



AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE

Wydział Elektrotechniki, Automatyki, Informatyki i Inżynierii Biomedycznej

Projekt dyplomowy inżynierski

Aplikacja mobilna optymalizująca zakupy książek w serwisie allegro.pl
Mobile application to optimize the process of book shopping at
allegro.pl

Autor:

Miłosz Szwedo

Kierunek studiów:

Informatyka

Opiekun pracy:

dr inż. Mirosław Gajer

Kraków, 2020

Uprzedzony o odpowiedzialności karnej na podstawie art. 115 ust. 1 i 2 ustawy z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (t.j. Dz.U. z 2006 r. Nr 90, poz. 631 z późn. zm.): „Kto przywłaszcza sobie autorstwo albo wprowadza w błąd co do autorstwa całości lub części cudzego utworu albo artystycznego wykonania, podlega grzywnie, karze ograniczenia wolności albo pozbawienia wolności do lat 3. Tej samej karze podlega, kto rozpowszechnia bez podania nazwiska lub pseudonimu twórcy cudzy utwór w wersji oryginalnej albo w postaci opracowania, artystycznego wykonania albo publicznie zniekształca taki utwór, artystyczne wykonanie, fonogram, wideogram lub nadanie.”, a także uprzedzony o odpowiedzialności dyscyplinarnej na podstawie art. 211 ust. 1 ustawy z dnia 27 lipca 2005 r. Prawo o szkolnictwie wyższym (t.j. Dz. U. z 2012 r. poz. 572, z późn. zm.): „Za naruszenie przepisów obowiązujących w uczelni oraz za czyny uchybiające godności studenta student ponosi odpowiedzialność dyscyplinarną przed komisją dyscyplinarną albo przed sądem koleżeńskim samorządu studenckiego, zwanym dalej «sądem koleżeńskim».”, oświadczam, że niniejszą pracę dyplomową wykonałem(-am) osobiście i samodzielnie i że nie korzystałem(-am) ze źródeł innych niż wymienione w pracy.

Serdecznie dziękuję ... tu ciąg dalszych podziękowań np. dla promotora, żony, sąsiada itp.

Spis treści

1. Wprowadzenie	7
1.1. Temat pracy	7
1.1.1. Stworzenie czegośtam	7
1.2. Cele pracy	7
1.2.1. Stworzenie czegośtam	7
1.3. Zawartość pracy	7
1.3.1. Zawiera cośtam	7
2. Streszczenie	9
2.1. Streszczenie	9
2.2. Summary	9
3. Projekt aplikacji	11
3.1. Architektura	11
3.2. Auth service	13
3.2.1. JSON Web Token	13
3.2.2. Autoryzacja, a autentykacja	14
3.3. Gateway	14
3.4. OffersFetcher	15
3.5. Zewnętrzne API	15
3.6. Baza danych	16
3.7. Aplikacja mobilna	17
4. Implementacja	19
4.1. Technologie	19
5. Użytkowanie	21
5.1. Struktura dokumentu	21
6. Podsumowanie	23
6.1. no jest to napisane i w ogóle	23

1. Wprowadzenie

Książki książki szukam ich i w ogóle

Allegro to cośtam

Ale problemem jest cośtam

1.1. Temat pracy

Tematem pracy jest aplikacja o charakterze mikroserwisowym czy co

1.1.1. Stworzenie czegośtam

1.2. Cele pracy

Celem poniższej pracy jest

1.2.1. Stworzenie czegośtam

1.3. Zawartość pracy

1.3.1. Zawiera cośtam

2. Streszczenie

2.1. Streszczenie

Lorem ipsum dolor sit ameth

2.2. Summary

Lorem ipsum dolor sit ameth (po angielsku)

3. Projekt aplikacji

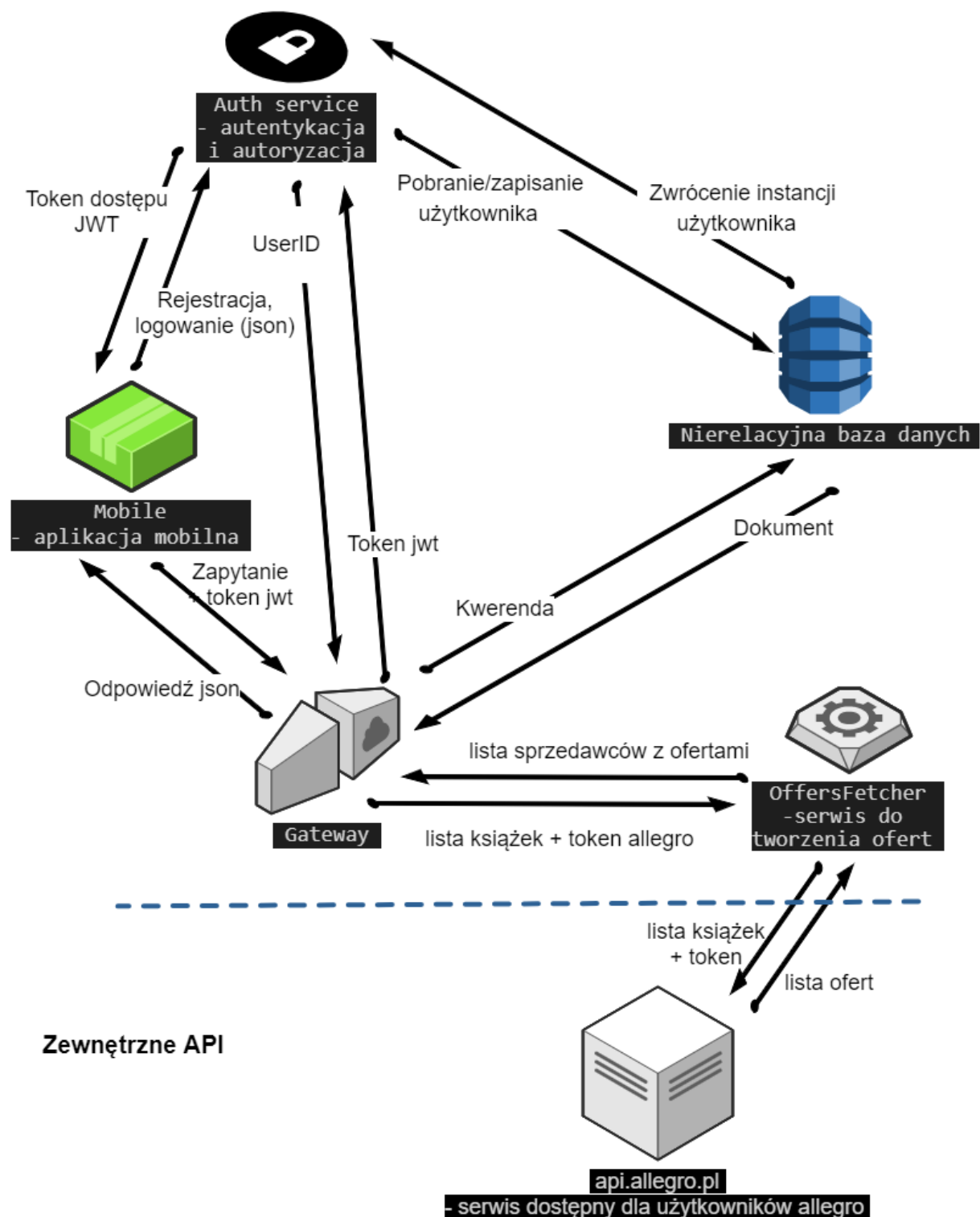
3.1. Architektura

Architektura aplikacji jest złożona z części mobilnej oraz czterech rozproszonych serwisów, z czego każdy występuje jako autonomiczna aplikacja z którą porozumiewanie odbywa się za pomocą protokołu HTTP. Warstwa prezentacyjna, porozumiewając się z pozostałymi serwisami zapewnia użytkownikowi płynną interakcję z systemem w celu osiągnięcia zamierzonych akcji dostępnych w obrębie funkcjonalności.

W ten sposób każda składowa część aplikacji może być niezależnie zarządzana. W momencie w którym pojedynczy element odpowiedzialny za szczególną usługę jest wyłączony, sama aplikacja może dalej działać wyłączając tylko funkcjonalności dostarczane przez niedostępny aktualnie serwis.

Takie podejście można określić mianem zorientowanym na usługi. Oznacza to, że przy tworzeniu systemu, spory nacisk kładziony jest na definiowanie spełniających wymagania użytkownika usług. Są one elementami oprogramowania zdolnymi do niezależnego funkcjonowania, udostępniającymi realizowane funkcje poprzez zdefiniowany interfejs.

HTTP (Hypertext Transfer Protocol), czyli “Protokół Przesyłania Danych Hiper-tekstowych to protokół warstwy aplikacji, odpowiedzialny za transmisję dokumentów hiper-medialnych, jak np. HTML. Został stworzony do komunikacji pomiędzy przeglądarkami, a serwerami webowymi, ale może być używany również w innych celach. HTTP opiera się na klasycznym modelu klient-serwer, gdzie klient inicjuje połączenie poprzez wysłanie żądania, następnie czeka na odpowiedź. HTTP jest protokołem bezstanowym, co oznacza, że serwer nie przechowuje żadnych danych (stanów) pomiędzy oboma żądaniami. (...)“[1]



Rys. 3.1. Struktura systemu

3.2. Auth service

Auth service dba o zachowanie bezpieczeństwa w całym systemie. Poprzez ekstrakcję funkcjonalności związanej z tworzeniem kont, logowaniem oraz zarządzaniem dostępem do pozostałych sektorów, gwarantuje niezawodną autentykację i autoryzację użytkownika pragnącego korzystać z aplikacji.

Informacje o kontach użytkowników przechowywane są w bazie danych, do której dostęp uzyskać można tylko za pomocą wygenerowanego przez nią, wewnętrznego klucza.

W celu swobodnego poruszania się po aplikacji należy uzyskać JWT(JSON Web Token). Aby pozyskać token należy się zarejestrować lub zalogować na ekranie logowania. Zapytanie utworzone w ten sposób zostanie wysłane do Auth service. W odpowiedzi przesłany zostanie wyżej wymieniony klucz dostępowy.

3.2.1. JSON Web Token

JSON Web Token to otwarty standard, który definiuje kompaktowy i samodzielny sposób na bezpieczny transfer danych. Poszczególne instancja składa się z trzech części oddzielonych kropkami w bezpośrednim formacie `xx..x.y..yy.zz..z`, gdzie poszczególne człony reprezentują: [2]

1. Header - nagłówek, zawierający dwie informacje:
 - typ tokenu, w tym przypadku "JWT"
 - algorytm szyfrujący(n.p. HMAC, SHA256 lub RSA)
2. Payload - lista wyrażeń opisujących szyfrowaną informację, w przypadku użytkownika - np jego login, czy email.
3. Signature - podpis stworzony poprzez zaszyfrowanie podanym w headerze algorytmem szyfrującym ciągu składającego się z
 - zakodowanego za pomocą Base64 (specjalnego kodowania transportowego) nagłówka i listy wyrażeń
 - sekretu, czyli unikalnego dla danych klucza.

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  
eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4  
gRG91IiwiaXNTb2NpYWwiOnRydWV9.  
4pcPyMD09olPSyXnrXCjTwXyr4BsezdI1AVTmud2fU4
```

Rys. 3.2. Przykładowy token jwt [2]

3.2.2. Autoryzacja, a autentykacja

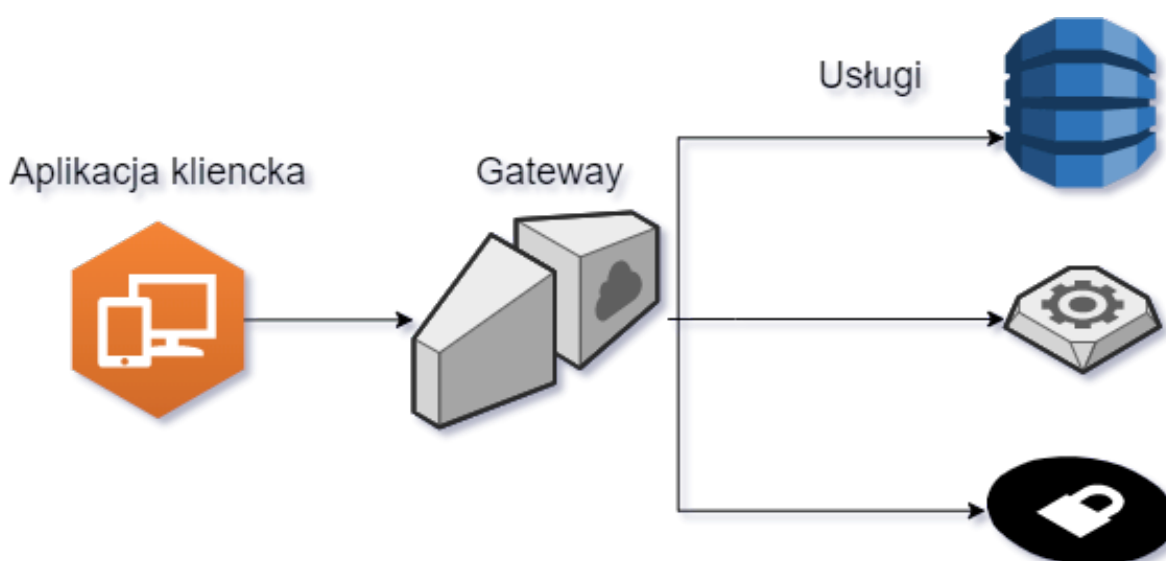
Warto implicity rozróżnić dwa bardzo ważne pojęcia związane z bezpieczeństwem aplikacji ze względu na częstotliwość z jaką są one mylone.

Autentykacja - często też w dwóch częściach jako identyfikacja i uwierzytelnienie. Polega na potwierdzeniu tożsamości, to znaczy określeniu, czy podmiot procesu jest tym za kogo się podaje. Na przypadku logowania, strona ufająca otrzymuje od użytkownika podstawue stwierdza, czy użytkownik może być pozytywnie zweryfikowany.

Autoryzacja to potwierdzenie, czy dany użytkownik jest uprawniony do skorzystania z konkretnego zasobu. Na tym etapie autentykacja została ewaluowana pozytywnie. Nie oznacza to jednak, że dany podmiot posiada dostęp w żądanym zakresie.

3.3. Gateway

Gateway to serwis zbudowany według podejścia zwanego wzorcem bramy interfejsu API[3]. Jest to element znajdujący się pomiędzy klientem a rozproszonymi usługami. Dzięki temu w prosty sposób można kontrolować wszelkie zapytania skierowane do poszczególnych serwisów. Jest to więc centralny punkt systemu, który ma na celu uproszczenie komunikacji warstwy prezentacyjnej z poszczególnymi usługami. Każde zapytanie wysłane do bramy zostaje zweryfikowane pod względem bezpieczeństwa. Następnie w zależności od potrzeb, modyfikowane, lub bezpośrednio przesłane dalej.



Rys. 3.3. Gateway - schemat

3.4. OffersFetcher

OffersFetcher to główna jednostka licząca w systemie. Usługa ta otrzymuje żądanie z listą książek oraz token dostępowy do REST API portalu Allegro. (3.5.) Dla każdej książki wykonywane jest odpowiednio zmodyfikowane zapytanie, którego rezultat jest przetwarzany i odkładany do odpowiedniej kolekcji, aby na koniec zostać wkomponowanym w pożądany rezultat. Analizowane są wszystkie obecnie dostępne w czasie rzeczywistym oferty sprzedaży w serwisie Allegro.pl.

Dane otrzymane w ten sposób są przetwarzane i grupowane po unikalnym identyfikatorze sprzedawcy. Serwis zwraca odpowiedź w postaci listy zbiorów przedmiotów, które wpisują się w pozycje otrzymane w zapytaniu.



Rys. 3.4. Poszukiwane książki i bazująca na nich przykładowa oferta

3.5. Zewnętrzne API

Źródłem danych dla ofert tworzonych w serwisie OffersFetcher (3.4.) jest Allegro REST API udostępnione przez Allegro.pl, czyli platformę transakcyjną on-line przedsiębiorstwa Allegro.pl. Portal ten umożliwia użytkownikom wystawianie na sprzedaż posiadanych przez nich przedmiotów oraz na korzystanie z ofert innych sprzedawców.

“Allegro REST API działa w oparciu o protokół HTTP (...) Autoryzacja realizowana jest w standardzie OAuth2.”[4]

REST API (**RE**presentational **S**tate **T**ransfer) to styl architektury oprogramowania w którym dane i funkcjonalności są odzwierciedlone poprzez Ujednolicone Identyfikatory Zasobów(w skrócie URI). Termin ten został stworzony przez Roy Fielding w 2000 roku[5].Dostęp uzyskiwany jest poprzez proste i jasno zdefiniowane operacje. Istnieje pięć obowiązkowych ograniczeń, które dokładnie definiują charakter tego podejścia:

- bezstanowość - każde zapytanie do serwera powinno zawierać wszystkie informacje potrzebne do jego zrozumienia.
- użycie buforownia podręcznego - jeżeli dane są lokalnie przechowywane, należy o tym bezpośrednio poinformować.
- system warstwowy - istnieje możliwość użycia wielu komponentów do poszczególnych funkcjonalności, które razem stanowią jedno API. Klient przeważnie nie jest w stanie określić, czy jego połączenie jest realizowane z serwerem końcowym czy którymś z pośredników.
- rozdział klienta od serwera - obie części powinno się być w stanie rozwijać osobno i niezależnie. Klient powinien jedynie znać URI, które może odpytywać.
- ujednolicony interfejs - należy deterministycznie zdefiniować i nie zmieniać adresów pod którymi dostępne będą zasoby.

[6]

3.6. Baza danych

Warstwa persystencyjna jako osobny i niezależny serwis ma zadanie utrzymywać stan aplikacji. Jest to ogromnie ważny element systemu, którego działanie niezbędne jest np dla Auth service(3.2) ze względu na posiadane informacje o użytkownikach, które używane są w celu autoryzacji i autentykacji. Oprócz danych dostępowych, dla każdego klienta przechowywane są również zbiory książek - posiadanych i poszukiwanych.

Złożone bazy danych można podzielić ze względu na struktury organizacji danych, które przechowują. Są to kolejno relacyjne, obiektowe, relacyjno-obiektowe, strumieniowe, temporalne, nierelacyjne (NoSQL). Najczęściej spotykane są nadal bazy relacyjne, gdzie dane występują pod postacią powiązanych między sobą tabel. Posiadają one wewnętrzne języki programowania, wykorzystujące zwykle SQL, służące do wykonywania zaawansowanych operacji. Popularność jednak zyskują ostatnio bazy nierelacyjne, czyli takie, które nie posiadają tabel ani relacji, tak więc i nie wykorzystują języka SQL. Nie jest również najczęściej wymagane, aby struktura danych była jednorodna, jak jest to podyktowane użyciem tabel.

3.7. Aplikacja mobilna

4. Implementacja

W rozdziale tym przedstawiono podstawowe informacje dotyczące struktury prostych plików \LaTeX . Omówiono również metody kompilacji plików z zastosowaniem programów *latex* oraz *pdflatex*.

4.1. Technologie

5. Użytkowanie

W rozdziale tym przedstawiono podstawowe informacje dotyczące struktury prostych plików \LaTeX a. Omówiono również metody kompilacji plików z zastosowaniem programów *latex* oraz *pdflatex*.

5.1. Struktura dokumentu

6. Podsumowanie

W rozdziale tym przedstawiono podstawowe informacje dotyczące struktury prostych plików \LaTeX . Omówiono również metody kompilacji plików z zastosowaniem programów *latex* oraz *pdflatex*.

6.1. no jest to napisane i w ogóle

Bibliografia

- [1] Autorzy MDN. <https://developer.mozilla.org/pl/docs/Web/HTTP>.
- [2] Auth0 Inc. <https://jwt.io/introduction/>.
- [3] Chris Richardson. *Apigateway*. <https://microservices.io/>.
- [4] *Allegro REST API*. <https://developer.allegro.pl/>.
- [5] Roy Thomas Fielding. "Architectural Styles and the Design of Network-based Software Architectures". PhD thesis. University of California, 2000.
- [6] <https://restfulapi.net/>. *Rest API*.