Andrzej Gębuś

Nr. Indeksu 106439

System zarządzania biblioteką

**Projekt zakończony**

Github: <https://github.com/eendrzmail/BD_2020_Gebus_Andrzej>

**Linki:**

Film z prezentacją aplikacji: <https://drive.google.com/file/d/1TSUk0roIByXYjsmoSnlr6L99NpeKdDUO/view?usp=sharing>

Maszyna Wirtualna:

<https://drive.google.com/file/d/1qNq9RnZ0yblp97RmRuux-lRfTvL-7QPV/view?usp=sharing>

Aplikacja desktopowa:

<https://drive.google.com/file/d/1J9jDxq739OyGWBpDdplYv5vy5RUv5pyK/view?usp=sharing>

Cały folder:

<https://drive.google.com/drive/folders/1b6V_bHL7TFC3xjDRjACEp9J8R71-ixuo?usp=sharing>

**Instrukcje uruchomienia na istniejącej maszynie**:

1. Pobrać maszynę wirtualną z linku dostępnego powyżej lub w repozytorium.
2. Zalogować się do maszyny wirtualnej danymi: login: student hasło: student
3. Włączyć xampp Control Panel (skrót na pulpicie) oraz uruchomić serwer Apache.
4. Interfejs graficzny znajduje się pod adresem przeglądarki internetowej localhost lub aplikacji desktopowej (skrót na pulpicie).

**Instrukcje uruchomienia na nowej maszynie**:

1. Zainstalować bazę danych Oracle.
2. Zaimportować bazę danych (plik do importu znajduje się w repozytorium: /sql/baza.sql)

(nazwa bazy danych: student, hostowana pod nazwą kosmos)

1. Zainstalować serwer xampp (<https://www.apachefriends.org/pl/index.html>)
2. (Jeśli nie zainstalowano z bazą danych) Zainstalować Oracle Instant Client
3. Odkomentować wtyczkę do Oracle w pliku konfiguracyjnym php:

• przejść do pliku xampp/php/php.ini

• w linii 922 odkomentować dodatek(usunąć ; z ;extension=oci8\_12c)

1. Przekopiować wszystkie pliki z repozytorium z katalogu /gui do folderu xampp/htdocs/ (usunąć instniejące)
2. Jeśli dane logowania do bazy danych różnią się od oryginalnych skonfigurować je w pliku /htdocs/api/conn.php.
3. Uruchomić xampp oraz serwer Apache
4. Interfejs graficzny znajduje się w przeglądarce pod adresem localhost

(Opcjonalne)

• Pobrać wersję desktopową aplikacji z linku dostępnego powyżej lub w repozytorium.

(Opcjonalne)

Aby samemu skompilować kod źródłowy należy:

1. Zainstalować środowisko Nodejs.
2. Zainstalować globalnie Angular cli: npm install -g @angular/cli
3. Pobrać kod źródłowy z repozytorium z katalogu /source\_code/
4. Zainstalować pakiet modułów (w folderze z kodem źródłowym): *npm install*
5. Zainstalować pakiet Angular Material (w folderze z kodem źródłowym): *ng add @angular/material*
6. (Opcjonalne) Serwer testowy można uruchomić za pomocą komendy *ng serve* (adres serwera testowego: *localhost:4200*)
7. (Opcjonalne) Aby skompilować projekt należy wykonać komendę *ng build (zbudowany projekt znajduje się w katalogu /dist/)*

# **Co znajduje się w repozytorium**

- katalog sql – zawiera pliki sql do importu bazy danych: baza.sql – całość bazy danych

tabele.sql – tabele bazy danych

insert.sql – dane bazy danych

funkcjonalności.sql – pl/sql

- katalog gui – Zawiera całą aplikację którą wystarczy wrzucić do katalogu htdocs w xampp

- katalog pass – zawiera plik z loginami i hasłami

- katalog source\_code – zawiera kod źródłowy z frameworka angular aplikacji końcowej

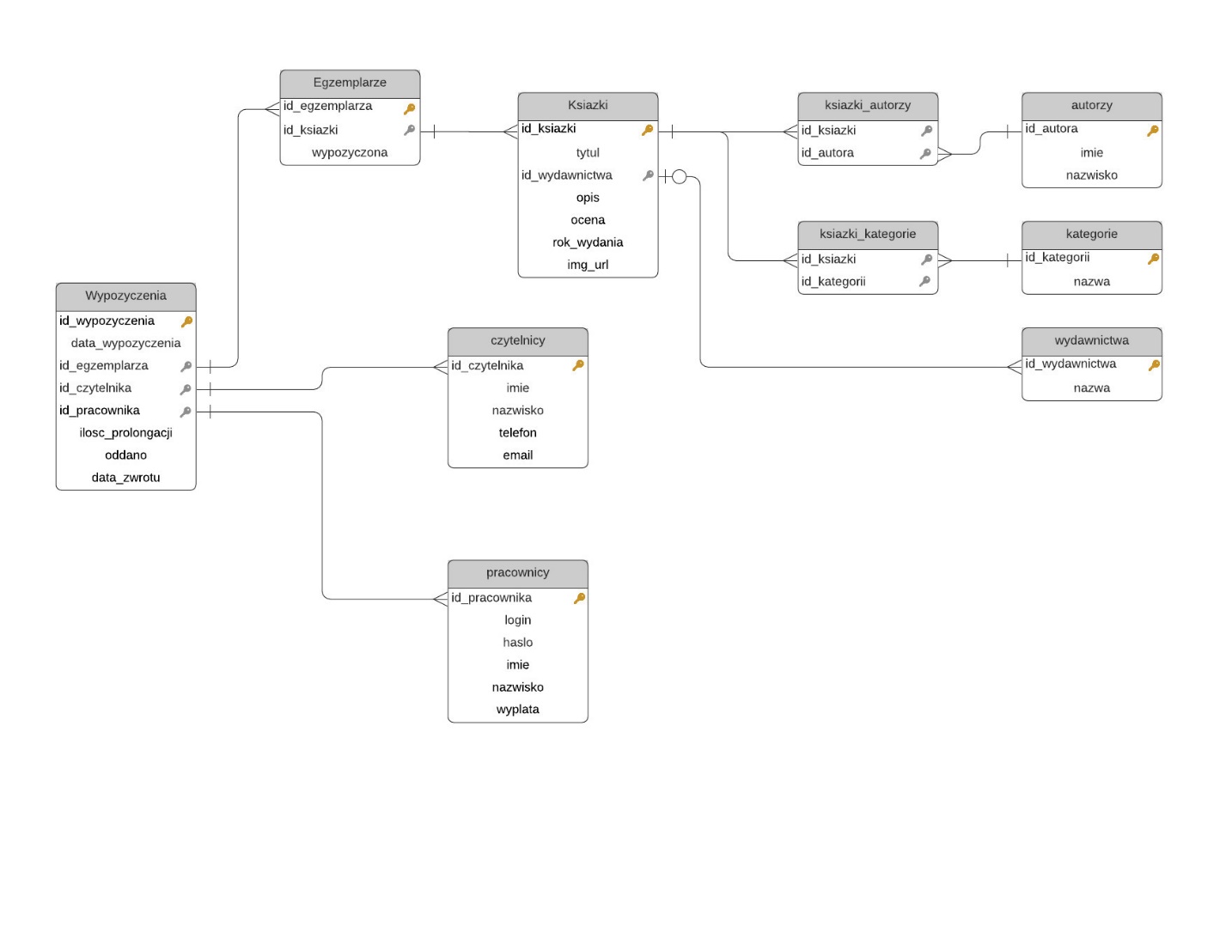
- katalog docs – zawiera dokumenty opisujące projekt oraz folder ze screenshotami aplikacji

# **Wstępny opis projektu**

Utworzenie aplikacji do obsługi biblioteki (dla pracowników) za pomocą następujących narzędzi:

* Angular (frontend)
* PHP (backend php łączące się z oracledb i wystawiające dane na zewnątrz w formie prostego rest API)
* OracleDB

Baza danych wykonana została na podstawie diagramu erd:



(Większa wersja w repozytorium githuba)

# **Założenia bazy danych:**

Książka:

* może mieć kilku autorów (tabela łącząca)
* może mieć kilka kategorii
* może mieć tylko jedno wydawnictwo
* może istnieć wiele egzemplarzy jednej książki

Wypożyczenia:

* Pracownik filii może na życzenie klienta prolongować daną książkę po jej wypożyczeniu na kolejne 7dni

Pracownicy:

* każdy pracownik ma login oraz hasło którym loguje się do systemu. Po zalogowaniu może dodawać oraz edytować wypożyczenia oraz książki

# **Przedstawienie projektu:**

## ***Frontend***

Aplikacja końcowa Została napisana na frameworku Angular. Podzielona jest na komponenty, serwisy oraz modele danych.

*Komponenty* odpowiedzialne są za wyświetlanie danych. Stosują wzorzec projektowy MVC. W kodzie źródłowym można to zauważyć po poszczególnych plikach:

.html – Odpowiada za wyświetlanie danych

.css – Odpowiada za wygląd wyświetlanych danych oraz wygląd całego komponentu

.ts – Kontroler. Odpowiada za obsługę np. przycisków oraz odbieranie danych z serwisów.

*Modele danych* to klasy odpowiadające tabelom z baz danych.

*Serwisy* służą do wysyłania oraz odbierania zapytań od serwera php oraz przesyłania otrzymanych danych do komponentów.

### Przykład działania:

Komponent ksiazka w konstruktorze importuje klasę route oraz serwis bookservice.

Po zainicjowaniu elementu pobiera parametr id z aktywnej ścieżki, po czym wywołuje funkcję getbook z pobranym wcześniej parametrem:

|  |
| --- |
| getbook(id:string){ |
|  | this.booksservice.getbook(id).subscribe((ret)=>{ |
|  | this.book=ret; |
|  | console.dir(this.book); |
|  | }) |
|  | } |

Funkcja wywołuje odpowiednią metodę z serwisu (getbook), po czym przypisuje dane do swojego pola book. Subscribe jest obiektem który reprezentuje przepływ danych. Oznacza to, że komponent wyświetli się od razu, a dane doczytają się asynchronicznie.

Metoda getbook w serwisie bookservice:

getbook(id:string):Observable<Ksiazka>{

const params=new HttpParams().set('id',id);

return this.http.get<Ksiazka>('http://localhost/api/ksiazki.php',{params});

}

Metoda tworzy parametr http z otrzymanych danych z argumentu, po czym za pomocą klienta http wysyła zapytanie do backendu oraz zwraca otrzymane dane w postaci klasy Ksiazka.

Następnie w widoku komponentu zostają wypisane dane przypisane do pola book oraz wyświetlone w odpowiednim miejscu.

Cała aplikacja końcowa dostępna jest również jako aplikacja desktopowa. Link do niej znajduje się na początku dokumentacji. Oczywiście do działania wymaga włączonego serwera apache oraz bazy danych.

## ***Backend***

Backend złożony jest z kilku ‘endpointów’, czyli adresów url do których odwołuje się aplikacja końcowa w celu pobrania danych z bazy danych. Każdy z poszczególnych endpointów wymaga podania konkretnego parametru http GET lub POST w celu zwrócenia odpowiednich danych. Całość backendu zlokalizowana jest w katalogu xampp/htdocs/api.

### Przykład działania:

Plik egzemplarze.php sprawdza czy jest podany parametr id

if (isset($\_GET['id'])) {

$q = 'select \* from egzemplarze where ID\_EGZEMPLARZA='.$\_GET['id'];

if ($conn){

$result= oci\_parse($conn,$q);

oci\_execute($result);

while($row = oci\_fetch\_array($result, OCI\_ASSOC+OCI\_RETURN\_NULLS)){

$temp=array(

"id\_egzemplarza"=>$row['ID\_EGZEMPLARZA'],

"ksiazka"=>json\_decode(file\_get\_contents("http://localhost/api/ksiazki.php?id=".$row['ID\_KSIAZKI'])),

"wypozyczona"=>$row['WYPOZYCZONA']

);

}

echo json\_encode($temp);

}

}

Jeśli tak, wysyła zapytanie do bazy danych o wybranie rekordu o id egzemplarza odpowiadającym podanemu. Następnie tworzy tymczasową tablicę asocjacyjną o polach odpowiadających bazie danych. Zamiast jednak trzymać samo id książki, wysyła zapytanie do endpointu książki.php o pobranie książki o id wczytanym z bazy danych. W ten sposób backend może zwrócić wszystkie informacje: sam egzemplarz, oraz książkę której podlega w tym samym zapytaniu. Dane z tymczasowej tabeli wystawiane są ‘na zewnątrz’ w formacie JSON.

W podobny sposób wykonane są wszystkie endpointy.

Opis wszystkich endpointów znajduje się w repozytorium w katalogu /docs/endpoints.md

## **Baza danych**

Baza danych jest zbudowana tak jak jest to pokazane w diagramie erd powyżej oraz w repozytorium.

Dodane są dwa widoki ułatwiające wyciąganie danych z tabeli łączącej.

Baza zawiera również następujące konstrukcje pl/sql:

- Sekwencje dla każdej tabeli inkrementujące wartość.

- Triggery odpowiadające za generowanie daty oddania z uwzględnieniem ilości prolongacji oraz za generowanie loginu użytkowników na podstawie dwóch pierwszych liter imienia i dwóch ostatnich nazwiska.

- Package z funkcją zwracającą top książek pod względem ocen

- Funkcję zwracającą informacje o zarobkach: sumę, medianę, średnią, odchylenie

# **Prezentacja projektu:**

Film z omówieniem aplikacji oraz pokazem funkcjnalności znajduje się pod linkiem:

<https://drive.google.com/file/d/1TSUk0roIByXYjsmoSnlr6L99NpeKdDUO/view?usp=sharing>

Screenshoty z aplikacji:

