Aplikacja web umożliwiająca organizowanie zbiórek pieniędzy

Organizacja i rozwój projektów Open Source
i
Projektowanie aplikacji internetowych

skład sekcji:

Dorian Barański (PAI, OiRPOS)
Bartosz Prusak (PAI, OiRPOS)

Webowa aplikacja umożliwiająca korzystającym użytkownikom zakładanie zbiórek pieniężnych oraz wspieranie innych osób, poprzez wpłaty na założone wcześniej zrzutki.

Wymagania funkcjonalne:

Ogólne wymagania dotyczące systemu oraz użytkownika:

- Rejestracja nowych użytkowników za pośrednictwem podania m.in. adresu email, hasła, daty urodzenia, imienia i nazwiska,
- logowanie się przez użytkownika do systemu za pomocą adresu email i hasła,
- edycja swojego profilu przez użytkownika - dotyczy zmiany adresu email lub imienia/nazwiska w wypadku pomyłki,
- system zapewnia weryfikację danych - na potrzeby uproszczonego systemu odbywałaby się weryfikacja wiekowa użytkowników przy rejestracji,
- użytkownik może dodać zrzutkę do listy obserwowanych zrzutek - w celu późniejszego filtrowania listy po takim parametrze.

Wymagania związane z instancją zrzutki:

- Zautoryzowany użytkownik ma możliwość założenia nowej zrzutki - podając m.in. nazwę zrzutki, opis, cel pieniężny - musi także przyporządkować zrzutkę do jednej z dostępnych kategorii,
- Zautoryzowany oraz anonimowy (niezalogowany) użytkownik może przeglądać aktualnie aktywne zrzutki oraz zrzutki niedawno zakończone - przez pewien wyznaczony od zakończenia czas,
- Użytkownik będący założycielem zrzutki może ją modyfikować - zmieniać opis, cel pieniężny,
- Założyciel zrzutki może także usunąć swoją zrzutkę, pod warunkiem, że jest zalogowany na swoje konto w serwisie,
- Zautoryzowany użytkownik ma możliwość dodawania komentarzy pod zrzutkami,
- Możliwość zgłaszania zbiórek, które wyglądają na niegodne zaufania. Po zgłoszeniu w zbiórce wyświetlona zostanie informacja o tym, ile użytkowników uważa tę zbiórkę za potencjalnie szkodliwą,
- Pod każdą zrzutką wyświetlona jest historia wpłat (z uwzględnieniem anonimowych darowizn).
- W aplikacji znajduje się sekcja z opisem działania zrzutek i samego portalu,
- W aplikacji jest możliwość wyszukiwania zrzutek według nazwy bądź filtrowanie z wykorzystaniem wybranego filtra.

Wymagania odnośnie dokonywania wpłat:

- możliwe jest wykonywanie wpłat dla użytkowników zautoryzowanych oraz anonimowych,
- możliwość ukrycia danych wpłacającego - wpłaty anonimowe lub wpłaty z wykorzystaniem pseudonimu w wypadku użytkowników posiadających zarejestrowane konto.

Kryteria akceptacji wybranych funkcjonalności:

- Niezalogowany użytkownik nie będzie miał dostępu do paneli/przycisków dostępnych tylko dla zautoryzowanych użytkowników, np. do tworzenia zrzutki,
- Po dokonaniu wpłaty na wybraną zrzutkę zmieni się zebrana dotychczas kwota oraz historia wpłat zostanie zaktualizowana o nowego darczyńcę.
- Podczas dokonywania wpłaty będzie widoczna opcja pozwalająca na anonimizację wpłaty. Po jej wybraniu w historii wpłat nie pojawi się imię i nazwisko darczyńcy, a jedynie adnotacja, że wpłata została dokonana anonimowo.
- Wybór kryterium sortowania zrzutek (np. według ilości wpłat) zmieni kolejność wyświetlania elementów.
- Po modyfikacji danych osobowych w panelu użytkownika i zatwierdzeniu, zmiany będą widoczne w profilu.

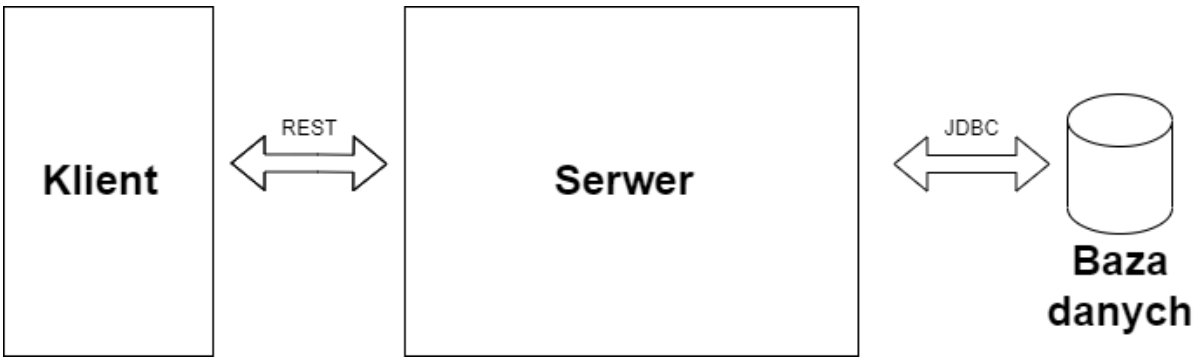
Wymagania niefunkcjonalne:

- Wykorzystanie technologii Spring Boot, PostgreSQL, Angular,
- Bezpieczne uwierzytelnianie z wykorzystaniem JWT Token,
- Bezpieczne przechowywanie wrażliwych danych użytkowników,
- Autoryzacja dostępu do różnych części systemu,
- Płynność przełączania podstron,
- Szybki czas reakcji strony na kliknięcie myszą,
- Łatwość użycia,

Architektura:

- System oparty o architekturę klient-serwer
- Serwer stanowi monolit udostępniający interfejs REST
- Serwer łączy się z zewnętrznym serwerem bazodanowym poprzez sterownik JDBC
- Autoryzacja użytkowników systemu odbywa się przy pomocy tokena JWT
- Po stronie klienta uruchamiana jest aplikacja przeglądarkowa wchodząca w interakcje z systemem poprzez żądania HTTP zgodne z REST wysyłane do serwera

Poglądowy schemat architektury:



Wybór architektury klient-serwer uzasadnia się oddzieleniem zapewniania usług dla klientów, a zgłaszaniem żądań obsługi. Współczesne urządzenia i przeglądarki internetowe są w stanie zapewnić odpowiednie zasoby do uruchomienia części prezentacyjnej aplikacji po stronie klienta.

Z racji na prototypowy charakter rozwiązania zdecydowaliśmy się na architekturę monolityczną - zastosowanie mikroserwisów zwiększyłoby stopień skomplikowania systemu oraz wydłużyłoby czas potrzebny na uzyskanie produktu z wystarczającą liczbą funkcji, z której mogą korzystać pierwsi użytkownicy. Z powodu ograniczonego grona odbiorców nie ma konieczności stosowania systemów równoważenia obciążenia.

Technologie:

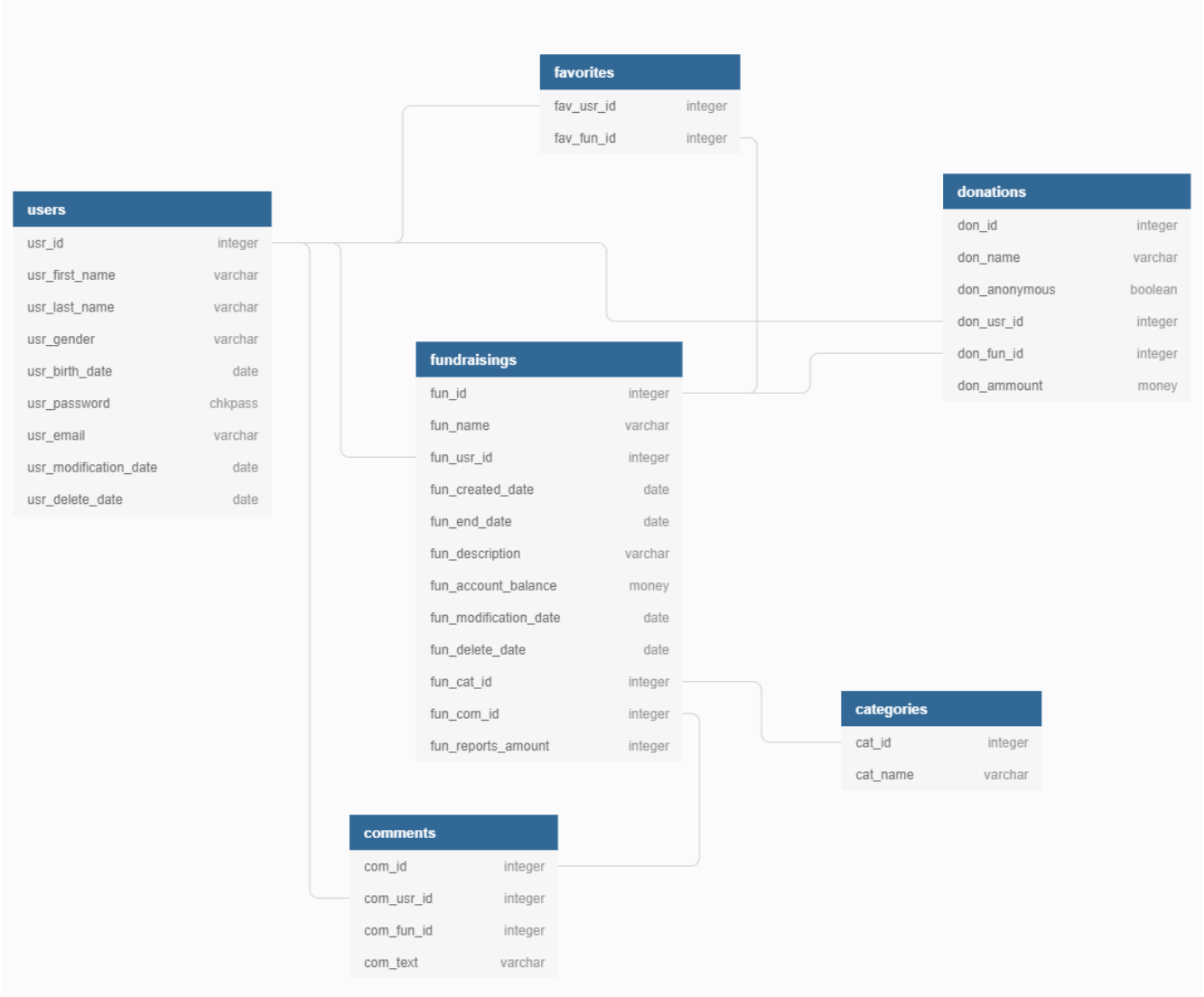
Skorzystamy z frameworków Spring Boot w języku Java oraz frameworku Angular po stronie frontend'owej projektu. Technologie wybrane zostały na podstawie doświadczenia naszego zespołu a także są powszechnie używane na obecnym rynku.

Alternatywą dla Spring Boot jest framework Django z języka Python natomiast tutaj próg wejścia mógłby być zbyt duży - przy dobrej bądź bardzo dobrej znajomości Spring Boot nieopłacalne jest uczenie się na nowo innego frameworku - problemy mogłoby stwarzać dostarczenie kodu o podobnej jakości jak w przypadku znanej technologii. Na podobnej zasadzie został odrzucony NodeJS.

Angular mógł zostać zastąpiony przez React lub Flutter - podobnie jak dla strony serwerowej również i tutaj wybór uzasadniony był doświadczeniem jakie posiadamy w korzystaniu z tych technologii.

W przypadku bazy danych skorzystamy z PostgreSQL. Alternatywami mogły być MariaDB, Oracle, Sqlite lub nierelacyjne bazy danych jak chociażby MongoDB. Postgres jest rozwiązaniem darmowym, dodatkowo możliwe jest łatwe umieszczenie bazy na darmowym hostingu elephantsql - pozwoli to uniknąć problemów z synchronizacją lokalnych środowisk podczas procesu rozwijania aplikacji.

Schemat bazy danych:



Projekt strony głównej aplikacji: