POLITECHNIKA WROCŁAWSKA

WYDZIAŁ INFORMATYKI I TELEKOMUNIKACJI

PROJEKT Z INTERNETOWYCH BAZ DANYCH

Bookshop - księgarnia internetowa

Termin zajęć: Poniedziałek, 12:15–14:30

Autor/Autorzy:

Kamil Dywan
Indeks: 254049
Paweł Kuźma
Indeks: 252778
Mateusz Urbańczyk
Indeks: 252808

Prowadzący zajęcia:

dr inż. Roman Ptak, W4N

Wrocław, 2022 r.

1. Wstęp

1.1. Cel projektu

Projekt, implementacja oraz wdrożenie księgarni internetowej składającej się z bazy danych, interfejsu użytkownika i serwera dla firmy „Bookshop'' znajdującej się na terenie Wrocławia oraz zajmującej się stacjonarną sprzedażą książek.

1.2. Plan projektu:

- Projektowanie:
    - Analiza SWOT
    - Dobór narzędzi
    - Wymagania funkcjonalne i niefunkcjonalne
    - Diagramy UML
    - Diagramy ERD
    - Prototypy interfejsu użytkownika
- Implementacja:
    - Relacyjna baza danych
    - Backend
    - Aplikacja kliencka (frontend)
- Testowanie:
    - Testy integracyjne backendu
    - Testy jednostkowe
    - Testy manualne interfejsu użytkownika
    - Testy automatyczne interfejsu użytkownika
- Wdrożenie

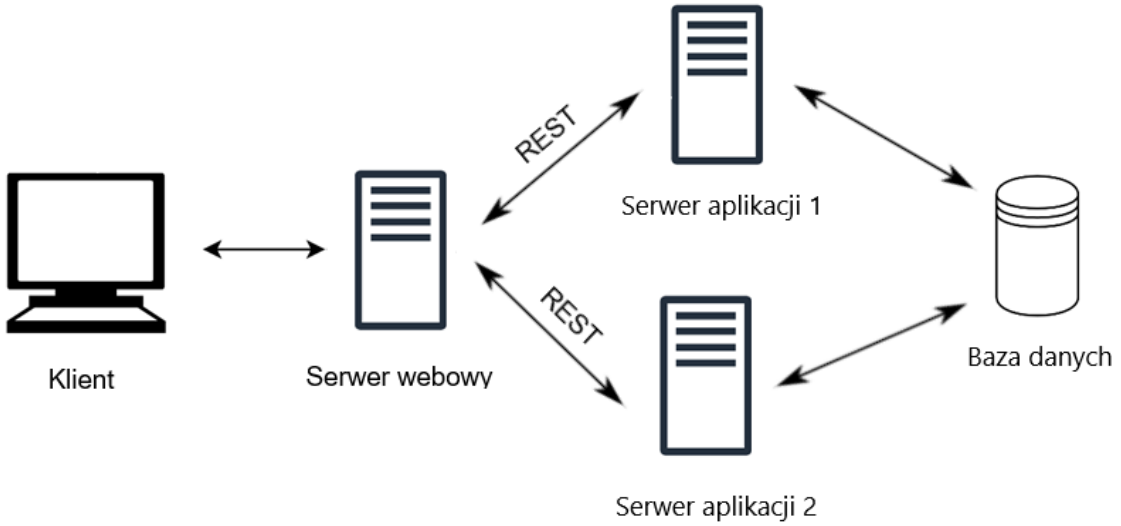
1.3. Zakres projektu:

Zakres projektu obejmuje stworzenie i wdrożenie strony internetowej, która ma za zadanie ułatwić pracownikom zarządzanie zasobami księgarni (np. określenie listy dostępnych książek oraz ustalenie ich cen) oraz umożliwić klientom przeglądanie i zamawianie książek przez internet, które zamówione będą mogły być zakupione w sklepie stacjonarnym. Powinna być również opcja założenia konta przez klienta, które to konto będzie umożliwiało użytkownikowi przeglądanie listy dokonanych zamówień, sprawdzenie statusu zamówień, czy wycofanie zamówień, które nie są jeszcze w trakcie realizacji.

1.3. Architektura aplikacji

Realizowany system jest rozproszonym serwisem webowym, który w dużym uogólnieniu można opisać jako system typu klient-serwer. Klient wysyła żądanie do serwera

aplikacji, a następnie serwer ten odpowiednio przetwarza otrzymane żądanie i zwraca klientowi odpowiedź, którą to później odpowiedź klient interpretuje i przedstawia użytkownikowi (w tym przypadku jest to GUI interfejsu webowego).



Zdecydowano się na zaimplementowanie systemu w postaci serwisu webowego, gdyż taka forma aplikacji zapewnia dużą wygodę użytkowania z powodu braku konieczności jej zainstalowania przez użytkownika, a z powodu tego, że założono, iż system powinien być dostępny dla każdego, kto posiada dostęp do Internetu, to nie ma wymogu instalowania aplikacji. Następnym powodem wybrania tego typu aplikacji jest to, że aplikacje webowe są obecnie najpopularniejszym typem systemu i implementacja takiej aplikacji pozwala na zdobycie wartościowego doświadczenia.

1.3.1 Baza danych

Baza danych to warstwa systemu odpowiedzialna za przechowywanie danych. Założono, że w systemie zostanie wykorzystana relacyjna baza danych (SQL). Jako bazę danych wybrano bazę relacyjną, gdyż bazy te charakteryzują się tym, że zapewniają integralność danych poprzez tworzenie relacji np. poprzez powiązanie tabel kluczem obcym, a w realizowanym systemie wykryto wiele przypadków, w których można by było utworzyć relacje np. klient jest powiązaniem między encjami użytkownika i zamówień.

1.3.2. Serwer aplikacji - Backend

Backend jest odpowiedzialny za przyjmowanie żądań od klienta, odpowiednie przetwarzanie tych żądań, wykonywanie pewnych operacji na danych przechowywanych w bazie danych na podstawie otrzymanych przez klienta danych i przekazywanie klientowi adekwatnej odpowiedzi. Backend udostępnia klientowi punkty końcowe (ang. API endpoints)

do których klient może przesłać żądanie. Punkty końcowe posiadają adresy URL np. http://localhost:9000/user, które składają się nazwy protokołu (http), nazwy hosta docelowego (localhost), portu docelowego (9000) oraz reszty, która identyfikuje punkt końcowy oraz jego przeznaczenie (dla przykładu punkt końcowy o końcówce adresu user związany jest z użytkownikami i klient może np. wysłać żądanie na ten endpoint do pobrania danych użytkownika o danym id). Ważnym elementem w przypadku backendu jest również wysłanie klientowi odpowiedniego statusu odpowiedzi oraz ewentualne dołączenie informacji zwrotnej do wiadomości, tak aby klient mógł odpowiednio zareagować na otrzymaną odpowiedź (np. w przypadku otrzymania negatywnej odpowiedzi, klient będzie mógł wyświetlić użytkownikowi komunikat o błędzie, a w przypadku otrzymania odpowiedzi negatywnej, klient będzie mógł wyświetlić komunikat o sukcesie). Warstwa ta jest w ścisłym powiązaniu z warstwą bazy danych.

1.3.2.1 Serwer aplikacji 1

Warstwa ta będzie odpowiedzialna za obsługę głównej części funkcji, które w założeniu ma oferować aplikacja webowa.

1.3.2.1 Serwer aplikacji 2

Warstwa ta będzie odpowiedzialna za przeglądanie listy dostępnych książek.

1.3.3. Klient - Frontend

Frontend jest odpowiedzialny za wysyłanie żądań do warstwy backendowej i następnie odpowiednie przetwarzanie oraz wyświetlanie danych otrzymanych w odpowiedzi od backendu. W systemie tym klientem jest strona internetowa renderowana po stronie użytkownika. Klient jest aplikacją typu SPA (ang. Single Page Application), co oznacza, że aplikacja składa się z jednej strony HTML, a jej zawartość jest zmieniana dynamiczne poprzez JavaScript. Zdecydowano się na aplikację typu SPA, a nie MPA (ang. Multi Page Application) głównie dlatego, że aplikacje SPA działają szybciej z powodu tylko jednej strony HTML, co jest w przypadku tego systemu konieczne, gdyż przewidziano, że aplikacja będzie się składać z wielu podstron. Kolejnymi kryteriami na korzyść SPA były łatwiejsze wdrożenie, gdyż wystarczą jedynie 3 pliki (HTML, CSS i JavaScript), łatwiejsze wprowadzenie dynamicznej zawartości oraz ułatwione tworzenie aplikacji.

1.3.3. Serwer WWW

Serwer obsługujący żądania protokołu http.

1.3.4 REST

Komunikacja w systemie między frontendem i backendem, frontendem i serwerem bezpieczeństwa oraz backendem i serwerem bezpieczeństwa odbywa się za pomocą REST. REST jest to sposób i format w jaki komunikuje się klient z serwerem. Serwer udostępnia klientowi punkty końcowe (end-pointy), do których klient może wysłać żądania http przesyłając przy tym jakieś dane np. tytuł wyszukiwanego artykułu. W skrócie komunikacja REST odznacza się następującymi cechami:

- Bezstanowość (nie przechowuje informacji o poprzednich wymianach wiadomości),
- Architektura klient-serwer,
- Jednolity interfejs komunikacyjny – dzięki temu możliwe jest np. komunikowanie się systemów zaimplementowanych w różnych językach programowania, czy zainstalowanych na różnych platformach (mobilna, czy aplikacja webowa)
- Wykorzystywanie protokołu HTTP

1.3.5 HTTP

Protokół warstwy aplikacji opisujący zasady przesyłania zawartości w Internecie. Protokół ten jest oparty na komunikacji typu klient-serwer, przy czym klientem jest najczęściej przeglądarka. Klientem wysyła żądanie do serwera, a serwer wysyła do klienta odpowiedź. Zawartość wiadomości HTTP można przede wszystkim podzielić na nagłówek (ang. Header) oraz ładunek (ang. Body). W nagłówku znajdują się pola typu klucz-wartość. Nagłówek żądania różni się od nagłówka odpowiedzi poprzez dostępne opcje, ale mają one również wspólne pola np. informacja o hoście docelowym, czy format wiadomości. W nagłówku żądania można wyróżnić jeszcze np. nagłówek Authorization, w którym umieszczane są poświadczenia uwierzytelnienia. W nagłówku odpowiedzi można za to wyróżnić m.in. status odpowiedzi, który np. informuje o tym, czy żądanie zostało wykonane pomyślnie (np. status 200), czy też wystąpił błąd, a jeśli wystąpił, to z jakiego powodu (np. status 401 oznaczający nieudane uwierzytelnienie użytkownika). Ładunek jest opcjonalnym polem, które stanowi treść wiadomości HTTP i np. w przypadku odpowiedzi może zawierać listę wyszukanych produktów.

1.3.6 OAuth 2.0

1.4. Techniczna wykonalność systemu

1.4.1. Projektowanie

a) Visual Paradigm:
   - Zakres: diagramy UML i ERD
   - Wersja: 17.0
   - Licencja: Academic license
   - Koszt: 0 zł

b) Figma:
   - Zakres: projektowanie makiet interfejsu użytkownika
   - Licencja: Freeware
   - Koszt: 0 zł

1.4.2. Implementacja

a) MySQL:
   - Zakres: stworzenie relacyjnej bazy danych
   - Wersja: 8.0.31
   - Licencja: GPL license
   - Koszt: 0 zł

b) Java:
   - Zakres: aplikacja serwerowa 1
   - Wersja: 17
   - Licencja: Oracle No-Fee Terms and Conditions
   - Koszt: 0 zł

c) Apache Maven:
   - Zakres: aplikacji serwerowa 1 - budowanie projektów Javowych
   - Wersja: 3.8.6
   - Licencja: Apache License 2.0
   - Koszt: 0 zł

d) Lombok:
   - Zakres: aplikacji serwerowa 1 - generowanie getterów, setterów, konstruktorów itp. dla projektów Javowych
   - Wersja: 1.18.24
   - Licencja: MIT
   - Koszt: 0 zł

e) Mapstruct:

- Zakres: aplikacji serwerowa 1 - generowanie mapowań obiektów dla projektów Javowych
- Wersja: 1.15.3
- Licencja: Apache License 2.0
- Koszt: 0 zł

f) Spring Boot:
- Zakres: framework Javy dla aplikacji serwerowej 1
- Wersja: 2.7.5
- Licencja: Apache License 2.0
- Koszt: 0 zł

g) Python:
- Zakres: aplikacja serwerowa 2
- Wersja: 3.10.8
- Licencja: Open Source
- Koszt: 0 zł

h) Django:
- Zakres: framework Pythona dla aplikacji serwerowej 2
- Wersja: 4.1.2
- Licencja: 3-clause BSD license
- Koszt: 0 zł

i) HTML:
- Zakres: aplikacja kliencka - zawartość stron
- Wersja: 5.3
- Licencja: Royalty-free license
- Koszt: 0 zł

j) CSS:
- Zakres: aplikacja kliencka - wygląd strony
- Wersja: 3
- Licencja: MIT
- Koszt: 0 zł

k) JavaScript:
- Zakres: aplikacja kliencka - logika strony
- Wersja: ECMAScript 2021
- Licencja:
- Koszt: 0 zł

l) React:

- Zakres: biblioteka JavaScript dla aplikacji klienckiej
- Wersja: 18.2.0
- Licencja: MIT
- Koszt: 0 zł

ł) IntelliJ IDEA Ultimate:
- Zakres: środowisko programistyczne dla aplikacji serwerowej 1
- Wersja: 2022.2.3
- Licencja: Trialware (dla celów edukacyjnych)
- Koszt: 0 zł

m) Visual Studio Code:
- Zakres: środowisko programistyczne do tworzenia frontendu
- Wersja: 1.72.2
- Licencja: MIT
- Koszt: 0 zł

n) MySQL Workbench:
- Zakres: zarządzanie MySQL
- Wersja: 8.0.22
- Licencja: GPL
- Koszt: 0 zł

o) GitHub:
- Zakres: repozytorium kodu
- Licencja: Free
- Koszt: 0 zł

1.4.3. Testowanie

a) JUnit:
- Zakres: testy integracyjne aplikacji serwerowej 1
- Wersja: 5.9.1
- Licencja: Eclipse Public License 2.0
- Koszt: 0 zł

b) REST-assured:
- Zakres: testowanie RESTowych serwisów aplikacji serwerowej 1
- Wersja: 5.2.0
- Licencja: Apache License 2.0
- Koszt: 0 zł

c) Postman:
   - Zakres: testowanie żądań HTTP dla aplikacji serwerowych
   - Wersja: 9.4
   - Licencja: OSS
   - Koszt: 0 zł

d) Selenium:
   - Zakres: automatyczne testy akceptacyjne aplikacji klienckiej
   - Wersja: 4.5.0
   - Licencja: Apache Software License (Apache 2.0)
   - Koszt: 0 zł

1.4.4. Wdrożenie

a) Google Cloud:
   - Zakres: Wdrożenie całej aplikacji
   - Licencja: Free license
   - Koszt: 0 zł

Wykorzystując własne umiejętności, posiadając wiedzę na temat tworzenia aplikacji webowych oraz posiadając odpowiednie narzędzia do wytwarzania oprogramowania, jesteśmy w stanie wykonać tę aplikację dla klienta w przeciągu 10 tygodni, które mamy. Wszystkie aktualnie wykorzystywane technologie są dla nas dostępne na licencjach darmowych lub akademickich.

1.5. Analiza SWOT

1.5.1 Identyfikacja czynników

	Pozytywne	Negatywne
Wewnętrzne	- znajomość wykorzystywanych technologii - możliwość pracy zdalnej - łatwa współpraca dzięki niewielkiemu rozmiarowi grupy	- brak renomy firmy - duża złożoność systemu - ograniczony czas na realizację
Zewnętrzne	- nawiązanie współpracy z innymi firmami w celu dalszego rozszerzania możliwości aplikacji	- ataki hackerów - zmiana przepisów prawnych - spadek popytu na

	- pojawienie się nowych technologii na rynku - potencjalne rozszerzenie asortymentu	papierowe książki

1.5.2 Ważność czynników

Mocne strony	**Waga**	**Słabe strony**	**Waga**
znaj. techn.	0,15	brak renomy	0,10
praca zdalna	0,35	złoż. systemu	0,45
niewiel. rozm. grupy	0,50	czas realizacji	0,45
	1,00		1,00
Szanse	**Waga**	**Zagrożenia**	**Waga**
współ. z inn. firm.	0,35	ataki hackerów	0,35
nowe technologie	0,35	zmiana przepisów	0,20
rozszerz. asort.	0,30	spadek popytu	0,45
	1,00		1,00

1.5.3 Analiza powiązań SWOT

Czy określona mocna strona pozwala wykorzystać daną szansę?

Szanse/ Mocne strony	[O1]	[O2]	[O3]	Waga	Liczba interakcji	Iloczyn wag i interakcji	Ranga
[S1]	0	0	1	0,15	1	0,15	3
[S2]	1	0	0	0,35	1	0,35	2
[S3]	1	0	0	0,5	1	0,5	1
Waga	0,35	0,35	0,3				
Liczba interakcji	2	0	1				

Iloczyn wag i interakcji	0,7	0	0,3				
Ranga	1	3	2				
Suma interakcji					6		
Suma iloczynów						2	

Czy określona mocna strona pozwala ograniczyć dane zagrożenie?

Zagrożenia / Mocne strony	[T1]	[T2]	[T3]	Waga	Liczba interakcji	Iloczyn wag i interakcji	Ranga
[S1]	1	0	0	0,15	1	0,15	2
[S2]	0	0	0	0,35	0	0	3
[S3]	1	1	0	0,5	2	1	1
Waga	0,35	0,35	0,3				
Liczba interakcji	2	1	0				
Iloczyn wag i interakcji	0,7	0,35	0				
Ranga	1	2	3				
Suma interakcji					6		
Suma iloczynów						2,2	

Czy określona słaba strona ogranicza możliwość wykorzystania danej szansy?

Szanse/ Słabe strony	[O1]	[O2]	[O3]	Waga	Liczba interakcji	Iloczyn wag i interakcji	Ranga
[W1]	1	0	0	0,15	1	0,15	3
[W2]	0	1	1	0,35	2	0,7	2
[W3]	1	1	1	0,5	3	1,5	1
Waga	0,35	0,35	0,3				
Liczba interakcji	2	2	2				
Iloczyn wag i interakcji	0,7	0,7	0,6				
Ranga	1	1	3				
Suma interakcji					12		
Suma iloczynów						4,35	

Czy określona słaba strona potęguje dane zagrożenie?

Zagrożenie / Słabe strony	[T1]	[T2]	[T3]	Waga	Liczba interakcji	Iloczyn wag i interakcji	Ranga
[W1]	0	0	1	0,15	1	0,15	3
[W2]	1	0	0	0,35	1	0,35	2
[W3]	1	1	0	0,5	2	1	1
Waga	0,35	0,35	0,3				
Liczba interakcji	2	1	1				

Iloczyn wag i interakcji	0,7	0,35	0,3				
Ranga	1	3	2				
Suma interakcji					8		
Suma iloczynów						2,85	

1.5.4 Analiza powiązań TOWS

Czy określona szansa potęguje daną silną stronę?

Szanse/ Mocne strony	[S1]	[S2]	[S3]	Waga	Liczba interakcji	Iloczyn wag i interakcji	Ranga
[O1]	0	1	0	0,15	1	0,15	3
[O2]	1	0	0	0,35	1	0,35	2
[O3]	1	0	0	0,5	1	0,5	1
Waga	0,35	0,35	0,3				
Liczba interakcji	2	1	0				
Iloczyn wag i interakcji	0,7	0,35	0				
Ranga	1	2	3				
Suma interakcji					6		
Suma iloczynów						2,05	

Czy określone zagrożenie ogranicza daną silną stronę?

Zagrożenia / Mocne strony	[S1]	[S2]	[S3]	Waga	Liczba interakcji	Iloczyn wag i interakcji	Ranga
[T1]	0	0	0	0,15	0	0	1
[T2]	0	0	0	0,35	0	0	1
[T3]	0	0	0	0,5	0	0	1
Waga	0,35	0,35	0,3				
Liczba interakcji	0	0	0				
Iloczyn wag i interakcji	0	0	0				
Ranga	1	1	1				
Suma interakcji					0		
Suma iloczynów						0	

Czy określona szansa pozwala osłabić (przezwyciężyć) daną słabą stronę?

Szanse/ Słabe strony	[W1]	[W2]	[W3]	Waga	Liczba interakcji	Iloczyn wag i interakcji	Ranga
[O1]	1	1	1	0,15	3	0,45	2
[O2]	0	0	0	0,35	0	0	3
[O3]	1	0	0	0,5	1	0,5	1
Waga	0,35	0,35	0,3				
Liczba interakcji	2	1	1				

Iloczyn wag i interakcji	0,7	0,35	0,3				
Ranga	1	2	3				
Suma interakcji					8		
Suma iloczynów						2,3	

Czy określone zagrożenie wzmacnia daną słabą stronę?

Zagrożenie / Słabe strony	[W1]	[W2]	[W3]	Waga	Liczba interakcji	Iloczyn wag i interakcji	Ranga
[T1]	0	1	1	0,15	2	0,3	3
[T2]	0	1	1	0,35	2	0,7	1
[T3]	1	0	0	0,5	1	0,5	2
Waga	0,35	0,35	0,3				
Liczba interakcji	1	2	2				
Iloczyn wag i interakcji	0,35	0,7	0,6				
Ranga	3	1	2				
Suma interakcji					10		
Suma iloczynów						3,15	

1.5.5 Wnioski oraz wybór strategii i uzasadnienie

Wyniki zbiorcze analizy SWOT/TOWS

Kombinacja	Wyniki analizy SWOT		Wyniki analizy TOWS		Zestawienie zbiorcze SWOT/TOWS	
	Suma interakcji	Suma iloczynów	Suma interakcji	Suma iloczynów	Suma interakcji	Suma iloczynów
Mocne strony [S]/ Szanse [O]	6	2	6	2,05	12	4,05
Mocne strony [S]/ Zagrożenia [T]	6	2,2	0	0	6	2,2
Słabe strony [W]/ Szanse [O]	12	4,35	8	2,3	20	6,65
Słabe strony [W]/ Zagrożenia [T]	8	2,85	10	3,15	18	6

Wyniki analizy strategicznej i wybór strategii

	Szanse	Zagrożenia
Mocne strony	Strategia agresywna	Strategia konserwatywna
	Liczba interakcji	Liczba interakcji
	12	6

	Ważona liczba interakcji	Ważona liczba interakcji
	4,05	2,2
Słabe strony	Strategia konkurencyjna	Strategia defensywna
	Liczba interakcji	Liczba interakcji
	20	18
	Ważona liczba interakcji	Ważona liczba interakcji
	6,65	6

Z powodu przeważających słabych stron i powiązanych z nimi szansami, zamierzamy przyjąć strategię konkurencyjną. Chcemy wykorzystać zależność pomiędzy naszymi słabymi stronami a szansami, jakie daje otoczenie, czyli przezwyciężyć szansami słabe strony.
Powinniśmy skupić się na wykorzystaniu szansy w postaci możliwości współpracy z innymi firmami w celu przezwyciężenia słabych stron w postaci braku renomy wśród klientów, dużej złożoności systemu czy ograniczonego czasu na wykonanie projektu.
Szansą dla kręgielni mogą być także pojawienie się nowych technologii oraz poszerzenie asortymentu o ebooki czy audiobooki.

1.6. Kosztorys

- oprogramowanie - 0 zł
- wynagrodzenie pracowników - 3 prac. * 60 h *50 zł/h = 9000 zł
- utrzymanie serwerów - 0 zł
- zakup domeny - 48,00 zł za pierwszy rok (bookshop.sklep.pl)
- hosting - 4354,56 zł (Google Cloud w pakiecie b2-7)

Całkowity, obliczony koszt wykonania systemu wynosi 13402,56 zł. Wartość ta mieści się w posiadanym budżecie, równym 20000 zł.

1.7. Harmonogram

Przygotowanie i uzasadnienie potrzeby realizacji. Analiza problemu.	24.10
Faza projektowa	07.11
Implementacja, testowanie i wdrożenie aplikacji	21.11
Prezentacja na forum publicznym i zaliczenie całości	05.12

1.8. Podział obowiązków

Kamil Dywan	Projektowanie, Baza danych, Aplikacja serwerowa 1, Frontend, Wdrożenie
Paweł Kuźma	Projektowanie, Baza danych, Aplikacja serwerowa 1, Frontend
Mateusz Urbańczyk	Projektowanie, Baza danych, Aplikacja serwerowa 2

2. Projektowanie

2.1 Role użytkowników

a) Użytkownik - rola przypisywana każdemu użytkownikowi w systemie. Użytkownik może przede wszystkim przeglądać ofertę księgarni.

   i) Niezalogowany użytkownik - rola, którą posiada każdy niezalogowany użytkownik. Użytkownik może założyć konto oraz się zalogować.
   ii) Zalogowany użytkownik - rola przypisywana każdemu użytkownikowi, który się zalogował. Użytkownik może przede wszystkim zmienić swoje dane osobowe oraz wylogować się.

1) Klient - rola przypisywana każdemu użytkownikowi, który zalogował się na konto klienta. Klient może przede wszystkim zarządzać swoim koszykiem i własnymi zamówieniami.
2) Pracownik - rola przypisywana każdemu użytkownikowi, który zalogował się na konto pracownika. Założono, że w systemie będą już utworzone konta pracowników. Pracownik może przede wszystkim zarządzać zasobami sklepu oraz zamówieniami.

2.2. Wymagania funkcjonalne

a) Wymagania związane ogólnie z użytkownikami:
- przeglądanie listy dostępnych książek.

b) Wymagania związane wyłącznie z niezalogowanym użytkownikiem:
- założenie konta,
- zalogowanie się.

c) Wymagania związane wyłącznie z zalogowanym użytkownikiem:
- wylogowanie się.

d) Wymagania związane wyłącznie z klientem:
- modyfikacja danych osobowych,
- dodanie i usunięcie książki z koszyka,
- zamówienie książek do odbioru w sklepie,
- sprawdzenie statusu własnego zamówienia,
- wycofanie własnego zamówienia niebędącego jeszcze w trakcie realizacji.

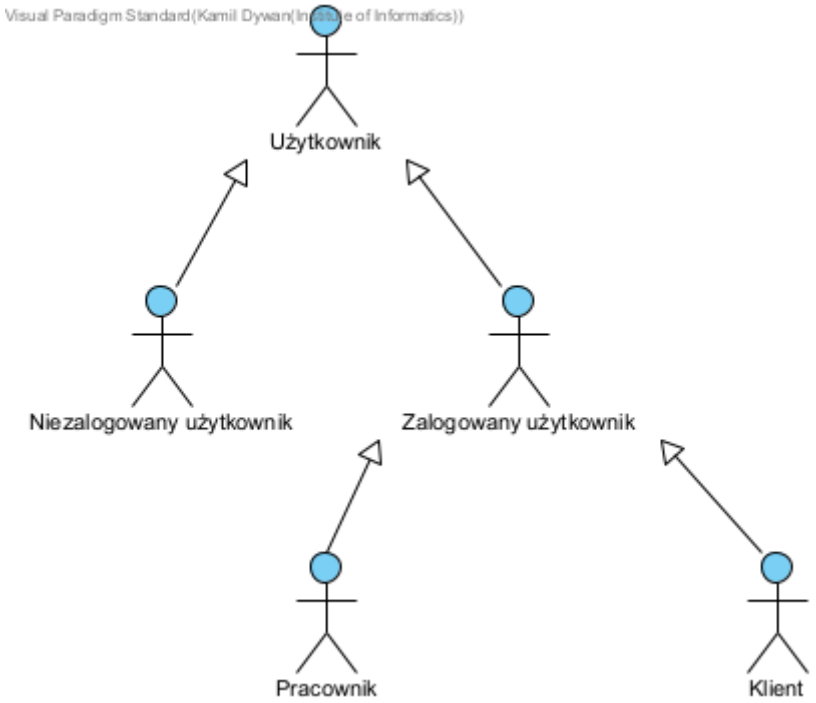
e) Wymagania związane wyłącznie z pracownikiem:
- przeglądanie listy książek,
- dodawanie, usuwanie i edytowanie parametrów książek będących w systemie,
- dodawanie i usuwanie książek z oferty dostępnej dla klientów,
- przeglądanie listy zamówień,
- zmiana statusu zamówienia.

2.3. Wymagania niefunkcjonalne

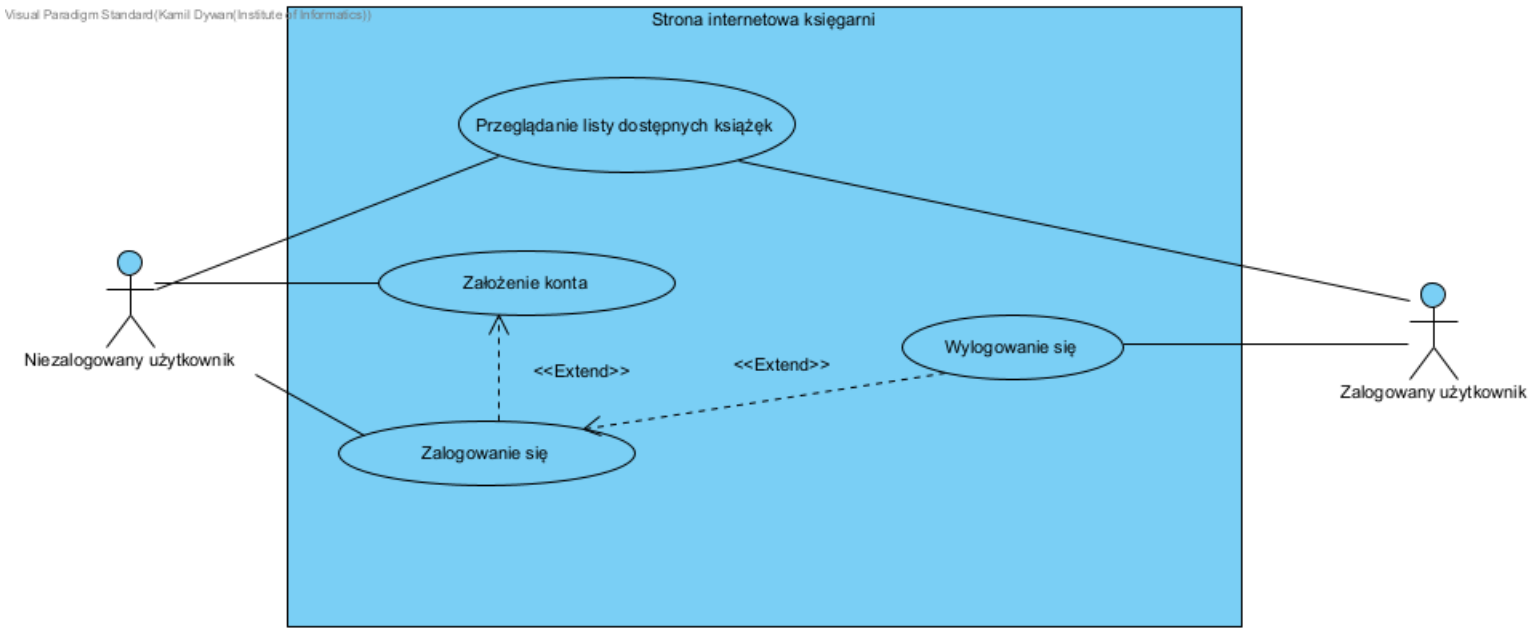
- Wymagania dotyczące bezpieczeństwa systemu:
    - tworzone hasła powinny spełniać następujące kryteria:
        - minimalna długość - 8,
        - co najmniej jedna mała litera,
        - co najmniej jedna duża litera,
        - co najmniej jedna cyfra.
    - bezpieczeństwo dostępu do systemu powinno być zapewnione poprzez użycie tokenów. Użytkownik nie będzie mógł skorzystać z danego endpointu albo wejść na daną podstronę, jeśli nie posiada wymaganej roli.
- Wymagania technologiczne:
    - komunikacja pomiędzym klientem i serwerami powinna odbywać się przy pomocy REST API,
    - dostęp do systemu będzie zabezpieczony poprzez protokół OAuth 2.0,
    - oprogramowanie serwera aplikacji 1 powinno być stworzone przy użyciu języka Java i frameworka Spring,
    - oprogramowanie serwera aplikacji 2 powinno być stworzone przy użyciu języka Python i frameworka django,
    - frontend aplikacji powinien być stworzony przy pomocy technologii HTML 5, CSS3 i Javascript z frameworkiem React,
    - baza danych powinna być stworzona przy użyciu technologii MySQL,
    - aplikacja powinna zostać przetestowana przy pomocy narzędzi JUnit, REST-assured, Postman, Selenium,
    - wdrożenie aplikacji powinno być wykonane w usłudze Google Cloud.

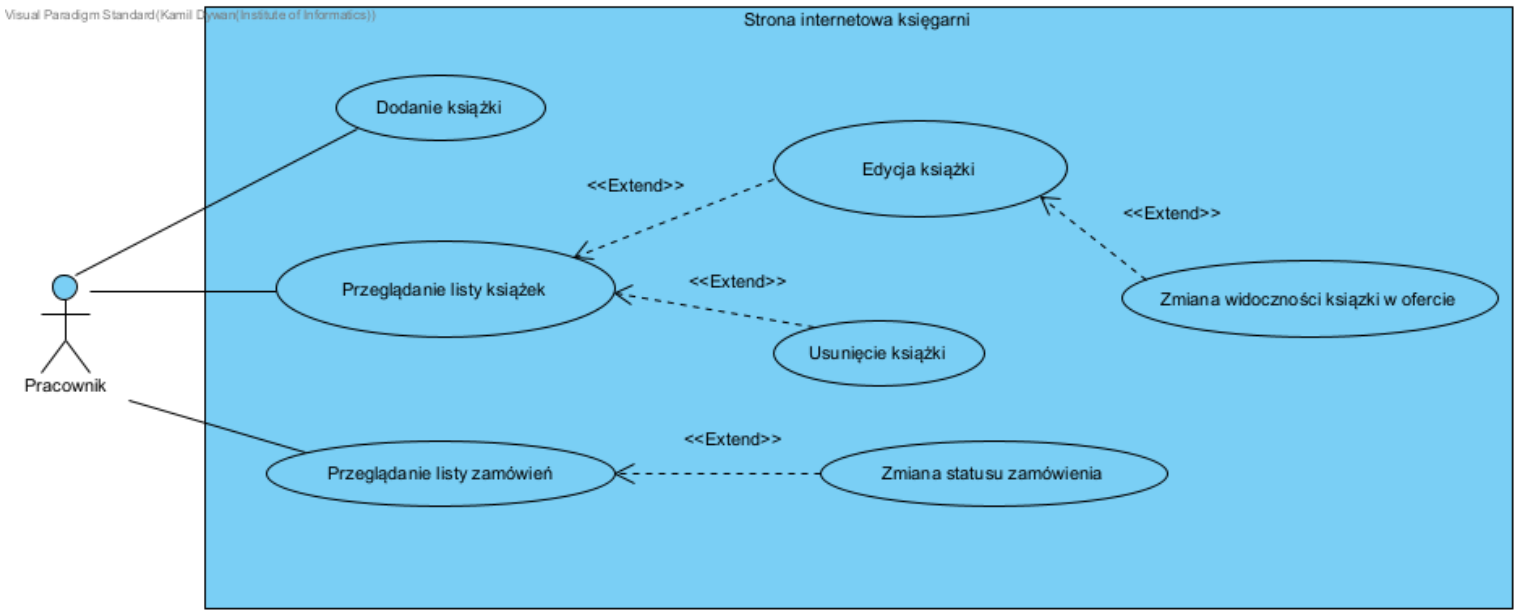
2.4. Diagramy przypadków użycia

Aktorzy:



Diagramy przypadków użycia dla aktorów




Visual Paradigm Standard(Kamil Dywan(Institute of Informatics))
Strona internetowa księgarni
Dodanie książki
Edycja książki
<<Extend>>
<<Extend>>
Przeglądanie listy książek
<<Extend>>
Zmiana widoczności książki w ofercie
Usunięcie książki
Pracownik
<<Extend>>
Przeglądanie listy zamówień
Zmiana statusu zamówienia

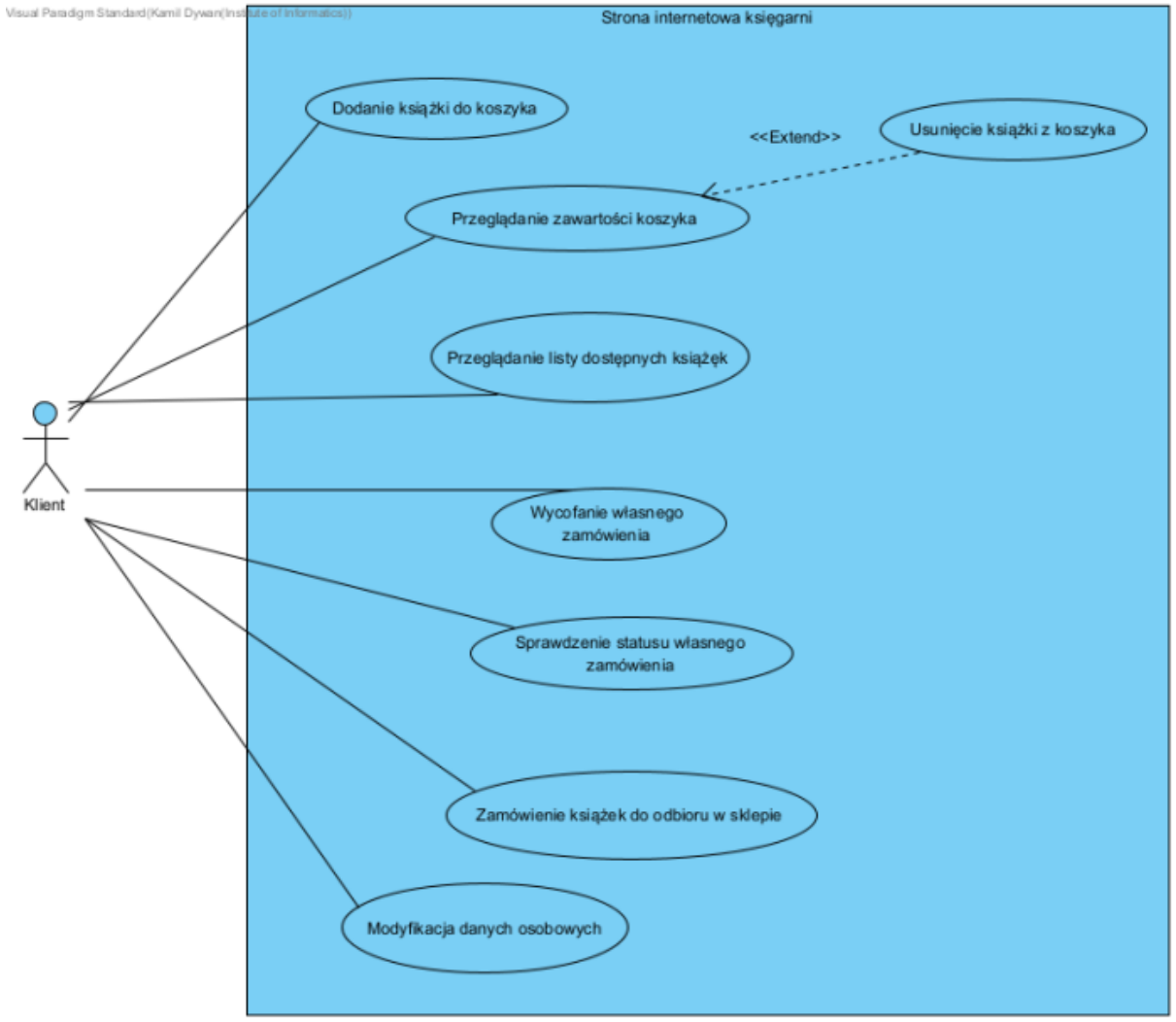
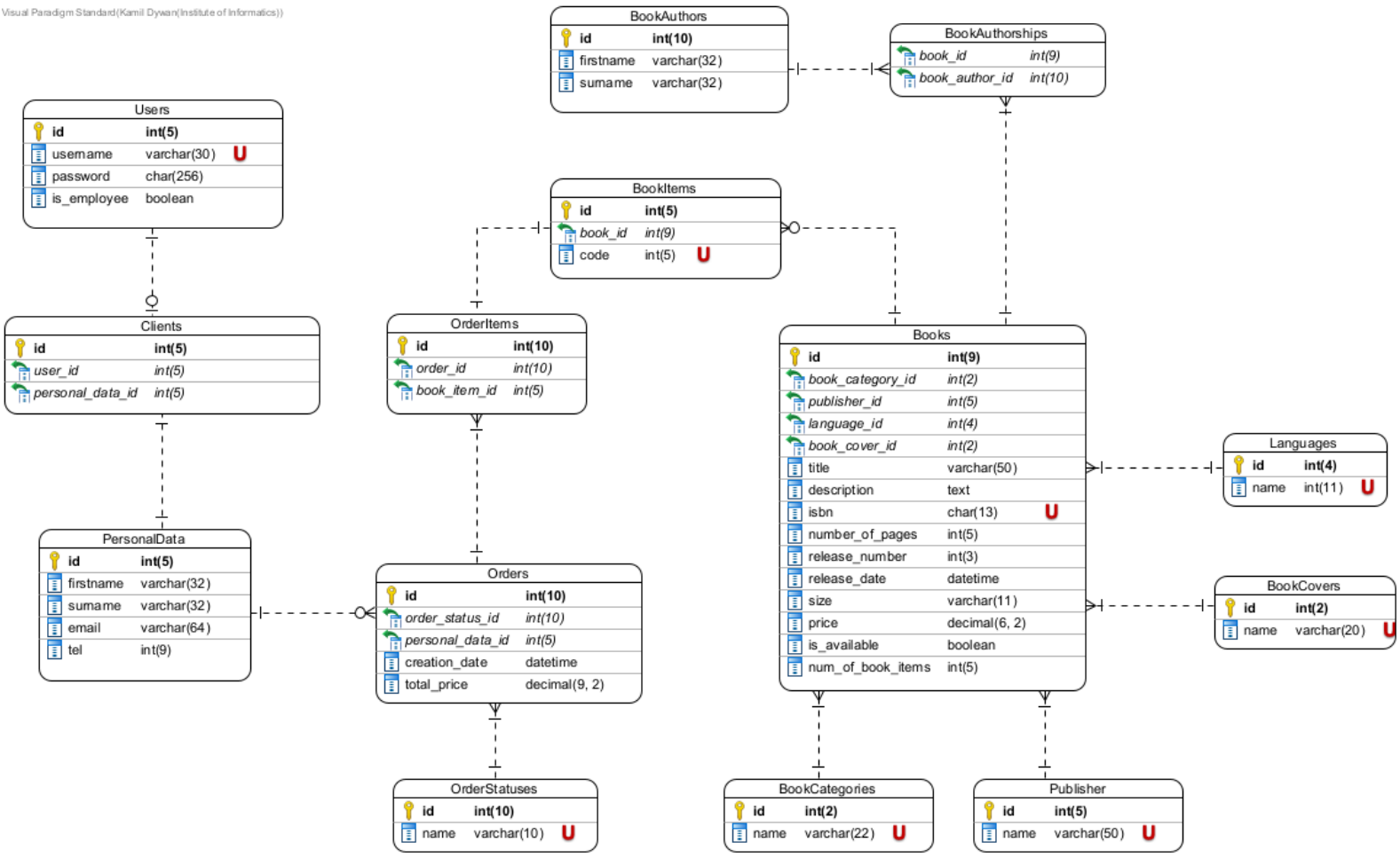

Visual Paradigm Standard(Kamil Dywan(Institute of Informatics))
Strona internetowa księgarni
Dodanie książki do koszyka
<<Extend>>
Usunięcie książki z koszyka
Przeglądanie zawartości koszyka
Przeglądanie listy dostępnych książęk
Klient
Wycofanie własnego zamówienia
Sprawdzenie statusu własnego zamówienia
Zamówienie książek do odbioru w sklepie
Modyfikacja danych osobowych

Diagram ERD



2.5. Endpointy serwerów

a) Endpointy dla serwera aplikacji 1:

- GET /orders - wyszukanie zamówień po kryteriach oraz paginacji,
- GET /orders/{clientId} - zwrócenie zamówień klienta o podanym id,
- POST /order - zamówienie książek znajdujących się w koszyku,
- PUT /order/{orderId} - edycja statusu zamówienia o podanym id przez pracownika,
- PUT /order/{orderId}/cancel - wycofanie zamówienia o podanym id,
- GET /book/{bookId} - wyszukanie książki po id,

- GET /books - wyszukiwanie książek po liście parametrów oraz paginacji,
- GET /books/available - wyszukiwanie dostępnych książek po liście parametrów oraz paginacji,
- POST /book - dodanie nowej książki,
- PUT /book/{bookId} - edycja parametrów książki o podanym id,
- DELETE /book/{bookId} - usunięcie książki o podanym id,
- PUT /user/{userId} - edycja danych użytkownika,
- PUT /client/{clientId} - edycja danych personalnych,
- POST /register - zarejestrowanie się,
- POST /token - aktualizowanie tokena dostępu,
- POST /login - zalogowanie się,
- DELETE /logout/{sessionId} - wylogowanie się.

b) Endpointy dla serwera aplikacji 2:
- GET /book/{bookId} - wyszukanie książki po id,
- GET /books - wyszukiwanie książek po kryteriach oraz paginacji,