

Sprawozdanie Node.js, MongoDB

Dominik Gronkiewicz- 118104

15 stycznia 2017

Spis treści

1	Wstęp	2
1.1	Technologie i narzędzia	2
2	Przebieg projektu	3
2.1	Użycie npm do konfiguracji projektu	3
2.2	Przygotowanie bazy danych	3
2.3	Utworzenie aplikacji Node.js oraz widoków	4
2.4	Widoki aplikacji	6
3	Podsumowanie i wnioski	9

1 Wstęp

Sprawozdanie opisuje przebieg oraz działanie projektu wykorzystującego nierałacyjną bazę danych MongoDB oraz Node.js. Tematem projektu jest system obsługi biblioteki. Aplikacja umożliwia użytkownikowi zarządzanie wypożyczonymi książkami oraz recenzjami do nich. Pisanie recenzji natępuje przy zwrocie i jest opcjonalne. Każda książka posiadająca co najmniej jedną recenzję ma też dodatkowy widok zawierający listę recenzji. Jeżeli użytkownik napisał recenzję, może ją też w tym widoku wykasować.

Projekt jest dostępny w wersji online.

- Kod aplikacji: <https://github.com/dongron/libraryNode>
- Url działającej aplikacji: <https://library-node-mongo.herokuapp.com/>

1.1 Technologie i narzędzia

- MongoDB
Nierałacyjna baza danych oparta na zapisie dokumentów w formacie JSON. Charakteryzuje się dużą skalowalnością, wydajnością oraz brakiem ściśle zdefiniowanej struktury obsługiwanych baz danych. Forma dokumentu umożliwia aplikacjom bardziej naturalne ich przetwarzanie, przy zachowaniu możliwości tworzenia hierarchii oraz indeksowania.
- Node.js
Node.js jest platformą wykorzystującą język JavaScript. Używa standardu języka Google Chrome V8. Posiada ogromną liczbę modułów.
- Express
Minimalistyczny, szybki w działaniu i prosty w użyciu framework webowy. Wykorzystuje protokół HTTP do komunikacji między serwerem aplikacji, a klientem.
- MongoDB Node.JS Driver
Sterownik do obsługi bazy danych MongoDB dla platformy Node.js. Wspiera metody callback i promise oraz typy standardu ES6.
- mustache
Mustache to silnik szablonów wykorzystywany głównie w HTML, plikach konfiguracyjnych oraz kodzie źródłowym. Nie zawiera operatorów warunkowych oraz pętli, wszystko tworzone jest z użyciem tagów
- body-parser
Biblioteka parsowania danych. Służy do pobierania danych z arkuszy HTML.
- Bootstrap
Popularny framework do tworzenia front-endu w HTML, CSS, JavaScript. Zawiera wiele predefiniowanych metod i zachowań dla elementów HTML o określonych klasach. Wspiera on styl Responsive Web Design.

- jQuery
Biblioteka JavaScript ułatwiająca pracę z elementami drzewa DOM. Ułatwia operacje na dokumencie HTML.

2 Przebieg projektu

2.1 Użycie npm do konfiguracji projektu

Za pomocą pakietu npm należy pobrać pakiety tworzące następujące zależności:

```
"dependencies": {
  "body-parser": "^1.15.2",
  "bootstrap": "^3.3.7",
  "consolidate": "^0.14.5",
  "express": "^4.14.0",
  "jquery": "^3.1.1",
  "mongodb": "^2.2.16",
  "mustache": "^2.3.0"
}
```

2.2 Przygotowanie bazy danych

Do działania aplikacji wymagane jest utworzenie odpowiednich struktur oraz wstępne wypełnienie bazy danych przykładowymi wartościami.

Uruchomienie bazy danych oraz konsolowego klienta w systemie operacyjnym opartym na Arch Linux odbywa się poleceniami:

```
systemctl start mongodb
```

Do operowania na bazie danych, najwygodniej jest użyć narzędzia graficznego jak np. MongoChef.

Należy dodać przykładowe dane dla użytkownika i książek:

```
db.users.insert({login: "test", password: "test"});
db.users.insert({_id:
  ObjectId("5063114bd386d8fadbd6b000"), login:
  "superman", password: "iloveflying"});

db.books.insert({name: "50 shades of Grey", author:
  "E.L. James", year: "2005", review: [
  {date: new Date("2016-12-12T12:00:00Z"), user_id:
    ObjectId("5063114bd386d8fadbd6b004"), content:
    "Worst book ever, not recommend. "},
```

```

    {date: new Date("2017-01-01T22:00:00Z"), user_id:
      "", content: "Quite good. I dont like fantasy.
        Readable."},
    {date: new Date("2008-05-28T07:00:00Z"), user_id:
      "", content: "I m not info romaces. Worst book
        ever. 1/10"},
    {date: new Date("2010-10-10T10:00:00Z"), user_id:
      "", content: "I love this book. Gery is my ideal
        model of man."};
  ], rentBy: ""});

db.books.insert({name:"30 doors down", author: "W.
  Gombowicz", year: "1956", review: "", rentBy: ""});

  Więcej komend dołączonych jest w pliku libraryMongoScript.js, który jest
  zawarty w plikach źródłowych projektu.

  Innymi przykładowo wykorzystanymi operacjami są kolejno: kasowanie re-
  kordu o podanej wartości; aktualizacja zawartości dokumentu; kasowanie elemen-
  tu o określonej nazwie; znalezienie jednego elementu o podanym parametrze:

db.books.remove({'name':'Kot w butach'});
db.books.update({"name": "Mikolajek"}, { $set:
  {"rentBy": "58666affbb569757a028012f"}});
db.users.findOne({login: "test"});

```

2.3 Utworzenie aplikacji Nojde.js oraz widoków

Kolejnym etapem jest utworzenie kodu aplikacji w pliku server.js. Serwer wykorzystuje metody get oraz post z biblioteki express, wykonuje operacje na bazie danych oraz przekazuje dane do widoków za pomocą mustache.

Przykładowa metoda get, w której pobierana jest lista recenzji dla książki o podanym id, utworzona zostaje z niej lista. W przypadku udanej operacji utworzona lista, inne użyte obiekty oraz widok w HTML zostają przekazane na adres zawierający id książki.

```

app.get('/reviews/:_id', function (req, res) {
  console.log(req.params._id);
  booksCollection.findOne({"_id": new
    ObjectId(req.params._id)}).then(
    item => {
      var book = {
        id: req.params._id,
        name: item.name,
        author: item.author
      }
      res.render(__dirname +
        "/views/"

```

```

        + 'bookDetails.html', {userId:
        User.id, book: book,
        reviews: item.review});
    },
    err => console.error(err)
  );
})

```

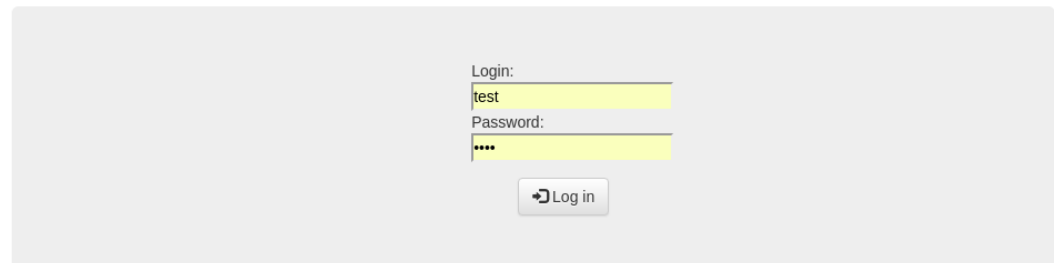
Przykładowy fragment widoku HTML, w którym lista recenzji jest przypisana pod elementy #reviews. Wyświetla on listę recenzji oraz przycisk umożliwiający usunięcie recenzji.

```

{{#reviews}}
<div class="listItem list-group-item">
  <div class="itemName">
    <div class="date">{{date}} </div>
    <div>
      <span class="review">
        {{content}} </span>
      <span>
        <button class="btn
          btn-danger btn-sm
          revUserId" type =
          "submit" value =
          "{{user_id}}"
          name="{{_id}}">
          <span
            class="glyphicon
            glyphicon-remove-circle"></span>
        </button>
      </span>
    </div>
  </div>
</div>
{{/reviews}}

```

2.4 Widoki aplikacji



Login:

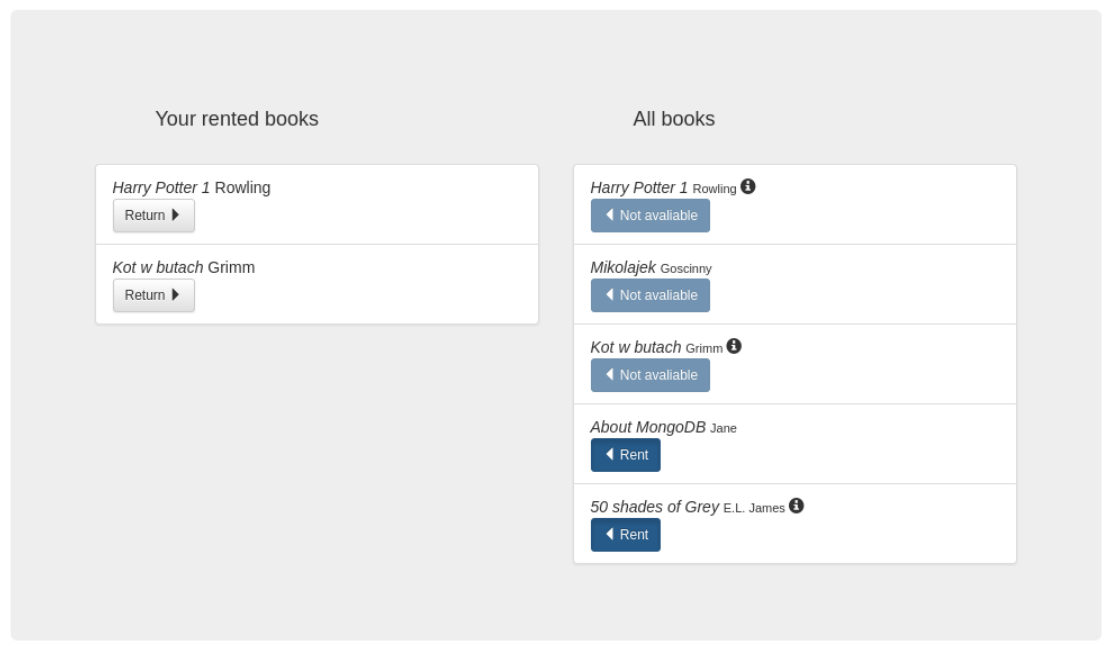
test

Password:

Log in

Rysunek 1: Widok logowania

test's books



Your rented books

Harry Potter 1 Rowling
Return ▶

Kot w butach Grimm
Return ▶

All books

Harry Potter 1 Rowling ⓘ
◀ Not available

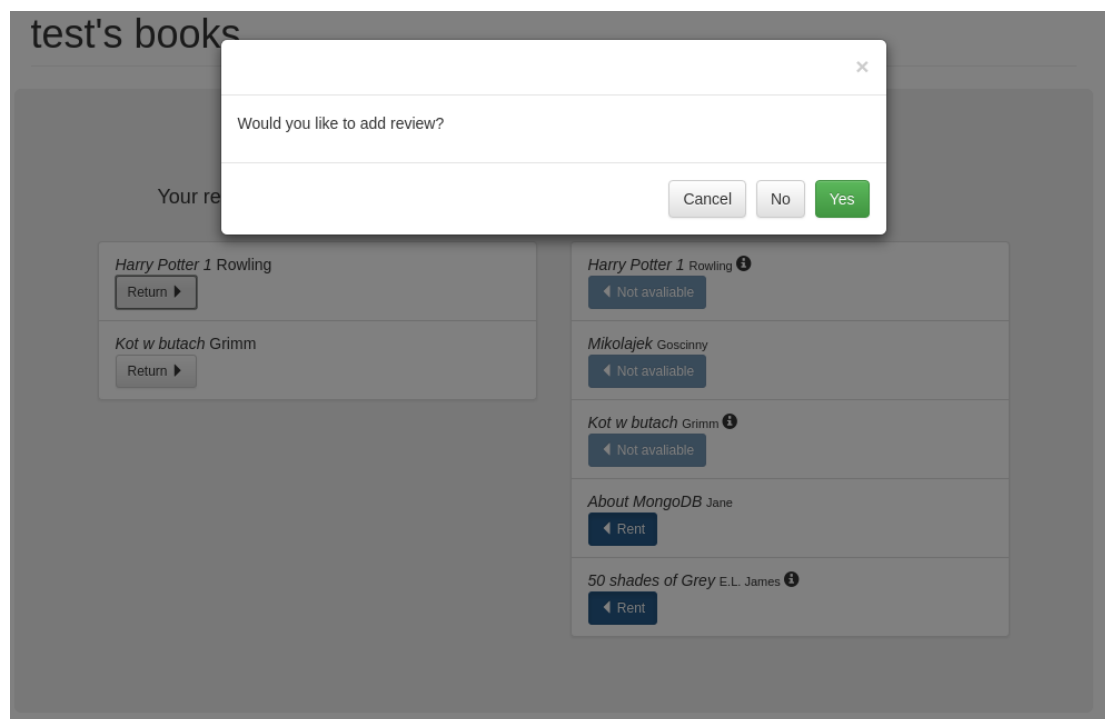
Mikolajek Goscimny
◀ Not available

Kot w butach Grimm ⓘ
◀ Not available

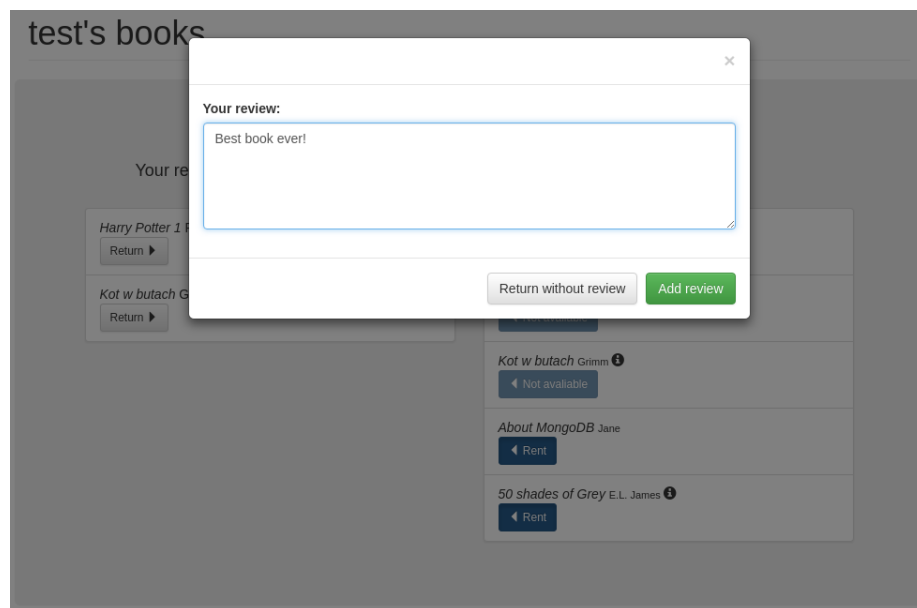
About MongoDB Jane
◀ Rent

50 shades of Grey E.L. James ⓘ
◀ Rent

Rysunek 2: Widok listy książek dostępnych oraz wypożyczonych. Gdy dostępna są recenzje dla wybranej książki, pojawia się przy niej dodatkowa ikona, która przekierowuje do zawierającego je widoku

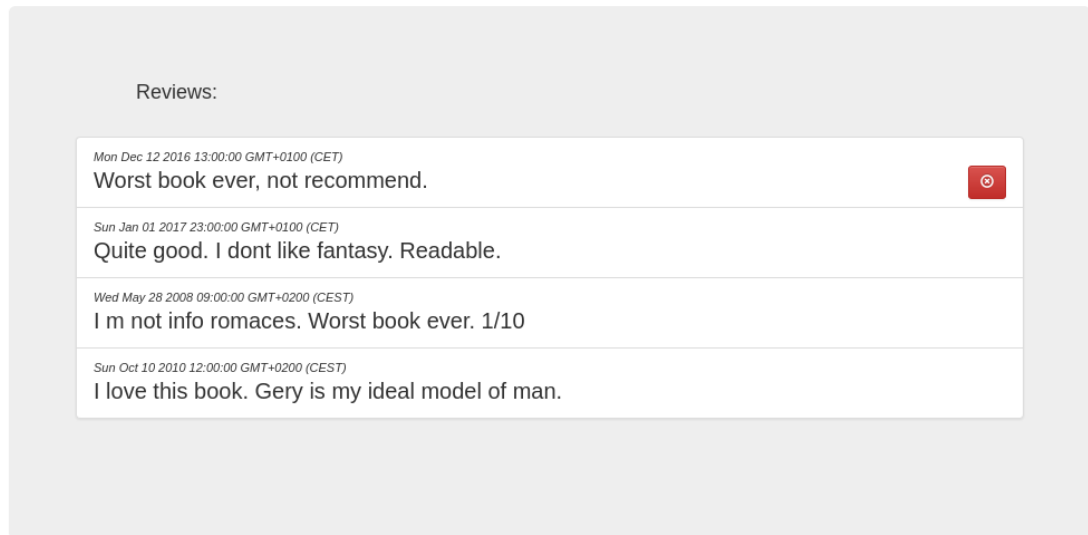


Rysunek 3: Dialog przy zwracaniu książki



Rysunek 4: Dialog dodawania recenzji

50 shades of Grey - E.L. James



Rysunek 5: Lista recenzji dla wybranej książki. Możliwość usunięcia własnej recenzji.

3 Podsumowanie i wnioski

Node.js jest wygodną w użyciu platformą do szybkiego i prostego tworzenia serwera aplikacji. W użyciu z bazą danych MongoDB, pozwala na utworzenie w pełni działającej aplikacji internetowej, która może posiadać nieujednoliconą strukturę elementów. Nierelacyjna baza danych podważa pominąć krok projektowania złożonych relacji.