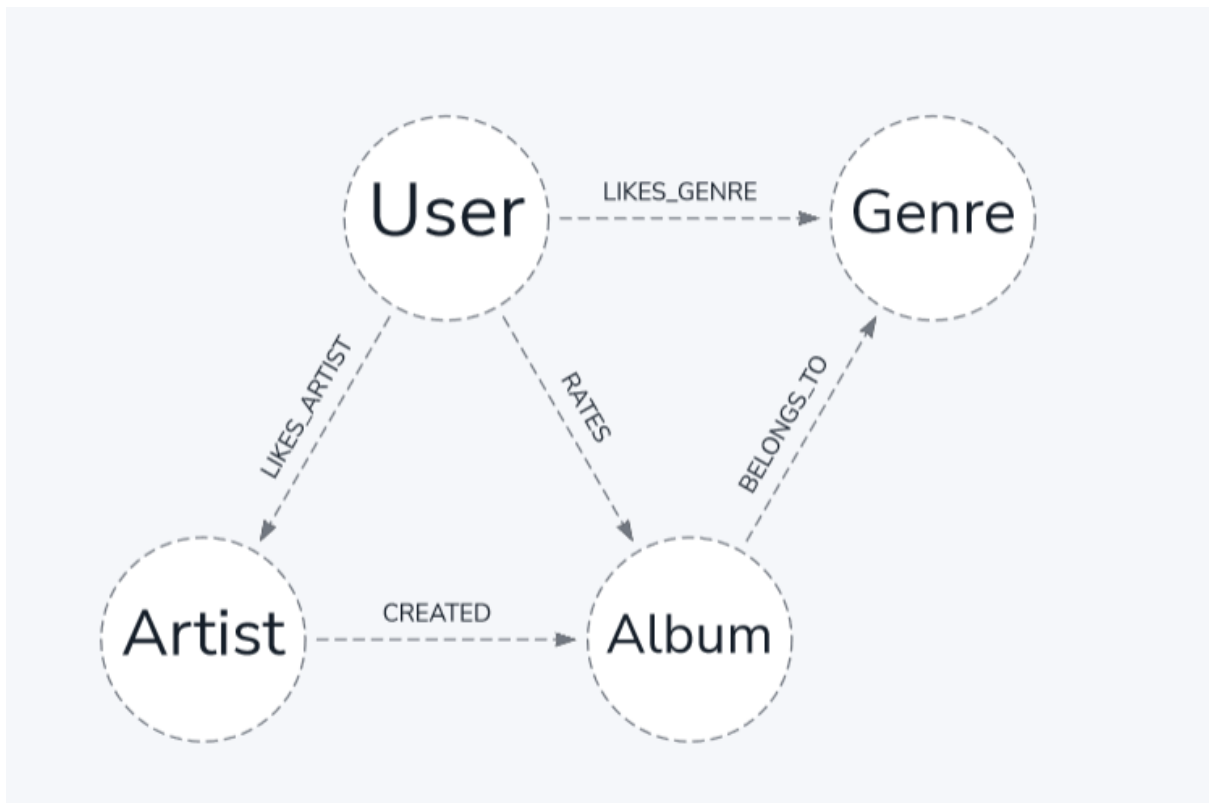
- POST /artists/like - polubienie wykonawcy
- POST /artists/dislike - odlubienie wykonawcy
- POST /artists/new - dodanie nowego wykonawcy

Rekomendacje:

- GET /recommendations/albums/:userID - albumy polecane dla konkretnego użytkownika

Baza danych

Struktura węzłów i relacji istniejących w bazie danych:



Węzły:

User:

```

id: string,
username: string,
password: string
  
```

Genre:

```

id: string,
name: string
  
```

Album:

```

id: string,
year: integer,
  
```

```
length: datetime,  
numTracks: integer,  
imageUrl: string,  
name: string
```

```
Artist:  
  id: string,  
  name: string
```

```
Relacje:  
(User->Genre)  
LIKES_GENRE:  
  id: string
```

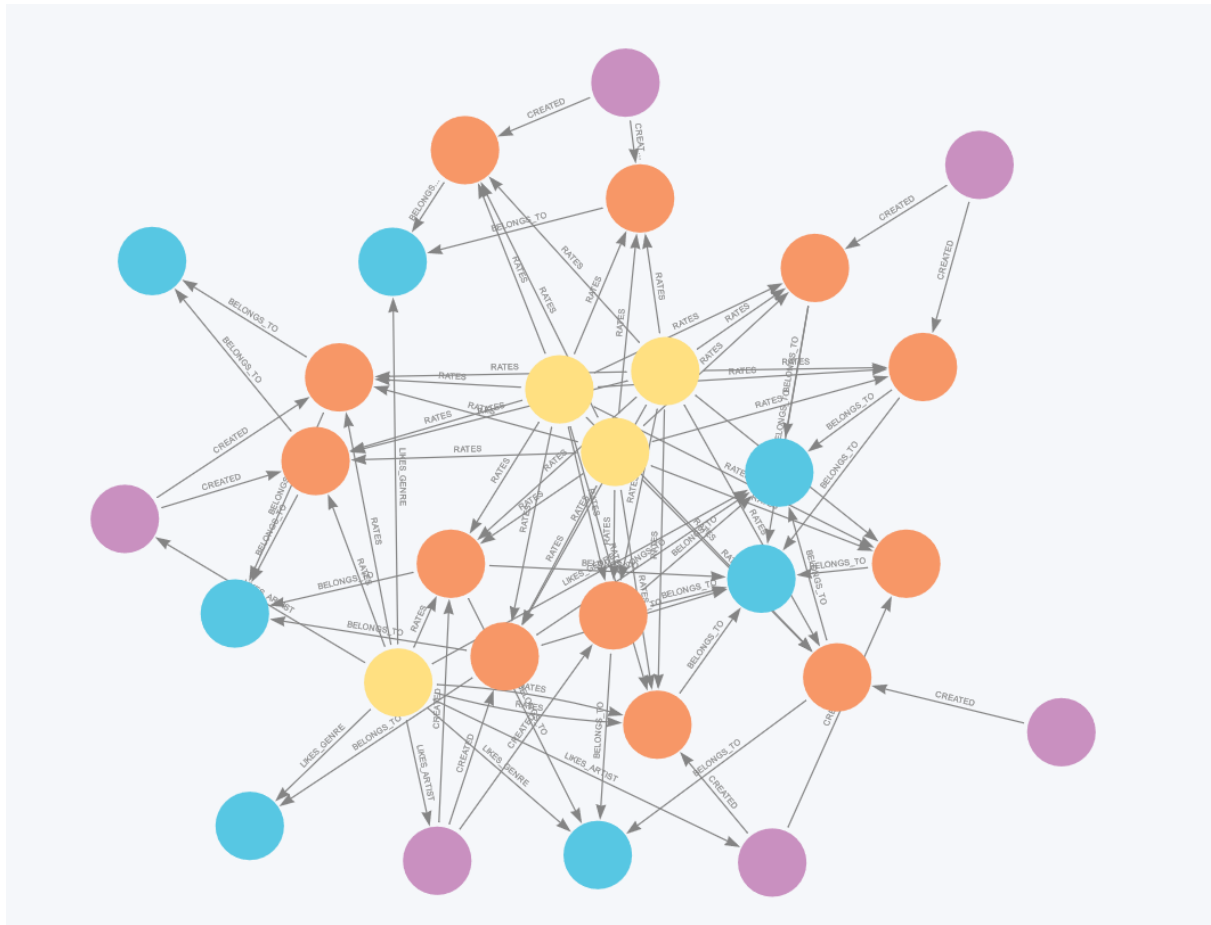
```
(User->Artist)  
LIKES_ARTIST:  
  id: string
```

```
(User->Album)  
RATES:  
  id: string,  
  rating: integer
```

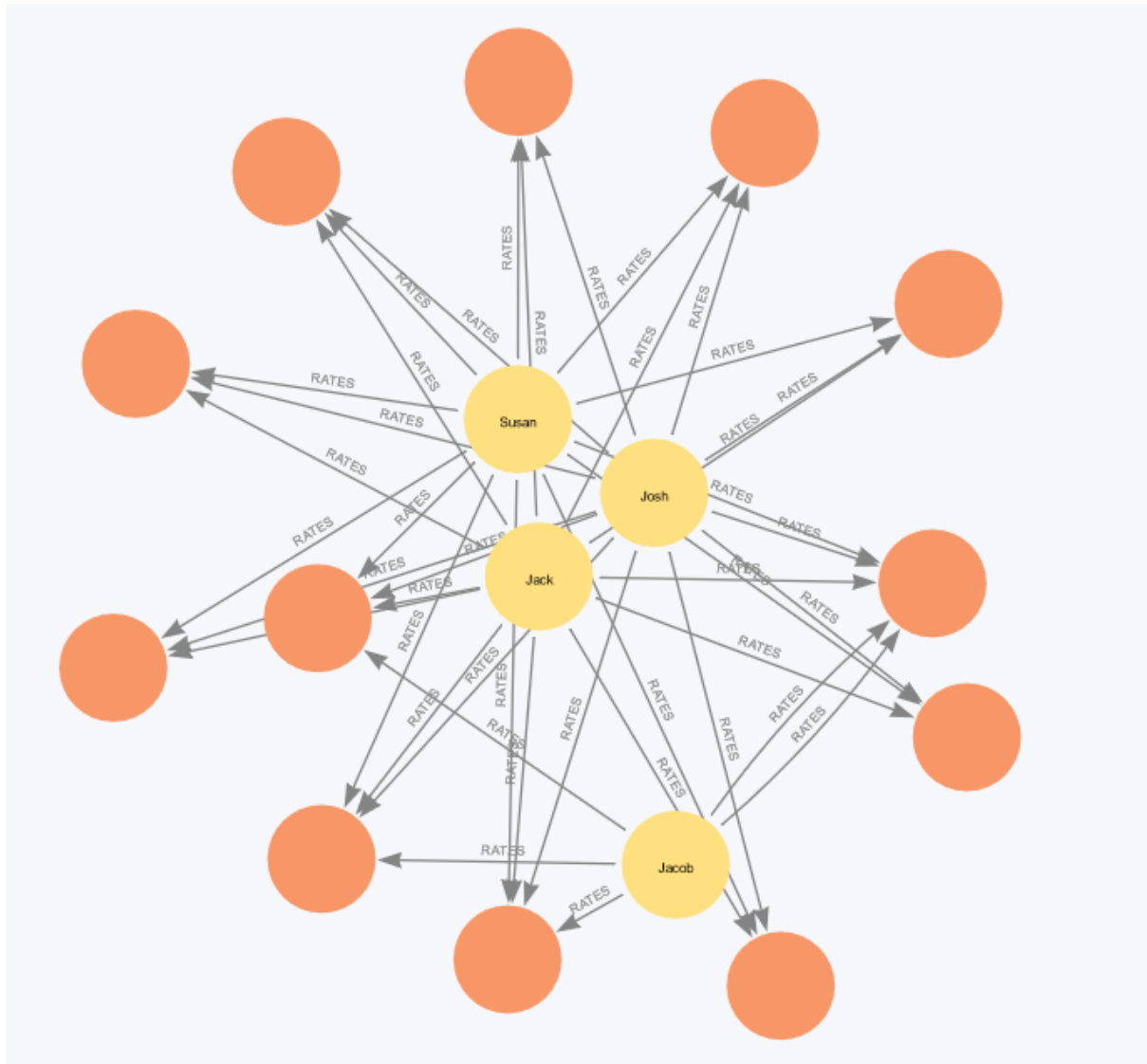
```
(Artist->Album)  
CREATED:  
  id: string
```

```
(Album->Genre)  
BELONGS_TO:  
  id: string
```

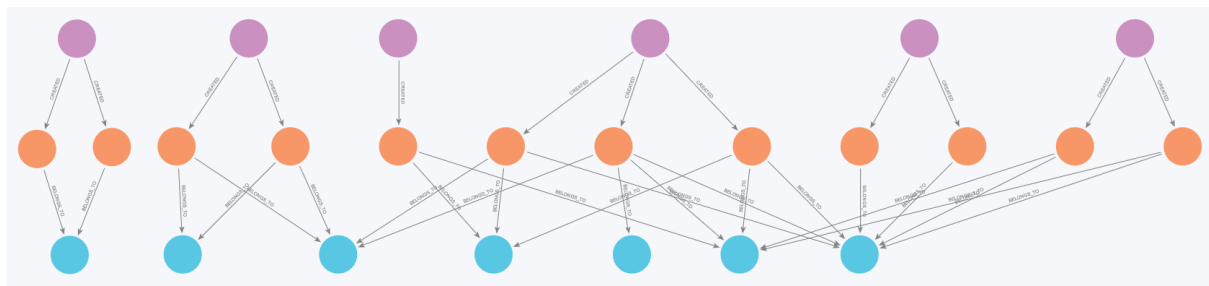
Wszystkie węzły i relacje obecne w bazie danych:



Użytkownicy i polubione przez nich albumy:



Wykonawcy, ich albumy oraz przynależność albumów do gatunków



Rekomendacja albumów

Rekomendacje albumów dla użytkownika opierają się o wykorzystanie metryki podobieństwa preferencji użytkowników. Jeśli dwóch użytkowników wystawi podobne oceny dla konkretnych albumów, to prawdopodobnie mają podobne upodobania muzyczne i słuchają podobnej muzyki, więc jeśli któryś z nich ocenia wysoko album nieoceniony jeszcze przez drugiego, to można ten album zarekomendować drugiemu z tych użytkowników.

Do rekomendacji wykorzystano współczynnik korelacji Pearsona, widoczny na poniższym wzorze.

$$r_{xy} = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^n (y_i - \bar{y})^2}}$$

Wdrożenie aplikacji

Aplikacja klienta i aplikacja serwerowa zostały zamieszczone na platformie Render. Baza danych działa na platformie Aura DB.