

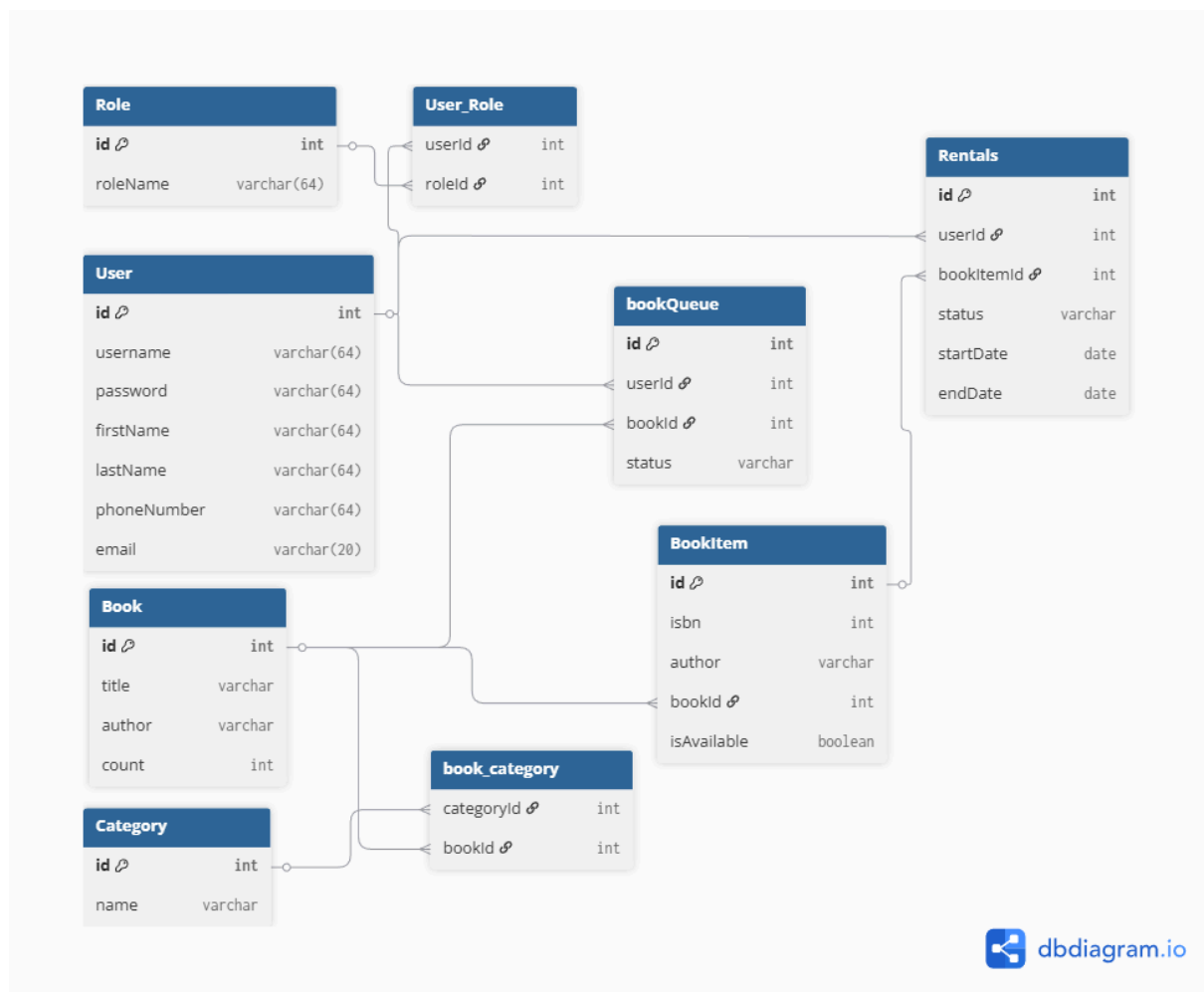
Technologie Obiektowe - Projekt

Skład zespołu

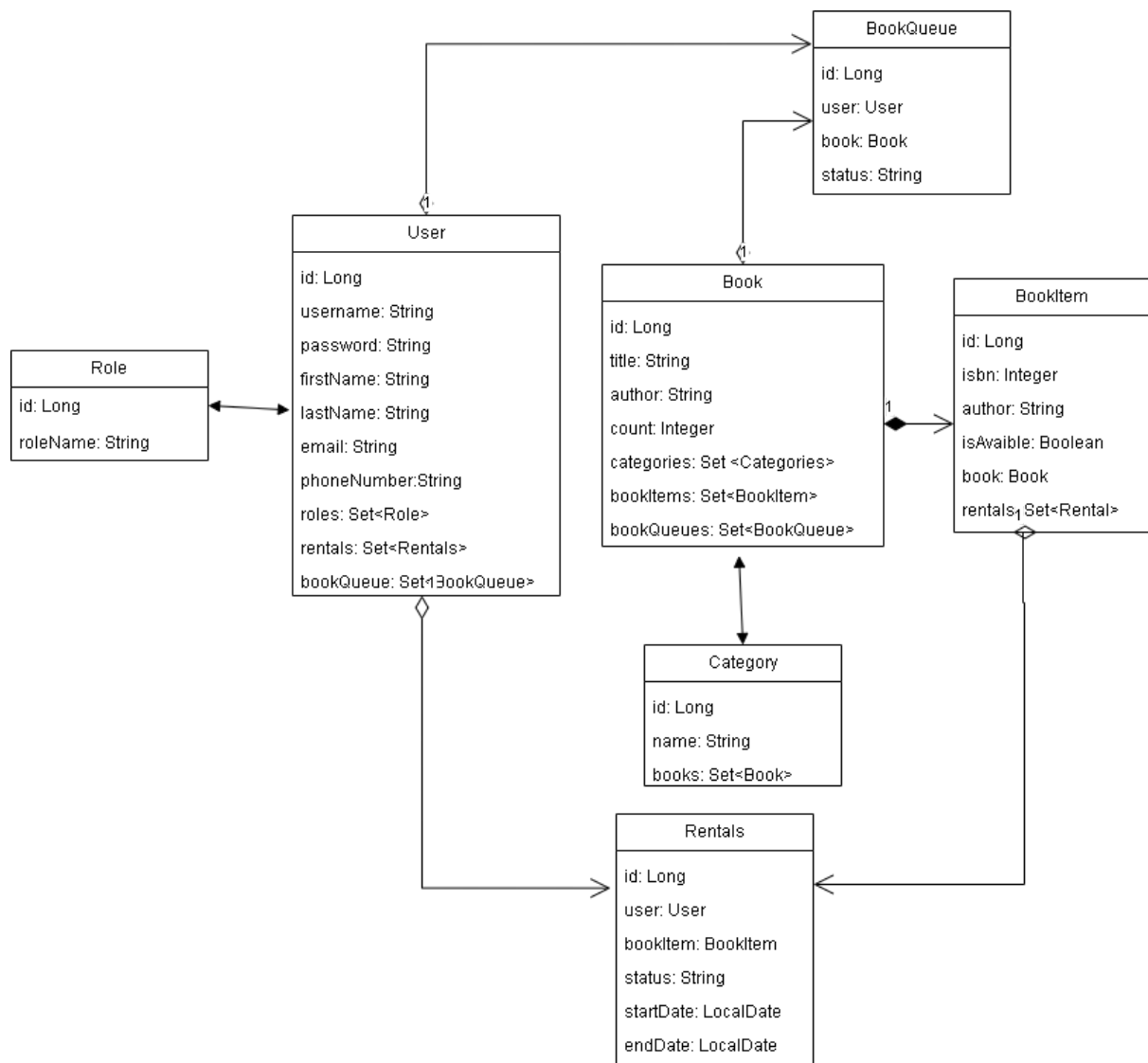
Maciej Sawiec, Bartosz Wójcik, Grzegorz Rybiński, Konrad Szymański

Pierwszy kamień milowy

Model bazodanowy aplikacji



Model obiektowy aplikacji



Wykorzystane technologie

Aplikacja została zbudowana z wykorzystaniem frameworka Spring Boot. Do zarządzania persystencją użyto JPA. Bezpieczeństwo - w tym autentykacja i autoryzacja - zostały zaimplementowane przy pomocy Spring Security oraz JSON Web Token. Warstwa frontendowa została napisana w środowisku React.

User

W związku z zarządzaniem tabelą użytkowników napisano klasy:

- User - jest to modelowa klasa odwzorowująca użytkowników w bazie danych.
- UserService - serwis odpowiedzialny za logikę biznesową dotyczącą zarządzania użytkownikami - w tym walidację.
- UserRepository - interface JPA zapewniający komunikację z bazą danych.
- Role - klasa modelowa POJO reprezentująca role należące do użytkowników.
- UserController - kontroler REST udostępniający endpointy służące do komunikacji z warstwą frontendową.

UserService

Klasa zapewnia metody pozwalające dodawać, usuwać oraz edytować użytkowników. Ponadto pozwala na modyfikację ról przypisanych do użytkowników. Klasa implementuje również metodę `loadUserByUsername` wymaganą do integracji serwisu z SpringSecurity. Przy dodawaniu użytkowników hasła są kodowane przy użyciu `BCryptPasswordEncoder` co zwiększa bezpieczeństwo aplikacji i chroni dane w przypadku potencjalnego wycieku.

UserController

Kontroler zapewnia endpointy do:

- rejestracji - nowy użytkownik jest dodawany do bazy.
- logowania - użytkownik jest uwierzytelniany - w przypadku sukcesu zwracany jest JWT.
- Odczytanie, edytowanie lub usuwanie użytkownika po id. Użytkownik posiada dostęp wyłącznie do swoich danych, chyba że posiada rolę administratora.
- Pobranie wszystkich użytkowników. Dostępne tylko dla administratora.

Security

Klasa `JwtUtil` pomaga w obsłudze JWT, udostępniając metody do tworzenia tokenów, ich weryfikacji oraz wydobywania nazwy użytkownika z tokenu.

Klasa `AuthTokenFilter` odpowiada za przechwytywanie tokenów i sprawdzanie ich poprawności.

Klasa `SecurityConfig` odpowiada za utworzenie `Security Filter Chain` – sekwencji filtrowania zapytań w Spring Security. Łączy powyższe klasy, określa które endpointy są ogólnie dostępne (logowanie i rejestracja), oraz konfiguruje mechanizm `Cross-Origin Resource Sharing`, umożliwiającą komunikację między backendem a frontendem.