

**WIEiK**

**Kierunek Informatyka w Inżynierii Komputerowej**

**Przedmiot: Inżynieria Programowania Projekt**



**Politechnika Krakowska  
im. Tadeusza Kościuszki**

**Prowadzący: prof.dr hab. inż. Sergii Telenyk**

**Temat projektu**

Tworzenie oprogramowania dla systemu zarządzania siecią  
bibliotek

**Opracowali:  
Beata Korzeniewska  
Mateusz Toczek  
Kraków**

## Spis treści

1. Cel projektu
2. Opis problemu
3. Założenia projektowe, zakres produktu
4. Specyfikacja techniczna
5. Wymagania funkcjonalne
6. Wymagania нефunkcjonalne
7. Procesy biznesowe
8. Analiza wybranych architektur i wybór właściwej
9. Wzorce architektoniczne
10. Opis użytkowników
  - 10.1.1. Pracownik
  - 10.1.2. Właściciel biblioteki
  - 10.1.3. Administrator
11. Analiza ryzyka
12. Diagram ER
13. Organizacja projektu
14. Architektura systemu
15. Błędy i usterki znalezione podczas testowania aplikacji
16. Rozwiązania błędów i usterek
17. Dokumentacja użytkownika
18. Aktualizacje i poprawki
19. Wnioski
20. Kod projektu

## 1. Cel Projektu

Celem projektu jest stworzenie oprogramowania dla systemu zarządzania siecią bibliotek. Celem oprogramowania będzie zmodernizowanie i pomoc użytkownikom w zarządzaniu zasobami bibliotecznymi, informacjami oraz personelem. Ponadto aplikacja ma na celu usprawnić śledzenie statystyk działalności bibliotecznej oraz szkolenie pracowników.

Celem projektu jest stworzenie oprogramowania wykorzystywanego do zarządzania siecią bibliotek. Przygotowane rozwiązanie ma być pomoc użytkownikom w zarządzaniu zasobami bibliotecznymi, informacjami oraz personelem. Całość oprogramowania ma stanowić wspólny, jednolity system realizujący zakres prac. Ponadto aplikacja ma na celu usprawnić śledzenie statystyk działalności bibliotecznej oraz szkolenie pracowników.

## 2. Opis Problemu

W obecnych czasach wiele przedsiębiorstw i instytucji decyduje się na korzystanie z programów ulepszających działalność. Wprowadzenie oprogramowania wykonującego czynności które wcześniej były pełnione przez pracowników, znacznie obniża koszty pracy oraz polepsza wydajność, czyniąc procedury szybszymi. Przekłada się to na zadowolenie klientów takich oto przedsiębiorstw, ponieważ stanowczo skraca się czas oczekiwania na usługę.

Wraz z rozwijającą się technologią i światem, popyt na informacje stale rośnie. Jednakże instytucje takie jak biblioteki, będące miejscem składowania książek i artykułów, wciąż pozostają nie zautomatyzowane. Klienci unikają korzystania z ich usług z uwagi na długi czas oczekiwania i brak przystępnej organizacji dóbr. Celem projektu jest zatem wyjść naprzeciw zapotrzebowaniom klientów oraz zapewnienie im odpowiednich narzędzi do zarządzania systemami bibliotecznymi poprzez odpowiednie oprogramowanie.

Oprogramowanie to umożliwia także zoptymalizowanie zarządzania użytkownikami, poprzez wgląd w grafik, harmonogram pracy i możliwość zobaczenia ilości przepracowanych godzin. Aplikacja zapewnia też krótkie kursy szkolenia z korzystania z niej dla pracowników. Prowadzenie monitoringu statystyk i badań pozwoli też franczyzobiorcy na celniejsze trafianie w gusta użytkowników z zakupem nowych dzieł do biblioteki, ponieważ umożliwia to wyznaczenie trendu wypożyczeń książek danego autora lub kategorii.

### 3. Założenia projektowe, zakres produktu

Oprogramowanie w założeniu ma być rozszerzalne, gdyż wraz z rozwojem technologii powinno się móc przystosowywać w działaniu do powstających nowych rodzajów mediów. W związku z pracą nad dużą bazą danych powinno ono być odpowiednio zoptymalizowane aby zapewnić użytkownikom korzystną płynność działania. Jako użytkownika aplikacji definiujemy osobę zarejestrowaną w systemie, posiadającą odpowiedni zbiór praw oraz obowiązków, w zależności od roli użytkownika wynikającą z jego umowy. Sposób zaprojektowania oprogramowania będzie umożliwiał równoległe korzystanie z niego wielu osobom, które będą miały możliwość przeglądania aplikacji deskopowej z przede wszystkim wygodnym, intuicyjnym i przejrzystym interfejsem graficznym zapewniającym szerokie pasmo przydatnych i użytecznych funkcjonalności. Założeniem projektowym jest także fakt iż czytelnik nie jest użytkownikiem. Dostęp do oprogramowania będą mieć tylko osoby pracujące, zatrudniające w bibliotece oraz administrator. Czytelnicy mają natomiast swoje profile zakładane przez użytkowników systemu, bez wglądu do nich.

### 4. Specyfikacja techniczna

Najważniejszym zadaniem oprogramowania jest usprawnianie działalności sieci bibliotek poprzez przechowywanie danych o dobrach w bazach i zarządzaniem istniejącymi już dobrami w przejrzysty sposób. Oprogramowanie pozwala także na stworzenie nowego systemu bibliotek od podstaw.

Głównym odbiorcą oprogramowania jest jej zarządca: poprzez rolę zarządcy biblioteki definiujemy Właściciela biblioteki lub sieci bibliotek wraz z całym zespołem administracyjnym. Zarządca posiada pełną kontrolę prac nad systemem, wykraczającą poza standardowych użytkowników. Pozostałą grupę odbiorców stanowią Pracownicy. Posiadają oni ogólnie zdefiniowany zakres praw przydzielany automatycznie przy dodawaniu nowego pracownika. Do zadań pracownika należy utrzymywanie porządku nad wirtualną biblioteką, obsługiwanie zapotrzebowań klientów oraz przestrzeganie harmonogramu pracownika.

W ramach oprogramowania wyróżniamy aplikację deskopową, pełniącą rolę komunikacji użytkownika z bazą danych poprzez interfejs graficzny oraz bazę danych przechowującą wszystkie informacje w ramach systemu bibliotek, wyposażoną w zestaw mechanizmów i funkcji do jej obsługi. Aplikacja posiada zestaw narzędzi usprawniających pracę biblioteki a użytkownik aplikacji może w każdej chwili podejrzeć aktualny stan oraz zawartość wybranej biblioteki. Wyszukiwanie i katalogowanie tytułów jest głównym celem aplikacji, dlatego podgląd zawartości dostępny jest dla każdego użytkownika bez konieczności logowania. Użytkownicy podstawowi mają dostęp wyłącznie do informacji powiązanych z kontem użytkownika: podgląd wypożyczeń, status konta oraz status dostępności wybranych tytułów. Pracownicy posiadają dostęp do statystyk oraz podsumowań, decydują o aktualnych zapotrzebowaniach biblioteki oraz rejestrują stan wypożyczeń. Dostęp do kalendarza wewnątrz aplikacji ułatwia pracę według harmonogramu, a codzienny organizator wyposażony w opcję przypomnień pozwala na łatwiejsze realizowanie zadań dla poszczególnego dnia. Większość korzyści wynikających z praw administratora dostępnych jest wewnątrz aplikacji do obsługi bazy danych. Pracownik posiada dostęp do dóbr przechowywanych w bazie, jednakże nie posiada on wszystkich praw do zarządzania bazą danych, w przeciwieństwie do zarządcy systemu, który posiada pełnię praw. Zadaniem administratora jest odpowiednia administracja danych oraz ich utrzymywanie i przetwarzanie. Posiada on też kontrolę nad

wszystkimi użytkownikami, zarówno pracownikami jak i właścicielami bibliotek. Właściciel bibliotek, niezależnie od ilości posiadanych bibliotek nie posiada dostępu do systemu zarządzania bazą danych. Pełny zakres praw i możliwości użytkowników oprogramowania, jak i wymagań, które ma spełniać system uwzględniony jest w ramach zestawu wymagań funkcjonalnych jak i нефункциональных.

## 5. Wymagania funkcjonalne

Właściciel bibliotek & Pracownik & Administrator:

- Utworzenie karty czytelnika
- Logowanie przy użyciu loginu i hasła
- Wylogowanie się
- Zmiana adresu email
- Wyszukanie tytułu dzieła
- Wyświetlenie gatunku i autora książki
- Ustawienie filtra wyszukiwania
- Wyświetlenie stanu książki/filmu/ audiobooka (Dostępna/ Wypożyczona )
- Wyświetlanie lokalizacji książki jeśli jest ona dostępna
- Wyświetlenie karty czytelnika oraz historii wypożyczenia
- Wyświetlanie lokalizacji bibliotek
- Możliwość sprawdzenia historii wypożyczania dzieła
- Możliwość zobaczenia historii wypożyczania książek przez czytelników
- Wprowadzenie do systemu wypożyczenia/ zwrotu dzieła przez czytelnika
- Możliwość zablokowania/odblokowania możliwości wypożyczania przez czytelnika
- Wyświetlenie grafiku pracownika
- Dodanie/Usunięcie czytelnika
- Dodanie/Usunięcie dzieła

Właściciel biblioteki & Administrator:

- Dodanie/ Usunięcie konta pracownika
- Wyświetlanie listy pracowników
- Wyświetlenie grafików pracowników
- Możliwość zmiany hasła do konta

Administrator:

- Dodanie / Usunięcie konta właściciela biblioteki
- Dodanie/ Usunięcie lokalizacji biblioteki

## 6. Wymagania нефункциональные

- Bezpieczeństwo bazy danych- dostęp tylko po zalogowaniu
- Synchronizacja danych – zmiany wprowadzone do systemu są widoczne dla innych urządzeń

- W każdym przypadku łączenia się aplikacji z serwerem musi istnieć zabezpieczenie, które wykryje błąd i powiadomi o tym użytkownika
- zgodność z podstawami prawnymi przechowywania danych wrażliwych zawodników – stworzenie Polityki RODO
- Szybki czas reakcji systemu bazodanowego na wysyłane do niego zapytania – krótszy niż 5 s
- Odporność na błędy logiczne takie jak między innymi pokazywanie wypożyczonej książki jako dostępnej- wiarygodność
- Oprogramowanie powinno być intuicyjne w użytkowaniu
- Rozszerzalność systemu – projekt oprogramowania i bazy danych umożliwiający dodanie nowych funkcji i ich pracę bez wprowadzania radykalnych zmian

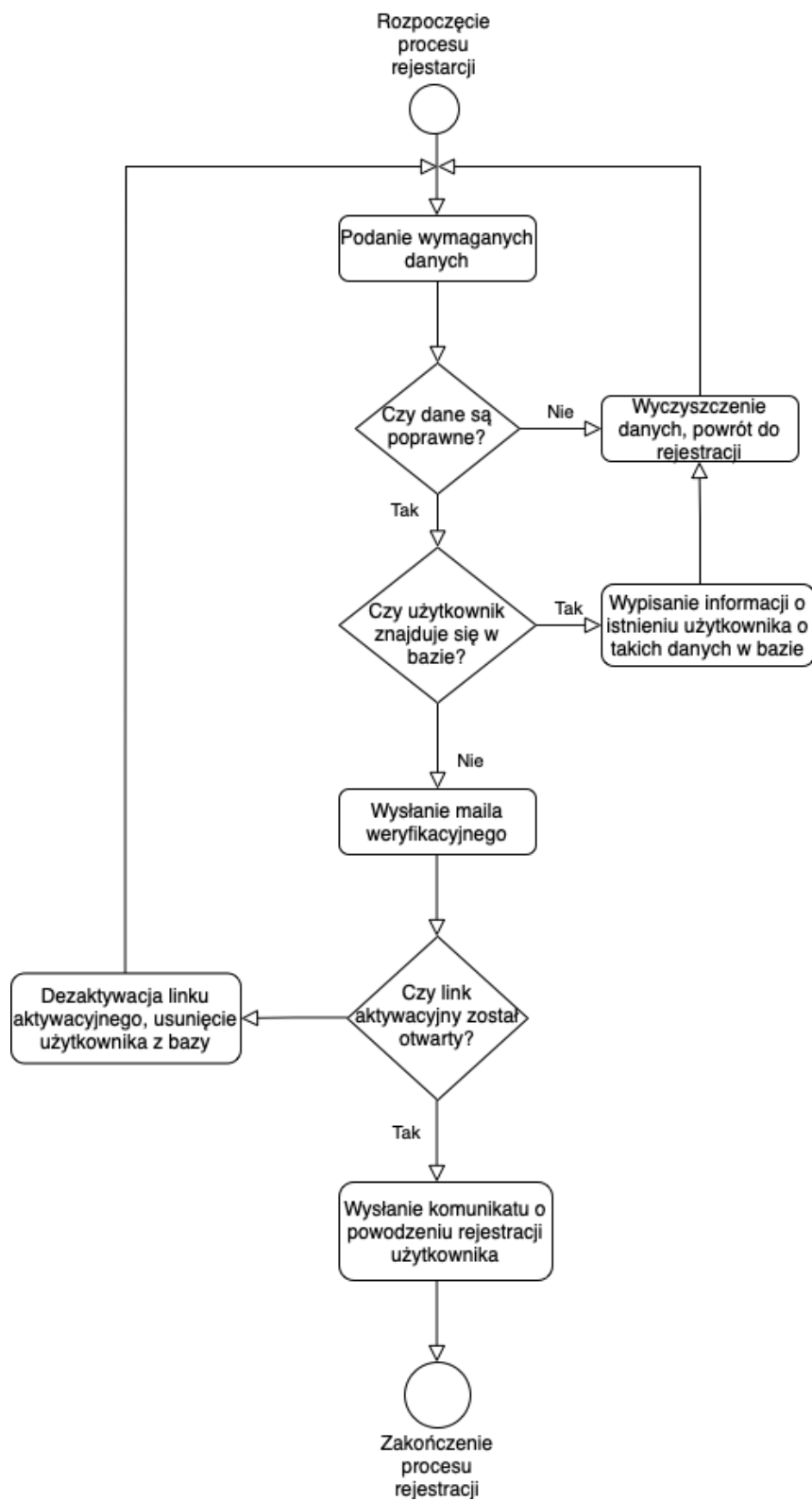
## 7. Procesy biznesowe

W ramach projektu, wyróżniamy następujące procesy:

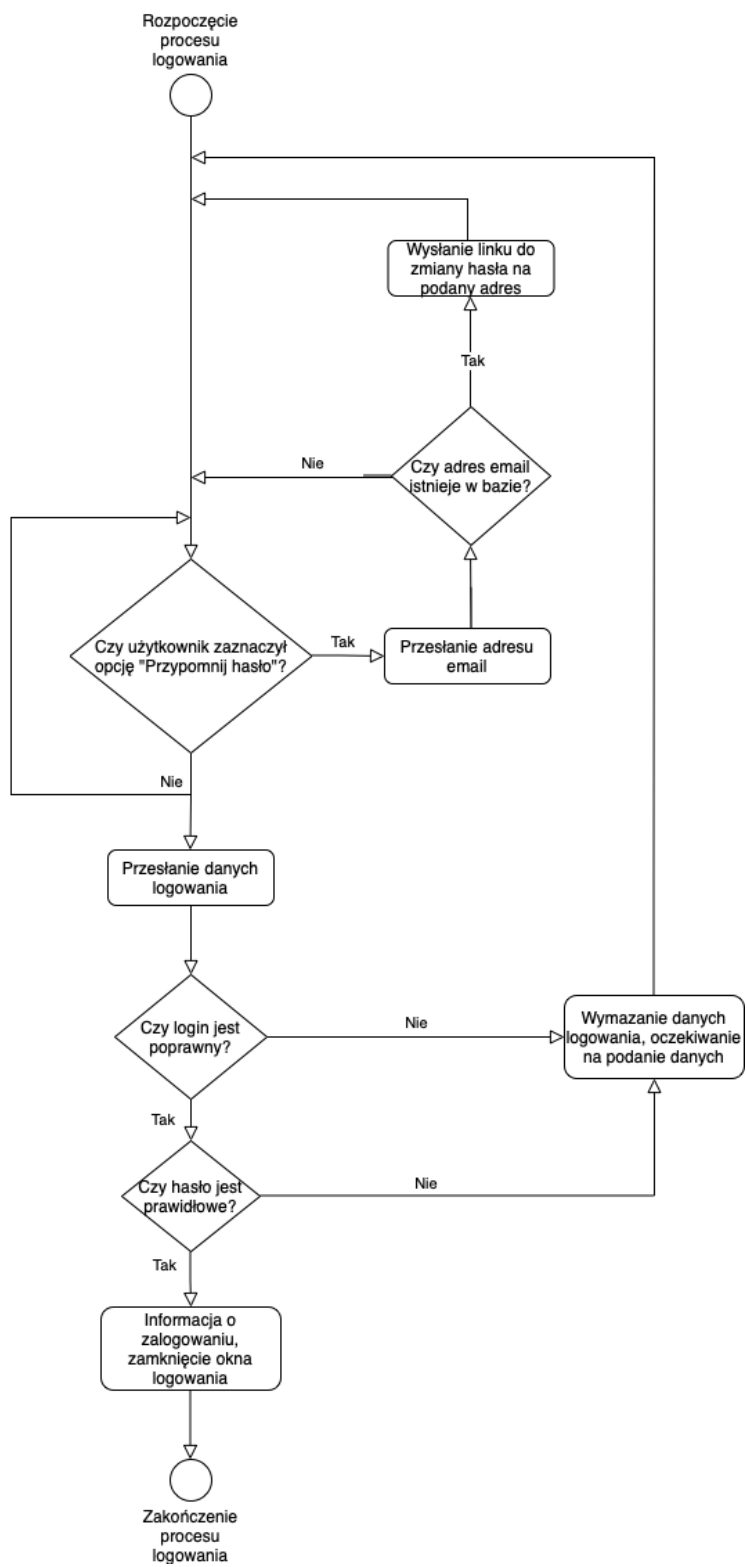
- rejestracja użytkownika,
- logowanie użytkownika,
- dodanie książki do bazy,
- wypożyczenie książki,

Każdy diagram odwzorowuje zachowanie aplikacji dla danej procedury, z podziałem na przypadki, w których podano błędną informację, lub podczas jej przeprowadzania napotkano wcześniej przewidziany błąd. Poszczególne diagramy zawierają również pola sygnalizujące początek oraz zakończenie realizacji procesu. Po zakończonym procesie następuje zwrócenie informacji o jej statusie realizacji (sukces lub niepowodzenie).

## Rejestracja użytkownika:

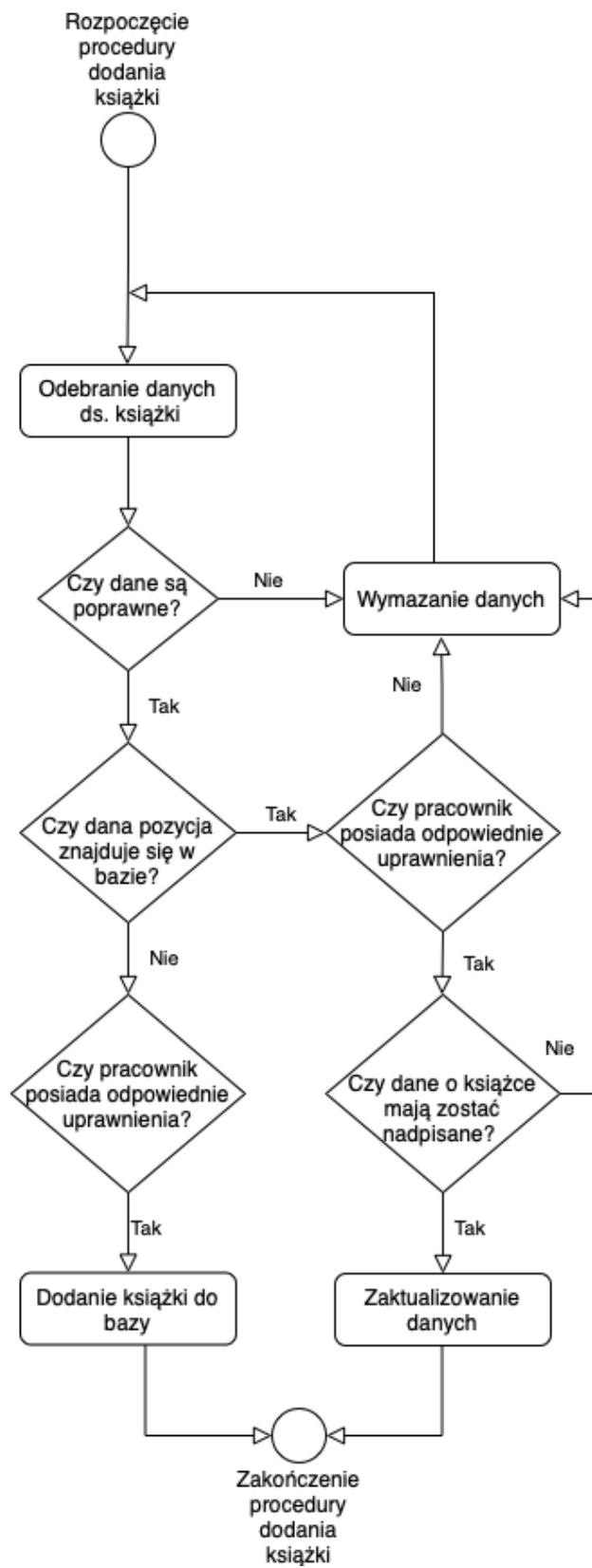


## Logowanie użytkownika:

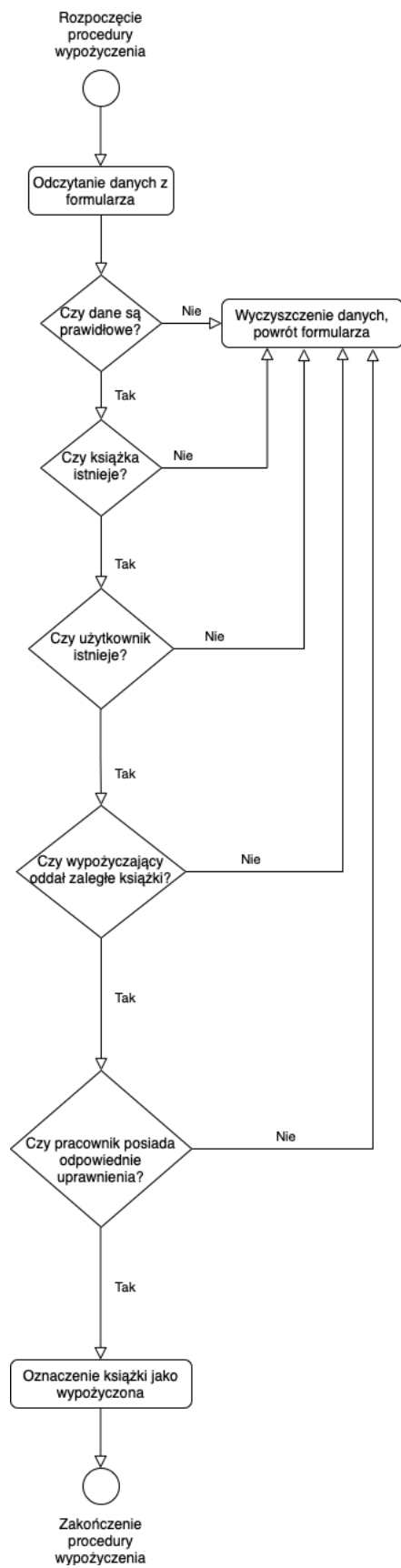




Dodanie książki do bazy:



## Wypożyczenie książki:



## 8. Analiza wybranych architektur i wybór właściwej

### Architektura monolityczna

Architektura monolityczna to tradycyjny model oprogramowania, tworzonego jako jednolita jednostka, która jest samodzielna i niezależna od innych aplikacji. Wszystkie funkcje oprogramowania są wdrażane jako pojedynczy plik, posiada też ona tylko jedną bazę danych. Architektura ta jest dużą, samodzielną strukturą, w której wszystko jest zbudowane liniowo. Daje to możliwość pełnej kontroli każdego aspektu i zadania jakie ma wykonać aplikacja. Jest zaprojektowana tak, aby zawierała wiele składników i aplikacji.

Zalety:

- łatwość określenia wymagań
- samowystarczalna jednostka wdrożeniowa
- wysoka wydajność gdyż system jest obciążony tylko jednym procesem
- duże większe bezpieczeństwo komunikacji niż w przypadku systemów rozproszonych
- prostota wykorzystania kodu w jednym miejscu

Wady:

- niską różnorodność usług
- brak modularności
- trudne i kosztowne wprowadzanie zmian
- integracja z innymi systemami może być utrudniona

### Architektura mikroserwisów

Jest to samowystarczalny model składający się z luźno powiązanych ze sobą modułów, gdzie każdy posiada swoją własną funkcjonalność i komunikuje się z innymi modułami tworząc większą, spójną całość. W przypadku mikroserwisów raz stworzona usługa może być stosowana wielokrotnie, jednakże każdy pojedynczy mikroserwis musi być samowystarczalny, oznacza to, że w przypadku systemu opartego o architekturę mikroserwisową musimy poświęcić znacznie więcej czasu na analizę wymagań oraz funkcjonalności.

Zalety:

- łatwa skalowalność aplikacji
- prosta dokumentacja
- możliwość łączenia wielu narzędzi programistycznych w ramach jednego projektu
- prostota aktualizacji i naprawy błędów

Wady:

- nieopłacalne dla mało skomplikowanych projektów

## **Architektura zorientowana na usługi**

Oznacza się modułowym podejściem do tworzenia oprogramowania poprzez komponenty umożliwiające komunikację z wykorzystaniem ustandaryzowanych protokołów. Wymagają silnego zarządzania reprezentacją usług, a jego implementacje są ograniczone lub włączone przez kontekst i muszą być opisane w ramach tych kontekstów.

Zalety:

- prosta w edycji
- niezależna od platformy
- oparta na serwisach zaawansowanych systemów

Wady:

- skomplikowana architektura
- duże koszty tworzenia i utrzymania
- konieczność walidacji wszystkich danych wejściowych przed podaniem ich do serwisów

## **Architektura komponentowa**

W architekturze komponentowej oprogramowanie budowane jest w oparciu o współpracujących ze sobą, niezależne komponenty programowe. Komponent to element oprogramowania posiadający dobrze wyspecyfikowany interfejs oraz zachowanie. Każdy komponent powinien realizować założoną funkcjonalność i dostarczać jej innym komponentom tylko za pośrednictwem własnego interfejsu. Komponenty traktuje się jako twory generyczne. Dopiero zbiór takich połączonych ze sobą komponentów stanowi rzeczywistą aplikację. Dany komponent może być wykorzystany w wielu aplikacjach. Warto zauważyć, że komponent to element gotowy do użycia (np. skompilowana wersja biblioteki).

## **Wybrana architektura**

Ze względu na liczne zalety projekt oparto zgodnie z założeniami architektury mikroserwisowej. Każdy pojedynczy mikroserwis, stanowiący podstawę działania aplikacji może być samowystarczalny, co pozwala na łatwiejszy podział pracy jak i zapewnia pewniejszą stabilność działania, gdyż w razie awarii pozostała część architektury nadal pozostaje w pełni sprawna. Jest to także odpowiednia architektura w przypadku ewentualnych aktualizacji oraz rozbudowy, a także umożliwia ona na łatwą i szybką naprawę błędów spowodowanych wadliwym działaniem aplikacji

## 9. Wzorce architektoniczne

### Wzorzec modułowy

Wzorcem modułowym nazywamy technikę kładącą nacisk na rozdzielnie funkcjonalności programu moduły, skłonne do wymiany. Każdy z nich musi zawierać wszystko co niezbędne do wykonania przez właściwy dla niego aspekt działalności. Każdy mikroservis w naszym projekcie składa się z odpowiednich dla niego modułów.

### Wzorzec komponentowo-łącznikowy

Aby wszystkie moduły które powstały po podzieleniu mikroservisów, jak zostało to ukazane powyżej, mogły działać poprawnie potrzebna jest im komunikacja i współdzielenie danych. Dlatego też rozważamy zadany nam wzorzec w kontekście przepływu danych. Aby umożliwić współpracę między modułami a backendem musieliśmy stworzyć odpowiednie do tego API. Informacje takie jak ID\_czytelnika są zdefiniowane globalnie, co umożliwia wszystkim modułom dostęp do niezbędnych informacji podczas wykonywania zdefiniowanych dla nich akcji – jak np. wypożyczenie książki, w tym celu potrzebujemy mieć dostęp do danych książki aby móc połączyć nowo utworzony rekord do odpowiedniego egzemplarzu dzieła. To wszystko jest możliwe dzięki naszemu łącznikowi oraz współdzieleniu pewnych danych , dlatego też nasz projekt spełnia wymagania wzorca komponentowo-łącznikowego.

### Wzorzec alokacji

Nasz projekt będący systemem informatycznym możemy wyróżnić 2 elementy składowe które współpracują aby system był kompletny.

- 1) Serwer baz danych
- 2) Hosting
- 3) Serwer plików

## 10. Opis użytkowników

### Pracownik

Jest to najbardziej podstawowy typ użytkownika, którego funkcje zawierają także użytkownicy: Właściciel biblioteki i Administrator. Przewidywane jest by było konta tego typu były najliczniejszymi. Użytkownik ma możliwość zobaczenia swojego grafiku. Ponadto ma dostęp do profili czytelników, które może zakładać, usuwać oraz m.in. wyświetlać informacje. Posiadają również kontrolę nad zbiorami dóbr, mogą je dodawać, usuwać lub wyświetlać ich informacje.

### Właściciel biblioteki

Właściciel biblioteki jest swego rodzaju menadżerem odpowiadającym za nią. Ma on wgląd w listę pracowników oraz ich grafików, może ich także dodawać i usuwać. W celu

monitoringu działalności biblioteki ma on także możliwość wyświetlania najczęściej wypożyczanych gatunków i autorów. Ponadto istnieje dla niego opcja wyświetlania danych kontaktowych do właścicieli innych bibliotek tej samej sieci. Posiada on także te same możliwości co użytkownik typu pracownik.

## Administrator

Administrator jest najwyższym typem użytkownika, zawiera w sobie możliwości pozostałych użytkowników oraz posiada umiejętność dodania/ usunięcia konta właściciela oraz lokalizacji biblioteki.

## 11. Analiza ryzyka

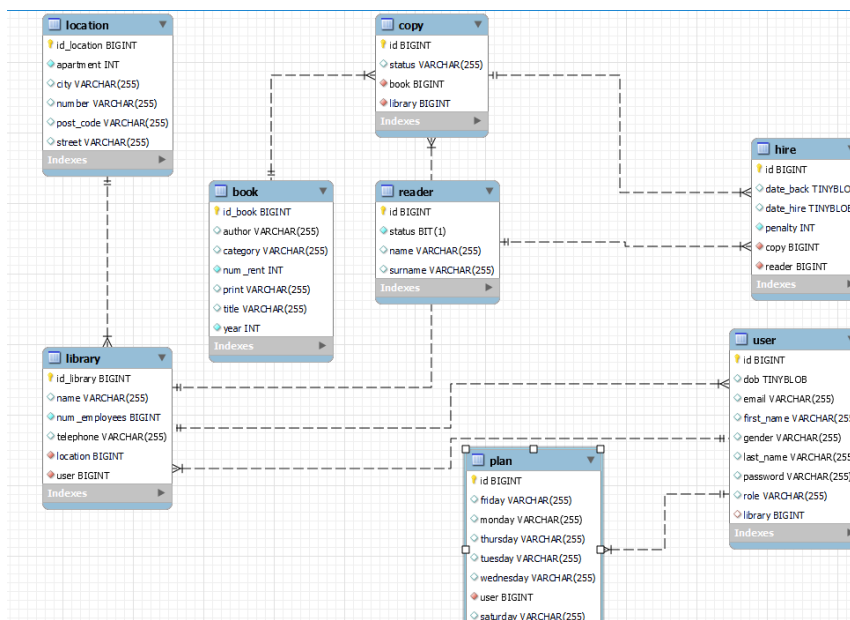
Z założenia program powinien pracować z wieloma różnymi typami danych o odmiennej złożoności, połączonymi licznymi relacjami. Może to skutkować bardzo złożonym systemem obsługi błędów, który nie jest w stanie obsłużyć wyjątków, które mogą wystąpić podczas interakcji użytkownika z aplikacją kliencką.

Przewidujemy przetwarzanie znacznej ilości danych, co zawsze wiąże się z możliwością utraty danych lub problemów z integralnością.

Raport 1.

Z powodu ograniczeń czasowych i zasobów możliwe jest, że niektóre cele związane z funkcjonalnością oprogramowania lub witryną internetową mogą nie zostać osiągnięte.

## 12. Diagram ER



Encje:

- 1) Czytelnik- mimo iż czytelnik sam w sobie nie jest użytkownikiem jest on swego rodzaju obiektem którego dane będą obsługiwać inni użytkownicy na zasadzie obchodzenia się z kartą czytelnika w bibliotekach pozbawionych cyfryzacji. Encja ta przechowuje dane o klientach biblioteki, takie jak data urodzenia, ich id, pesel, imię, nazwisko i status, który określa czy osoba może wypożyczyć książki czy nie.
- 2) Książka- encja ta zawiera isbn, tytuł, wydawnictwo, rok wydania, kategorie, oraz ilość wypożyczeń. Ostatni atrybut jest dodany aby móc monitorować statystyki wypożyczeń i z łatwością określać trend w wypożyczaniu książek.
- 3) Biblioteka- encja ta określa lokalizację, właściciela oraz numer telefonu dla każdej biblioteki należącej do systemu. Zawiera w sobie także informacje o ilości pracowników.
- 4) Egzemplarz- w encji egzemplarz przechowywane są dane określające czy egzemplarze danej książki są dostępne i określające lokalizację biblioteki skąd książka została wypożyczona lub gdzie się znajduje.
- 5) Grafiki - określa dla każdego użytkownika tryb pracy oraz liczbę dni wolnych.
- 6) Lokalizacja- encja przechowuje informacje o adresach, które są łączone poprzez klucze obce z innymi encjami takimi chociażby jak biblioteka czy czytelnik.
- 7) Użytkownik- jest to jedna z najbardziej rozbudowanych encji w bazie. Posiada ona informacje o lokalizacji użytkownika czyli o jego adresie. Ponadto zawiera informacje osobiste takie jak imię, nazwisko, pesel, płeć. Aby rozróżnić typy użytkowników tabela ta posiada atrybut o nazwie typ, będący enum. Z uwagi na możliwość określenia użytkownika jako pracownika, encja ta przechowuje również klucz obcy do grafiku oraz klucz obcy id\_przełożonego.
- 8) Wypożyczenie- ostatnią encją jest encja wypożyczenie zawierająca dane na temat czytelnika wypożyczającego, nr katalogowy książki wskazujący na konkretny egzemplarz książki, datę wypożyczenia i datę zwrotu oraz karę.

## 13. Organizacja projektu

Aby zrealizować zdecydowaliśmy się wybrać rodzaje techniki jego realizacji. Koniecznym było wybranie takich które zawierają zestaw wszystkich czynności, funkcji i zadań zarówno technicznych, jak i menedżerskich, wymaganych do zaspokojenia terminów i warunków realizacji tego projektu.

**Projekt inżynierii oprogramowania i systemu** – czynność ta wymaga określenia i spełnienia ustanowionych celów i planów, mieszcząc się w określonym budżecie oraz czasie wyznaczonym przez datę startu i datę jego ukończenia. Jego celem jest zaspokojenie wymagań funkcjonalnych i нефункциональных projektu wyszczególnione w uzgodnieniach projektowych na samym początku. Zużywa on zasoby.

**Projekt oprogramowania** –podczas tego procesu powstaje koncepcja wewnętrznej logiki systemu, jego architektura, interfejsy (w tym użytkownika). Obecnie stosuje się metody obiektowe, więc projekt często stanowi abstrakcję, dość dokładna jednak, przyszłego systemu.

**Projekt wstępny** – proces w którym tworzy się jest ciało normatywne, które zbiera szereg środków dla określonej funkcji, opracowuje program i chce, aby najpierw został zatwierdzony przez radę ministrów jako projekt ustawy, który w przypadku ostatecznego zatwierdzenia po wszystkich etapach byłby ustawą .Projekt wstępny jest wynikiem przebiegu procesów projektowania wstępnego produktu.

**Projektowanie do kosztu** – jest to proces zarządzania projektami, w której powinniśmy pokładać ufność w utrzymaniu projektu w budżecie określonym w harmonogramie. Oznacza to, że każdą potrzebę monitorujemy pod kątem istotności i ustalamy ściśle cele kosztowe projektowania i realizacji każdego działania w celu analizy (oszacowania) procesu projektowego (przykładem może być serwer, który zdecydowaliśmy się zakupić do projektu). Aby to zrobić, zaplanowaliśmy implikacje kosztowe (między 3 a 5 procent), jednocześnie szukając wykonalnego kompromisu między strategią operacyjną, zakresem i harmonogramem.

**Projektowanie szczegółowe** – proces weryfikacji, podczas którego usuwamy błędy, następuje rozszerzenie projektu wstępnego oprogramowania aby posiadał więcej szczegółowych opisów logiki przetwarzania, struktur danych oraz definicji danych do tego stopnia, gdzie projekt jest wystarczająco szczegółowy, aby mógł zostać wdrożony. Jest on wynikiem szczegółowego procesu projektowania, również uważany za opis specyfikacji projektu szczegółowego.

W naszym zespole realizacja projektu polegała na zaprojektowaniu każdego elementu projektu, ustaleniu podziału prac i ich terminów, spisywaniu raportów częściowych i końcowej dokumentacji dla użytkownika.

W trakcie realizacji projektu kierowaliśmy się czynnikami takimi jak:

- zakres projektu
- analiza zysków i strat
- nie ograniczanie oprogramowania na wprowadzanie zmian, mających udoskonalić jego działanie
- opracowanie strategii
- w trakcie rozwijania projektu ewolucja modelu
- korzystanie z różnego rodzaju dokumentacji i artykułów w celu utworzenia jak najnowszego oprogramowania
- oryginalność,
- działanie nastawione na dotarcie do niszy konsumenckiej
- podejmowanie uzasadnionego działania zbliżającego projekt do celu
- wyraźne rozplanowanie szczegółowo wszystkich kierunków działań i środków, za pomocą których można osiągnąć założony cel,
- wdrożenie dokładnego planu działania



## 14. Architektura systemu

Architekturę zastosowaną w projekcie stanowi połączenie aplikacji desktopowej wraz z bazą danych dostępną wewnątrz serwera bazy danych. Aplikacja umożliwia przeglądanie zasobów biblioteki, a po zalogowaniu umożliwia realizację procedur niezbędnych dla pracy pracownika biblioteki oraz zarządzania bibliotekami dla ich właścicieli lub administratorów. Aplikacja jest połączona bezpośrednio z bazą danych, do której wysyła wszystkie zapytania i czeka na odpowiedź.

**Aplikacja desktopowa:** czytelna, przejrzysta oraz szybka aplikacja napisana w języku programowania Java, pozwalająca na zarządzanie bibliotekami oraz pracownikami. Jest ona dostosowana do potrzeb użytkownika, posiada kalendarz, terminarz, listę książek oraz bibliotek, a także zbiór najważniejszych informacji związanych z zakresem prac pracownika. Jest także prosta w obsłudze, a jej sposób integracji z resztą projektu pozwala na liczne usprawnienia i modyfikacje. Proces rozbudowy aplikacji nie wpływa w żaden sposób na działanie bazy danych.

**Baza danych:** dostępna w ramach serwera (publicznego lub prywatnego), pozwala na wykonywanie zapytań i otrzymywania odpowiedzi w postaci zwróconych danych wewnątrz aplikacji. Wewnątrz bazy znajdują się informacje dotyczące bibliotek, książek oraz pracowników i właścicieli. Bezpośredni dostęp do bazy danych w pełnym zakresie jej funkcjonalności posiadają wyłącznie administratorzy. Baza danych jest ściśle powiązana z aplikacją. Serwer lokalny, na którym znajduje się aplikacja stanowi środowisko XAMPP, wraz ze wsparciem systemu MySQL, w oparciu o który baza danych została utworzona.

Zaletą zastosowanej architektury jest jej prostota oraz funkcjonalność. Podział architektury na dwie osobne części pozwala na niezależne funkcjonowanie obydwu serwisów, a także prostą rozbudowę oraz szybkie eliminowanie błędów. Architektura intuicyjnie dzieli także rolę poszczególnych użytkowników, dla przykładu pracownik posiada dostęp wyłącznie do części aplikacji (lub minimalny dostęp umożliwiający jedynie odczyt danych wewnątrz bazy danych), natomiast dla kontrastu administrator serwisowy posiada pełny dostęp zarówno do wewnętrznej struktury aplikacji jak i do bazy danych, w ramach której jest uprawniony do modyfikowania i przetwarzania dostępnych treści.

## 15. Błędy i usterki znalezione podczas testowania aplikacji

Podczas pracy nad projektem w tym w trakcie testowania go mieliśmy okazję napotkać kilka błędów i usterek które po wnikliwej analizie udało się rozwiązać. Jednym z nich był problem tworzenia w bazie encji o powtarzających się krotkach i sposobie połączenia ich przy użyciu javy.

Na samym początku podczas tworzenia podstawowych klas w przyjęliśmy że klucze główne będą się znajdowały w klasach przedstawiających encje będąc wyrażone przez typy prymitywne.

```
@Entity (name="Library")
@Table(name="Library")
public class Library {

    @Id
    @GeneratedValue(strategy=GenerationType.IDENTITY)
    @Column(name = "id", updatable = false, nullable = false)
    private long id;

    private int idLocation;
    private int ownerId;
    private String telephone;
    private int numEmployees;

    public long getId() {
        return id;
    }
}
```

*Przykład klasy Library przed zastosowaniem mapowania*

Po zauważeniu błędów i usterek jakie niosło za sobą to rozwiązanie zdecydowaliśmy się użyć elementów biblioteki javax.persistence.

```
@Entity (name="Library")
@Table(name="Library")
public class Library {

    @Id
    @GeneratedValue(strategy=GenerationType.IDENTITY)
    @Column(name = "idLibrary", updatable = false, nullable = false)
    private long idLibrary;
    private String name;
    private String telephone;

    @ManyToOne(optional=false)
    @JoinColumn(name="User",referencedColumnName="id")
    private User owner;

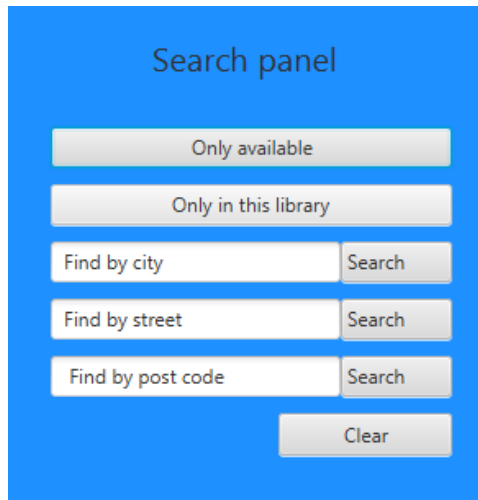
    @ManyToOne(optional=false)
    @JoinColumn(name="location",referencedColumnName="idLocation")
    private Location location;

    private int numEmployees;
}
```

*Przykład klasy Library po zastosowaniu mapowania z JpaRepository*

## 16. Rozwiązania błędów i usterek

Znalezioną usterką było błędne sformułowanie komendy do przeszukiwania po encji. Było ono potrzebne do Search Panel w widoku listy egzemplarzy dla książki.



### Search Panel

Program kończył się niepowodzeniem wyrzucając następujący error:

```
... 58 more
Caused by: java.lang.IllegalArgumentException: Parameter value [1] did not match expected type [com.codetreatise.bean.Library (n/a)]
    at org.hibernate.jpa.spi.BaseQueryImpl.validateBinding(BaseQueryImpl.java:897)
    at org.hibernate.jpa.internal.QueryImpl.access$000(QueryImpl.java:61)
    at org.hibernate.jpa.internal.QueryImpl$ParameterRegistrationImpl.bindValue(QueryImpl.java:235)
```

Rozwiązaniem okazało się odwołanie do atrybutu klasy mapowanej z klasą odpowiedzialną za poszukiwaną encję.

```
@FXML
void searchThisLibrary(ActionEvent event) {
    if(!myBean.getRoleUser().contains("Admin")) {
        List<Copy> copys = copyService.findByLibrary(myBean.getIdLibrary());
        loadUserDetails(copys);
    }
}
```

Wykorzystanie źle sformułowanego polecenia w Controllerze

```
-----
List<Copy> findByLibrary(long idLibrary);
List<Copy> findByStatus(String status);
```

Źle określone polecenie w pliku Repository

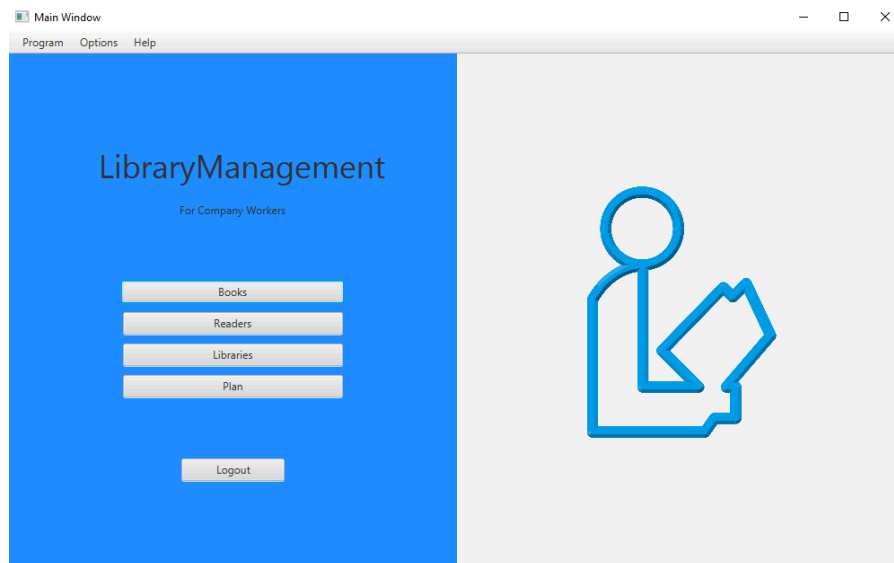
```
List<Copy> findByLibrary_idLibrary(long idLibrary);
```

Poprawione polecenie w pliku Repository

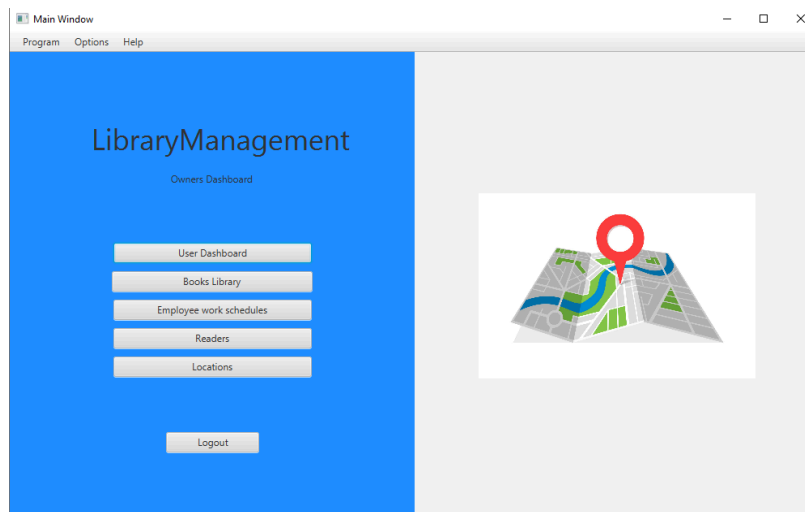
## 17. Dokumentacja użytkownika

Program posiada 3 możliwe typy użytkowników, takie jak właściciel biblioteki (Owner), administrator(Admin) oraz pracownik(User). Dla każdego z nich wyświetla się inne okno startowe .

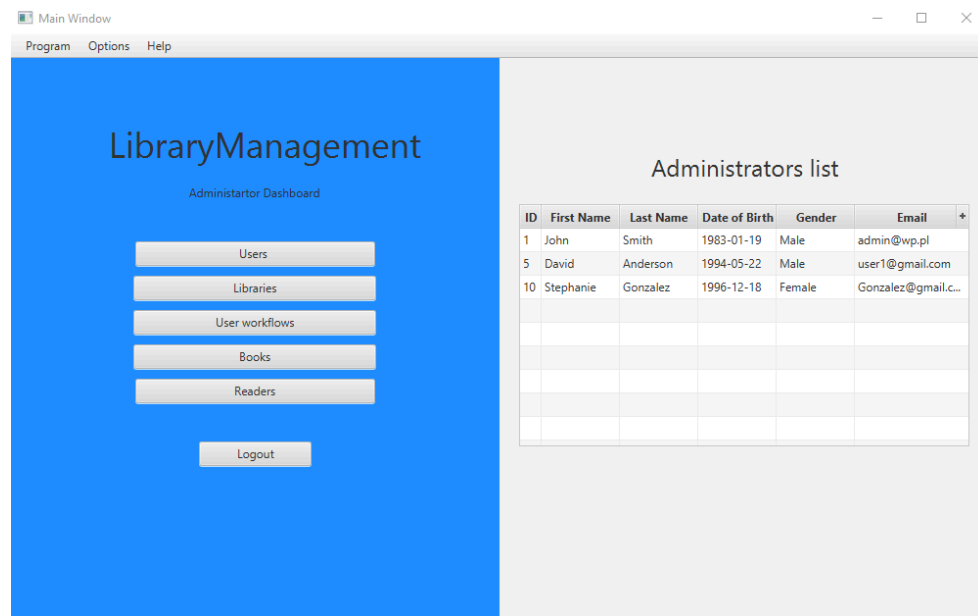
Okno startowe dla pracowników



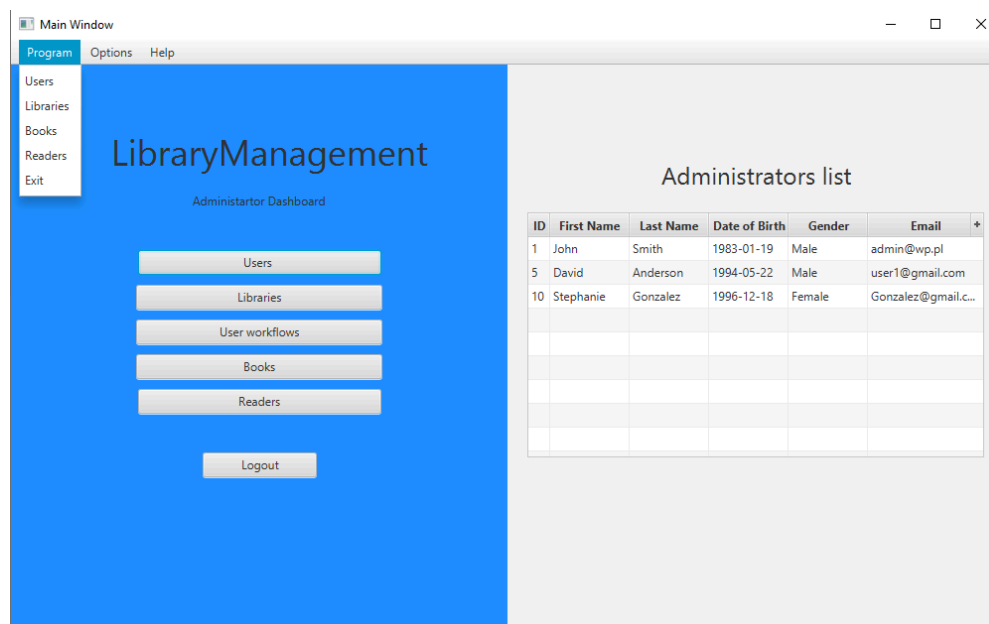
Okno startowe dla właścicieli



## Okno startowe dla administratorów



Każde z okien ma na górnym pasku menu z rozwijanymi opcjami, których zawartość różni się w zależności od kategorii użytkownika.



Są one następujące:

-**Books** -okno w które wchodzimy gdy chcemy wypożyczyć czytelnikowi egzemplarz książki.

-**Libraries**- okno z listą wszystkich bibliotek należących do sieci.

-**Readers**- okno z danymi wszystkich czytelników zarejestrowanych w sieci bibliotek.

**-Users-** dostępne tylko z poziomu administratora i właściciela okno zawierające informacje o użytkownikach.

**-Exit-** funkcja odpowiedzialna za natychmiastowe opuszczenie programu.

Ponadto w każdym z okien głównych posiadamy możliwość przejścia do grafików:

- w przypadku pracownika do jego własnego
- w przypadku właściciela do grafików jego pracowników
- w przypadku administratora do grafików wszystkich użytkowników

## Okno user

Do okna użytkownika dostęp mają tylko właściciele i administratorzy. Można wejść do niego z poziomu okna startowego dla tych dwóch typów. zawartość okna jest zależna od uprawnień wyświetlającego. Administrator systemu widzi w tabeli wszystkich użytkowników, natomiast właściciele tylko i wyłącznie swoich pracowników i siebie samych. Z tego poziomu istnieje także możliwość zmiany hasła i maila użytkownika. Po lewej stronie okna widnieje niebieski panel.

**Add New User**

User ID -

First Name

Last Name

Date of Birth

☒ Male ☐ Female

Select Role

Email

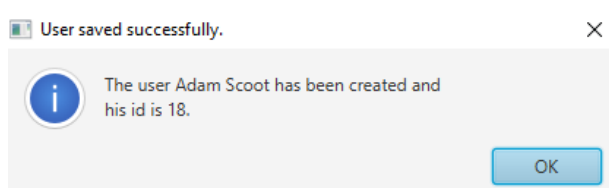
Password

Reset Save

ID	First Name	Last Name	Date of Birth	Gender	Role	Email	Edit
1	John	Smith	1983-01-19	Male	Admin	admin@wp.pl	
2	Michael	Johnson	1998-01-22	Male	Owner	Owner2@onet.pl	
3	Sarah	Brown	1996-06-12	Female	User	User@gmail.com	
4	Jessica	Rodriguez	1998-06-28	Female	Owner	Rodriguez@wwq.com	
5	David	Anderson	1994-05-22	Male	Admin	user1@gmail.com	
6	Elizabeth	Thomas	1999-08-20	Female	Owner	Thomas@yahoo.com	
7	Jennifer	Hernandez	1995-01-26	Female	User	pudzian@hotmail.com	
8	James	Moore	2003-09-25	Male	User	Moore@gmail.com	
9	Christopher	Thompson	1999-04-05	Male	User	Thompson@gmail.c...	
10	Stephanie	Gonzalez	1996-12-18	Female	Admin	Gonzalez@gmail.com	
11	Joseph	Harris	2003-11-29	Male	User	Harris@gmail.com	
12	Andrew	Lewis	1993-04-09	Male	User	Lewis@gmail.com	
13	Melissa	Young	1993-05-30	Female	Owner	Young@gmail.com	

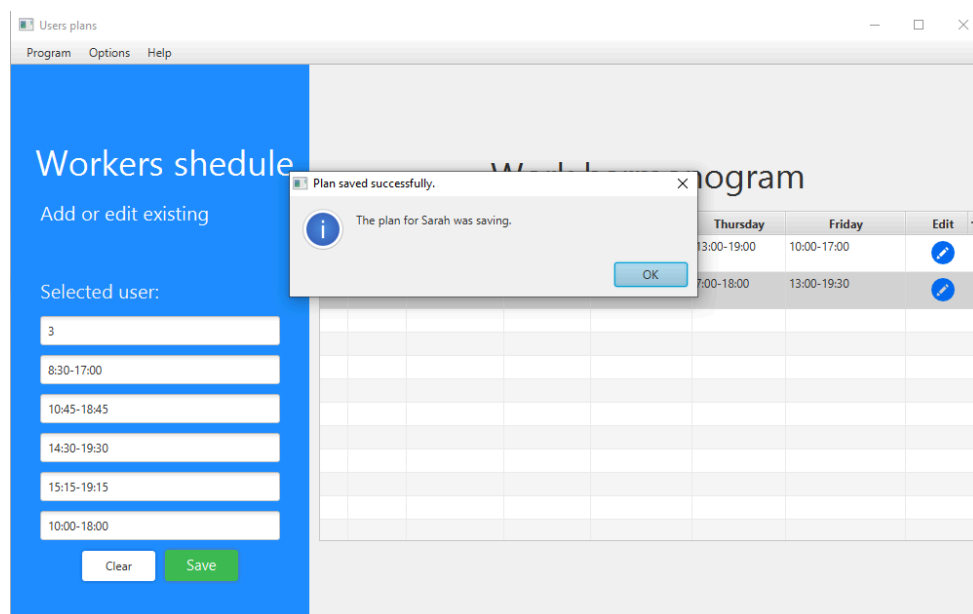
Dzięki polom do wpisywania tekstu możemy dodać nowego użytkownika, wpisując odpowiednie wartości. Aby je zapisać należy kliknąć na przycisk **Save**. W przypadku podania nieprawidłowych danych pojawi się okienko z komunikatem wskazującym na błąd. Wówczas należy kliknąć przycisk **Okey** i poprawić podane dane.

Podczas przebywania w tym widoku możemy również edytować użytkowników poprzez kliknięciem myszką na niebieską ikonkę przy prawej krawędzi okna programu. Po jej naciśnięciu w panelu po lewej strony pokażą się dane wybranego użytkownika, które można zedytować i zapisać. Podczas edycji użytkownika wyświetlany jest jego numer ID. Po poprawnie wykonanej procedurze dodawania, usuwania i edycji komunikat.



## Grafik użytkowników

Z poziomu okien głównych właściciela i administratora mamy również możliwość wejścia w okno grafików pracy dla pracowników, jednakże tak jak w przypadku okna user wyświetlanej tabeli jest zależna od uprawnień użytkownika. Właściciel jest w stanie zobaczyć i edytować krotki należące do pracowników jego biblioteki, podczas gdy administrator ma dostęp do całej encji zawierającej grafiki wszystkich pracowników pracujących dla określonej sieci bibliotek.



Dodawanie i edycja grafików przebiega w analogiczny sposób jak edycja wiersza w oknie użytkownik. A po poprawnym wprowadzeniu zmian lub zapisaniu nowych wartości występuje komunikat.

## Okno kolekcji książek sieci bibliotek

Okno kolekcji książek sieci bibliotek jest dostępne dla każdego użytkownika i jest także jedną z opcji dostępnych w górnym panelu na większości stron. Zawier on informacje na temat wszystkich możliwych tytułów książek które są zarejestrowane w sieci bibliotek.



Dodawanie i edycja już istniejących w bazie książek odbywa się w analogiczny sposób jak edycja czy zapisywanie danych w oknie użytkownika. A po poprawnie wykonanej procedurze użytkownik jest informowany komunikatem o sukcesie.

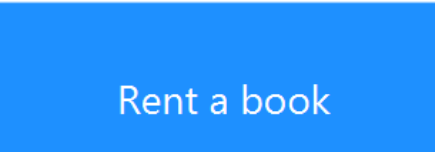




Po naciśnięciu na określony egzemplarz pokazuje się opcja **Rent**, odpowiedzialna za przeniesienie nas do okna przeprowadzającego proces wypożyczenia książki.

## Okno Wypożyczenia

Okno wypożyczenia jest małym oknem który jest odpowiedzialny tylko i wyłącznie za umieszczenie w bazie informacji o nowym wypożyczeniu egzemplarza książki przez użytkownika którego id podajemy w odpowiednim polu tekstowym. Aby potwierdzić proces klikamy przycisk **Accept** , w przypadku jeśli chcemy powrócić do poprzedniego okna klikamy przycisk **Rent**.



## Okno wszystkich bibliotek

Okno biblioteki zawiera dane na temat wszystkich bibliotek zawierających dane wszystkich bibliotek należących do sieci. Zmiany w nim mogą wprowadzać jedynie administratorzy.

[illegible]

## Okno czytelników

Okno czytelników jest oknem dostępnym z poziomu wszystkich użytkowników, przechowuje ono informacje o czytelnikach. Panele boczne odpowiedzialne są za dodanie czytelnika oraz jego blokowanie czy odblokowywanie. Aby tego dokonać należy po wprowadzeniu poprawnych danych kliknąć zielony przycisk z nazwą procedury którą chcemy wykonać.

ID	First Name	Last Name	Status
1	Mason	Green	true
2	Emma	Smith	true
3	Olivia	Brown	true
4	Isabella	Jones	true
5	Noah	Garcia	true
6	Sophia	Martinez	true
7	Jacob	Rodriguez	false
8	Charlotte	Perez	false
9	Aiden	Turner	true
10	Joshua	Campbell	true
11	Ava	Parker	true
12	Alexander	Evans	false
13	Scarlett	Edwards	true
14	Nathaniel	Collins	false
15	Hailey	Stewart	false
16	Brandon	Henderson	false
17	Emily	Cox	true
18	Sophia	Lee	false

## Okno historii wypożyczeń

Okno przedstawiające historię wypożyczeń umożliwia śledzenie stanu wypożyczonych książek. W umieszczonym po lewej stronie panelu wyszukiwania zamieszczone są pola tekstowe służące do wyszukiwania po określonych wartościach. Okno jest dostępne dla każdego użytkownika jednakże, wyświetlana zawartość jest różna w zależności od uprawnień. Pracownik i właściciel może wyświetlać wypożyczenia obejmujące tylko jego bibliotekę. Administrator zaś ma pełny dostęp do wszystkich wypożyczeń. Panel z przyciskami przy prawej krawędzi okna służy do ustawienia filtra wyświetlania wypożyczeń.

Refresh table						
	ID	Reader	Copy	dateHire	dateBack	penalty +
1	2		1	2023-01-17	2023-01-17	0
2	3		1	2023-01-17	2023-01-17	1
3	3		2	2023-01-17	2023-01-17	4
4	4		3	2023-01-17	2023-01-17	1
5	5		2	2023-01-17	2023-01-17	2
6	3		2	2023-01-17	2023-01-17	3
7	3		1	2023-01-17	2023-01-17	23
8	2		2	2023-01-17	2023-01-17	1
9	4		1	2023-01-17	2023-01-17	21
10	4		3	2023-01-17	2023-01-17	33

Zwrócić książkę można na dwa sposoby:

- klikając w wybrane wypożyczenie i jeśli nie jest ono zakończone zostaniemy przeniesieni do okna zwrotu z wypełnionymi wartościami.
- Wchodząc w **Options** i wybierając **Return** na panelu górnym aplikacji, wtedy zostaniemy przeniesieni do okna zwrotu z polami czekającymi na wpisanie odpowiednich wartości.

## Okno zwrotu

Okno zwrotu jest odpowiedzialne za edycję danych w systemie. Zmienia on wypożyczenie jako ukończone dodając do niego datę zwrotu. W polach tekstowych należy podać id czytelnika zwracającego egzemplarz oraz id tego egzemplarzu. Aby zakończyć procedurę należy nacisnąć przycisk **Accept**, w przypadku chęci anulowania procesu zwracania książki należy wcisnąć przycisk **Cancel**.

Return.fxml

Return of books

idReader

idCopy

Cancel

Accept

## 18. Aktualizacje i poprawki

Aplikacja będzie stale poprawiana i uaktualniana. Proces ciągłego rozwijania aplikacji podczas jej działania może prowadzić do niektórych trudności i błędów, jednak te są nieodłącznym i koniecznym elementem. Oczekiwane korzyści z wprowadzenia aplikacji na rynek jak najszybciej przewyższają potencjalne niedogodności. Po wdrożeniu aplikacji użytkownicy będą mogli zgłaszać opinie i sugestie co pozwoli na lepsze i szybsze rozwijanie systemu.

## 19. Wnioski

Program który stworzyliśmy podczas projektu uczelnianego na przedmiot inżynierie programowania jest kompletny i w pełni działający. Aplikacja ma na celu wspomóc zarządzanie siecią bibliotek dlatego też jej funkcjonalności są zaprojektowane właśnie w tym celu. Jego architektura oparta jest na mikroserwisach, co umożliwia łatwe rozwijanie i aktualizację aplikacji. Ta architektura pozwala także na szybsze i skuteczniejsze rozwiązywanie problemów systemowych. Okazało się, że obawy dotyczące ograniczonego czasu i zasobów na realizację projektu były przesadzone, ponieważ udało się stworzyć stabilny system spełniający podstawowe wymagania. Aplikacja jest skonstruowana z modułów połączonych ze sobą w jedną kompatybilną całość. System jest gotowy do wdrożenia w podstawowej formie, co zaspokoi potrzeby użytkownika, a kolejne moduły będą stopniowo do niego dopasowywane. Za pomocą odpowiednich narzędzi programistycznych, bibliotek i frameworków, czas potrzebny na stworzenie rozbudowanego projektu został skrócony.