

Dokumentacja

Expenses Splitter

Autorzy: Jakub Cisło i Filip Czyż

Spis Treści

- 1.Opis programu
- 2.Zakres Funkcjonalności
- 3.Diagramy
- 4.Wymogi Techniczne
- 5.Technologie użyte do implementacji
- 6.Testy
- 7.Podział pracy

1.Opis programu

Expenses Splitter to aplikacja umożliwiająca rozliczenia grupowych wydatków. Użytkownicy mogą dodawać wydatki, za które płacili oraz oznaczać, kto ma oddać pieniądze za poniesione koszty. Aplikacja sama rozliczy, kto komu powinien przekazać pieniądze, aby wyrównać rachunki oraz zminimalizuje liczbę potrzebnych transakcji.

Aplikacja będzie wykonana w technologii webowej. Frontend zostanie wykonany z wykorzystaniem frameworka Angular, natomiast backend w języku C#.

Z programu można korzystać pod adresem <http://expenses.programowaniezpasja.pl>.

Aplikacja jest inspirowana istniejącymi rozwiązaniami:

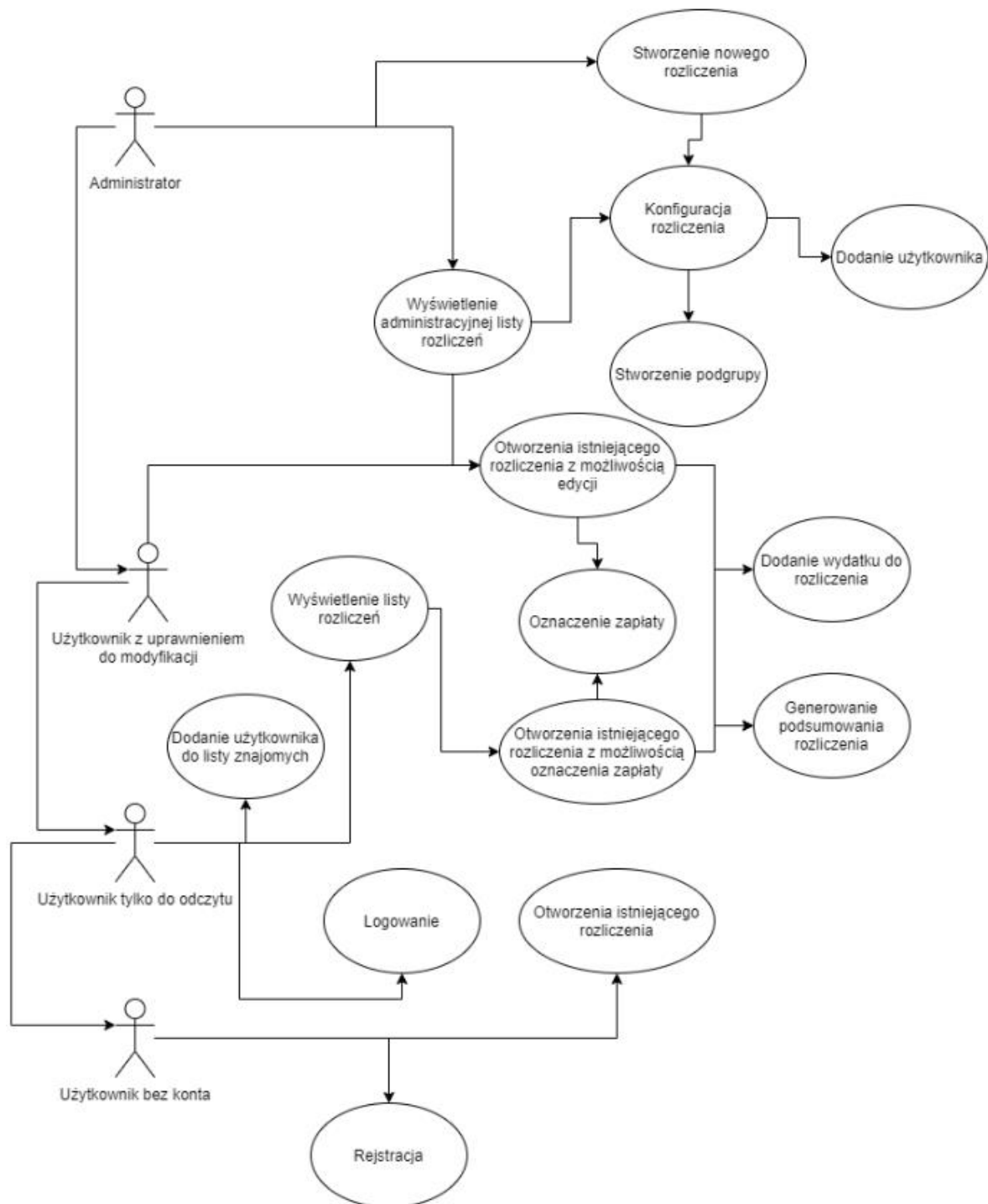
- <https://www.tricount.com>
- <https://www.kittysplit.com>

2.Zakres Funkcjonalności

- Stworzenie nowego rozliczenia
- Użytkownicy mają dostęp do rozliczenia za pomocą linku
 - Każdy może coś zmienić w rozliczeniu
- Konfiguracja osób w rozliczeniu
 - Liczba osób i imiona/nicki
 - Podgrupy w ramach głównej grupy (np. podróżowanie w 2 samochodach)
- Dodawanie wydatków do rozliczenia
 - Równomierny podział pomiędzy wszystkich w grupie
 - Równomierny podział pomiędzy zaznaczone osoby
- Opłacanie wydatku w rozliczeniu
 - Oznaczenie, że wydatek został zapłacony
- Widok podsumowania rozliczenia
 - Kto zapłacił
 - Kto ma jaki dług i względem kogo
- Logowanie do aplikacji
 - Lista wszystkich rozliczeń dla zalogowanego użytkownika
 - Role/uprawnienia użytkowników
 - Administrator oraz użytkownicy z uprawnieniem tylko do odczytu
 - Użytkownicy z uprawnieniami do oznaczenia uregulowania długu
 - Użytkownicy z uprawnieniami do modyfikacji wydatków

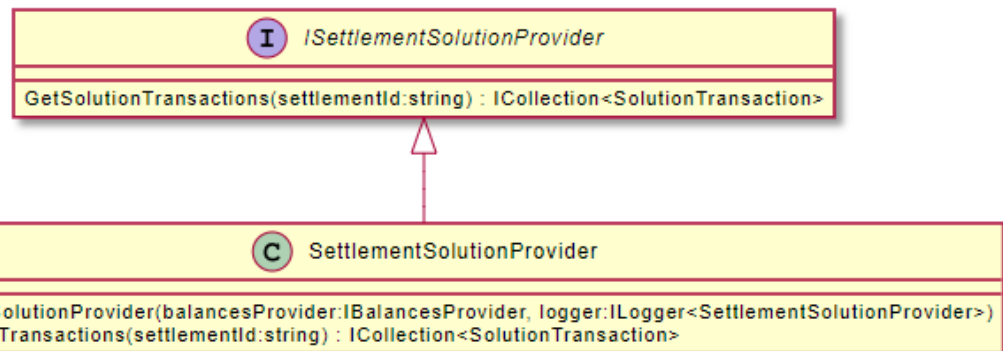
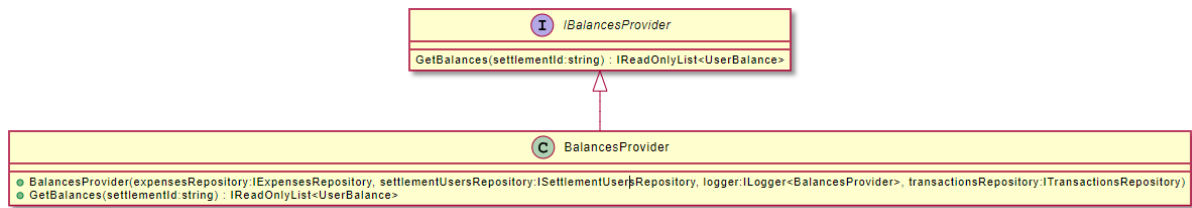
3.Diagramy

Diagram przypadków użycia

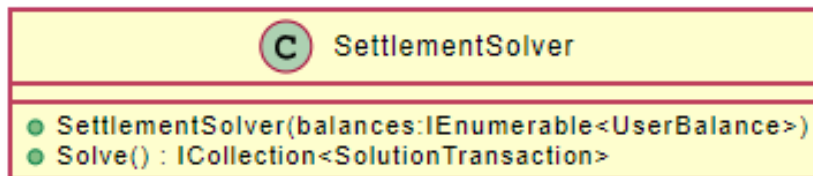


Diagramy klas automatycznie wygenerowane przez Visual Studio:

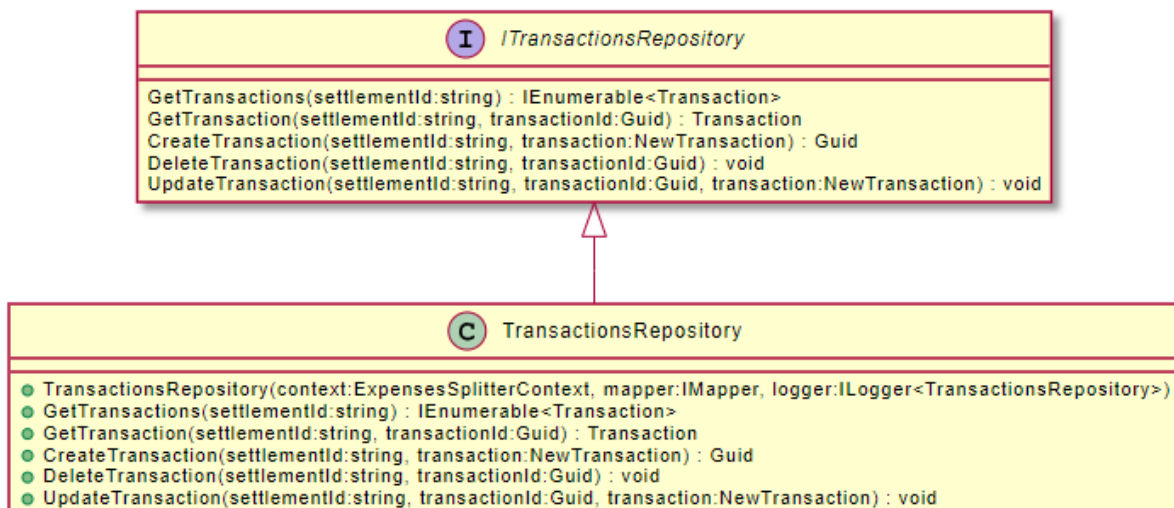
Providers

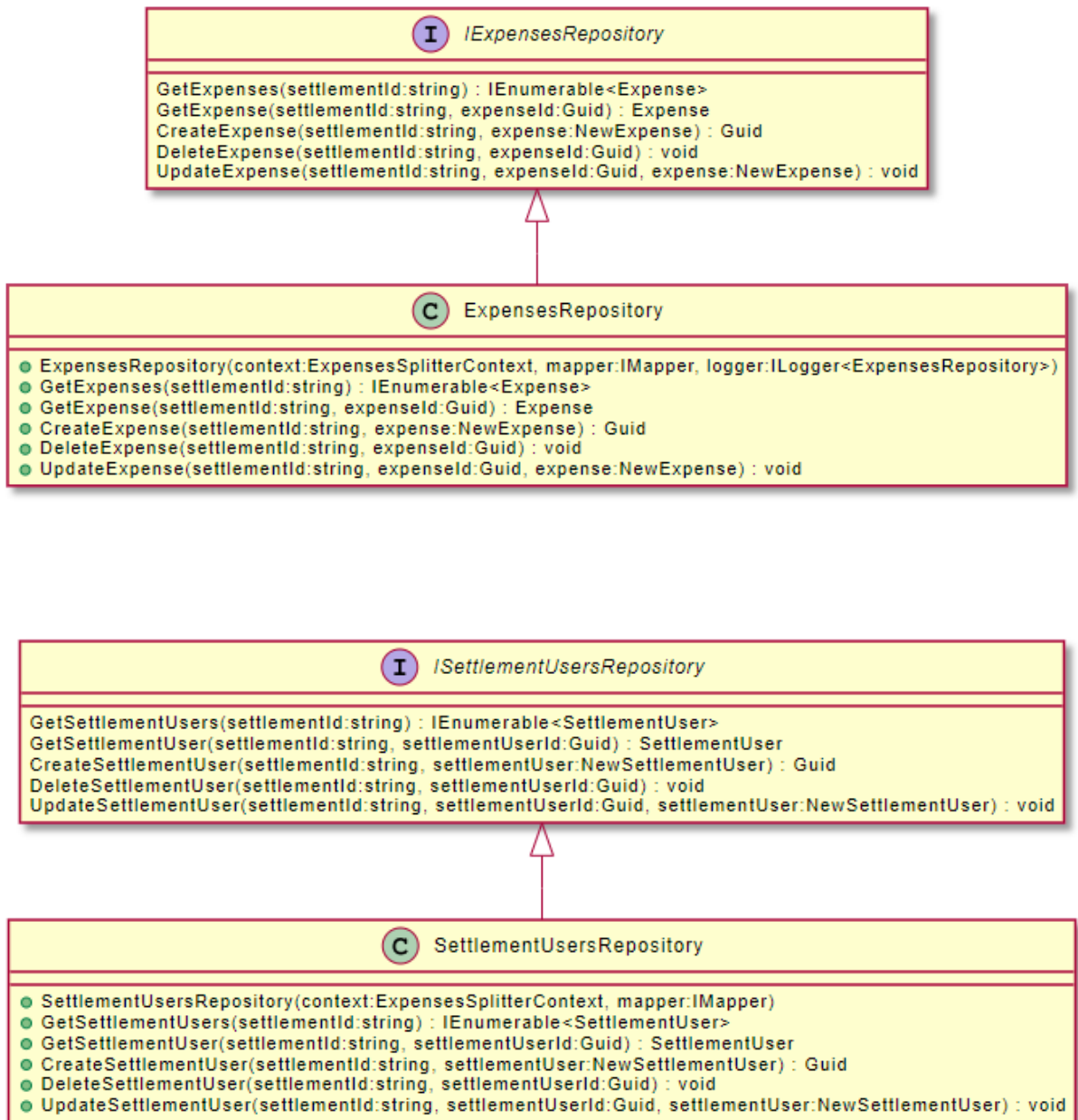


Algorithms

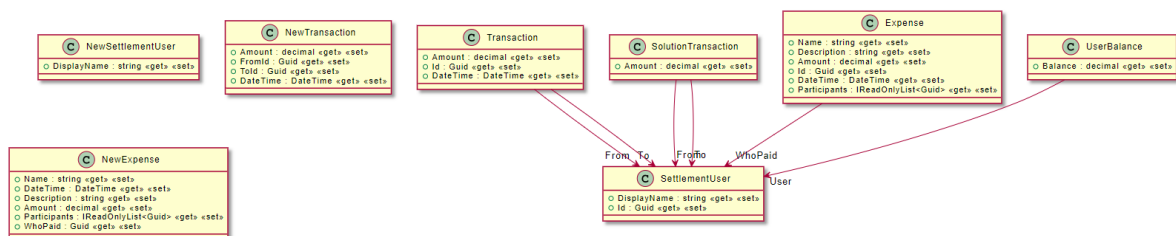


Repositories

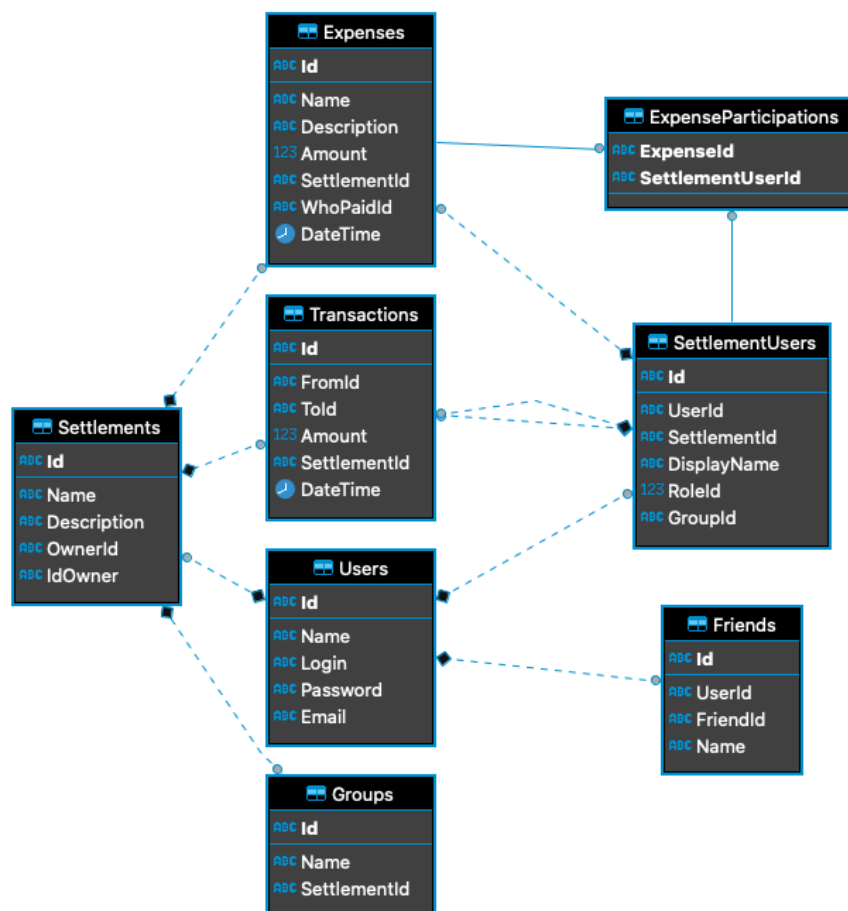




Models



Baza Danych



4.Testy

Testy do aplikacji zostały napisane z wykorzystaniem biblioteki XUnit. Do mockowania użyliśmy biblioteki NSubstitute. Testy znajdują się w projekcie ExpensesSplitter.WebApi.Tests.

```
ExpensesSplitter.WebApi.Tests (11 tests) Success
  ExpensesSplitter.WebApi.Tests (11 tests) Success
    Algorithms (6 tests) Success
      SettlementSolverTests (6 tests) Success
        Solve_EverybodyReturnsMoneyToOnePerson Success
        Solve_OnePersonReturnsMoneyToEverybodyElse Success
        Solve_QuiteComplexScenario Success
        Solve_SimpleScenario Success
        Solve_WhenBalancesAreZero_ReturnsEmptyList Success
        Solve_WhenSumOfBalancesIsNotZero Success
    Providers (5 tests) Success
      BalancesProviderTests (5 tests) Success
        GetBalances_RoundsBalances Success
        GetBalances_SplitsExpenses Success
        GetBalances_SplitsExpensesBetweenAll Success
        GetBalances_SplitsExpensesBetweenParticipants Success
        GetBalances_WhenTransactionsExist Success
```

5.Wymagania Techniczne

Aby szybko rozpocząć pracę nad projektem wystarczy skorzystać z dockera.

Zbudowanie obrazów

> docker-compose build

Uruchomienie

> docker-compose up

Wszystkie komponenty składające się na projekt zostaną uruchomione w trybie developmentu - zmiany w plikach źródłowych będą monitorowane, a odpowiednie komponenty rekompilowane: zarówno po stronie web-api jak i frontendu.

Migracje EF

Po zmianie modeli dla bazy danych, wystarczy wydać następujące polecenia:

> docker-compose stop web-api

> docker-compose run --rm web-api dotnet ef migrations add MigrationName

> docker-compose run --rm web-api dotnet ef database update

> docker-compose restart web-api

Podczas uruchamiania web-api automatycznie aplikowane są wszystkie migracje.

6.Technologie użyte do implementacji

- Docker
- Frontend - Angular
- Backend – C#
- Bootstrap

7.Podział pracy

Jakub Cisko	Filip Czyż
Implementacja dockera	Rejestracja i logowanie
Dodawanie wydatków do rozliczenia	Zarządzanie znajomymi
Równomierny podział wydatków pomiędzy wybrane osoby	Tworzenie rozliczeń
Widok proponowanych transakcji do rozliczenia	Tworzenie podgrup w ramach głównej grupy
Widok bilansu w rozliczeniu	Zarządzanie użytkownikami w rozliczeniu
Oznaczenie opłacenia wydatku w rozliczeniu	Lista rozliczeń na stronie głównej