**Niedokończona!!!!**

1. **Wstęp**

Aplikacja służy do wyświetlania, wyszukiwania, sortowania, dodawania i usuwania danych o książkach z bazy danych *ksiegozbior*. Dane książek użytkownik może wprowadzić samodzielnie lub skorzystać z dodawania poprzez ISBN – dane są wtedy pobierane z bazy danych Biblioteki Narodowej.

1. **Instrukcja użytkowania**

Strona główna

Jako pierwsza pokazuje się strona główna, która wyświetla wszystkie książki z bazy danych. Obok danych książki znajduje się przycisk „Usuń”.



Dane można sortować (rosnąco lub malejąco: po tytule, po autorze, po roku wydania) za pomocą komponentu po lewej stronie. Sortowanie bierze pod uwagę wyszukaną frazę.

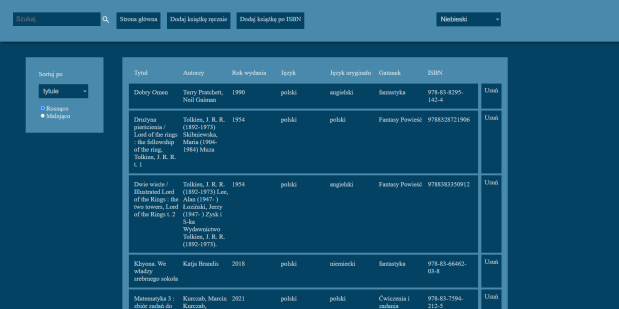


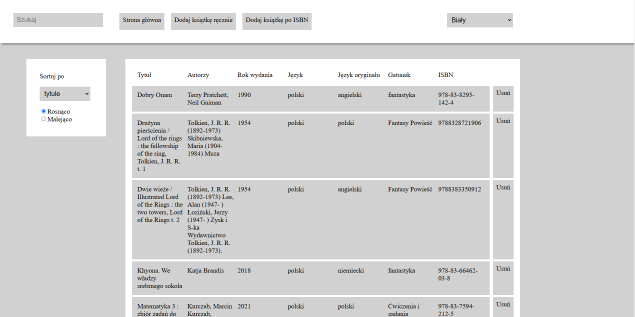
Menu

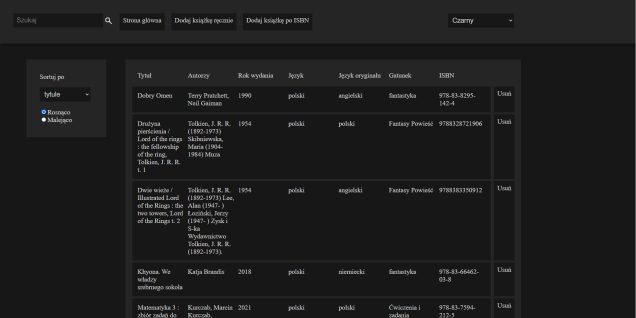


W menu znajduje się:

* Wyszukiwarka, dzięki której użytkownik może znaleźć interesujące go książki. Wyszukiwać można nie tylko tytuły, ale całość lub fragmenty wszystkich pól, np. fragment opisu.
* Przycisk „Strona główna”, który przekierowuje do strony głównej wyświetlającej wszystkie książki.
* Przycisk „Dodaj książkę ręcznie” przekierowuje do formularza, w którym użytkownik podaje dane książek.
* Przycisk „Dodaj książkę po ISBN” przekierowuje do formularza, w którym użytkownik podaje ISBN dodawanej książki.
* Lista rozwijana, która służy do wybierania motywu kolorystycznego strony (czerwony, niebieski, biały, czarny).



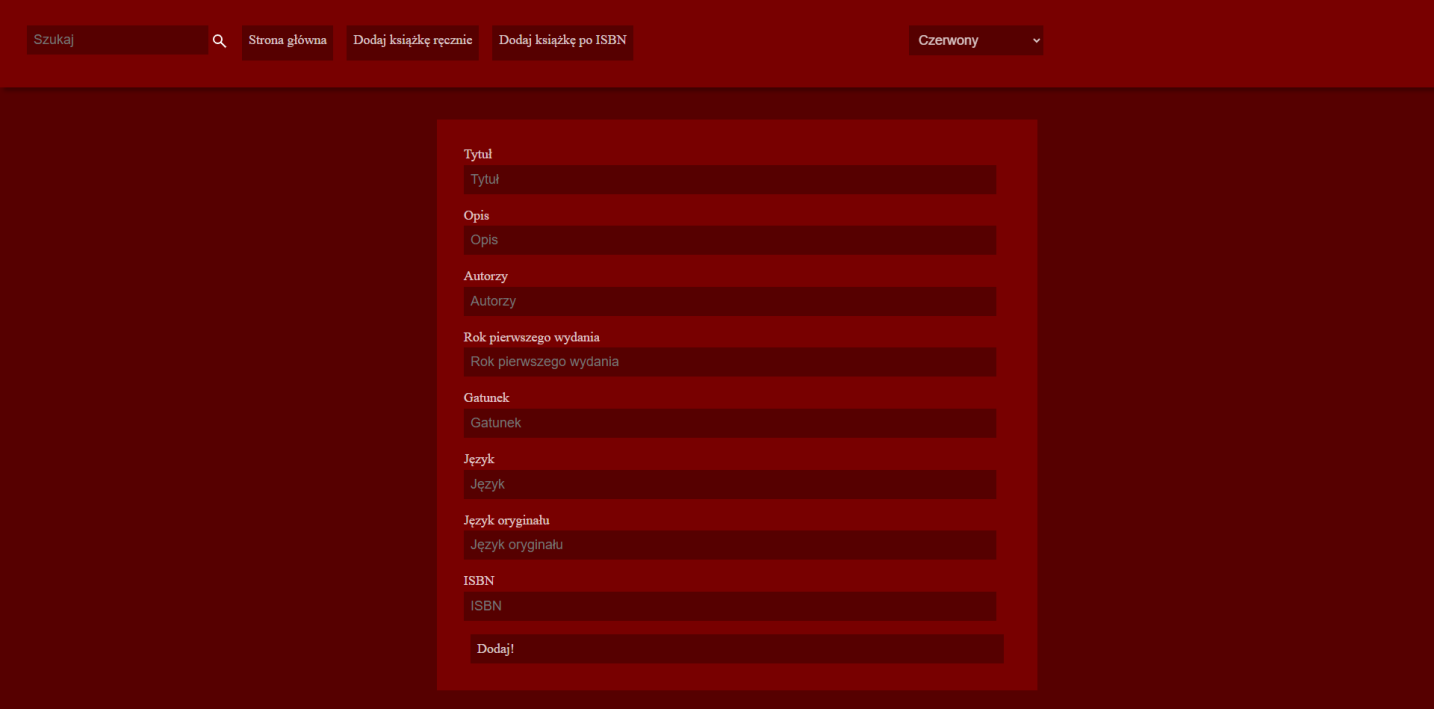




Samodzielne dodawanie książki

Pola formularza: Tytuł, Opis, Autorzy, Rok pierwszego wydania, Gatunek, Język, Język oryginału, ISBN.

Pola „Tytuł” oraz „ISBN” muszą zostać uzupełnione, żeby książka została dodana. Po naciśnięciu przycisku „Dodaj!” książka zostaje dodana jeżeli spełnione zostały powyższe warunki oraz ISBN nie występuje jeszcze w bazie danych.



Dodawanie książek po ISBN

Pola formularza: ISBN. Po naciśnięciu przycisku „Dodaj!” książka o danych pobranych z Biblioteki Narodowej zostaje dodana jeżeli podany ISBN nie istnieje jeszcze w bazie danych aplikacji oraz jest w bazie Biblioteki Narodowej.

1. **Baza danych**

Baza danych aplikacji *ksiegozbior* jest relacyjną bazą danych MySql.

Tabele:

* *autorzy*: *id* (int), *autorzy* (text)
* *gatunki*: *id* (int), *nazwa* (varchar)
* *jezyki*: *id* (int), *jezyk* (text)
* *książki*: *id* (int), *tytul* (text), *opis* (text), *oznaczenie\_odpowiedzialności* (int), *rok\_wydania* (pierwszego) (int), *gatunek\_id* (int), *jezyk* (wydania, które zostało wpisane do bazy danych) (int), *oryginalny\_jezyk* (int), *ISBN* (varchar)

- *oznaczenie\_odpowiedzialności* to pole zawierające klucz obcy, którym jest id autorów z tabeli *autorzy*

- *gatunek\_id* to pole zawierające klucz obcy, którym jest id gatunku z tabeli *gatunki*

- *jezyk*: to to pole zawierające klucz obcy, którym jest id języka z tabeli *jezyk*

-*oryginalny\_jezyk*: to to pole zawierające klucz obcy, którym jest id języka z tabeli *jezyk*

Kwerenda zwracająca id, tytuł, opis, autorów (słownie), rok wydania, gatunek(nazwa), język (słownie), oryginalny język (słownie) oraz ISBN:

*select ksiazki.id, tytul, autorzy, opis, rok\_wydania, jezyk2.jezyk as jezyk, jezyk1.jezyk as jezyk\_oryginal, nazwa, ISBN from ksiazki join autorzy on ksiazki.oznaczenie\_odpowiedzialnosci = autorzy.id join jezyki as jezyk1 on ksiazki.oryginalny\_jezyk = jezyk1.id join jezyki as jezyk2 on ksiazki.jezyk = jezyk2.id join gatunki on ksiazki.gatunek\_id = gatunki.id;*

1. **Back-end**

Back-end został wykonany w Node.js z użyciem frameworku Express. Do jego uruchomienia należy w terminalu wpisać npm exec nodemon. Działa na porcie 8000.

Moduły:

- myslq – do połączenia i obsługi bazy danych

- cors – do udostępnienia danych klientowi

- body-parser – do pobrania danych z formularza

Funkcje:

- checkIfExistsAddShowId

Argumenty: query1 – zapytanie zwracające sprawdzane dane, query2 – zapytanie dodające te dane do bazy danych

Funkcja sprawdza czy rekord z podanymi informacjami istnieje już w bazie danych. Jeżeli takich danych w bazie jeszcze nie ma, to je dodaje. W obu przypadkach zwracana jest obietnica zawierająca id rekordu już istniejącego lub dodanego.

- addBook

Argumenty: title, authors, genre, langOriginal, description, year, lang, ISBN, res

Asynchroniczna funkcja za pomocą funkcji *checkIfExistsAddShowId* sprawdza czy w bazie danych istnieją podani w argumentach autorzy, języki, gatunek i w razie potrzeby to dodaje. Id powyższych rekordów zostają zapisane do odpowiednich zmiennych. Następnie sprawdzana jest obecność książki z podanym w argumentach ISBN. Jeżeli taka książka jest w bazie danych *ksiegozbior,* to jest wysyłana informacja „Książka istnieje” w formacie JSON. W przeciwnym przypadku książka jest dodawana (wykorzystane zmienne z argumentów oraz zmienne zawierające klucze obce (id autorów, id języków, id gatunku)). Jeżeli książka została dodana pomyślnie, wysyłana jest informacja „Dodano książkę” w formacie JSON, jeżeli wystąpiły błędy, wysyłana jest informacja „Błąd”.

Endpointy:

- /order/:orderBy/:desc – GET

Wysyła rekordy pobrane z tabel bazy danych: *książki*, *autorzy*, *gatunki*, *jezyk* jako tablicę obiektów w formacie JSON. Dane są posortowane według parametrów *orderBy* i *desc.*

- /search/:searchVal/:orderBy/:desc – GET

Wysyła rekordy, których jakiekolwiek pole zawiera wyszukaną frazę (parametr *searchVal*), pobrane z tabel bazy danych: *książki*, *autorzy*, *gatunki*, *jezyk* jako tablicę obiektów w formacie JSON. Dane są posortowane według parametrów *orderBy* i *desc.*

- /addForm – POST

Przy użyciu funckji *addBook* dodawana jest do bazy danych książka, której dane są obiektem odbieranym za pomocą POST (wysłane przez klienta).

- /addJson/:isbn – GET

Przy użyciu funckji *addBook* dodawana jest do bazy danych książka, której dane w formacie JSON są pobierane za pomocą *fetch* z bazy danych Biblioteki Narodowej. Jest ona dodawana pod warunkiem, że w Bibliotece Narodowej jest książka z podanym w parametrze *isbn*. W przypadku jej braku wysyłany jest komunikat „Nie ma takiej książki w bazie Biblioteki Narodowej” w formacie JSON.

- /deleteBook/:bookId – GET

Z bazy danych usuwana jest książka, której id zostało podane w parametrze *bookId*. Do bazy wysyłane jest zapytanie, po którego wykonaniu zwracana jest informacja „Usunięto” w formacje JSON.

1. **Front-end**

Front-end został wykonany w React. Do jego uruchomienia należy w terminalu wpisać npm run dev. Po pobraniu projektu z GitHuba w folderze nie ma node\_modules.

Komponenty:

App

- Stany: *homeContent* – numer determinujący jaki komponent jest wyświetlany jako główna strona, *searchContent* – wyszukiwana fraza*, books* – tablica przechowująca książki z bazy danych

- Funckje:

* goHome

Argumenty: brak

Ustawia wartość *homeContent* na 0.

* addISBN

Argumenty: brak

Ustawia wartość *homeContent* na 1.

* addByHand

Argumenty: brak

Ustawia wartość *homeContent* na 2

Argumenty: brak

* search

Argumenty: searchC (wyszukiwana fraza)

Ustawia wartość *homeContent* na 0, ustawia wartość *SearchContent* na *searchC.*

* getBooks

Argumenty: *search* – wyszukiwana fraza, *order* – pole, według którego są sortowane książki , *sort* – kolejność sortowania

W zależności od tego, czy *search* jest puste do zmiennej *path* przypisywany jest adres do odpowiedniego endopointu na serwerze (czy wyszukiwanie jest brane pod uwagę, czy też nie). Następnie za pomocą *fetch* z serwera pobierane są dane książek (tablica obiektów) i są zapisywane z stanie *books*.

Komponent zwraca komponent *Menu* oraz div o id *page\_content* z komponentem *PageContent.*

Menu

- Propsy:

*addISBN* – funkcja z komponentu *App*,

*addByHand* – funkcja z komponentu *App*,

*goHome* – funkcja z komponentu *App*,

*searchVal* – funkcja z komponentu *App* (*search*),

*searchFunc –* funkcja z komponentu *App* (*getBooks*)

- Stany: *searchContent* – wyszukiwana fraza

Komponent zwraca div o id *menu*, a w nim

PageContent

- Propsy:

*page* – stan z komponentu *App* (*homeContent*),

*books* - prze stan z komponentu *App* (*books*),

*getBooks* – funkcja z komponentu *App*,

*search* - stan z komponentu *App* (*searchContent*)

W zależności od wartości propsu *page* komponent zwraca:

* Komponent *Sort* i komponent *Home* w przypadku 0.
* Komponent *AddISBN* w przypadku 1.
* Komponent *AddByHand* w przypadku 2.

Sort

- Propsy: *sortFun* – funkcja z komponentu *App* (*getBooks*), *search* - stan z komponentu *App* (*searchContent*)

- Stany: *sortState* - pole, według którego są sortowane książki, *orderState* - kolejność sortowania

Komponent zwraca div o id *sort*, w którym znajduje się formularz z rozwijaną listą select i

Home

BookShort

Delete

BookInfo

AddByHand

AddISBN

Alert

Propsy: *text* – treść komunikatu

Komponent zwraca div o klase *alert* z treścią komunikatu