Szkoła Główna Gospodarstwa Wiejskiego

w Warszawie

Wydział Zastosowań Informatyki i Matematyki

Sebastian Łyszkowski

Numer albumu: 191698

Aplikacja wspierająca korzystanie z biblioteki oraz wspomagająca jej pracę

Application supporting the use of the library

and supporting her work

Praca dyplomowa inżynierska

na kierunku – Informatyka

Praca wykonana pod kierunkiem

dr. Andrzeja Zembrzuskiego

Instytut Informatyki Technicznej

Katedra Systemów Informatycznych

Zakład Symulacji Komputerowych

Warszawa, 2021 rok

**Oświadczenie promotora pracy**

Oświadczam, że niniejsza pracazostała przygotowana pod moim kierunkiem i stwierdzam, że spełnia warunki do przedstawienia tej pracy w postępowaniu o nadanie tytułu zawodowego.

Data .................................... Podpis promotora pracy ...................................................

**Oświadczenie autora pracy**

Świadom odpowiedzialności prawnej, w tym odpowiedzialności karnej za złożenie fałszywego oświadczenia, oświadczam, że niniejsza praca dyplomowazostałanapisanaprzeze mnie samodzielnie i nie zawieratreści uzyskanych w sposób niezgodny z obowiązującymi przepisami prawa, w szczególności ustawą z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (Dz. U. Nr 90 poz. 631 z późn. zm.)

Oświadczam, że przedstawiona praca nie była wcześniej podstawą żadnej procedury związanej z nadaniem dyplomu lub uzyskaniem tytułu zawodowego.

Oświadczam, że niniejsza wersja pracy jest identyczna z załączoną wersją elektroniczną. Przyjmuję do wiadomości, że praca dyplomowa poddana zostanie procedurze antyplagiatowej.

Data ..................................... Podpis autora pracy .............................................

**Streszczenie**

**Aplikacja wspierająca korzystanie z biblioteki oraz wspomagająca jej pracę**

Celem niniejszej pracy było wytworzenie aplikacji umożliwiającej pracownikom biblioteki zdalne zarządzanie nią, a czytelnikom zdalne korzystanie z niej. Aplikacja została napisana w języku C# z wykorzystaniem platformy .NET. Przy tworzeniu baz danych do aplikacji zostało zastosowane podejście ”Code First”. Praca zaczyna się od wstępu teoretycznego poświęconego historii i działaniu technologii wykorzystanych przy tworzeniu aplikacji, takich jak: C#, .NET, Entity Framework. Główną częścią pracy jest omówienie funkcjonalności programu oraz ich implementacji. Na końcu pracy znalazła się ocena jakości wytworzonej aplikacji, w której zostały przedstawione zalety i wady stworzonej aplikacji oraz omówienie perspektyw jej rozwoju.

Słowa kluczowe – C#, .NET, Entity Framework, Windows Forms, MVC.

**Summary**

**Application supporting the use of the library and supporting her work**

The aim of this work was to create an application that would allow library employees to remotely manage it, and readers to use it remotely. The application was written in C# using the .NET platform. The "Code First" approach was applied to the application when creating the databases. The work begins with a theoretical introduction devoted to the history and usage of technologies used in the development of applications, such as: C#, .NET, Entity Framework. The main part of the work is to discuss the features of the program and their implementation. At the end of the work, the quality of the created application was assessed, in which the advantages and disadvantages of the created application were presented, as well as the prospects for its development.

Keywords – C#, .NET, Entity Framework, Windows Forms, MVC.

**Spis treści**

[1. Wstęp 9](#_Toc59010185)

[1.1. Cel i ogólny opis projektu 9](#_Toc59010186)

[1.2. Omówienie zawartości pracy 10](#_Toc59010187)

[2. Omówienie tematyki pracy 11](#_Toc59010188)

[2.1. Istniejące rozwiązania na rynku 11](#_Toc59010191)

[2.1.1. Bibliotekarz.NET 12](#_Toc59010195)

[2.1.2. Prolib 13](#_Toc59010196)

[2.2. Porównanie aplikacji 13](#_Toc59010204)

[3. Techniczne omówienie pracy 14](#_Toc59010205)

[3.1. Omówienie technologii wykorzystanych w projekcie 14](#_Toc59010209)

[3.2. Omówienie struktury aplikacji 18](#_Toc59010210)

[3.2.1. Omówienie struktury bazy danych 20](#_Toc59010211)

[3.2.2. Omówienie logiki aplikacji 21](#_Toc59010212)

[3.2.3. Omówienie widoku aplikacji 22](#_Toc59010213)

[4. Interfejs użytkownika 22](#_Toc59010214)

[4.1. Przygotowanie aplikacji do użycia 23](#_Toc59010215)

[4.2. Ogólne omówienie funkcjonalności z poziomu użytkownika 23](#_Toc59010216)

[4.2.1. Logowanie 24](#_Toc59010217)

[4.2.2. Rejestracja 25](#_Toc59010218)

[4.2.3. Menu główne 27](#_Toc59010219)

[4.2.4. Zakładka z informacjami o bibliotece 29](#_Toc59010220)

[4.2.5. Zakładka z użytkownikami 31](#_Toc59010221)

[4.2.6. Zakładka z książkami 32](#_Toc59010222)

[4.2.7. Dodawanie książki 34](#_Toc59010223)

[4.2.8. Edycja książki 35](#_Toc59010224)

[4.2.9. Zakładka z wypożyczeniami 36](#_Toc59010225)

[4.2.10. Zakładka z płatnościami 38](#_Toc59010226)

[5. Stworzenie aplikacji 40](#_Toc59010227)

[5.1. Omówienie najważniejszych implementacji 40](#_Toc59010228)

[5.2. Omówienie problemów napotkanych podczas tworzenia aplikacji 40](#_Toc59010229)

[6. Podsumowanie 40](#_Toc59010230)

[6.1. Ocena jakości i perspektywy rozwoju aplikacji 40](#_Toc59010231)

[6.2. Wnioski 40](#_Toc59010232)

[7. Bibliografia 40](#_Toc59010233)

# Wstęp

Postępujący nieustanie rozwój techniki przyczynił się do coraz większej automatyzacji wielu czynności człowieka, takich jak zamawianie jedzenia przez Internet, zarządzanie urządzeniami elektronicznymi w domu za pomocą aplikacji czy korzystanie z biblioteki przy użyciu aplikacji.

W swoim codziennym życiu czytam dużo książek oraz korzystam z biblioteki. Dzięki temu, że miałem często do czynienia z biblioteką wiem co powinna implementować aplikacja wspierająca korzystanie z niej i jej pracę. Wiem czego oczekują czytelnicy biblioteki od dobrze działającej aplikacji.

Moje zainteresowania oraz rozwój techniki spowodował, że postanowiłem stworzyć aplikację wspierającą korzystanie z biblioteki oraz wspomagająca jej pracę jako temat mojej pracy inżynierskiej. Chciałem też sprawdzić moje umiejętności nabyte podczas studiów poprzez stworzenie takiej aplikacji.

## Cel i ogólny opis projektu

Celem niniejszej pracy jest zaprojektowanie oraz stworzenie aplikacji, która wspierałaby pracę bibliotekarzy w zarządzaniu biblioteką i ułatwiałaby korzystanie z niej czytelnikom. Aplikacja jest przeznaczona na najpopularniejszy system operacyjny dla komputerów stacjonarnych – Windows 10. Aplikacja musi posiadać przejrzysty oraz intuicyjny interfejs, aby ludzie młodzi, jak i starsi nie mieli problemu z korzystaniem z aplikacji.

Aplikacja musi posiadać dwa typy kont: konto administratora (bibliotekarza) i konto zwykłego użytkownika (czytelnika). Konta posiadają różne uprawnienia, konto administratora posiada wszystkie funkcje konta użytkownika oraz specjalne dodatkowe funkcje administratora.

Zwykły użytkownik posiada następujące funkcje:

* przeglądanie bazy książek,
* rezerwowanie książek,
* przeglądanie historii swoich wypożyczeń,
* przeglądanie płatności.

Administrator oprócz funkcji zwykłego użytkownika posiada też:

* edytowanie informacji o bibliotece,
* zmienianie ustawień działania biblioteki,
* edytowanie tabeli książek i sygnatur,
* przeglądanie bazy danych,
* nakładanie oraz zdejmowanie blokad z kont użytkowników,
* zatwierdzanie płatności,
* zatwierdzanie wypożyczeń i zwrotów.

Aplikacja jest połączona z bazą danych, która posiada sześć tabel, które zostaną omówione w późniejszym rozdziale. Jako język bazodanowy wybrałem T-SQL używany w platformie Microsoft SQL Server, z którego korzystałem do zarządzania bazą danych.

## Omówienie zawartości pracy

W rozdziale 2 omawiam tematykę pracy jako zagadnienie związane z działaniem biblioteki. Opisuję jakie czynności się wykonuje w bibliotece, i które są implementowane w innych rozwiązaniach tego typu na rynku. Przedstawiam dwa inne rozwiązania istniejące na rynku. Krótko je opisuje i informuje o najważniejszych zaimplementowanych funkcjach. Następnie porównuje aplikacje z rynku z moją wizją aplikacji. Zwracam uwagę na najważniejsze różnice i podobieństwa.

W rozdziale 3 omawiam tematykę pracy z technicznego punktu widzenia. Przedstawiam ogólne założenia aplikacji. Najważniejszą częścią rozdziału jest omówienie technologii, które zostaną wykorzystane w aplikacji. Po omówieniu technologii przedstawiam ogólną strukturę rozwiązania, a następnie omawiam ją bardziej szczegółowo z podziałem na najważniejsze części.

Rozdział 4 przedstawia stworzoną aplikację. Na początku jest krótka informacja o sposobie instalacji i pierwszej konfiguracji aplikacji. Następnie jest pokazany przegląd wszystkich funkcjonalności aplikacji z poziomu użytkownika wraz z rysunkami przedstawiającymi wygląd aplikacji.

W rozdziale 5 pokazuje najważniejsze implementacje oraz przedstawiam problemy, które napotkałem podczas tworzenia aplikacji. Omawiam w szczegółowy sposób jak je rozwiązałem w mojej aplikacji.

Rozdział 6 jest podsumowaniem mojej pracy. Zawiera ocenę mojego rozwiązania oraz przemyślenia na temat przyszłych ulepszeń aplikacji. Na końcu rozdziału znajdują się wnioski wysunięte podczas tworzenia pracy.

# Omówienie tematyki pracy

W korzystaniu z biblioteki zawiera się wiele czynności takich jak: wypożyczanie i zwracanie książek, przeglądanie dostępnych książek, płacenie kar za przetrzymywanie książki, przejrzenie książki przed wypożyczeniem i sprawdzenie czy się czytelnikowi podoba. Aplikacja nie może implementować każdej z tych czynności, dlatego należy ocenić jakie funkcjonalności powinna mieć aplikacja, które są najważniejsze, a które mniej ważne. Do tego przyda się na pewno przegląd dostępnych rozwiązań na rynku, który został wykonany w dalszej części pracy. Pokazuje on jakie funkcje najczęściej posiadają rozwiązania o podobnej tematyce.



## Istniejące rozwiązania na rynku

Obecnie na rynku istnieje wiele różnorodnych rozwiązań o tej samej tematyce. Warto przed stworzeniem własnej aplikacji zwrócić uwagę na te rozwiązania i upewnić się, czy zaproponowane przez nas rozwiązanie już istnieje na rynku. Warto też sprawdzić, jakie funkcjonalności powinna implementować dobra aplikacja, aby odpowiadała na żądania użytkowników. Warto zwrócić uwagę na powtarzające się funkcjonalności w innych rozwiązaniach, bo warto też mieć je zaimplementowane w własnym rozwiązaniu. Dlatego warto sprawdzić, co oferuje rynek i porównać to z naszym rozwiązaniem.



### Bibliotekarz.NET



Rys. 1. Wygląd aplikacji Bibliotekarz.NET.

Źródło: Strona internetowa autora aplikacji[[1]](#_Bibliografia).

Bibliotekarz.NET to platforma rozwiązań bibliotecznych autorstwa Marka Ryńskiego. Powstała w 2004 roku, a najnowsza wersja (1.6.0.7) pochodzi z 2014 roku. Program posiada intuicyjny interfejs (Rys. 1) oraz charakteryzuje się wysoką wydajnością i bezpieczeństwem. Aplikacja jest całkowicie darmowa. System wspiera pracę bibliotek szkolnych, uczelnianych, firmowych, kościelnych, publicznych oraz własnych niedużych księgozbiorów. Twórca uważa, że system cechują następujące zalety:

* “dodawanie, usuwanie i edycja czytelników oraz publikacji,
* rozbudowane wyszukiwanie czytelników oraz publikacji,
* wypożyczanie, zwracanie, prolongowanie publikacji,
* możliwość korzystania z kilku baz danych,
* określanie lokalizacji, stanu oraz tematyki publikacji,
* statystyki graficzne,
* wycofywanie (ubytkowanie) publikacji,
* raporty PDF, XLS, HTML: zaległości, wypożyczeń, itp.,
* obsługa kodów kreskowych,
* pobieranie opisów książek na podstawie numerów ISBN z Internetu,
* wydruki: bazy publikacji, listy zaległości, listy czytelników i inne,
* zabezpieczenie programu przed osobami nieupoważnionymi,
* archiwizacja kopii bezpieczeństwa bazy,
* obsługa skontrum,
* powiadamianie czytelników via e-mail o zaległościach.”[[1]](#_Bibliografia)

### Prolib

Prolib to system informatyczny stworzony przez informatyków firmy Max Elektronik S.A. (aktualnie Sygnity Business Solutions). Aplikacja jest przeznaczona do obsługi i zarządzania różnego rodzaju bibliotek. Prolib jest komercyjnym rozwiązaniem wciąż rozwijanym przez dużą firmę i jest używany w wielu placówkach (na czas pisania pracy jest to 285). Firma gwarantuje polskojęzyczny system wsparcia i usługi serwisowe. System posiada takie funkcjonalności jak:

* “Szybka i wieloaspektowa obsługa czytelników.
* Pozyskiwanie, opracowywanie, wyszukiwanie i ocena informacji.
* Wsparcie dla administracyjnych i informatycznych obszarów działalności biblioteki.
* Gromadzenie, opracowywanie, udostępnianie tradycyjnych i cyfrowych zasobów bibliotek.
* Optymalizacja pracy innych działów instytucji. Integracja systemu bibliotecznego z innymi systemami, np. działu kadr, księgowości, dziekanatu i działu IT.
* Umożliwienie użytkownikom samodzielnego i efektywnego zarządzania eksploatacją systemu bibliotecznego, jego skalowalności i adaptacji do aktualnej sytuacji biblioteki w odpowiedzi na rozwój i zmiany organizacyjne instytucji.”[[2]](#_Bibliografia)



## Porównanie aplikacji

Analizując istniejące rozwiązania dostępne na rynku, należy przyznać, że są dobrymi rozwiązaniami, ale jak każde rozwiązanie, mimo że posiada wiele zalet, posiada też wady. Bibliotekarz.NET jest dobrym i darmowym rozwiązaniem, ale nie jest już utrzymywane, a ostatnia wersja pochodzi z 2014 roku. Czyli jest to dość stare rozwiązanie i aplikacja mogła stać się nieaktualna. W przeciwieństwie do wcześniejszej aplikacji, aplikacja Prolib jest dużym i stale rozwijanym rozwiązaniem, ale jest rozwiązaniem komercyjnym i wiele mniejszych bibliotek nie może sobie pozwolić na taką aplikację.

W przeciwieństwie do Prolib moje rozwiązanie nie będzie rozwiązaniem komercyjnym, dzięki czemu aplikacja będzie dostępna dla jak najszerszego grona użytkowników oraz bibliotek, także dla tych mniejszych. Niestety ze względu na posiadane zasoby moja aplikacja nie będzie tak mocno rozwinięta jak Prolib, która była tworzona przez wielu informatyków oraz później utrzymywana przez dużą firmę. W porównaniu do aplikacji Bibliotekarz.NET, moja aplikacja będzie korzystała z nowocześniejszych rozwiązań oraz mam w planach pewien okres czasu na dalszy rozwój aplikacji. Analizując różnice między rozwiązaniami uważam, że moja aplikacja znajdzie swoje miejsce na rynku oraz użytkowników.

# Techniczne omówienie pracy

Przed stworzeniem aplikacji warto poświęcić czas na zaplanowanie jej. Warto sprawdzić jakie technologie będę najlepiej pasowały do naszego rozwiązania oraz przybliżyć informacje o nich w pracy. Bardzo ważną częścią tworzenia aplikacji jest planowanie. Dobrze jest zaplanować przed przystąpieniem do pracy ogólną strukturę aplikacji, wybrać pasujące wzorce projektowe i stworzyć schemat bazy danych wykorzystywanej w rozwiązaniu. W tym rozdziale zostanie omówione wszystko to o czym napisałem przed chwilą.



## Omówienie technologii wykorzystanych w projekcie

W tworzeniu aplikacji w głównej mierze wykorzystuję język C#. Jest to wysokopoziomowy, zorientowany obiektowo język programowania ogólnego przeznaczenia stworzony przez firmę Microsoft w roku 2000 i jest nadal rozwijany. Najnowszą wersją języka C# na czas pisania pracy jest C# 9.0 (premiera w październiku 2020 roku) i właśnie tą wersję wykorzystuję w aplikacji. Język C# jest bardzo podobny do języka Java, o czym świadczy stwierdzenie autorów książek o języku C++ „*Java and C# are almost identical programming languages.*”[[3]](#_Bibliografia). Głównymi cechami języka C# są:

* Obiektowość z dziedziczeniem – przekazywanie cech klasy bazowej do klasy podrzędnej. Tak jak w języku Java, dziedziczenie opiera się na jednej klasie nadrzędnej, po której dziedziczą inne klasy. W języku C# jest to System.Object, po której dziedziczy reszta klas.
* Odśmiecacz pamięci (ang. *Garbage Collection*) – system zarządzania pamięcią, usuwa referencje do nieużywanych obiektów, dzięki czemu w większości przypadków programista nie musi się troszczyć o zwalnianie pamięci poprzez wywoływanie destruktorów.
* Typy ogólne (ang. *Generics*) – Typy ogólne umożliwiają dostosowanie metody, klasy, struktury lub interfejsu do precyzyjnego typu danych, na którym działa. Korzyści płynące z typów ogólnych to zwiększone wykorzystanie kodu i bezpieczeństwo typów.[[4]](#_Bibliografia)
* Delegaty i zdarzenia – odpowiednik wskaźników w C++, delegaty przechowują referencję do metody lub grupy metod i nie posiadają implementacji.
* Bogata biblioteka klas BCL (*Base Class Library*), pozwalająca na rozwój aplikacji konsolowych, okienkowych (WinForms, WPF), bazodanowych (ADO.NET), sieciowych (System.Net) oraz internetowych (ASP.NET).

Aplikacja jest tworzona w oparciu o platformę .NET. Po języku C#, platforma .NET jest najważniejszą częścią pracy. .NET to platforma programistyczna stworzona przez firmę Microsoft, obejmująca środowisko uruchomieniowe i wiele różnych bibliotek umożliwiających tworzenie aplikacji.

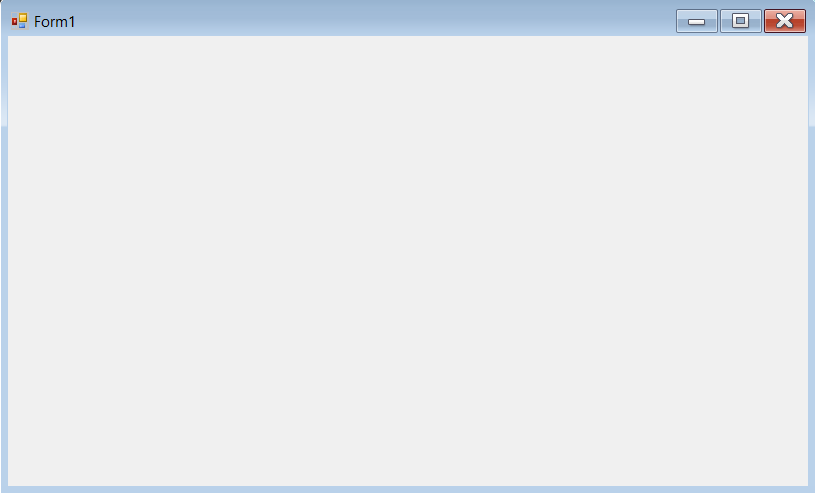


Rys. 2. Komponenty oraz rozwój platformy .NET.

Źródło: Wikipedia[[5]](#_Bibliografia).

Najnowsza wersja .NET Framework to 4.8 (aplikacja używa wersji 4.7.2), a pierwsza wersja została wydana w 2002. Od początku istnienia platforma .NET została rozbudowana o wiele komponentów (Rys. 2). Główne komponenty platformy .NET używane w projekcie to: WinForms, LINQ i Entity Framework (przestał być komponentem od 2013 roku).

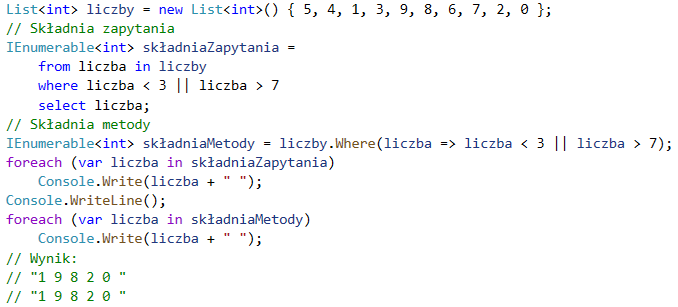
WinForms to graficzny interfejs użytkownika zawarty w .NET Framework, który umożliwia natywny dostęp do elementów interfejsu graficznego systemu Microsoft Windows. Korzystając z środowiska programistycznego takiego jak Visual Studio, można tworzyć aplikacje WinForms, które wyświetlają informacje, żądają danych wejściowych od użytkowników i komunikują się z komputerami zdalnymi przez sieć. W Windows Forms formularz to wizualna powierzchnia, na której wyświetlane są informacje dla użytkownika. Zwykle tworzysz aplikacje Windows Forms, dodając kontrolki do formularzy i opracowując odpowiedzi na akcje użytkownika, takie jak kliknięcia myszą lub naciśnięcia klawiszy. Kontrolka to dyskretny element interfejsu użytkownika (UI), który wyświetla dane lub akceptuje wprowadzanie danych.[[6]](#_Bibliografia) Podstawowy wygląd aplikacji z użyciem tej aplikacji jest przedstawiony poniżej (Rys. 3).



Rys. 3. Podstawowy wygląd aplikacji stworzonej z wykorzystaniem biblioteki WinForms.

Źródło: Opracowanie własne

Kolejnym ważnym komponentem platformy .NET używanym w projekcie jest technologia LINQ (ang. *Language INtegrated Query*), która umożliwia zadawanie pytań na obiektach. Dzięki LINQ można łatwo operować na zbiorach danych. Istnieją trzy sposoby na tworzenie zapytań w LINQ: składnia zapytania (ang. *query syntax*), składnia metody (ang. *syntax method*) i połączenie obu składni. W projekcie wykorzystujemy LINQ do operacji na zbiorach z bazy danych.



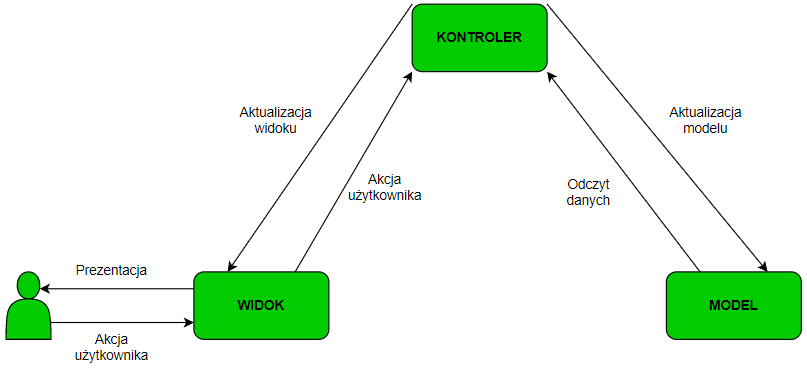
Rys. 4. Przykład wykorzystania LINQ na dwa sposoby: składanie zapytań i składanie metod.

Źródło: Opracowanie własne

W 2008 roku do platformy .NET (do wersji 3.5 SP1) dołączył Entity Framework i był jej integralną częścią aż do 2013 roku, kiedy Entity Framework 6 stał się oddzielną technologią. EF (Entity Framework) jest platformą mapowania obiektowo – relacyjnego (ang. *Object-Relational Mapping – ORM*). Kilka głównych cech EF to:

* Wieloplatformowość – Entity Framework Core może być używany na systemach Windows, Linux i Mac.
* Działania na bazach danych – EF umożliwia korzystanie z LINQ i zapytań SQL do operacji na zbiorach danych w bazach danych.
* Śledzenie zmian – EF śledzi zmiany w instancjach encji, które muszą zostać zgłoszone do bazy danych.
* Zapisywanie zmian – EF wykonuje operacje INSERT, UPDATE, DELETE w zależności od zmiany w bazie danych podczas wywołania metody SaveChanges().
* Migracje – EF oferuje zestaw komend do migracji wywoływanych w CLI (ang. *Command Line Interface*) lub konsoli menadżera pakietów NuGet, aby móc tworzyć i zarządzać schematami baz danych.

## Omówienie struktury aplikacji

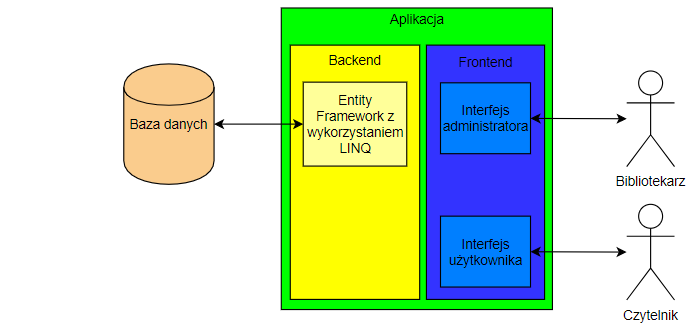


Rys. 5. Wzorzec MVC w mojej aplikacji

Źródło: Opracowanie własne.

W tworzeniu aplikacji zastosowałem wzorzec architektoniczny MVC ( ang. *Model-View-Controller*), Model-Widok-Kontroler to wzorzec stosowany przy organizowaniu struktury aplikacji posiadającej graficzny interfejs użytkownika (ang. *GUI – Graphical User Interface*). MVC zakłada podział aplikacji na trzy części:

* Model – centralny komponent wzorca. Dynamiczna struktura danych, niezależna od interfejsu użytkownika. Zarządza bezpośrednio danymi i logiką.
* Widok – dowolna reprezentacja informacji. Wyświetla pewną część modelu w ramach graficznego interfejsu użytkownika.
* Kontroler – akceptuje dane wejściowe od użytkownika i reaguje na jego działania, zarządzając zmianami modelu i widoku.



Rys. 6. Ogólna struktura aplikacji.

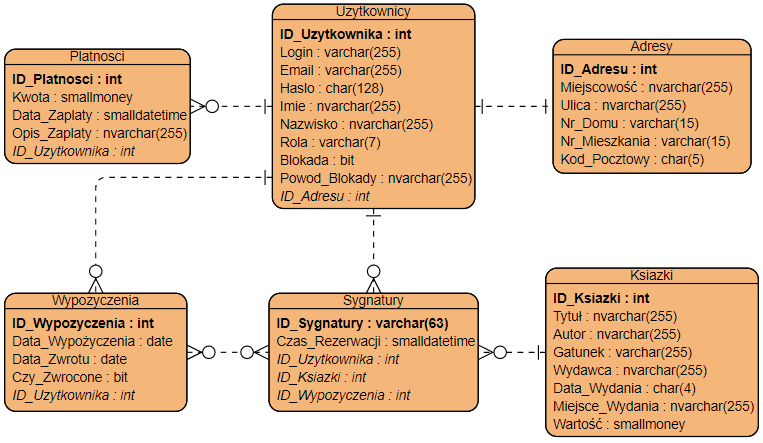
Źródło: Opracowanie własne.

Na stworzenie aplikacji składają się trzy główne zadania. Pierwszym z nich jest stworzenie bazy danych do przechowywania danych z biblioteki. Baza jest tworzona za pomocą Entity Framework podejściem „*Code First*”, czyli najpierw tworzymy klasy i relacje między nimi w kodzie projektu, a potem podczas pierwszego uruchomienia aplikacji Entity Framework zainicjuje bazę danych. Entity Framework ułatwia też migracje bazy danych w razie potrzeby wprowadzenia jakichkolwiek zmian w bazie. Wykorzystanie Entity Framework dzięki tym rozwiązaniom pozwala zaoszczędzić czas na tworzeniu danych. Do operacji na zbiorach z bazy danych będzie wykorzystywany LINQ.

Drugim zadaniem jest stworzenie całej logiki aplikacji oraz implementacja wszystkich funkcji pozwalających na korzystanie z aplikacji (patrz. [Cel i ogólny opis projektu](#_Cel_i_ogólny)). To tutaj zostaną zaimplementowane operacje na bazie danych z wykorzystaniem Entity Framework i LINQ. Muszą zostać obsłużone wszystkie wyjątki, żeby aplikacja działała bez błędów.

Trzecim zadaniem jest stworzenie widoku aplikacji zgodnego z systemem Windows 10, który jest podzielony na dwie części: interfejs użytkownika i interfejs administratora. Bibliotekarz ma dostęp do interfejsu administratora, a czytelnik do interfejsu użytkownika. Interfejs musi być przejrzysty i intuicyjny.

### Omówienie struktury bazy danych



Rys. 7. Diagram relacji encji (ang. Entities Relationship Diagram).

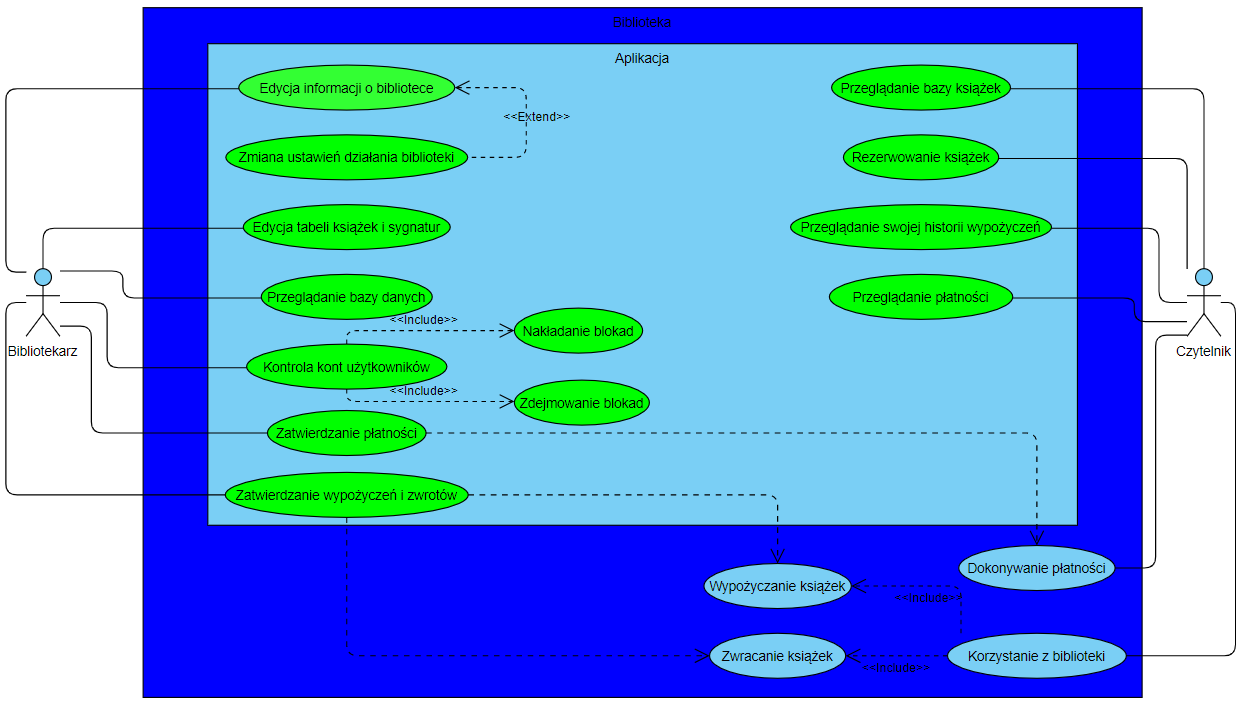
Źródło: Opracowanie własne.

Zanim przejdę do omawiania struktury bazy danych użytej w aplikacji, wyjaśnię kilka pojęć związanych z bazami danych, żeby ułatwić zrozumienie dalszej części tekstu. Tymi pojęciami są:

* Klucz podstawowy (ang. *Primary Key*) – to kolumna (lub zestaw kolumn, wtedy mamy do czynienia z kluczem złożonym) w tabeli, która posiada atrybut klucza podstawowego, a jej wartość jednoznacznie identyfikuje rekord w tabeli. Wartość tej kolumny musi być unikatowa dla każdego rekordu w tabeli.
* Klucz obcy (ang. *Foreign Key*) – to kolumna lub zestaw kolumn w tabeli, która posiada atrybut klucza obcego i oznacza wartość jednoznacznie identyfikującą rekord w innej tabeli będącej w relacji z daną tabelą. Bardzo często jest to klucz podstawowy drugiej z tabel.
* Relacja jeden do jednego (ang. *one to one*) – jest to relacja między tabelami tej samej bazy danych oznaczająca, że jeden rekord z pierwszej tabeli może być powiązany z jednym rekordem z drugiej tabeli.
* Relacja jeden do wielu (ang. *one to many*) – jest to relacja między tabelami tej samej bazy danych oznaczająca, że jeden rekord z pierwszej tabeli może być powiązany z jednym lub wieloma rekordami z drugiej tabeli.
* Relacja wiele do wielu (ang. *many to many*) – jest to relacja między tabelami tej samej bazy danych oznaczająca, że jeden lub wiele rekordów z pierwszej tabeli może być powiązany z jednym lub wieloma rekordami z drugiej tabeli.

Skoro najważniejsze pojęcia zostały wyjaśnione możemy przystąpić do omówienia struktury bazy danych. Baza danych projektu posiada sześć tabel powiązanych ze sobą relacjami. Schemat relacji tabel jest przedstawiony na rysunku nr. 7. Aby ułatwić zrozumienie diagramu przedstawię informacji zawartych na diagramie. Tabela Użytkownicy jest w relacji z tabelą Sygnatury, Wypozyczenia i Platnosci jeden do wielu oraz w relacji jeden do jednego z tabelą Adresy. W tabeli Uzytkownicy kolumna ID\_Uzytkownika to klucz podstawowy (pogrubiona czcionka), a kolumna ID\_Adresu to klucz obcy (napisane kursywą). Tabela Sygnatura jest w relacji wiele do wielu z tabelą Wypozyczenia. Diagram (Rys. 6) przedstawia też wybór typu danych dla odpowiednich kolumn takich jak: bit, char, nvarchar, int.

### Omówienie logiki aplikacji



Rys. 8. Diagram przypadków użyć (ang. *Use Case Diagram*)

Źródło: Opracowanie własne.

Aplikacja musi umożliwiać użytkownikom korzystanie z wszystkich wspomnianych w ogólnym opisie projektu funkcji, tak jak przedstawia rysunek nr. 8. Korzystanie z aplikacji musi być dostępne dopiero po zalogowaniu. Hasła do kont powinny być zabezpieczone poprzez zaszyfrowanie. Jako funkcję do hashowania wybrałem SHA-512, ponieważ należy do zestawu funkcji SHA-2, który jest znacznie bardziej bezpieczny niż funkcje z zestawu SHA-1. SHA-512 tworzy 64 bitowy ciąg znaków.

### Omówienie widoku aplikacji



Rys. 9. Graficzny interfejs użytkownika stworzony w aplikacji.

Źródło: Opracowanie własne.

Stworzyłem swój własny graficzny interfejs. Pozbyłem się domyślnego obramowanie aplikacji i zaimplementowałem własne zamykanie, maksymalizowanie i minimalizowanie aplikacji i ruszanie okna aplikacji. Ikony wykorzystane w aplikacji pochodzą z strony internetowej[[7]](#_Bibliografia) z m. in. darmowymi ikonami. Uważam, że interfejs jest intuicyjny, co za tym idzie łatwy w obsłudze.

# Interfejs użytkownika

Główną częścią tego rozdziału będzie omówienie wszystkich funkcjonalności aplikacji z poziomu użytkownika. Będzie też krótko wyjaśnione w jaki sposób korzystać z aplikacji, aby wszystko działało prawidłowo.

## Przygotowanie aplikacji do użycia

W tym podrozdziale krótko opiszę przygotowanie aplikacji do wykorzystywania w bibliotece. Moja aplikacja jest darmowa, dlatego jest udostępniona w serwisie GitHub na moim koncie. Biblioteka może ją stamtąd[[8]](#_Bibliografia) pobrać. Razem z aplikacją będzie znajdowała się bardziej szczegółowa instrukcja przygotowania aplikacji. Biblioteka przed udostępnieniem aplikacji czytelnikom musi stworzyć połączenie z bazą danych, w której będą umieszczane dane wykorzystywane przez aplikację. Aby to zrobić administrator musi zmienić connectionString, który jest ciągiem znaków definiującym połączenie. Żeby to zrobić administrator otwiera plik App.config, który znajduje się w folderze Widok. Tam znajduje miejsce, gdzie jest zdefiniowany connectionString i go zmienia na taki, który pozwoli połączyć się z odpowiednią bazą danych. Następnie należy skompilować projekt instalacyjny w rozwiązaniu i na tym kończy się przygotowanie. Od tego momentu biblioteka może opublikować plik instalacyjny dla czytelników. Sam proces instalacji aplikacji jest bardzo prosty i intuicyjny przypominający instalację innych aplikacji.



Rys. 10. Rysunek przedstawia zawartość pliku App.config wraz z zaznaczoną częścią zawierającą connectionString.

Źródło: Opracowanie własne.

## Ogólne omówienie funkcjonalności z poziomu użytkownika

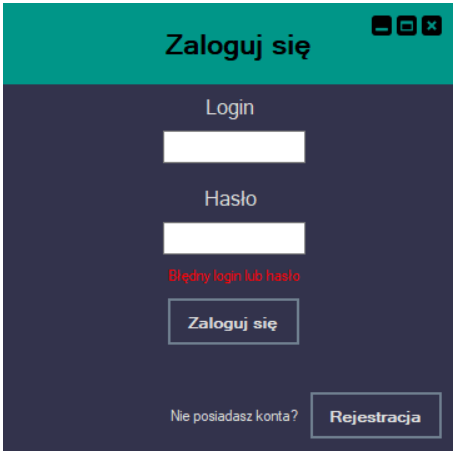
W tym podrozdziale zostaną omówione i wyjaśnione wszystkie funkcjonalności zaimplementowane w aplikacji z poziomu użytkownika. Rozdział będzie zawierał rysunki wyglądu aplikacji przy odpowiedniej funkcjonalności, aby ułatwić orientowanie się w aplikacji.

### Logowanie



Rys. 11. Okno logowania do aplikacji.

Źródło: Opracowanie własne.

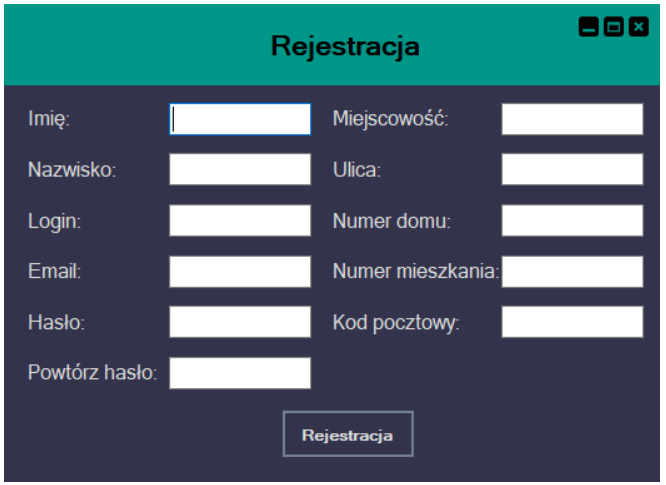


Rys. 12. Obraz okno logowania przy błędnych danych.

Źródło: Opracowanie własne.

Po uruchomieniu aplikacji na środku ekrany pojawi się okno logowania (Rys. 11). Logujemy się za pomocą loginu i hasła. W razie błędnego podania danych pojawi się odpowiedni komunikat (Rys. 12).

### Rejestracja



Rys. 13. Okno rejestracji do aplikacji.

Źródło: Opracowanie własne.



Rys. 14. Okno rejestracji przy nieprawidłowych danych.

Źródło: Opracowanie własne.

Jeśli użytkownik nie ma konta w aplikacji, to może je utworzyć przechodząc z okna logowania za pomocą przycisku „Rejestracja”. Wtedy pojawi się okno rejestracji (Rys. 13), w którym trzeba podać takie informacje jak: imię, nazwisko, login email, hasło, potwierdzenie hasła, miejsce zamieszkana (miejscowość, ulica, numer domu, numer mieszkania i kod pocztowy). W razie nie prawidłowych informacji kolor nazwy danych obok miejsca do wpisania danych zmieni kolor na czerwony (Rys. 14). W większości przypadków aplikacja wymaga co najmniej jedno znakowy ciąg. W przypadku hasła wymaga co najmniej 8 znaków, a kod pocztowy musi pasować do wzoru CC-CCC (C – cyfra).

### Menu główne



Rys. 15. Widok meny głównego dla administratora.

Źródło: Opracowanie własne.



Rys. 16. Widok meny głównego dla czytelnika.

Źródło: Opracowanie własne.



Rys. 17. Okno po zalogowaniu się na zablokowane konto.

Źródło: Opracowanie własne.

Po zalogowaniu się do aplikacji pojawi się okno z menu głównym (Rys. 15, 16), które w zależności od rodzaju konta będzie różniło się dostępem do tabeli kont użytkowników, do którego dostęp ma administrator, a zwykły czytelnik nie. Po lewej stronie okna znajdują się przyciski, po których naciśnięciu pojawiają się odpowiednie zakładki. W aplikacji znajduje się pięć zakładek:

* Biblioteka – zawiera informację o bibliotece, administrator może je edytować.
* Użytkownicy – wyświetla użytkowników i pozwala nimi zarządzać administratorowi.
* Książki – wyświetla książki z bazy, pozwala na ich rezerwację i zmiany w tabeli.
* Wypożyczenia – wyświetla historię wypożyczeń i pozwala administratorowi na tworzenie wypożyczeń i zwrotów.
* Płatności – wyświetla historię płatności i pozwala administratorowi na tworzenie płatności.

Kliknięcie w logo „*LibNet*” w lewym, górnym rogu spowoduje zamknięcie zakładki i powrót do początkowego widoku. Na górze aplikacji znajduje się zegar z dokładną datą oraz godziną. Jeśli konto, na które się zalogowaliśmy jest zablokowane, wtedy pojawi się informacja w początkowej zakładce (Rys. 17). W przyszłości dość pusta zakładka początkowa przyda się przy rozwoju aplikacji, o którym napiszę w późniejszym rozdziale.

### Zakładka z informacjami o bibliotece



Rys. 18. Okno z informacjami o bibliotece z poziomu administratora.

Źródło: Opracowanie własne.

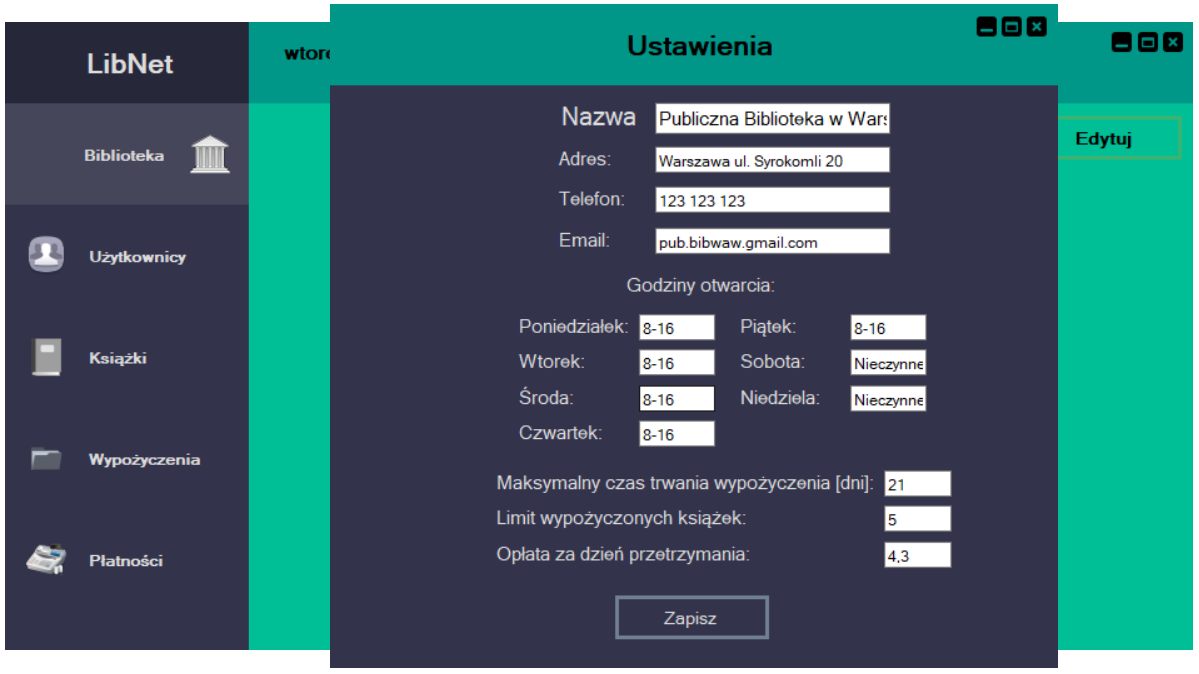


Rys. 19. Okno z informacjami o bibliotece z poziomu czytelnika.

Źródło: Opracowanie własne.

Jedną z podstawowych funkcjonalności aplikacji jest wyświetlanie informacji o bibliotece (Rys. 18, 19). Ta zakładka wyświetla podstawowe informacje i bibliotece. Administrator może te informacje edytować. Aby przejść do edycji informacji o bibliotece należy wcisnąć przycisk „*Edytuj*”.

#### Edycja informacji o bibliotece

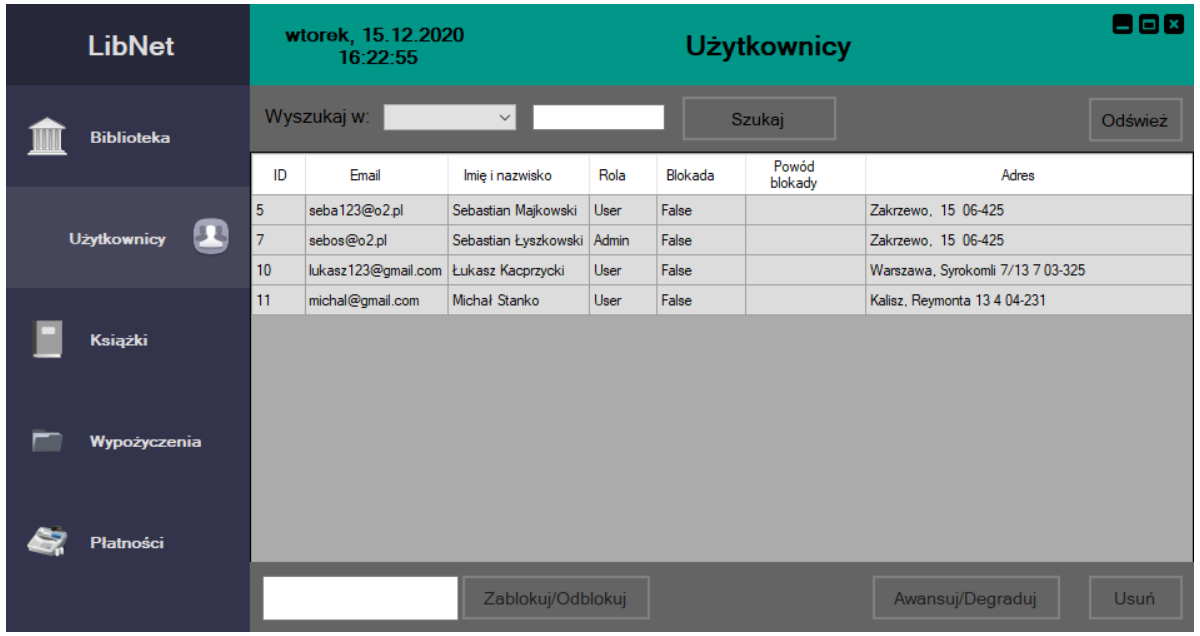


Rys. 20. Okno edycji informacji o bibliotece.

Źródło: Opracowanie własne.

Administrator może edytować informacje o bibliotece. Dla ułatwienia podczas edycji miejsca do wpisania nowych informacji są zapełniane aktualnymi informacjami. Mimo możliwości edycji tych informacji, zalecam nie wykorzystywać tej opcji zbyt często, gdyż informacje są przechowywane w pliku, który posiada każdy egzemplarz. Dlatego zmiany w informacjach wymagają opublikowania aktualnego pliku lub całej aplikacji do pobrania.

### Zakładka z użytkownikami

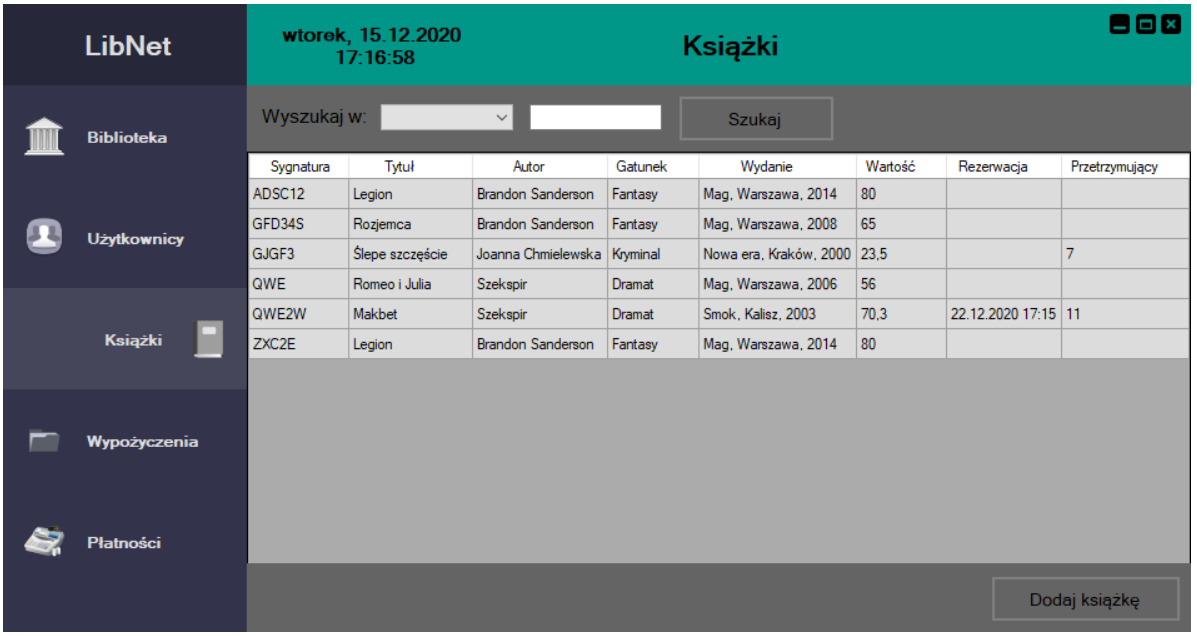


Rys. 21. Okno wyświetlające tabelę użytkowników.

Źródło: Opracowanie własne.

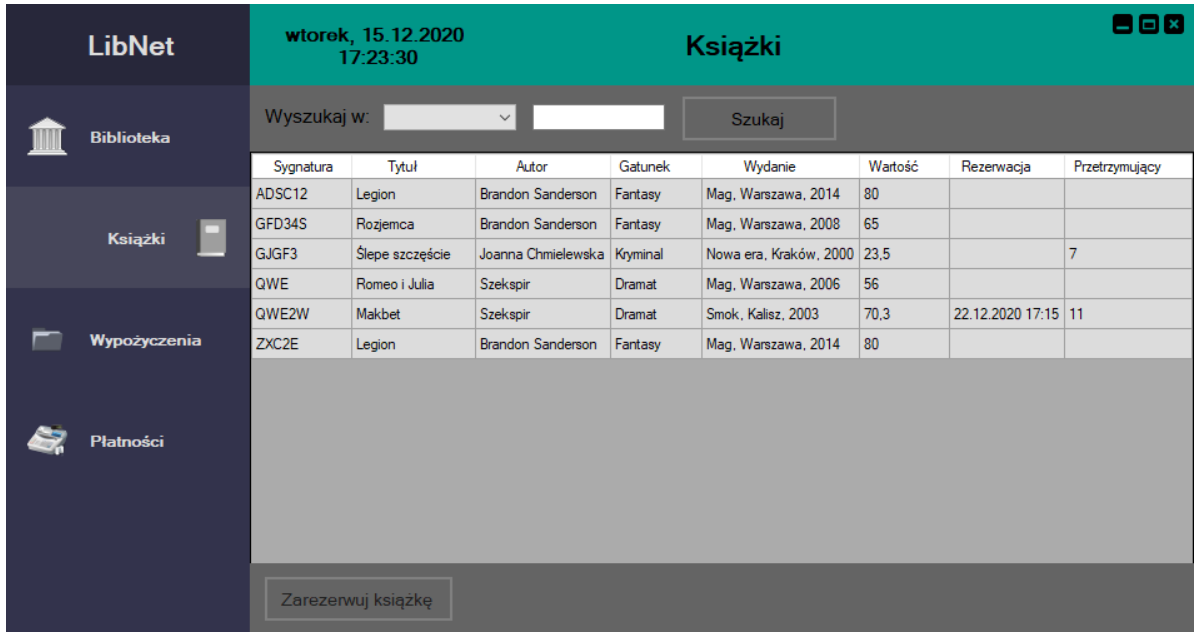
Administrator ma dostęp do zakładki „*Użytkownicy*”, gdzie może przeglądać bazę użytkowników. Może zablokować lub odblokować konto, dodatkowo może podać powód blokady. W razie zablokowanego konta, użytkownik nie będzie mógł rezerwować książek ani ich wypożyczać. Inne czynności jakie można tutaj wykonać to: nadawanie lub odebranie uprawnień administratora i usuwanie konta. Aby wykonać, którąś z tych czynności należy najpierw zaznaczyć w tabeli, o którego użytkownika chodzi. W innym wypadku przyciski pozostaną zablokowane. W górnej części znajdują się filtry, gdzie można wybrać, w której kolumnie chcę się znaleźć daną frazę. Przycisk „*Odśwież*” służy do sprawdzania kont, czy powinny zostać zablokowane za przetrzymywanie książki. System automatycznie zablokuje konta, które przetrzymują zbyt długo książki, co ułatwia pracę bibliotekarza.

### Zakładka z książkami



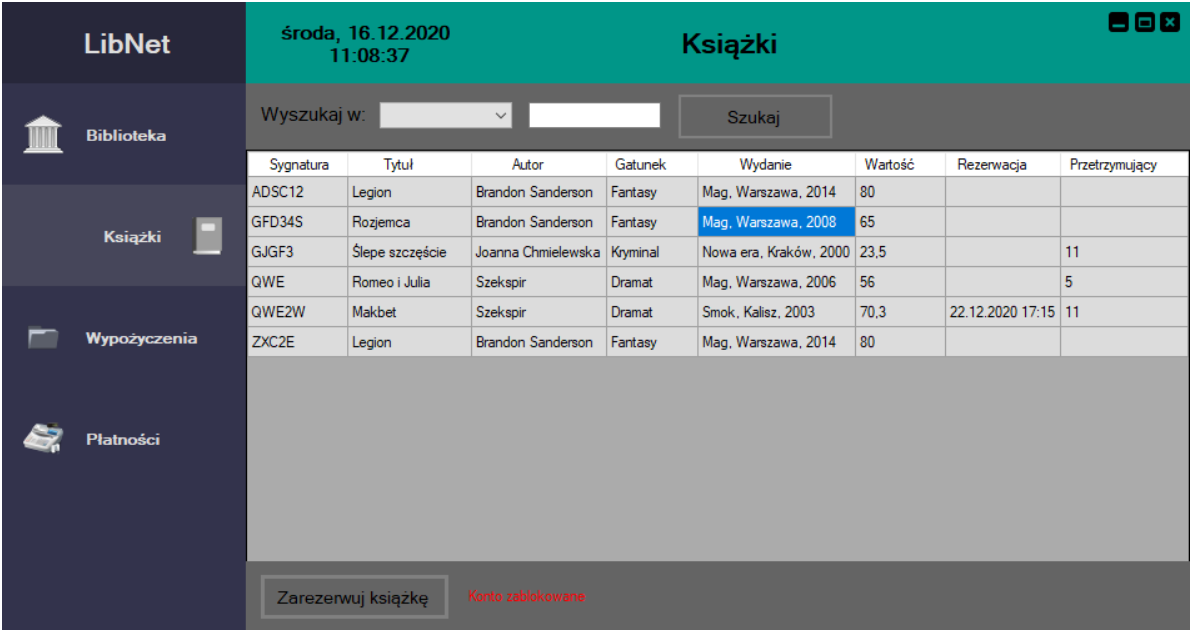
Rys. 22. Okno z zakładką z książkami z poziomu administratora.

Źródło: Opracowanie własne.



Rys. 23. Okno z zakładką z książkami z poziomu czytelnika.

Źródło: Opracowanie własne.

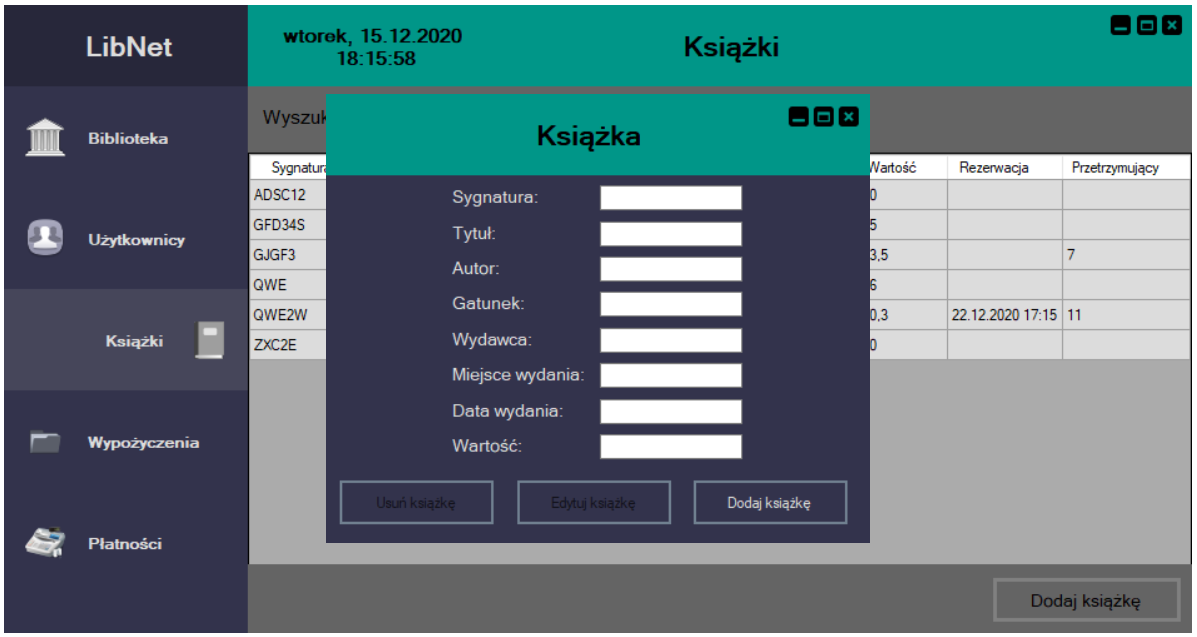


Rys. 24. Okno przy próbie rezerwacji książki na zablokowanym koncie.

Źródło: Opracowanie własne.

Kolejną funkcjonalnością aplikacji jest przeglądanie tabeli książek. Tak jak w poprzedniej zakładce znajdują się przyciski do przeszukiwania tabeli, które działają tak samo jak poprzednie. Administrator w tej zakładce może dodawać książki (egzemplarze), edytować je i usuwać. Do dodawania książki służy przycisk „*Dodaj książkę*”. Jeśli chce usunąć książkę lub edytować ją musi dwukrotnie kliknąć na odpowiednią książkę w tabeli. Czytelnik w tym miejscu może zarezerwować książkę, aby móc później ją pożyczyć bez obawy o to, że ktoś już ją wcześniej wypożyczył. Aby zarezerwować klikamy na odpowiednią książkę w tabeli (zaznaczamy), a potem naciskamy przycisk „*Zarezerwuj książkę*”. Rezerwacja trwa 7 dni. Jeśli konto, z którego próbujemy zarezerwować książkę jest zablokowane, wtedy pojawi się odpowiednia informacja.

### Dodawanie książki



Rys. 25. Okno dodawania książki.

Źródło: Opracowanie własne.

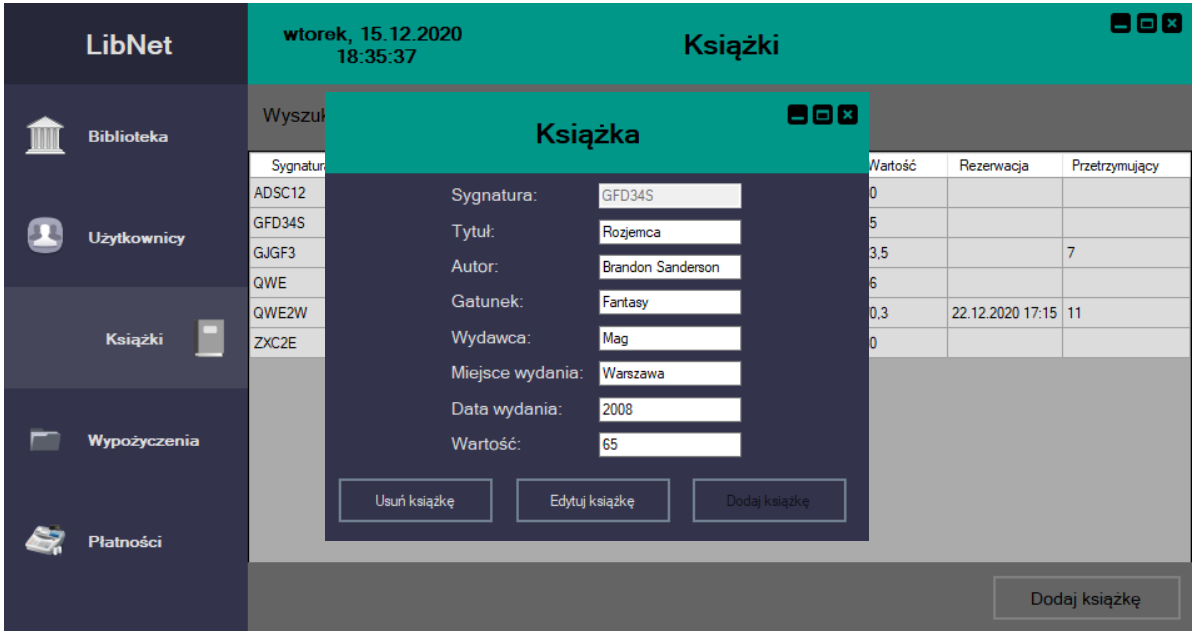


Rys. 26. Okno przy próbie dodania książki z nieprawidłowymi danymi.

Źródło: Opracowanie własne.

Gdy naciśnie się przycisk dodawania książki pojawi się okno (Rys. 25), w którym trzeba podać odpowiednie dane i zatwierdzić dodanie książki. Jeśli dane nie będą prawidłowe tekst obok miejsca do wpisywania danych zmieni kolor na czerwony (Rys. 26). Ogólnie sprawdza to tylko czy dane nie są puste. Tylko w przypadku sygnatury, gatunku i daty wydania trzeba spełnić dodatkowe wymagania. Sygnatura musi być unikatowa, różna od tego co znajduje się w bazie oraz nie może zawierać znaków specjalnych. Gatunek nie może zawierać znaków specjalnych, a data wydania musi być liczbą z zakresu 1000 – 2999. Przy dodawaniu książki, przyciski do edycji i usunięcie książki są wyłączone.

### Edycja książki

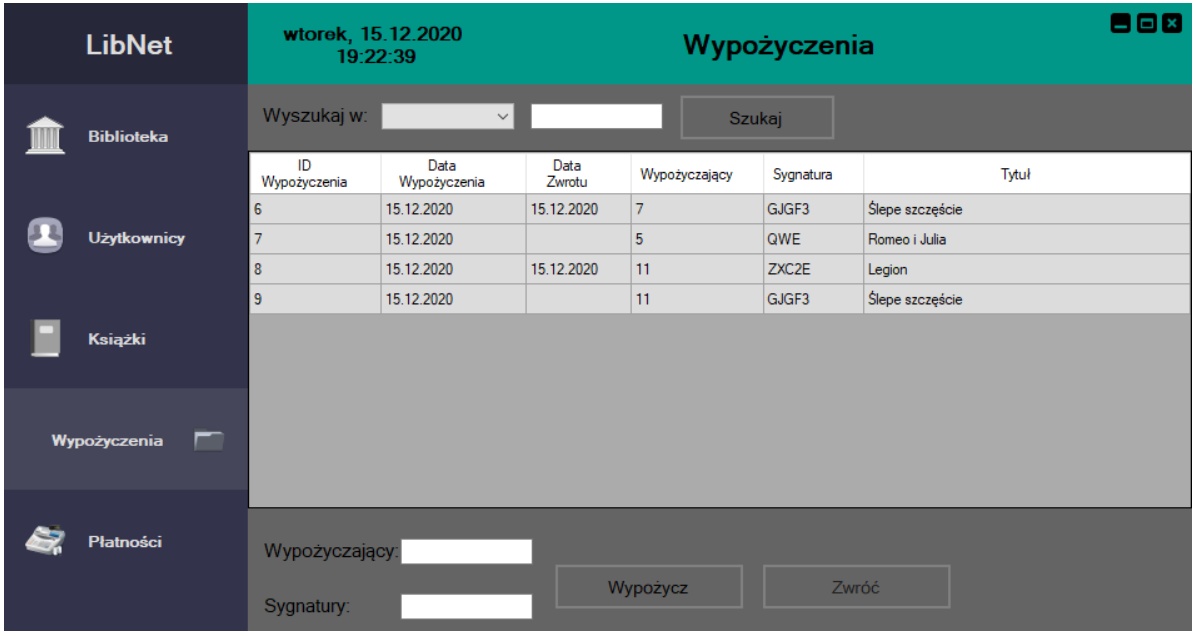


Rys. 27. Okno edycji książki.

Źródło: Opracowanie własne.

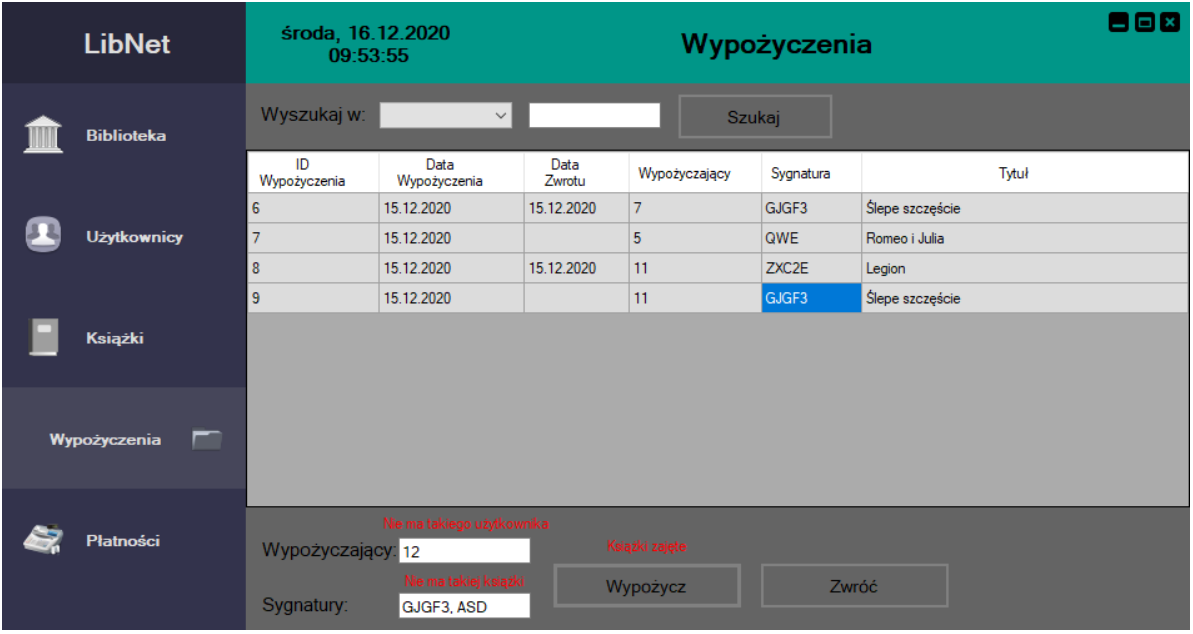
Jeśli chcemy zmienić dane książki bądź ją usunąć musimy kliknąć dwukrotnie na odpowiednią książkę w tabeli i ukaże nam się okno do edycji książki. Wszystkie pola do wypełnienia będą na początku uzupełnione aktualnymi informacjami, a my może je zmieniać. Dane muszą spełniać takie same wymagania jak przy dodawaniu książki, w innym przypadku tekst zmieni kolor na czerwony tak jak przy dodawaniu książki. Przy edycji książki, przycisk do dodawania książki jest wyłączony.

### Zakładka z wypożyczeniami



Rys. 28. Okno z tabelą wypożyczeń z poziomu administratora.

Źródło: Opracowanie własne.



Rys. 29. Okno z tabelą wypożyczeń z poziomu administratora z nieprawidłowymi danymi przy tworzeniu wypożyczenie.

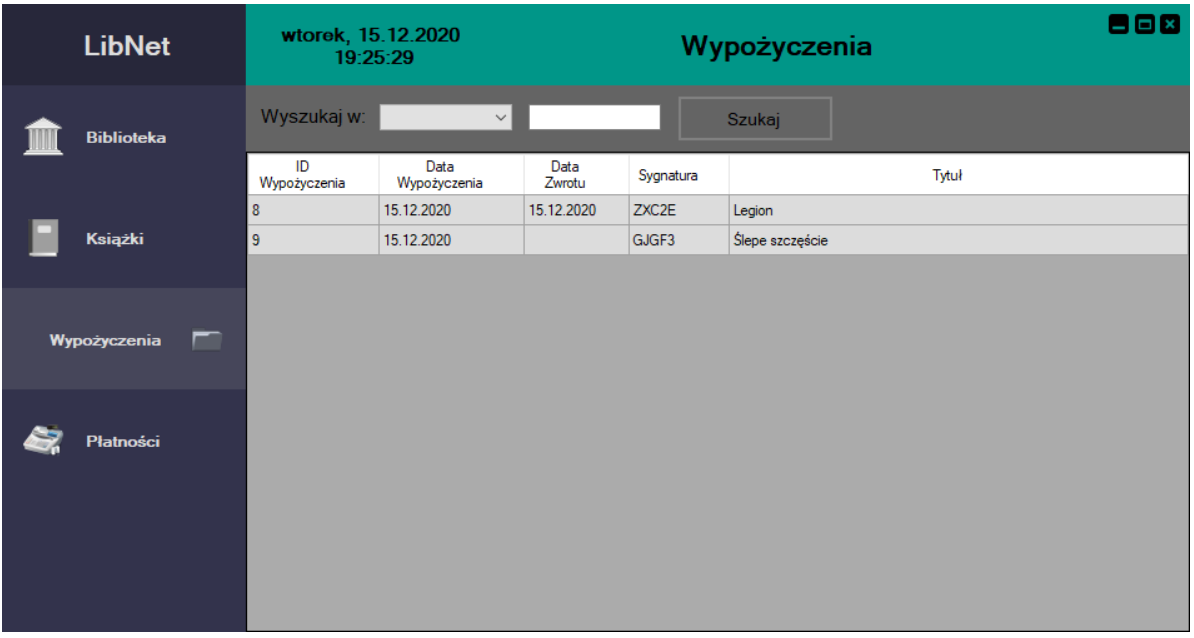
Źródło: Opracowanie własne.



Rys. 30. Okno przy próbie wypożyczenia książki na zablokowane konto.

Źródło: Opracowanie własne.

W zakładce z wypożyczeniami administrator tworzy wypożyczenie i zatwierdza zwroty (Rys. 28). W górnej części jak zwykle znajdują się filtry. Żeby stworzyć wypożyczenie należy podać ID osoby wypożyczającej i sygnatury książek, które chce wypożyczyć. Przy liczbie książek większej niż 1, sygnatury oddzielamy tylko przecinkiem. W razie podanych nieprawidłowych danych, aplikacja o tym ogólnie poinformuje. Jak widać na rysunku 29, książka GJGF3 jest aktualnie wypożyczona, a książka ASD nie istnieje, tak samo jak użytkownika o numerze ID 12. Wypożyczenia na zablokowane konto są niemożliwe (Rys. 30). Kiedy administrator chce zatwierdzić zwrot, klika na odpowiednie wypożyczenie (zaznacza je), wtedy przycisk zostanie włączony i wystarczy go nacisnąć.



Rys. 31. Okno z tabelą wypożyczeń z poziomu czytelnika.

Źródło: Opracowanie własne.

Czytelnik z tej zakładce ma dostęp tylko do historii swoich wypożyczeń, którą może filtrować. Jak można zauważyć na rysunkach 28 i 31, drugi rysunek pokazuje okno dla czytelnika o numerze ID 11. Nie pokazuje temu czytelnikowi wypożyczeń innych użytkowników.

### Zakładka z płatnościami



Rys. 32. Okno z zakładką z płatnościami z poziomu administratora.

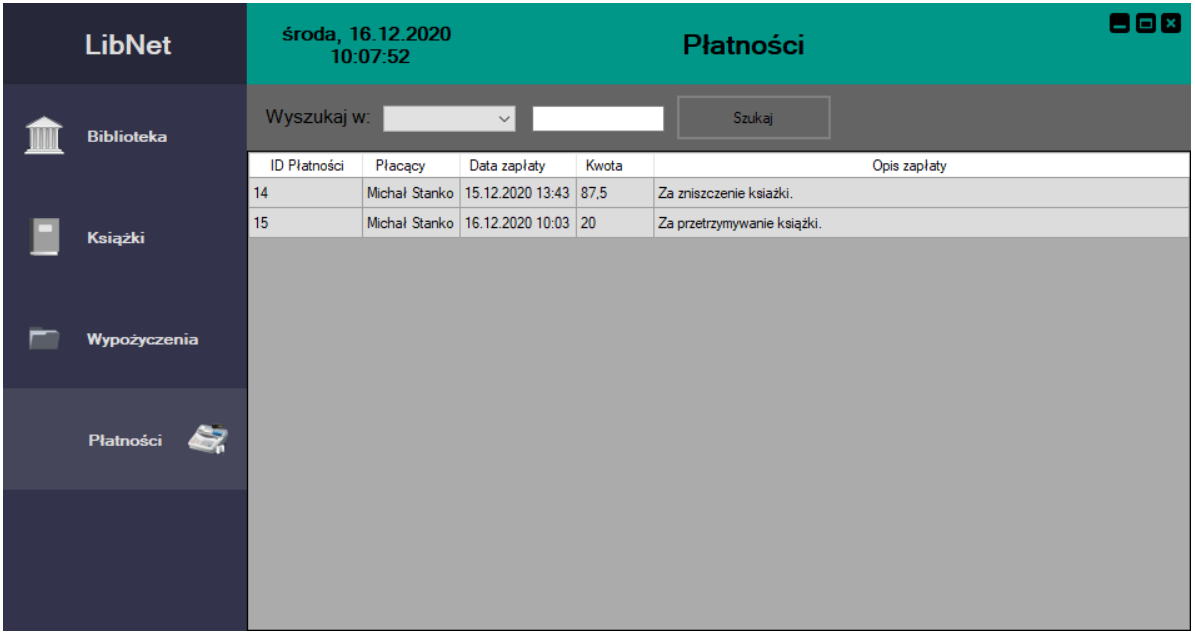
Źródło: Opracowanie własne.



Rys. 33. Okno z zakładką z płatnościami z poziomu administratora z nieprawidłowymi danymi przy tworzeniu płatności.

Źródło: Opracowanie własne.

Ostatnią zakładką w mojej aplikacji jest zakładka z płatnościami. Jak w każdej zakładce w górnej części znajdują się filtry. Administrator może tutaj przeglądać wszystkie płatności oraz tworzyć płatności (Rys. 32). Żeby stworzyć płatność należy podać numer ID osoby płacącej, opcjonalnie opis zapłaty i kwotę zapłaconą. W razie nieprawidłowych danych, aplikacja o tym poinformuje (Rys. 33).



Rys. 34. Okno z zakładką z płatnościami z poziomu czytelnika.

Źródło: Opracowanie własne.

W zakładce z płatnościami czytelnik może przeglądać swoją historię płatności oraz ją filtrować. Nie ma dostępu do płatności innych użytkowników.

# Stworzenie aplikacji

## Omówienie najważniejszych implementacji

## Omówienie problemów napotkanych podczas tworzenia aplikacji

# Podsumowanie

## Ocena jakości i perspektywy rozwoju aplikacji

## Wnioski

# Bibliografia

1. “*Bibliotekarz.NET*”, autorski blog Marka Ryńskiego, <https://bibliotekarz.net/> (dostęp 25.11.2020).
2. “*Rozwiązania dla bibliotek - Prolib*”, strona internetowa z ofertami firmy Sygnity Business Solutions, <https://sygnitysbs.pl/oferta/system-dla-bibliotek-prolib> (dostęp 25.11.2020).
3. Klaus Kreft, Angelika Langer. “*After Java and C# - what is next?*”. <https://www.artima.com/weblogs/viewpost.jsp?thread=6543> (dostęp 19.11.2020).
4. “*Typy ogólne w .NET*”, Dokumentacja platformy .NET firmy Microsoft, <https://docs.microsoft.com/pl-pl/dotnet/standard/generics/> (dostęp 19.11.2020).
5. “*.NET Framework component stack*”, Wikipedia, wolna encyclopedia, <https://en.wikipedia.org/wiki/.NET_Framework> (dostęp 19.11.2020).
6. “*Windows Forms overview*”, Dokumentacja platformy .NET firmy Microsoft, <https://docs.microsoft.com/pl-pl/dotnet/desktop/winforms/windows-forms-overview?view=netframeworkdesktop-4.8> (dostęp 06.12.2020).
7. “*Icon Archive*”, strona internetowa z ikonami, <https://www.iconarchive.com/> (dostęp 06.12.2020).
8. „*GitHub*”, strona internetowa z kodami źródłowymi aplikacji,

Wyrażam zgodę na udostępnienie mojej pracy w czytelniach Biblioteki SGGW   
w tym w Archiwum Prac Dyplomowych SGGW

.................................................................

*(czytelny podpis autora pracy)*