Imiona, nazwiska: Michał Michalski (k78), Marek Marcinkowski (k78), Bartosz Sikorski (k79)

Dokumentacja Techniczna Projektu

MyBookLife

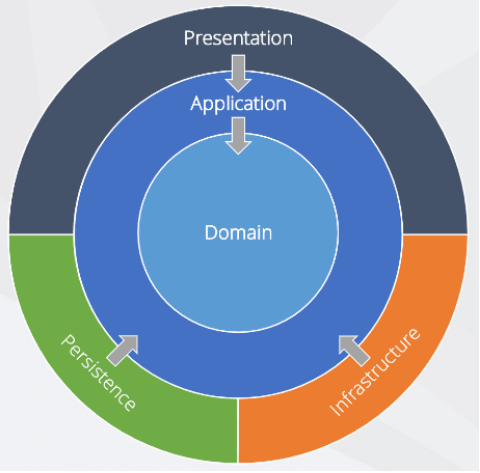
1. Wprowadzenie

Aplikacja MyBookLife jest aplikacją webową stworzoną w technologii .NET 6 w architekturze klient-serwer, z wykorzystaniem wzorca projektowego MVC, clean architecture. Celem aplikacji jest umożliwienie użytkownikom śledzenia swojego postępu czytania książek oraz zarządzania informacjami na temat przeczytanych pozycji.

2. Architektura

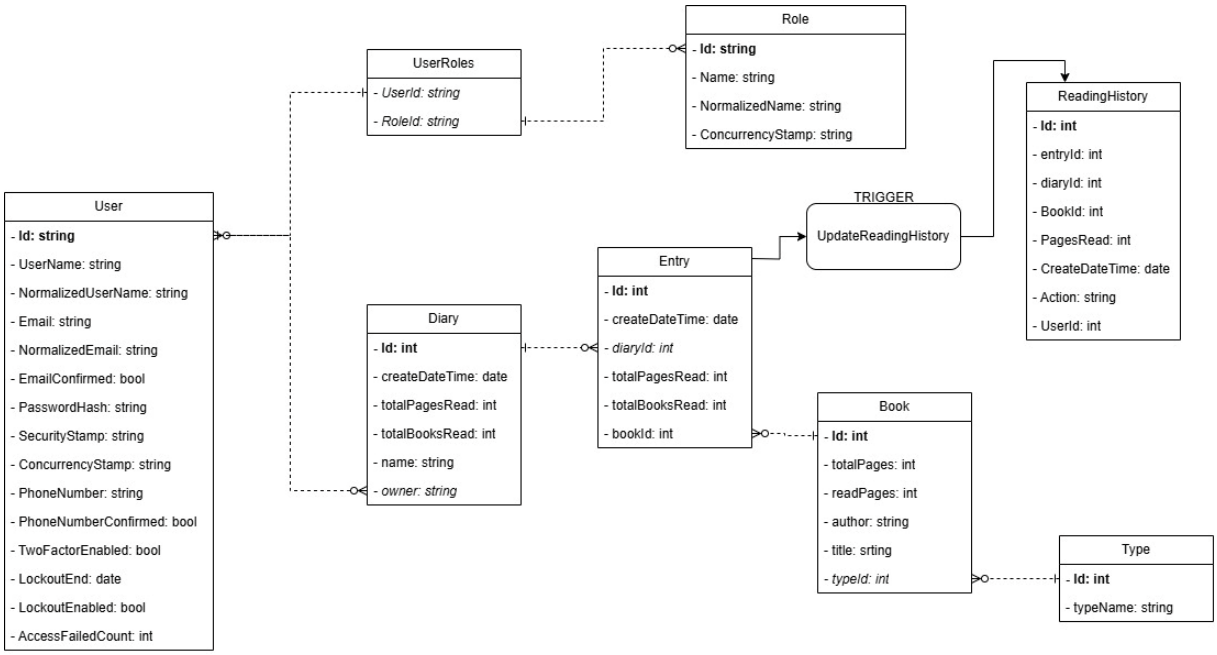
Projekt MyBookLife został zbudowany zgodnie z zasadami clean architecture (onion architecture), co oznacza podział na warstwy: Presentation, Application, Domain oraz Infrastructure.

- *Domain Layer* (MyBookLife.Domain): Reprezentuje podstawowe encje oraz logikę biznesową. W tej warstwie znajdują się modele encji i interfejsy repozytoriów.
- *Application Layer* (MyBookLife.Application): Zawiera logikę biznesową aplikacji. W tej warstwie znajdują się serwisy aplikacyjne obsługujące operacje na książkach, postępach czytania, oraz autentykacji.
- *Presentation Layer* (MyBookLife.Web): Odpowiada za obsługę interfejsu użytkownika i prezentację danych. W tej warstwie znajduje się projekt MVC, zawierający kontrolery, widoki oraz modele.
- *Infrastructure Layer* (MyBookLife.Infrastructure): Odpowiada za implementację dostępu do danych, logikę dostępu do zewnętrznych źródeł danych, jak baza danych czy systemy autentykacji.



3. Baza Danych

- Do przechowywania danych aplikacji MyBookLife wykorzystuje bazę danych SQL (SQL Server Managment Studio 19). W projekcie zastosowano Entity Framework Core jako ORM (Object Relation Mapping) dla łatwej obsługi bazy danych.
- Został utworzony trigger o nazwie "UpdateReadingHistory" na tabeli "Entries", który automatycznie uruchamia się po operacjach INSERT i UPDATE. Trigger ten zapisuje informacje o zmianach w historii czytania do tabeli „ReadingHistory”, rejestrując szczegóły dotyczące wpisu, książki, użytkownika oraz rodzaju wykonanej operacji.
- Diagram UML dla bazy danych aplikacji:



4. Funkcjonalności Aplikacji

- Dodawanie nowego pamiętnika: Użytkownik ma możliwość dodawania nowego pamiętnika, przypisując mu unikalną nazwę.
- Dodawanie wpisów do pamiętnika: Użytkownik może dodawać wpisy z postępami w czytaniu, zawierające informacje takie jak książka, liczba przeczytanych stron, data wpisu itp.
- Przegląd pamiętników: Użytkownik ma dostęp do listy swoich pamiętników, z możliwością szybkiego przełączania między nimi.
- Dodawanie Książek: Użytkownik może dodawać książki, podając takie informacje jak tytuł, kategoria, autor itp. Dane te są przechowywane w bazie danych.
- Zapisywanie Postępów: Aplikacja umożliwia użytkownikowi zapisywanie postępów w czytaniu, takich jak liczba przeczytanych stron danego dnia w ramach konkretnej książki.
- Usuwanie Wpisów: Użytkownik ma możliwość usuwania wpisów dotyczących książek oraz zapisanych postępów czytania.
- Aktualizacja Wpisów: Funkcjonalność pozwalająca na aktualizację informacji o książkach oraz postępach czytania.
- Widok Postępów: Aplikacja oferuje użytkownikowi widoki prezentujące ogólny postęp czytania, ilość przeczytanych książek, łączną liczbę przeczytanych stron itp.

- Widok ciekawych statystyk opisujących wszytskie postępy, książki, wpisy i inne informacje związane z całą aktywnością użytkownika na stronie.
- Rejestracja i Logowanie: Aplikacja wymaga, aby użytkownicy mieli konta. Dostęp do funkcji wymaga autentykacji.
- Dla administratora: Przegląd listy wszystkich użytkowników

5. Bezpieczeństwo

Aby zapewnić bezpieczeństwo użytkowników, aplikacja MyBookLife wykorzystuje mechanizmy uwierzytelniania i autoryzacji. Hasła są przechowywane w bezpieczny sposób (hashowanie).

6. Inne Technologie i Biblioteki

- Entity Framework Core: Do obsługi dostępu do bazy danych.
- Identity Framework: Do obsługi procesu rejestracji i logowania użytkowników.
- Automapper: Biblioteka zewnętrzna do mapowania obiektów pomiędzy warstwami aplikacji.

7. Podsumowanie

Projekt MyBookLife to kompleksowa aplikacja webowa stworzona w technologii .NET, zaprojektowana z myślą o miłośnikach książek, którzy pragną śledzić swój postęp czytelniczy. Dzięki zastosowaniu zasad clean architecture, aplikacja charakteryzuje się wysoką łatwością utrzymania, skalowalnością oraz przejrzystą organizacją kodu.

Kluczowe cechy projektu:

Funkcjonalność

- Śledzenie postępu czytania książek
- Zarządzanie osobistą biblioteką
- Możliwość dodawania recenzji i ocen przeczytanych pozycji

Architektura

- Wykorzystanie clean architecture
- Podział na warstwy: prezentacji, logiki biznesowej i dostępu do danych
- Łatwa rozszerzalność i modyfikacja poszczególnych komponentów

Obszary do rozwinięcia:

Mimo solidnych podstaw, projekt MyBookLife ma potencjał do dalszego rozwoju. Jednym z kluczowych elementów, który nie został jeszcze zaimplementowany, jest system potwierdzania konta użytkownika za pomocą adresu e-mail. Wdrożenie tej funkcji znacząco zwiększyłoby bezpieczeństwo aplikacji i wiarygodność zarejestrowanych użytkowników.

Inne potencjalnie możliwe rozszerzenia:

- Integracja z zewnętrznymi bazami książek (np. Google Books API)
- Funkcje społecznościowe, takie jak dzielenie się recenzjami czy tworzenie grup czytelniczych
- Personalizowane rekomendacje książek na podstawie historii czytania

Projekt MyBookLife, dzięki swojej solidnej architekturze, stanowi doskonałą bazę do dalszego rozwoju i dodawania nowych funkcjonalności, które mogą znacząco wzbogacić doświadczenie użytkowników pasjonujących się literaturą.

Projekt dostępny w github pod adresem: https://github.com/michalmichalskii/MyBookLife

Instrukcja do uruchomienia projektu

Instrukcja do uruchomienia projektu

Pobieranie Projektu z GitHuba:

1. Otwórz przeglądarkę internetową i przejdź do strony projektu na GitHubie: MyBookLife GitHub Repository.
2. Skorzystaj z przycisku "Code" i wybierz opcję "Download ZIP" lub skopiuj link do repozytorium i użyj go w terminalu/git bash do sklonowania repozytorium na swoim komputerze.

Pobieranie ASP.NET w wersji 6.0:

1. Przejdź na oficjalną stronę ASP.NET: ASP.NET.
2. Znajdź i pobierz najnowszą wersję ASP.NET 6.0, zgodną z systemem operacyjnym Twojego komputera.
3. Postępuj zgodnie z instrukcjami instalacyjnymi dostępnymi na stronie, aby zainstalować ASP.NET w wersji 6.0.

XAMPP:

1. Pobierz aplikację XAMPP na swój komputer i włącz moduł Apache

Włączanie Projektu za pomocą Visual Studio 2022:

1. Otwórz Visual Studio 2022 na swoim komputerze.
2. Wybierz opcję "Open a project or solution" z ekranu startowego Visual Studio.
3. Przejdź do katalogu, w którym został pobrany projekt z GitHuba, i otwórz plik projektu .sln.

Dostosowanie połączenia z bazą danych do Swojego Komputera:

1. Zainstaluj aplikację SqlSerwer ze strony https://www.microsoft.com/pl-pl/sql-server/sql-server-downloads
2. Zainstaluj aplikację SQL Server Management Studio ze strony https://learn.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver16
3. Utwórz baze danych pod nazwą "BookLifeDb"

4. W projekcie MyBookLife otwórz plik appsettings.json znajdujący się w projekcie MyBookLife.Web.
5. Znajdź sekcję dotyczącą połączenia do bazy danych (DefaultConnection).
6. Dostosuj ustawienia połączenia: "DefaultConnection": "Server=SERVER_NAME(tutaj_zmien);Database=BookLifeDb;Trusted_Connection=True;"
7. Upewnij się, że baza danych o nazwie podanej w konfiguracji istnieje na Twoim serwerze bazy danych.
8. Wykonaj migrację danych

Dostosowywanie triggera:

1. W bazie danych (SQL Server Managment Studio) utwórz tabelę za pomocą nowego query z kodem:

```
USE [BookLifeDb]
GO

/****** Object:  Table [dbo].[ReadingHistory]    Script Date: 19/12/2024 10:23:32 ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[ReadingHistory](
   [Id] [int] IDENTITY(1,1) NOT NULL,
   [EntryId] [int] NOT NULL,
   [DiaryId] [int] NOT NULL,
   [BookId] [int] NOT NULL,
   [PagesRead] [int] NOT NULL,
   [CreateDateTime] [datetime2](7) NOT NULL,
   [Action] [varchar](10) NOT NULL,
   [UserId] [nvarchar](450) NULL,
PRIMARY KEY CLUSTERED
(
   [Id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON,
OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
```

2. W nowym query utwórz trigger za pomocą poniższego kodu:

```
USE [BookLifeDb]
```

```
GO

/****** Object:  Trigger [dbo].[UpdateReadingHistory]    Script Date: 19/12/2024 10:24:15 ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TRIGGER [dbo].[UpdateReadingHistory]
ON [dbo].[Entries]
AFTER INSERT, UPDATE
AS
BEGIN
        --wyłącza wysyłanie komunikatów o liczbie zmodyfikowanych wierszy
  SET NOCOUNT ON;

  INSERT INTO ReadingHistory (EntryId, DiaryId, BookId, UserId, PagesRead, CreateDateTime, Action)
        --Te linie wybierają dane do wstawienia. i odnosi się do tabeli inserted, która zawiera nowe lub zaktualizowane wiersze.
  SELECT
    i.Id,
    i.DiaryId,
    i.BookId,
    u.Id,
    i.PagesRead,
    i.CreateDateTime,
            --CASE określa, czy operacja to INSERT (gdy d.Id jest NULL) czy UPDATE.
    CASE
      WHEN d.Id IS NULL THEN 'INSERT'
      ELSE 'UPDATE'
    END
        --Te linie łączą tabelę inserted z tabelą deleted. Dla operacji INSERT, deleted będzie pusta.
  FROM
    inserted i
  LEFT JOIN
    deleted d ON i.Id = d.Id
        --To złączenie pobiera Id użytkownika z tabeli AspNetUsers, łącząc pole Owner z Entries z UserName z AspNetUsers.
  INNER JOIN
    [dbo].[AspNetUsers] u ON i.Owner = u.UserName;
END;
GO
```

ALTER TABLE [dbo].[Entries] ENABLE TRIGGER [UpdateReadingHistory]

GO

Logowanie przez Google:

1. Wejdź na stronę https://console.cloud.google.com/projectcreate i utwórz nowy projekt o dowolnej nazwie
2. Po utworzeniu wejdź w zakładkę "Ekran zgody OAuth"
3. Wybierz "Z zewnątrz" i naciśnij przycisk utwórz
4. Wypełnij formularz podając najlepiej nazwę aplikacji "MyBookLife", podając swój adres email w sekcji: "Informacje o aplikacji" oraz "Dane kontaktowe dewelopera". Następnie naciśnij "Zapisz i Kontynuuj"
5. W zakładce "Zakresy" nic nie zmienaj, naciśnij "Zapisz i Kontynuuj"
6. W zakładce "Użytkownicy testowi" nic nie zmienaj, naciśnij "Zapisz i Kontynuuj"
7. W sekcji "Podsumowanie" sprawdź dane i przejdź do panelu wybierając przycisk "Powrót do panelu"
8. W bocznym menu wybierz "Dane logowania", następnie utwórz dane logowania za pomocą przycisku "Utwórz dane logowania"->"Identyflkator klienta OAuth"
9. Wybierz typ aplikacji: "Aplikacja inetrnetowa", podaj dowolną nazwę i naciśnij przycisk "Utwórz"
10. W Visual Studio włącz aplikację i skopiuj adres strony
11. Do pola "Autoryzowane identyfikatory URI przekierowania"->"Dodaj URI" wklej adres strony + "/signin-google" (Na przykład będzie to "https://localhost:7101/signin-google")
12. Naciśnij przycisk "Utwórz"
13. Skopiuj "Identyfikator klienta"
14. W Visual Studio otówrz terminal, przejdź do folderu "MyBookLife.Web" i wpisz "dotnet user-secrets set "Authentication:Google:ClientId" "twój skopiowany klucz"", naciśnij enter
15. z przeglądarki skopiuj "Tajny klucz klienta"
16. W Visual Studio otówrz terminal, przejdź do folderu "MyBookLife.Web" i wpisz "dotnet user-secrets set "Authentication:Google:ClientSecret" "twój skopiowany tajny klucz"", naciśnij enter
17. Odkomentuj poniższy fragment kodu z pliku "Program.cs":

    //builder.Services.AddAuthentication().AddGoogle(googleOptions =>

    //{

```
//    googleOptions.ClientId = configuration["Authentication:Google:ClientId"];
//    googleOptions.ClientSecret = configuration["Authentication:Google:ClientSecret"];
//});
```

Uruchamianie Projektu:

1. Wybierz projekt MyBookLife.Web jako główny projekt w Visual Studio.
2. Naciśnij przycisk "Start" lub skorzystaj z klawisza F5, aby uruchomić aplikację.
3. Otwórz przeglądarkę internetową i przejdź do adresu https://localhost:PORT, gdzie PORT to numer portu, na którym została uruchomiona aplikacja