

Projektowanie portali biznesowych

Informatyka III niestacjonarne

Prowadzący: dr Piotr Lasek

Autor projektu: Klaudiusz Florek

Temat Projektu: Projekt realizujący podstawową funkcjonalność typu CRUD

Tytuł Projektu: myWallet

Data Oddania: 21/04/2020

Zakres Projektu:  Przygotowanie UI, budowa logiki aplikacji, synchronizacja z bazą danych

Technologie: React.js, Node.js, Express.js, MongoDB

Podgląd projektu: <https://ancient-meadow-10141.herokuapp.com/>

1.Cel aplikacji i ogólne informacje

Projekt myWallet jest aplikacją typu MERN.

Umożliwia dodawanie transakcji przy podaniu jej nazwy oraz wartości. Użytkownik może sprawdzać aktualny stan swojego wirtualnego salda, śledzić historię transakcji, które przeprowadził, a także je usuwać.

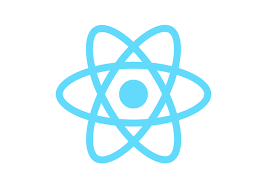
UI aplikacji zostało zbudowane w technologii React.js, strona serwerowa przy pomocy środowiska uruchomieniowego Node.js (javascript) oraz framework’a Express.js, a jako bazę danych wykorzystano mongoDB.

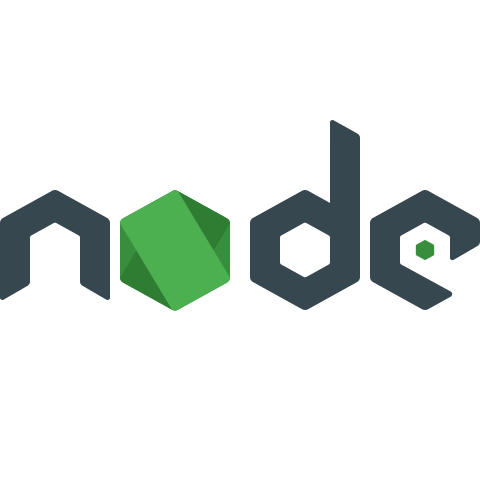
Aplikacja korzysta z podstawowych zapytań http:

**GET** (http://localhost:1410/api/v1/transactions)

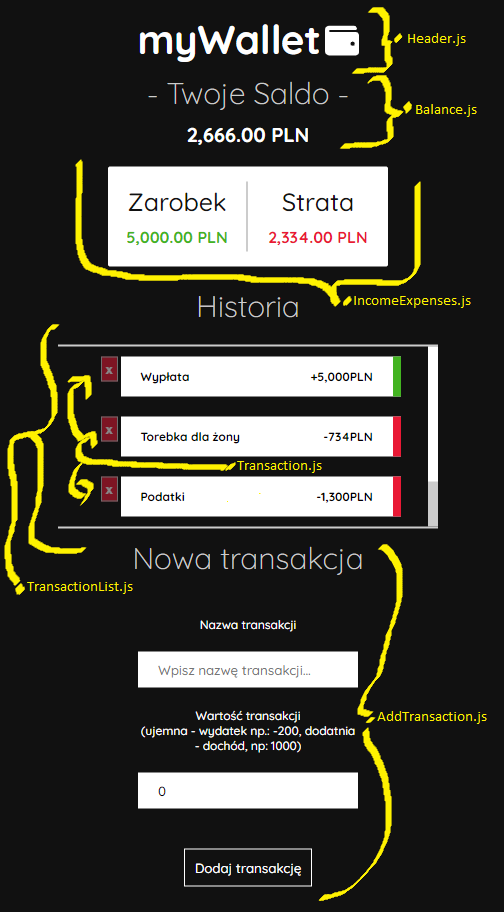
**POST** (http://localhost:1410/api/v1/transactions)

**DELETE** (http://localhost:1410/api/v1/transactions/:id)



2.Widok główny aplikacji



3.Podział aplikacji na komponenty

UI aplikacji składa się z sześciu komponentów tj. Header.js, Balance.js, IncomeExpenses.js, TransactionList.js, Transaction.js i AddTransaction.js oraz głównego komponentu App.js, który pełni rolę spajającą całą aplikację.

Header.js odpowiedzialny jest za wyświetlenie logotypu aplikacji.

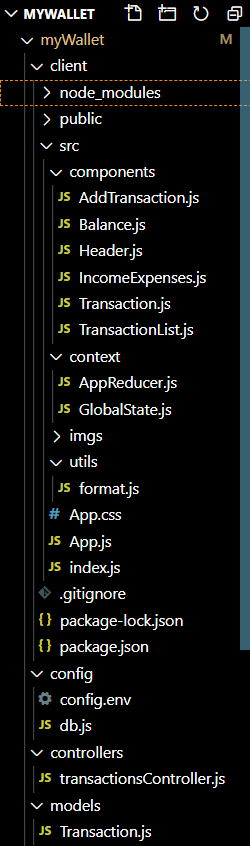
Balance.js odpowiedzialny jest za podsumowanie i wyświetlenie salda.

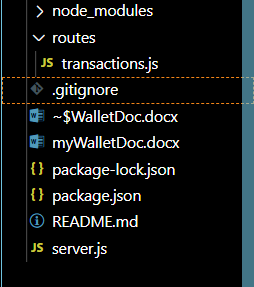
IncomeExpenses.js odpowiedzialny jest za obliczenie i wyświetlenie kolejno zarobku oraz strat, które wynikły z dodanych transakcji w komponencie TransactionList.js.

Transaction.js odpowiada za przetrzymywanie danych odnoszących się do każdej transakcji oraj jej wyświetlenie.

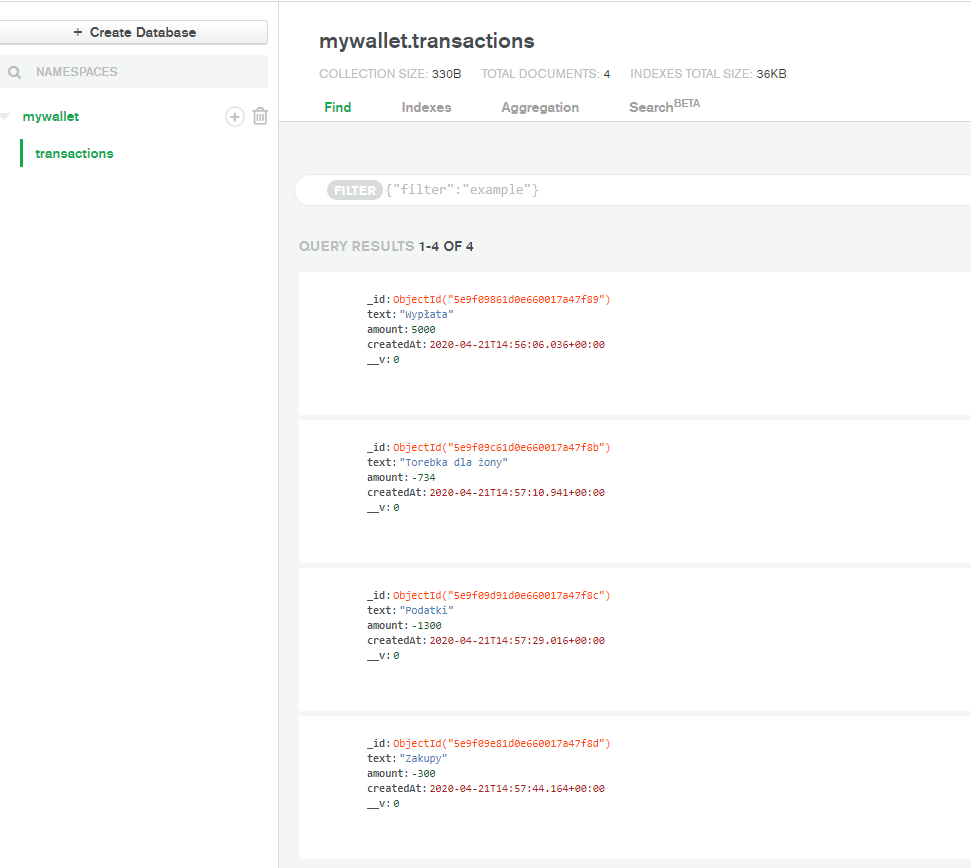
AddTransaction.js jest komponentem, który pełni rolę formularza pozwalającego dodać nową transakcję.

4.Struktura aplikacji



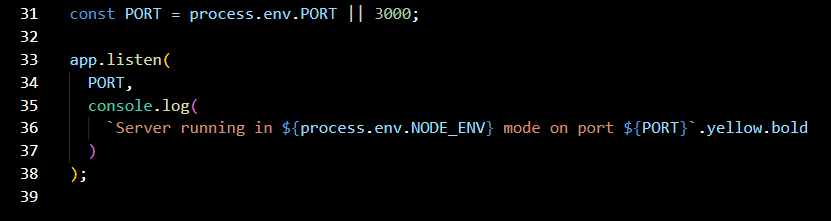


5.Podgląd bazy danych



*Dodane rekordy w tabeli transactions.*

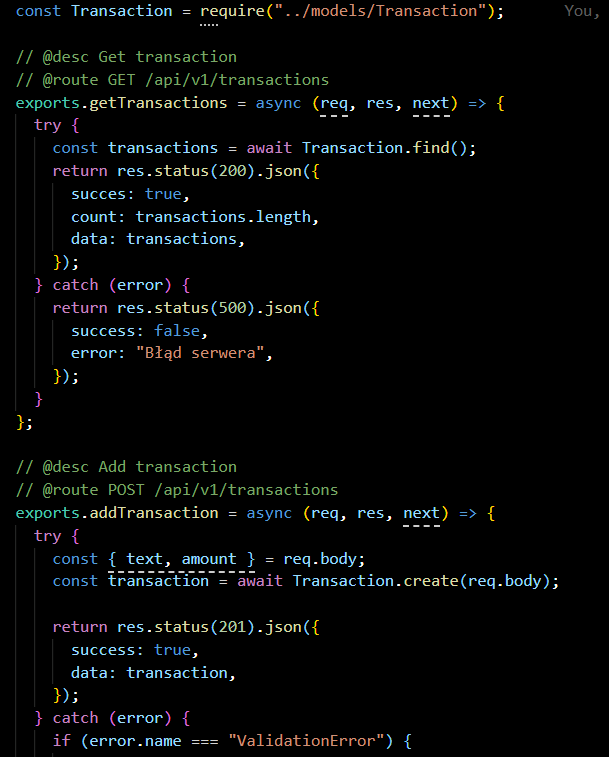
6.Strona serwerowa aplikacji



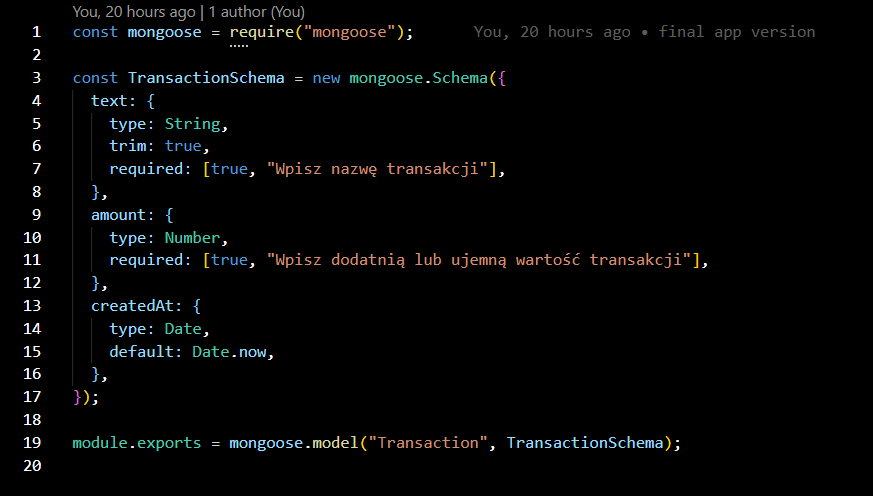
*W skrypcie server.js tworzymy serwer, konfigurujemy port, ustawiamy adres url i łączymy się z bazą danych za pomocą wcześniej utworzonej metody connectDB.*



*W skrypcie db.js definiujemy metodę connectDB, a w niej jak ma odbywać się połączenie z bazą danych. Następnie export ujemy ją przy pomocy module.exports, by użyć jej w skrypcie server.js*



*W skrypcie transactionsController definujemy api naszej aplikacji, obsługujemy ewentualne błędy na zapytania oraz walidujemy zgodność przesyłanych danych, których typ (schemat) zbudowany został w skrypcie Transaction.js*



*Transaction.js, który zawiera model danych.*



*GlobalState.js – skrypt, w którym przechowywane są informacje o globalnym stanie aplikacji oraz metodach odpowiedzialnych za podstawowe funkcjonalności.*