**Do sprawdzenia jeszcze**

**Spis treści:**

1. **Wstęp ………………….……………...… strona 1**
2. **Instrukcja użytkowania ….……... strona 1**
3. **Baza danych …………………..…..…. strona 4**
4. **Back-end ………………….……………. strona 5**
5. **Front-end ………………….….…..…… strona 7**
6. **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. Aktualnie aplikacja działa tylko lokalnie.

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.

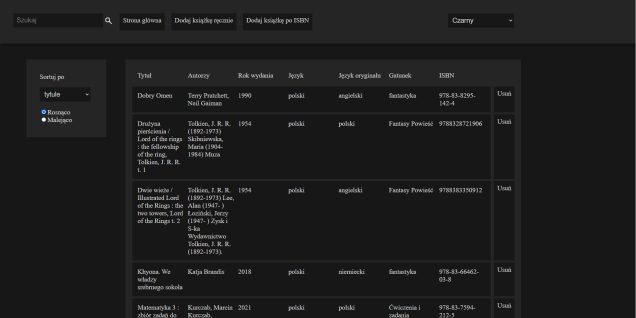
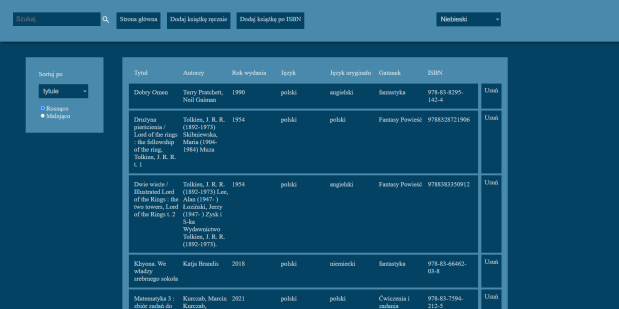


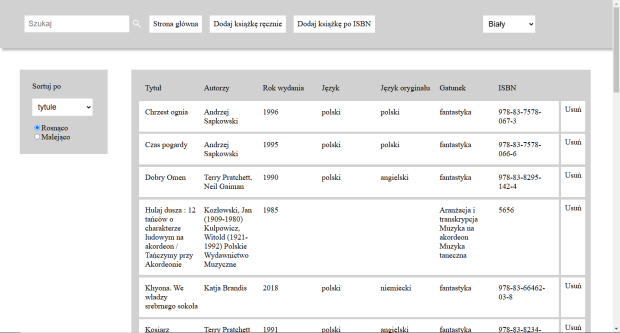
Menu



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).

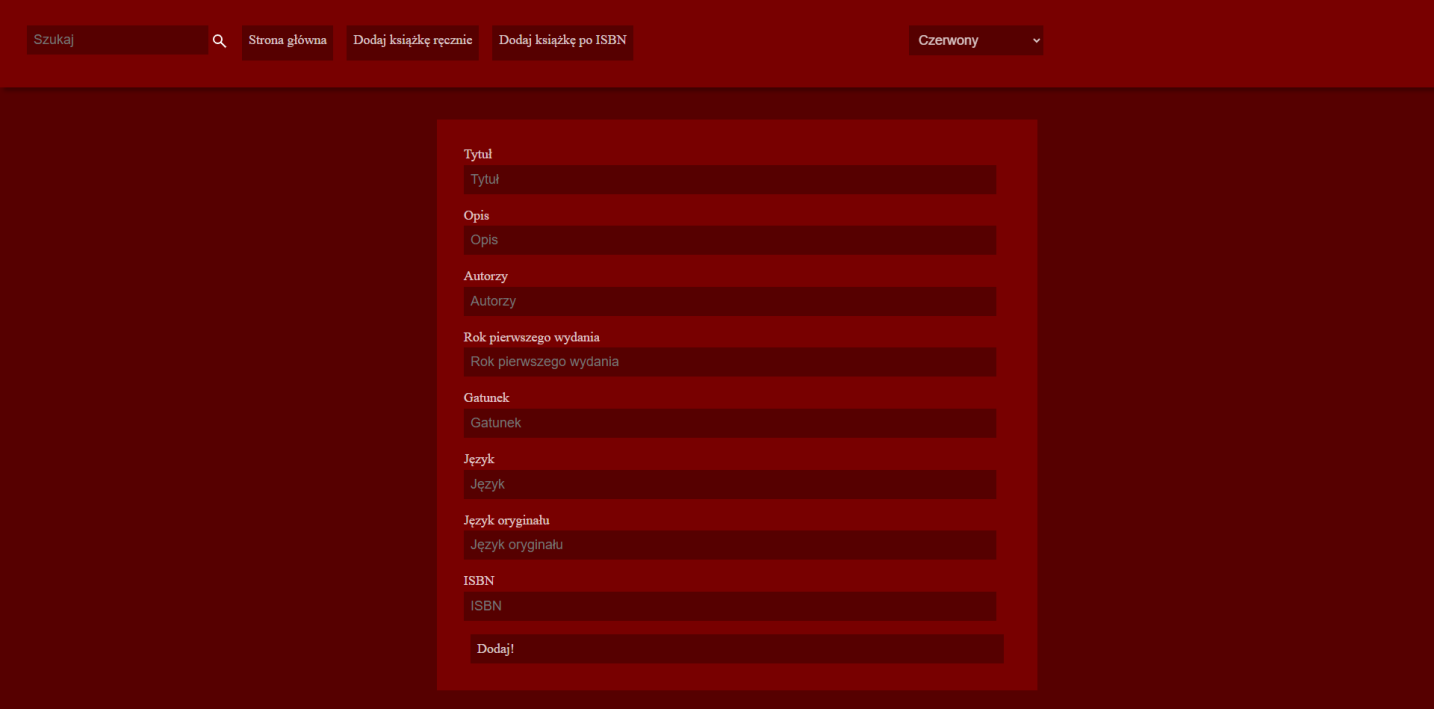




Samodzielne dodawanie książki

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 książka o podanym 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 książka o podanym 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)
* *ksiazki*: *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)

Relacje:

* *oznaczenie\_odpowiedzialności* - pole tabeli *ksiazki* zawierające klucz obcy, którym jest id autorów z tabeli *autorzy*
* *gatunek\_id* - pole tabeli *ksiazki* zawierające klucz obcy, którym jest id gatunku z tabeli *gatunki*
* *jezyk*: to - pole tabeli *ksiazki* zawierające klucz obcy, którym jest id języka z tabeli *jezyk*
* *oryginalny\_jezyk*: - pole tabeli *ksiazki* 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 pobierania danych w formacie JSON z Biblioteki Narodowej używana jest funckja *fetch.* 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 w bazie danych 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 komunikacji z serwerem używana jest funckja *fetch.* Do jego uruchomienia należy w terminalu wpisać npm run dev. Po pobraniu projektu z GitHuba w folderze może nie być *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

- Funkcje:

* 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: „/order/:orderBy/:desc” lub „/search/:searchVal/:orderBy/:desc” (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

- Właściwości:

*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:

* div o klasie *textBox* zawierający:

- pole tekstowe (input text) do wyszukiwania książek. Po wpisaniu czegoś do inputa (zdarzenie onChange) stan *searchContent* jest ustawiany na wartość inputa. Po wciśnięciu klawisza *Enter* (zdarzenie onKeyDown) wywoływana jest funkcja *searchVal* ze stanem jako argumentem (w komponencie *App* stan odpowiadający za wartość strony zmienia się na 0 i stan przechowujący wyszukiwaną frazę zmienia się na wyszukiwanie).

- div o id *Glass*, w którym znajduje się wektorowa grafika przedstawiająca lupę. Po kliknięciu diva (zdarzenie onClick) dzieje się to samo, co po naciśnięciu *Entera.*

* div o klasie *addBtn* i treści „Strona główna” to przycisk nawigujący, po którego kliknięciu (zdarzenie onClick) wyszukiwanie jest czyszczone oraz poprzez funkcję *goHome* wyświetlana jest główna strona (komponent *Home*).
* div o klasie *addBtn* i treści „Dodaj książkę ręcznie” to przycisk nawigujący, po którego kliknięciu (zdarzenie onClick) wyszukiwanie jest czyszczone oraz poprzez funkcję *addByHand* wyświetlany jest formularz do dodawania książek (komponent *AddByHand*).
* div o klasie *addBtn* to przycisk nawigujący, po którego kliknięciu (zdarzenie onClick) wyszukiwanie jest czyszczone oraz wyświetlany jest formularz do dodawania książek po ISBN (komponent *AddISBN*).
* div o klasie *change\_theme* zawiera rozwijaną listę select, która służy do wybrania motywu kolorystycznego. Po wybraniu elementu listy (zdarzenie onChange) zmienna *theme* jest ustawiana na wartość elementu i na tej podstawie zmienne w CSS są ustawiane na odpowiednie kolory.

PageContent

- Właściwości:

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

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

*getBooks* – funkcja z komponentu *App*,

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

W zależności od wartości właściwości *page* komponent zwraca:

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

Sort

- Właściwości:

*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 zawierający:

* rozwijaną listę select, po kliknięciu której (zdarzenie onClick) stan *orderState* jest ustawiany na wartość wybranego elementu oraz pobierane są posortowane książki z bazy danych za pomocą funkcji *sortFun*.
* dwa elementy radio, w zależności od tego, które pole zostanie zaznaczone (zdarzenie onClick) stan *sortState* jest ustawiany na wartość wybranego elementu oraz pobierane są posortowane dane z bazy danych za pomocą funkcji *sortFun*.

Home

Właściwości:

books – tablica obiektów z książkami z bazy danych (stan z komponentu *App*),

getBooks – funkcja pobierająca książki z bazy danych z komponentu *App*

Komponent zwraca div o id *home* zawierający

* div o klasie *header*, w którym znajduje się nagłówek (Tytuł, Autorzy, Rok wydania, Język, Język oryginału, Gatunek, ISBN).
* iterowaną za pomocą funkcji *map* tablicę *books*. Jej elementy są przekazywane do właściwości komponentu *BookShort*.

BookShort

Właściwości:

*bookData* – obiekt, dane jednej książki,

*getBooks* – funkcja pobierająca książki z bazy danych z komponentu *App*

Stany:

*displayDelete* – domyślnie przechowuje wartość false, determinuje czy komponent *Delete* jest wyświtlany,

*showInfo* - domyślnie przechowuje wartość false, determinuje, czy komponent *BookInfo* jest wyświetlany.

Funkcje:

*stopShowing* – ustwia wartość *showInfo* na false,

*displayNone* – ustawia wartość *displayDelete* na false

Komponent zwraca div o klasie *bookData* zawierający:

* div o klasie *bookShort*, w którym wyświetlane są dane książki. Po jego kliknięciu wartość stanu *showInfo* jest ustawiana na true,
* div o klasie *deleteBtn* z napisem „Usuń”. Po jego kliknięciu (zdarzenie onClick) wartość stanu *displayDelete* jest ustawiana na true
* jeżeli *displayDelete* ma wartość true to jest wyświetlany jest komponent *Delete*
* jeżeli *showInfo* ma wartość true to jest wyświetlany jest komponent *BookInfo*

Delete

Właściwości:

*Id* – id książki do usunięcia,

*title* – tytuł książki do usunięcia,

*displayNone* – stan z komponentu *BookShort*,

*getBooks* – stan z komponentu *BookShort*

Komponent zwraca div o klasie *delete*, w którym znajduje się div o klasie *deleteWindow*. Po jego kliknięciu (zdarzenie onClick) z użyciem *Id*  tworzony jest odpowiedni adres do endopintu „/deleteBook/:bookId”. Za pomocą funkcji *fetch* odpowiednie dane są usuwane.

BookInfo

Właściwości: *bookData* – obiekt, dane książki, *onClick* – funkcja z komponentu *BookShort* (stopShowing)

Stany: *hide* – przyjmuje wartość „” lub „Ukryj opis”

Komponent zwraca div o klasie *bookInfo,* który zawiera opis książki oraz stan *hide.* Po jego kliknięciu (zdarzenie *onClick*) stan *showInfo* z komponentu *BookShort* zostaje ustawiony na false (funkcja *onClick*), po najechaniu na niego myszą (zdarzenie onMouseOver) stan *hide* zostaje ustwiony na „Ukryj opis” (napis się wyświetli), a po cofnięciu myszy (zdarzenie onMouseOut) stan *hide* zostaje ustwiony na „” (napis zniknie).

AddByHand

Właściwości: brak

Stany:

*alert* – domyślnie ma wartość false, determinuje czy pod formularzem zostaje wyświetlony komunikat.

*title* – tablica z dwoma elementami, pierwszy to tytuł, a drugi może przyjąć wartość „red”

*isbn* - tablica z dwoma elementami, pierwszy to isbn, a drugi może przyjąć wartość „red”

*desc* – opis książki

*authors* - autorzy książki

*year* – rok wydania książki

*genre* – gatunek książki

*lang* – język książki

*langOg* – oryginalny język książki

Komponent zwraca div o klasie *add* , a w nim:

* formularz z polami do wypełnienia (tytuł, opis, autorzy, rok wydania, gatunek, język, oryginalny język) (input text i input number).

-Kiedy coś zostaje wpisane do pól formularza (zdarzenie onChange), wartość odpowiedniego stanu zostaje ustawiona na wartość inputa (np. wartość pola podpisanego jako Autorzy -> stan *authors*).

- Po wyjściu z pól tytułu i ISBN (zdarzenie onBlur) nic nie wpisawszy, pola te zmieniają kolor tła na czerwony (backgroundColor w stylach przyjmuje wartość drugiego elementu stanu *title* lub *isbn* – „” lub „red”).

* div o klasie *addBtn* i treści „Dodaj!”, po kliknięciu którego (zdarzenie onClick), jeżeli pierwsze elementy stanów *title* i *isbn* nie są puste, tworzony jest obiekt *bookData* z polami takimi jak stany komponentu i ich wartości (w przypadku *title* i *isbn* wartości ich pierwszych elementów). Następnie za pomocą funkcji *fetch* obiekt *bookData* zostaje wysłany do serwera w celu dodania danych książki do bazy danych. Serwer odsyła informację zwrotną, czy udało się dodać książkę do bazy danych i jest ona zapisywana do stanu *alert.* W przypadku nieuzupełnienia pól tytułu lub ISBN wartość stanu *alert* zostaje ustawiona na „Uzupełnij pola” i zostaje wyświetlony komponent *Alert*, a drugi element stanu *title* lub *isbn* zostaje ustawiony na „red” i odpowiednie pola zmieniają kolor tła na czerwony.

AddISBN

Właściwości: brak

Stany:

*alert -* domyślnie ma wartość false, determinuje czy pod formularzem zostaje wyświetlony komunikat.

Komponent zwraca div o klasie *add,* a w nim:

* formularz z jednym polem input number. Kiedy coś zostanie do niego wpisane (zdarzenie onChange) do zmiennej *isbn* zostaje przypisana wartość inputa.
* div o klasie *addBtn* o treści „Dodaj!”. Po jego kliknięciu (zdarzenie onClick), jeżeli zmienna *isbn* nie jest pusta, za pomocą funkcji *fetch* do serwera wysyłany jest ISBN. Serwer odsyła informację zwrotną, czy udało się dodać książkę do bazy danych i jest ona zapisywana do stanu *alert.*
* jeżeli stan *Alert* zawiera informację zwrotną, renderowany jest komponent *Alert.*

Alert

Właściwości:

*text* – treść komunikatu

Komponent zwraca div o klasie *alert* z treścią komunikatu.