

# Koncepcja i implementacja systemu do zarządzania budżetem domowym

(Concept and implementation of home budget management system)

Mateusz Kmita

Praca inżynierska

**Promotor:** dr inż. Leszek Grocholski

Uniwersytet Wrocławski  
Wydział Matematyki i Informatyki  
Instytut Informatyki

2 stycznia 2022



## **Streszczenie**

Celem pracy było stworzenie koncepcji i implementacja systemu służącego do zarządzania budżetem domowym. Motywacją do opracowania takiego systemu było dostarczenie prostego i użytecznego rozwiązania pomagającego użytkownikom w prowadzeniu osobistego budżetu. Do zrealizowania tego celu użyto nowoczesnych oraz popularnych technologii i narzędzi, które pomagają w stworzeniu aplikacji dostępnej z poziomu przeglądarki internetowej.

---

The aim of the work was to develop the concept and implementation of home budget management system. The motivation of the work was to provide a simple and useful tool



# Spis treści

<b>1. Wprowadzenie</b>	<b>7</b>
1.1. Opis zagadnienia i motywacje autora . . . . .	7
1.2. Przykłady innych rozwiązań . . . . .	7
1.2.1. You Need a Budget (YNAB) . . . . .	8
1.2.2. Mint . . . . .	8
<b>2. Opis systemu</b>	<b>11</b>
2.1. Sposób instalacji i dostępu do systemu . . . . .	11
2.2. Opis funkcji systemu . . . . .	11
2.2.1. Zarządzanie kontami . . . . .	11
2.2.2. Wyświetlanie operacji . . . . .	13
2.2.3. Rejestrowanie operacji . . . . .	13
2.2.4. Wyświetlanie budżetów . . . . .	13
2.2.5. Tworzenie budżetów . . . . .	13
2.2.6. Wyświetlanie statystyk . . . . .	13
<b>3. Użyte rozwiązania technologiczne</b>	<b>15</b>
3.1. Część serwerowa . . . . .	15
3.1.1. Spring . . . . .	16
3.1.2. Schemat bazy danych . . . . .	16
3.1.3. Inne technologie użyte w części serwerowej . . . . .	16
3.2. Część kliencka . . . . .	16
3.2.1. Biblioteka React . . . . .	16

<b>4. Podsumowanie</b>	<b>17</b>
4.1. Możliwości dalszej rozbudowy . . . . .	17
<b>Bibliografia</b>	<b>19</b>

# Rozdział 1.

## Wprowadzenie

### 1.1. Opis zagadnienia i motywacje autora

W ramach tej pracy zajmuję się zagadnieniem prowadzenia budżetu domowego, które polega na podziale posiadanych pieniędzy pomiędzy obszary, na które chcemy je przeznaczyć. Z jednej strony budżet domowy wymaga poświęcenia dodatkowego czasu na stworzenie oraz jego regularne prowadzenie. Z drugiej strony możemy w zamian liczyć na większą kontrolę nad stanem naszego portfela. Przy założeniu, że taki budżet prowadzony jest regularnie, mamy wiedzę o tym, ile pieniędzy możemy jeszcze wydać. Może to pozwolić nawet na uniknięcie długów. Osoba prowadząca budżet domowy i historię wydatków może także łatwiej analizować swoje nawyki konsumenckie oraz je poprawiać, aby móc efektywniej zarządzać swoimi pieniędzmi w przyszłości.

Zdecydowałem się na stworzenie aplikacji, która będzie dostarczała wszystkie wymagane podstawowe funkcjonalności do zrealizowania opisanego wyżej zadania oraz będzie na tyle prosta, aby nowy użytkownik mógł jak najmniejszym kosztem zacząć prowadzić swój budżet domowy. W dalszej części pracy opisałem jakie funkcje udostępnia aplikacja. Następnie skupiłem się na przedstawieniu i uzasadnieniu mojego rozwiązania pod względem technologicznym. W końcowej części pracy podsumowałem zrealizowane zadania oraz podałem przykłady sposobów na dalsze ulepszenie aplikacji.

### 1.2. Przykłady innych rozwiązań

Użytkownicy zainteresowani tematyką prowadzenia budżetu domowego mają duży wybór dostępnych rozwiązań. Istnieją zarówno aplikacje dostępne przez przeglądarkę, tradycyjne programy i aplikacje na urządzenia mobilne. Wiele programów

dostępnych jest jednocześnie na wielu rodzajach urządzeń dla wygody użytkowników. Ja skupiłem się na rozwiązaniach, które dostępne są przez przeglądarkę, gdyż są najbardziej zbliżone do rozwiązania proponowanego przeze mnie.

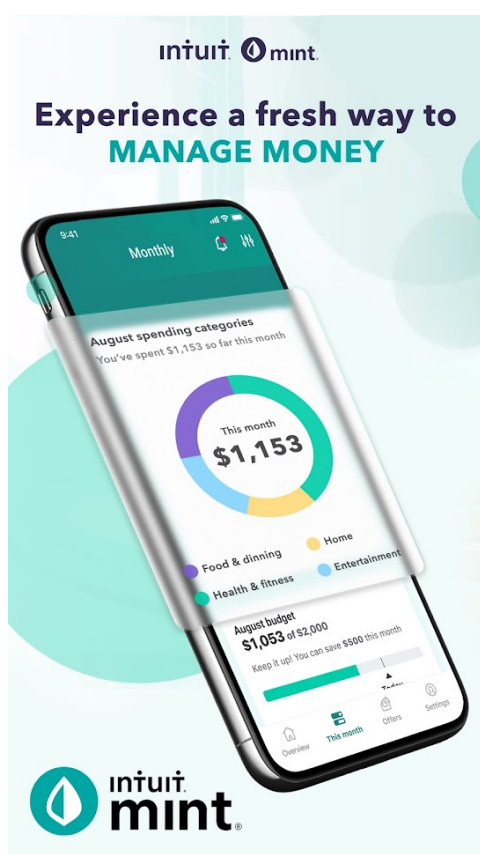
### 1.2.1. You Need a Budget (YNAB)

Aplikacja You Need a Budget [1] jest dostępna przez przeglądarkę internetową oraz w postaci aplikacji na urządzenia mobilne na systemy operacyjne Android i iOS. Jest to program płatny, który kosztuje 14,99\$ miesięcznie lub 89,99\$ rocznie [2]. Ideą tej aplikacji jest nie tyle zautomatyzowanie całego procesu zarządzania budżetem i całkowite wyręczenie użytkownika, co nauczenie go jak taki budżet tworzyć, więc aplikacja YNAB wymaga od użytkownika poświęcenia czasu na ręczne tworzenie budżetu. Filozofia tworzenia budżetu domowego w tej aplikacji polega na rozdzielaniu pieniędzy co do grosza na nasze cele. Program ten posiada użyteczną funkcję automatycznego pobierania wydatków z konta bankowego.

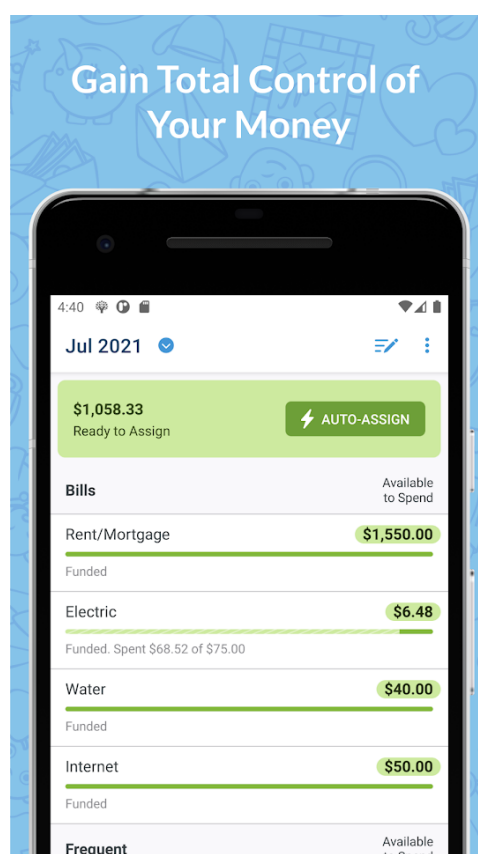
### 1.2.2. Mint

Mint [3] to darmowy program dostępny przez przeglądarkę internetową oraz jako aplikacja na urządzenia mobilne na systemy operacyjne Android i iOS. Aplikacja dostępna jest za darmo, ale wyświetla użytkownikom reklamy ofert od swoich partnerów biznesowych. To rozwiązanie posiada wiele funkcji automatyzujących cały proces zarządzania budżetem. Automatycznie pobiera wydatki z połączonego konta bankowego i przypisuje je do odpowiednich kategorii. Posiada też dodatkowe funkcje oprócz prowadzenia budżetu takie jak przypominanie o płatnościach za subskrypcje i rachunki lub wysyłanie użytkownikom porad, jak mogą lepiej zarządzać pieniędzmi. Dostępne są także informacje o zdolności kredytowej, funkcje związane z inwestowaniem oraz inne funkcjonalności dotyczące pieniędzy. Mamy więc tutaj więcej funkcji niż w poprzednim rozwiązaniu.





(a) Mint



(b) You Need A Budget

Rysunek 1.1: Inne rozwiązania dotyczące prowadzenia budżetu osobistego



## Rozdział 2.

# Opis systemu

### 2.1. Sposób instalacji i dostępu do systemu

### 2.2. Opis funkcji systemu

Koncepcja aplikacji polega na założeniu, że użytkownik będzie samodzielnie regularnie rejestrował w aplikacji wszystkie swoje wydatki oraz wpływy oraz ręcznie tworzył budżet rozdzielając pieniądze pomiędzy różne kategorie. Program będzie te informacje zapisywał oraz obliczał środki dostępne w ramach budżetu.

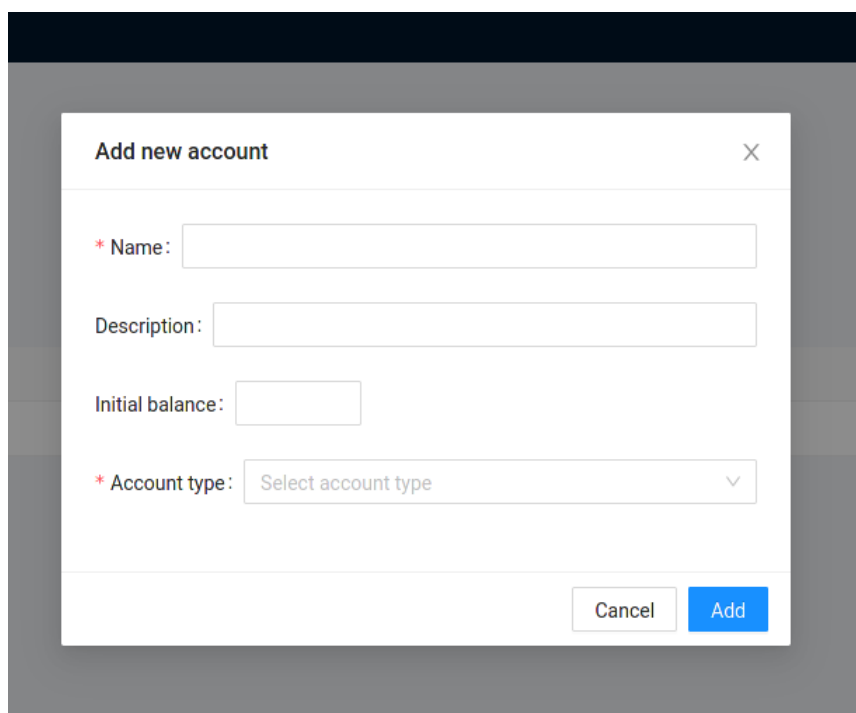
Poniżej wytłumaczyłem podstawowe pojęcia używane w aplikacji.

- **Transakcje** - operacje pieniężne, które są wydatkami lub wpływami z zewnętrznych źródeł. Te operacje mogą wpływać na stan budżetu.
- **Transfery** - operacje pieniężne pomiędzy dwoma różnymi kontami, które nie wpływają na budżet, ale dotyczą przeniesienia środków wewnątrz portfela użytkownika
- **Kategoria transakcji** - podział wszystkich transakcji na różne kategorie dotyczące tej samej dziedziny. Ocena do jakiej kategorii należy transakcja leży całkowicie po stronie użytkownika.
- **Budżet** - budżet to comiesięczny podział wpływów pomiędzy różne kategorie wraz z wydatkami w tym miesiącu oraz informacją o środkach dostępnych jeszcze w ramach każdej z kategorii.

#### 2.2.1. Zarządzanie kontami

Użytkownik posiada możliwość podziału swojego portfela na różne konta. Ta funkcja pozwala śledzić ilość pieniędzy ulokowanych w różnych źródłach. Dla wygody użytkownika konta są podzielone pomiędzy cztery różne rodzaje: rozliczeniowe,

oszczędnościowe, gotówkę oraz inne. Lista kont wyświetlana jest w panelu nawigacyjnym po lewej stronie ekranu. Możemy stamtąd przejść do listy transakcji lub transferów przypisanych do każdego z kont. Na górze strony będą wtedy wyświetlane informacje o koncie, którego ekran dotyczy. Znajdziemy tam nazwę konta, opis, rodzaj i aktualny stan środków na koncie. Obok tych informacji znajdują się przyciski “Usuń konto” oraz “Edytuj konto” służące do usuwania konta i edycji danych o nim. Aby dodać nowe konto należy nacisnąć na przycisk “Dodaj konto” w panelu nawigacyjnym. Wyświetla się wtedy formularz, w którym należy podać nazwę i rodzaj konta oraz opcjonalnie opis i początkowy stan tego konta.

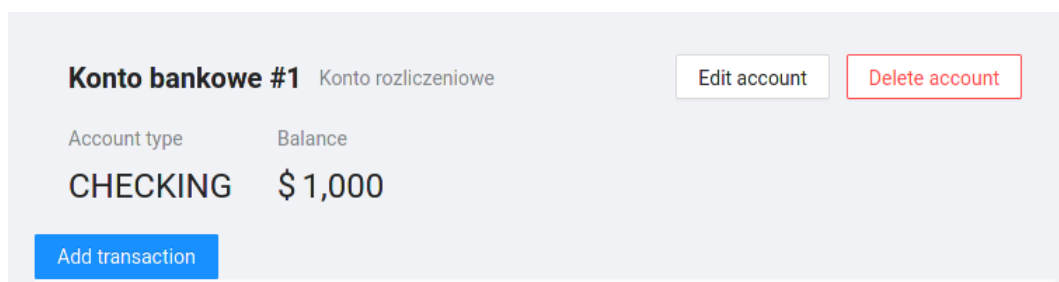


The image shows a modal form titled "Add new account". It includes a close button (X) in the top right corner. The form contains the following fields:

- \* Name:
- Description:
- Initial balance:
- \* Account type:

At the bottom right, there are two buttons: "Cancel" and "Add".

Rysunek 2.1: Formularz tworzenia nowego konta



The image shows a card-like view for an account. It contains the following information:

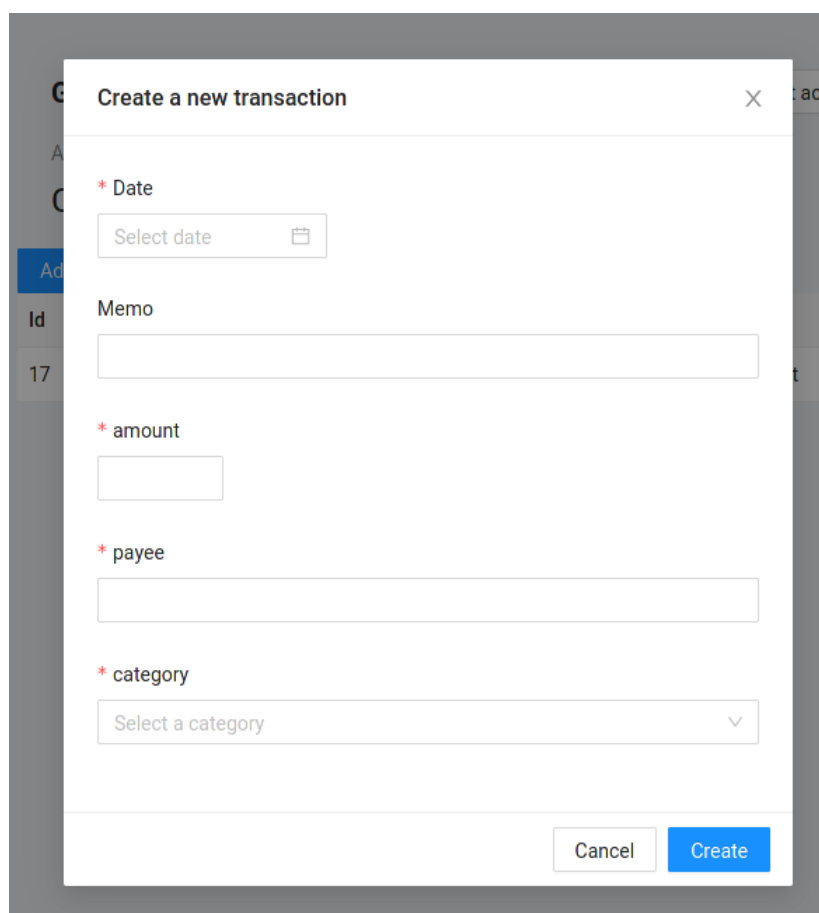
- Konto bankowe #1** Konto rozliczeniowe
- Buttons: Edit account, Delete account
- Account type: CHECKING
- Balance: \$ 1,000
- Button: Add transaction

Rysunek 2.2: Widok informacji o koncie

### 2.2.2. Wyświetlanie operacji

### 2.2.3. Rejestrowanie operacji

Zgodnie z koncepcją aplikacji, użytkownik powinien regularnie rejestrować w programie swoje wydatki, przychody i transfery pieniędzy. Aby to zrobić, należy nacisnąć na przycisk “Add transaction” lub “Add transfer”. Są one widoczne na ekranach transakcji i transferów dla danego konta. Pierwszy z nich służy do zapisywania transakcji, a drugi do zapisywania transferów na danym koncie. Naciśnięcie tych przycisków powoduje wyświetlenie okienka z formularzem. Należy w nim podać dane o operacji. Niektóre pola są wymagane i oznaczone są one znakiem czerwonej gwiazdki. Pozostałe pola są opcjonalne.



The image shows a modal window titled "Create a new transaction" with a close button (X) in the top right corner. The form inside contains the following fields:

- \* Date**: A text input field with the placeholder "Select date" and a calendar icon.
- Memo**: A text input field.
- \* amount**: A text input field.
- \* payee**: A text input field.
- \* category**: A dropdown menu with the placeholder "Select a category" and a downward arrow.

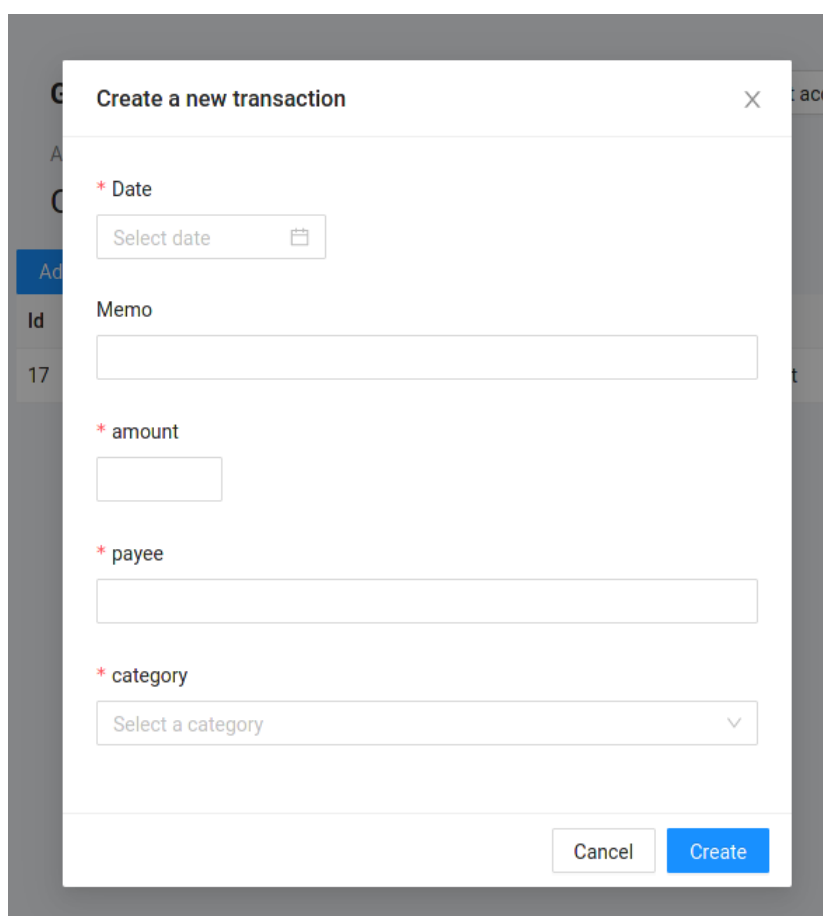
At the bottom right of the form are two buttons: "Cancel" and "Create".

Rysunek 2.3: Formularz dodawania nowej transakcji

### 2.2.4. Wyświetlanie budżetów

### 2.2.5. Tworzenie budżetów

### 2.2.6. Wyświetlanie statystyk



The image shows a modal dialog box titled "Create a new transaction" with a close button (X) in the top right corner. The dialog contains several input fields, each preceded by a red asterisk indicating it is required:

- \* Date**: A date picker field with the placeholder text "Select date" and a calendar icon.
- Memo**: A text input field.
- \* amount**: A numeric input field.
- \* payee**: A text input field.
- \* category**: A dropdown menu with the placeholder text "Select a category" and a downward arrow icon.

At the bottom right of the dialog, there are two buttons: a "Cancel" button and a blue "Create" button.

Rysunek 2.4: Formularz dodawania nowego transferu pomiędzy kontami

## Rozdział 3.

# Użyte rozwiązania technologiczne

Program został podzielony na dwie części. Pierwsza z nich to część serwerowa. Odpowiada ona za logikę programu i przechowywanie danych w trwałej pamięci. Druga część to część kliencka. Jej głównym zadaniem jest prezentacja danych otrzymanych od części serwerowej użytkownikowi. Komunikacja między tymi dwiema warstwami odbywa się przez protokół HTTP. W dalszej części tego rozdziału dokładniej opisuję obie te warstwy programu.

### 3.1. Część serwerowa

Część serwerowa programu została napisana z użyciem języka Java. W kodzie źródłowym możemy wydzielić trzy warstwy. Pierwsza z nich to warstwa dostępu do danych. Ta część programu komunikuje się z bazą danych i tłumaczy jej relacyjny system typów na architekturę obiektową języka Java. Druga warstwa to tzw. warstwa serwisowa. Udostępnia ona główną logikę działania programu i komunikuje się z jednej strony z warstwą dostępu do danych, aby zapisywać informacje w trwałej pamięci, a z drugiej strony udostępnia wyższej warstwie zbiór dostępnych do wykonywania w ramach logiki programu operacji. Tą wyższą warstwą jest warstwa kontrolerów. Kontrolery to obiekty wyspecjalizowane do odbierania żądań ze świata zewnętrznego. Korzystają z warstwy serwisów, aby zlecone żądania wykonać. Kontrolery udostępniają bezstanowy interfejs API dostępu do programu. Aplikacja stworzona jest z wykorzystaniem architektury Representational State Transfer (REST) [4], więc udostępnia zasoby (zasobem jest np. konto, transakcja, budżet), na których można wykonywać operacje zdefiniowane przez protokół HTTP, które powodują zmianę stanu tych zasobów.

### 3.1.1. Spring

Główną technologią użytą w tworzeniu części serwerowej jest framework Spring [5]. Jest to obecnie najpopularniejszy framework do tworzenia aplikacji w języku Java [6]. Pod nazwą Spring kryje się tak naprawdę wiele różnych bibliotek. Cały framework podzielony jest więc na moduły rozwiązujące różne problemy [7] takie jak udostępnianie kontenera Inversion of Control (IoC) [8], tworzenie aplikacji internetowych, tworzenie aplikacji dla środowiska rozproszonego uruchamianych w chmurze, zabezpieczanie aplikacji, dostęp do baz danych i inne. Udostępnianie kontenera IoC jest jedną z głównych funkcji tego frameworka, a implementacja tego wzorca w postaci zasady Dependency Injection jest używana także w mojej aplikacji.

W moim rozwiązaniu użyłem między innymi modułu Spring MVC [9], który używany jest w warstwie prezentacji danych. Dzięki temu rozwiązaniu mogłem z łatwością odczytywać żądania przesyłane do kontrolerów w aplikacji z zewnątrz przez protokół HTTP. Spring MVC bazuje na technologii Servlet API i ułatwia definiowanie zasobów w architekturze REST. Wystarczy stworzyć metody i zdefiniować jakie adresy i metody HTTP mają one obsługiwać. Spring MVC zajmie się za nas niskopoziomowymi kwestiami obsługi żądań takimi jak odbieranie danych, tłumaczenie ich na obiekty języka Java i wysyłanie odpowiedzi.

Aby ułatwić sobie tworzenie aplikacji użyłem także modułu Spring Boot [10]. Spring Framework wymaga konfiguracji używanych elementów poprzez pliki XML, kod Java lub adnotacje w języku Java [11]. Twórcy tej biblioteki zauważyli, że większość programistów wybiera przy tworzeniu programów podobne opcje konfiguracji biblioteki Spring. Aby ułatwić z nią pracę, Spring Boot automatyzuje proces konfiguracji, zarówno elementów dotyczących frameworka Spring jak i innych bibliotek, udostępniając zdefiniowane wcześniej konfiguracje tych elementów. Pozwala to o wiele szybciej rozpocząć tworzenie programów nieprzejmując się pisanem konfiguracji dopóki nie będziemy mieli żadnych specyficznych wymagań. Dzięki temu będziemy mieli mniej powtarzalnego kodu w naszej aplikacji.

### 3.1.2. Schemat bazy danych

### 3.1.3. Inne technologie użyte w części serwerowej

## 3.2. Część kliencka

### 3.2.1. Biblioteka React



## Rozdział 4.

# Podsumowanie

### 4.1. Możliwości dalszej rozbudowy



# Bibliografia

- [1] Strona aplikacji You Need A Budget. <https://www.youneedabudget.com/>.
- [2] Cennik używania aplikacji You Need A Budget. <https://www.youneedabudget.com/pricing/>.
- [3] Strona aplikacji Mint. <https://mint.intuit.com/>.
- [4] Roy Thomas Fielding. *Architectural Styles and the Design of Network-based Software Architectures*. PhD thesis, University of California, Irvine, 2000.
- [5] Framework Spring. <https://spring.io/>.
- [6] Raport popularności narzędzi używanych w ekosystemie tworzenia aplikacji w języku Java. <https://snyk.io/jvm-ecosystem-report-2021/>.
- [7] Moduły frameworka Spring. <https://spring.io/projects>.
- [8] Wzorzec Inversion of Control i zasada Dependency Injection. <https://www.martinfowler.com/articles/injection.html>.
- [9] Dokumentacja Spring MVC. <https://docs.spring.io/spring-framework/docs/current/reference/html/web.html>.
- [10] Spring Boot. <https://spring.io/projects/spring-boot>.
- [11] Adnotacje w języku Java. <https://docs.oracle.com/javase/tutorial/java/annotations/>.