Dokumentacja Techniczna projektu

Programowanie Zaawansowane

Grupa K78

“Projekt aplikacji bibliotecznej LibreX”

Autorzy: Mateusz Pytlos, Kacper Andraszyk

Krótki opis projektu: Projekt będzie miał na celu stworzenie prostej, aczkolwiek w pełni funkcjonalnej aplikacji, do wykorzystania przykładowo w bibliotekach, ma ona być swego rodzaju dziennikiem, który będzie rejestrował użytkowników wypożyczających produkt z danego zbioru.
Chcemy aby nasza aplikacja, dzięki swojej budowie mogła służyć w wielu miejscach, zawrzemy tam między innymi ogólny zbiór prac, które może przeglądać użytkownik, będzie miał on możliwość wyszukania danej książki po jej gatunku oraz autorze, następnie użytkownik będzie mógł daną pracę zarezerwować oraz wypożyczyć. Dzięki utworzonej podstronie będzie mógł on przeglądać książki, które wypożyczył oraz sprawdzić czas pozostały do oddania danej książki.
Aplikacja będzie automatycznie nadawała kary użytkownikom, którzy przekroczą czas wymagany do oddania książki, dodatkowo utworzymy konto z uprawnieniami administratora, które będzie miało dostęp do takich funkcjonalności jak dodanie książki do zbioru, usunięcie jej, ręczne blokowanie książki oraz anulowanie kary nadanej danemu użytkownikowi.

1. Użyte technologie -
    a) **.NET wersja 8.0**
    b) **Entity Framework Core z bazą SQLite w wersji 8.0.11**

2. Instrukcja pierwszego uruchomienia -
a) Sklonuj repozytorium ze strony GitHub w programie VS
b) Przed pierwszym uruchomieniem w konsoli menadżera pakietów użyj komend
   Add-Migration <nazwa migracji>
   Update-Database

c) Uruchom aplikację i przejdź do zakładki logowanie (aby przetestować działanie roli administratora) lub rejestracja (aby przetestować działanie całej aplikacji z perspektywy użytkownika
d) Dane logowania administratora: E-mail - admin@librex.com Hasło - Admin123!

3. Struktura Projektu

a) Sekcja **Areas** - Sekcja odpowiedzialna między innymi za wygląd stron Logowania oraz rejestracji, znajdują się tam tzw. Widoki wymienionych wcześniej zakładek
b) Sekcja **Controllers** - Sekcja zawierająca kontrolery odpowiedzialne za działanie poszczególnych funkcjonalności przykładowo
    - **AdminController** - Funkcje administracyjne
    - **BooksController** - Kontroler odpowiedzialny za rezerwacje książek
    - **LoansController**- Kontroler odpowiedzialny za rezerwacje oraz wypożyczanie książek
    - **ReservationsController**- Kontroler pomocniczy dla identyfikacji użytkownika przy rezerwacjach

c) Sekcja **Data** - Sekcja odpowiedzialna za przechowywanie informacji dotyczących bazy danych, między innymi tworzonych migracji, zawiera również plik odpowiedzialny za połączenie z bazą.
d) Sekcja **Models** - Zawiera klasy reprezentujące dane aplikacji, są to m.in. modele takiej jak:
    - **ApplicationUser** - Model który jest rozwinięciem Identity, pomaga w identyfikacji użytkownika
    - **Book** - Dane książki w bazie
    - **BookReservationViewModel** - Informacje dotyczące książki przy rezerwacji
    - **ErroViewModel** - Obsługa błędów
    - **Loan** - Informacje o wypożyczeniu
    - **LoanViewModel** - Dodatkowe informacje o wypożyczeniu, pomagające rozróżnić je od siebie, stworzone dla umożliwienia dodania nowych funkcjonalności
e) Sekcja **Views** - Odpowiedzialna za przechowywanie widoków dla poszczególnych funkcjonalności takich jak wypożyczenia, rezerwacje, panel administracyjny.

4.Modele użyte w projekcie -

a) **ApplicationUser -** Model, który dodaje dodatkową zmienną dla identyfikacji użytkownika w postaci FullName tzn. Imię i nazwisko, ma on

funkcję podobną do ID użytkownika, umożliwił stworzenie roli administratora

b) **Book** - Model, opisujący książkę w bazie danych, znajdujemy tu takie informacje jak:
   - ID - Id książki
   - Title - Tytuł książki z walidacją w postaci maksymalnej ilości znaków wynoszącej 100
   - Autor - Imię i nazwisko autora z walidacją w postaci maksymalnej ilości znaków wynoszącej 100
   - Genre- Gatunek z walidacją w postaci maksymalnej ilości znaków wynoszącej 50
   - IsBlocked - Informacja czy książka jest zablokowana (bool)
   - IsReturned - Informacja czy książka została zwrócona (bool)
   - IsBorrowed - Informacja czy książka jest wypożyczona (bool)

c) **BookReservationViewModel -** Model wspomagający funkcjonalności rezerwacji:
   - Book - Dodatkowa klasa książki
   - IsReserved - Informacja czy książka jest zarezerwowana (bool)
   - IsReservedByCurrentUser - Informacja czy książka jest zarezerwowana przez zalogowanego użytkownika (bool)
   - IsBorrowed - Dodatkowa klasa informująca czy książka jest wypożyczona (bool)

d) **ErrorViewModel -** Służy do przekazywania informacji o błędach typowych dla stron internetowych, usprawnia działanie widoków

e) **Loan -** Zawiera informacje o wypożyczeniu takie jak:
   - Id - ID wypożyczenia
   - UserId - ID użytkownika
   - User - Użytkownik
   - BookId - ID książki
   - Book - Książka
   - LoanDate - Data wypożyczenia
   - DueDate - Data zwrotu
   - IsOverdue - Funkcja odpowiedzialna za określenie czy książka nie została oddana w terminie (bool)
   - IsReturned - Informuje czy książka została zwrócona
   - PenaltyFee - Wysokość kary (decimal)
   - IsBorrowed - Dodatkowa klasa informująca o wypożyczeniu książki (bool)

f) **LoanViewModel -** W dużej mierze powtórzenie z modelu Loan, zawiera dodatkowo informacje takie jak:

- BookTitle - Tytuł wypożyczonej książki
- BookAuthor - Autor wypożyczonej książki
- BookGenre - Gatunek wypożyczonej książki
- Email - Email użytkownika
- ReservedByUserId - Rezerwacja przez użytkownika z danym ID
- LoanId - ID wypożyczenia

5. Kontrolery użyte w projekcie -

a) Metody w AdminController

Nazwa	Metoda HTTP	Parametr	Opis Działania	Zwracane dane
IndexAsync	GET	Brak	Wyświetla główny widok panelu administracyjnego z nazwą aktualnie zalogowanego użytkownika.	Widok strony głównej panelu (View).
BlockBook	POST	int id	Blokuje książkę o podanym identyfikatorze, ustawiając jej status na IsBlocked = true.	Przekierowanie do widoku książek (RedirectToAction).
CancelPenalty	POST	int loanId	Anuluje karę za wypożyczenie o podanym identyfikatorze, ustawiając wartość PenaltyFee = 0.	Przekierowanie do widoku wypożyczeń (RedirectToAction).
AddBook (GET)	GET	Brak	Wyświetla formularz do dodawania nowej książki.	Widok formularza (View).
AddBook (POST)	POST	Book Model (parametry modelu książki)	Dodaje nową książkę do bazy danych, jeśli model przesłanych danych jest poprawny.	Przekierowanie do widoku zarządzania książkami lub formularz z błędami.
ManageBooks	GET	Brak	Pobiera listę wszystkich książek z bazy danych.	Widok listy książek (View).
ManagePenalties	GET	Brak	Pobiera listę wypożyczeń z przekroczonym terminem zwrotu.	Widok listy wypożyczeń z karami (View).
DeleteBook	POST	int od	Usuwa książkę o podanym identyfikatorze z bazy danych.	Przekierowanie do widoku zarządzania książkami.

b) Metody w BooksController

Nazwa	Metody HTTP	Parametry	Opis działania	Zwracane dane
Index	GET	Brak	Wyświetla listę książek z informacją o dostępności i statusie rezerwacji dla użytkownika.	Widok listy książek (View).
Reserve	POST	int id	Rezerwuje książkę o podanym identyfikatorze dla zalogowanego użytkownika.	Przekierowanie do widoku listy książek (RedirectToAction).
CancelReservation	POST	int id	Anuluje rezerwację książki o podanym identyfikatorze dla zalogowanego użytkownika.	Przekierowanie do widoku listy książek (RedirectToAction).
Create	POST	book book (parametry modelu książki)	Dodaje nową książkę do bazy danych, jeśli dane modelu są poprawne.	Przekierowanie do widoku listy książek (RedirectToAction).

c) Metody w HomeController

Nazwa	Metody HTTP	Parametry	Opis działania	Zwracane dane
Index	GET	Brak	Wyświetla stronę główną aplikacji.	Widok strony głównej (View).
Privacy	GET	Brak	Wyświetla stronę z polityką prywatności.	Widok strony prywatności (View).
Error	GET	Brak	Wyświetla stronę błędu z informacjami o identyfikatorze żądania.	aWidok strony błędu (View) z modelem ErrorViewModel.

d) Metody w LoansController

Nazwa	Metody HTTP	Parametry	Opis działania	Zwracane dane
Index	GET	Brak	Pobiera listę aktywnych wypożyczeń książek. Administrator widzi wszystkie, użytkownik tylko swoje.	Widok z listą wypożyczeń (LoanViewModel).
Reserve	POST	int bookId	Rezerwuje książkę dla użytkownika. Sprawdza dostępność książki i	Przekierowanie z informacją o sukcesie/błędzie.

			unikalność rezerwacji.	
CancelReservation	POST	int loanId	Anuluje rezerwację użytkownika. Usuwa blokadę na książce i rezerwację w bazie.	Przekierowanie z informacją o sukcesie/błędzie.
Borrow (GET)	GET	Brak	Pobiera listę dostępnych książek do wypożyczenia i użytkowników.	Widok formularza do wypożyczenia (BorrowFormViewModel).
Borrow (POST)	POST	int bookId, string userId	Wypożycza książkę dla użytkownika. Tworzy nowe wypożyczenie i aktualizuje status książki.	Przekierowanie z informacją o sukcesie/błędzie.
OverdueLoans	GET	Brak	Pobiera listę wypożyczeń, które są przeterminowane.	Widok z listą przeterminowanych wypożyczeń.
ReturnBook	POST	int id	Oznacza wypożyczoną książkę jako zwróconą. Aktualizuje status książki i wypożyczenia w bazie.	Przekierowanie z informacją o sukcesie/błędzie

e) Metody w ReservationsController

Nazwa	Metody HTTP	Parametry	Opis działania	Zwracane dane
Index	GET	Brak	Pobiera listę rezerwacji dla zalogowanego użytkownika. Wykorzystuje nazwę użytkownika jako filtr.	Widok z listą rezerwacji (Loan).

6. Użytkownicy i Administratorzy

Użytkownicy mają podstawowe funkcjonalności takie jak:

- Stworzenie konta
- Zmiana hasła
- Wyświetlanie książek
- Rezerwowanie książek
- Anulowanie rezerwacji
- Wypożyczanie książek

- Sprawdzanie daty wypożyczenia oraz daty zwrotu książek
- Sprawdzanie nałożonych kar

Administratorzy mają takie same funkcjonalności oraz dodatkowo:

- Dodawanie książek
- Edycja książek
- Usuwanie książek
- Zarządzanie karami

7. Ciekawe funkcjonalności zawarte w naszym projekcie:
   - Automatyczne nakładanie kar w przypadku przekroczenia terminu zwrotu
   - Możliwość rezerwacji książki, przez użytkownika, aczkolwiek ostateczną decyzję co do wypożyczenia podejmuje administrator
   - Administrator może wypożyczyć książkę tylko użytkownikowi który ją zarezerwował, dopiero po anulowaniu rezerwacji przez użytkownika, może wypożyczyć ją innej osobie
   - Administrator potwierdza fakt oddania książki w panelu administracyjnym, dzięki czemu użytkownik nie może oszukać systemu, fałszywie oznaczając książkę jako zwróconą.