Wojciech Dreslerski

Jakub Oksiak

Expense tracker – dokumentacja

**Opis działania aplikacji ExpenseTracker**

Aplikacja **ExpenseTracker** to narzędzie umożliwiające zarządzanie wydatkami w ramach rodziny. Aplikacja oferuje funkcjonalność logowania, zarządzania rolami użytkowników oraz monitorowania wydatków.

**Logowanie i role użytkowników:**

Po uruchomieniu aplikacji każdy użytkownik może się zalogować. Domyślnie, po instalacji aplikacji, tworzony jest użytkownik z rolą **admin**, którego dane logowania można znaleźć w pliku README. Administrator posiada pełną kontrolę nad uprawnieniami innych użytkowników.

**Role użytkowników:**

Aplikacja obsługuje dwie podstawowe role użytkowników:

1. **Admin** – ma pełną kontrolę nad zarządzaniem rolami.
2. **Głowa rodziny** – użytkownik przypisany do tej roli może dodawać nowe wydatki do wspólnego konta rodziny, a także przeglądać oraz analizować wszystkie wydatki związane z całą rodziną. Głowa rodziny ma pełny wgląd w budżet rodzinny.
3. **Członek rodziny** – użytkownik przypisany do tej roli może dodawać swoje wydatki na wspólne konto rodziny, jednak nie ma uprawnień do przeglądania wydatków innych członków rodzin. Może jedynie śledzić i analizować swoje własne wydatki.

**Funkcjonalności:**

* **Logowanie** – użytkownicy mogą się logować do aplikacji, wykorzystując dane logowania.
* **Zarządzanie użytkownikami** – admin może przypisać odpowiednie role do zarejestrowanych użytkowników (głowa rodziny lub członek rodziny).
* **Dodawanie wydatków** – głowa rodziny oraz członkowie rodziny mogą dodawać wydatki, które będą przypisane do wspólnego budżetu rodziny.
* **Analiza wydatków** – użytkownicy w zależności od przypisanej roli mają dostęp do funkcji analizy wydatków, z możliwością przeglądania danych o swoich osobistych wydatkach lub wszystkich wydatkach rodziny.

**Użyte technologie:**

* .NET 8
* ASP.NET
* Entity Framework
* Blazor
* Microsoft SQL Server

**Jak uruchomić aplikacje:**

* Otwórz plik ExpenseTracker.sln.
* Wpisz komendę Update-Database w konsoli menedżera pakietów.

**Struktura projektu**

* **Controllers** – Zawiera kontrolery, które odpowiadają za logikę aplikacji, zarządzając żądaniami i odpowiedziami użytkownika oraz komunikacją z danymi.
* **Models** – Przechowuje klasy reprezentujące dane aplikacji, które są wykorzystywane do operacji na bazie danych oraz do przesyłania danych między warstwą logiki a widokami.
* **Views** – Zawiera pliki widoków, które odpowiadają za wygląd i prezentację strony internetowej, generując dynamiczny HTML, który jest wyświetlany użytkownikowi.
* **Views/Shared** – Przechowuje wspólne elementy widoków, takie jak nagłówki, stopki, lub inne fragmenty, które są używane w różnych widokach aplikacji.
* **wwwroot** – Zawiera pliki statyczne, takie jak pliki CSS, JavaScript, obrazy oraz inne zasoby, które są bezpośrednio dostępne dla użytkownika w przeglądarce.
* **Program.cs** – Plik konfiguracyjny aplikacji, w którym definiowane są serwisy, middleware oraz ustawienia związane z uruchomieniem aplikacji.

**Modele w projekcie**

Model **Expense** reprezentuje wydatki w systemie. Przechowuje informacje o poszczególnych transakcjach finansowych użytkownika. Zawiera następujące właściwości:

* + **Id** – Unikalny identyfikator wydatku (klucz główny).
  + **Name** – Nazwa wydatku, wymagana, o długości maksymalnej 100 znaków.
  + **Amount** – Kwota wydatku, wymagany, musi być większa niż 0.
  + **Date** – Data wydatku, domyślnie ustawiona na bieżącą datę.
  + **UserId** – Identyfikator użytkownika, do którego należy wydatek. Jest kluczem obcym odnoszącym się do tabeli użytkowników.
  + **User** – Odwołanie do użytkownika, który dodał wydatek (relacja z **IdentityUser**).

Model **UserDto** to obiekt DTO, który przechowuje podstawowe informacje o użytkownikach. Jest wykorzystywany głównie w celu przesyłania danych użytkowników między warstwą aplikacji a interfejsem użytkownika.

Zawiera następujące właściwości:

* **UserId** – Unikalny identyfikator użytkownika.
* **UserName** – Nazwa użytkownika (zwykle odpowiada za nazwę wyświetlaną w aplikacji).
* **Role** – Rola przypisana do użytkownika (np. admin, głowa rodziny, członek rodziny).

**1. ExpenseController**

* **Index**
  + **Opis**: Wyświetla listę wydatków użytkownika lub całej rodziny. Umożliwia filtrowanie wyników według nazwy, daty początkowej i końcowej, oraz paginację wyników.
  + **Parametry**:
    - searchTerm (string) – fraza do wyszukiwania w nazwach wydatków (opcjonalny).
    - startDate (DateTime?) – data początkowa do filtrowania wydatków (opcjonalna).
    - endDate (DateTime?) – data końcowa do filtrowania wydatków (opcjonalna).
    - page (int) – numer strony do paginacji (domyślnie 1).
    - pageSize (int) – liczba wyników na stronę (domyślnie 10).
  + **Zwraca**: Widok z listą wydatków.
* **Create**
  + **Opis**: Renderuje widok do tworzenia nowego wydatku.
  + **Parametry**: Brak.
  + **Zwraca**: Widok do tworzenia nowego wydatku.
* **AddExpense**
  + **Opis**: Dodaje nowy wydatek do bazy danych, przypisując go do zalogowanego użytkownika.
  + **Parametry**:
    - expense (Expense) – obiekt zawierający dane nowego wydatku.
  + **Zwraca**: Przekierowanie do akcji **Index** (lista wydatków).

**2. HomeController**

* **Index**
  + **Opis**: Wyświetla stronę główną aplikacji.
  + **Parametry**: Brak.
  + **Zwraca**: Widok strony głównej.

**3. StatsController**

* **Index**
  + **Opis**: Wyświetla statystyki wydatków z podziałem na miesiące dla wybranego roku. Umożliwia przeglądanie wydatków z różnych lat.
  + **Parametry**:
    - year (int) – rok, dla którego mają być wyświetlone statystyki (domyślnie bieżący rok, jeśli wartość to -1).
  + **Zwraca**: Widok z danymi statystyk wydatków.

**4. UserController**

* **Index**
  + **Opis**: Wyświetla listę wszystkich użytkowników w systemie z przypisaną rolą.
  + **Parametry**: Brak.
  + **Zwraca**: Widok z listą użytkowników i przypisanymi rolami.
* **AssignRole (GET)**
  + **Opis**: Wyświetla formularz przypisania roli dla wybranego użytkownika.
  + **Parametry**:
    - userId (string) – identyfikator użytkownika, którego rolę chcemy przypisać.
  + **Zwraca**: Widok z formularzem przypisania roli.
* **AssignRole (POST)**
  + **Opis**: Przypisuje rolę do użytkownika, aktualizując jego rolę w systemie.
  + **Parametry**:
    - model (UserDto) – obiekt zawierający dane użytkownika i rolę, którą chcemy przypisać.
  + **Zwraca**: Przekierowanie do akcji **Index** (lista użytkowników).