

Krzysztof Kogut

Wspólna Kasa

Licencyjny projekt programistyczny

Spis treści

Spis treści.....	2
Wstęp	4
Podstawowe funkcjonalności	4
Wymagania niefunkcjonalne	4
Założenia.....	4
Baza danych.....	5
Schemat bazy danych	5
Tabele	5
AspNetUsers	5
Groups	5
Expenses.....	5
Transfers.....	6
Integracja aplikacji mobilnych	7
Architektura aplikacji internetowej.....	8
Schemat komponentów	8
Klasy opisujące model danych.....	9
Dodatkowe klasy	9
Warstwa prezentacji.....	9
Architektura sprzętowa	10
Interfejs graficzny użytkownika	11
Przypadki użycia i scenariusze testowe	14
Podejście do testów w projekcie.....	14
Przypadki użycia	14
Zarejestrowanie użytkownika	14
Logowanie	15
Tworzenie grupy	16
Zmiana nazwy grupy	16
Usunięcie grupy	16
Dodanie wydatku.....	16
Edytowanie wydatku	16
Usunięcie wydatku	16
Dodanie przelewu.....	16
Opis implementacji.....	18
Użyte technologie.....	18
Schemat przepływu	18

Kontrolery.....	18
Serwisy.....	19
Podsumowanie	20
Spełnienie wymagań niefunkcjonalnych	20
Łatwość obsługi	20
Możliwość integracji z aplikacjami mobilnymi	20
Uwierzytelnienie za pomocą Facebooka	20
Możliwe zastosowania	20
Dalszy rozwój aplikacji.....	20
Literatura	21

Wstęp

Wspólna Kasa to system do zapisywania wydatków pomiędzy różnymi osobami. Dzięki niemu wspólne zakupy do mieszkania, wyjazdy, imprezy czy zbiórki pieniędzy nie będą sprawiać kłopotów ani powodować konfliktów podczas ich rozliczania.

Podstawowe funkcjonalności

1. Tworzenie grup osób.

Użytkownicy aplikacji mają możliwość zakładania nowych i dołączania do istniejących grup. Osoby w grupie mają możliwość wspólnych rozliczeń. Nie ma ograniczeń na ilość tworzonych przez użytkownika grup.

2. Rejestrowanie wydatków.

Po założeniu grupy, jej członkowie mają możliwość dodawania wydatków. Wydatek można rozłożyć na wszystkich lub tylko wybranych członków grupy.

3. Rejestrowanie przelewów.

Użytkownicy mają możliwość rozliczania się przez zarejestrowanie przelewu do innego użytkownika. Wspólna Kasa nie jest pośrednikiem płatności, wpis o przelewie służy do poprawnego wyliczania bieżących sald.

4. Pokazywanie obecnego salda dla każdego z członków grupy (ile ma oddać i komu).

Każdy użytkownik może zobaczyć ile ogólnie (licząc wszystkie grupy, do których należy) ma zwrócić lub jaki dostanie zwrot. Ponadto wewnątrz każdej z grup, do której należy, użytkownik może zobaczyć bilans z każdym z członków grupy.

Wymagania нефunkcjonalne

- Łatwość obsługi. Wykonywanie poszczególnych akcji, takich jak dodawanie wydatków, czy tworzenie grup powinno być zgodnie ze standardami tworzenia nowoczesnych aplikacji internetowych.
- Architektura systemu powinna pozwolić na łatwe zintegrowanie systemu Wspólna Kasa z aplikacjami mobilnymi.
- Uwierzytelnienie za pomocą Facebooka.

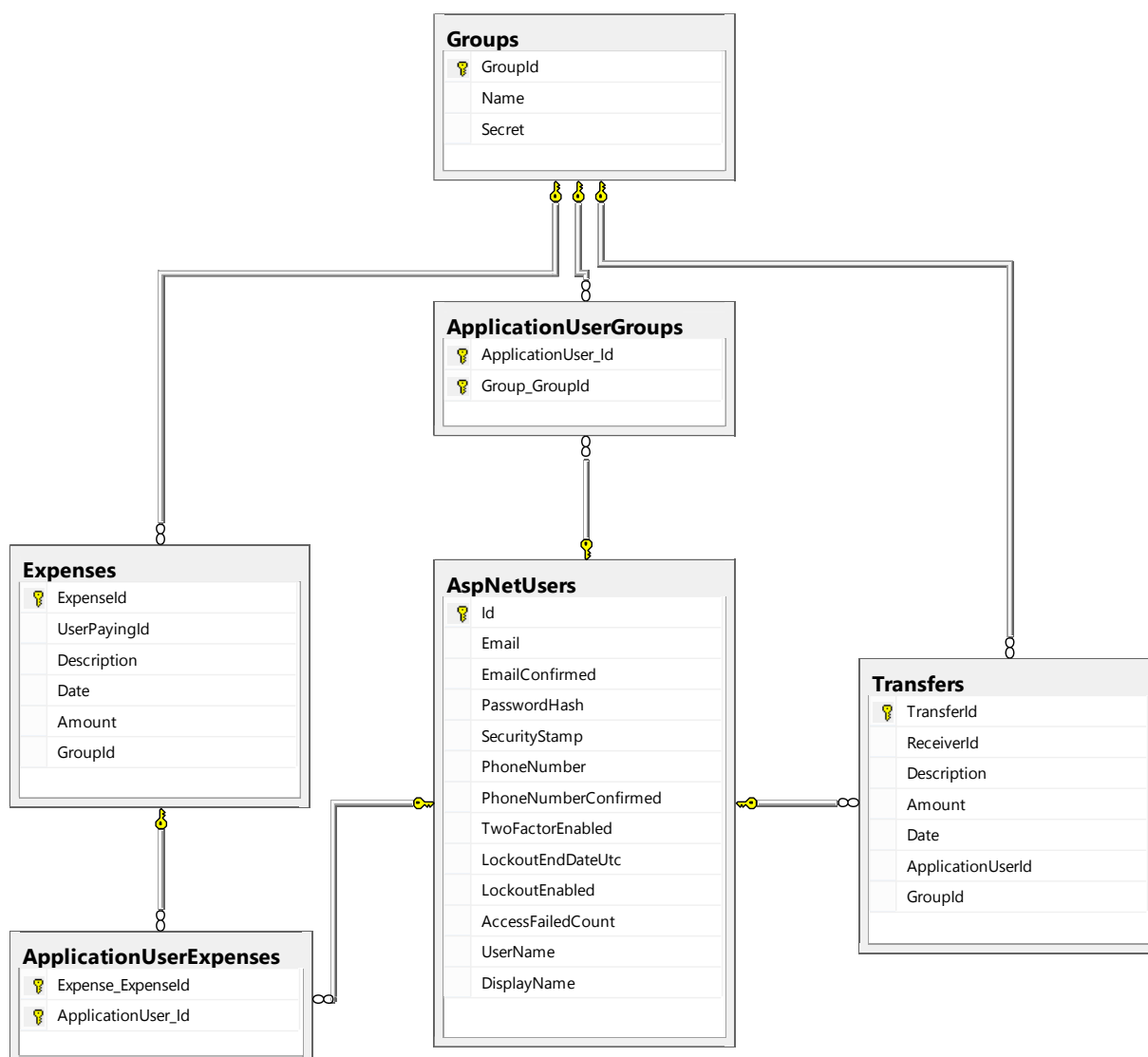
Założenia

Użytkownik musi posiadać komputer z dostępem do Internetu przeglądarkę oraz nowoczesną przeglądarkę internetową.

Baza danych

Schemat bazy danych

Schemat 1: Schemat bazy danych



Tabele

AspNetUsers

Tabela zawiera informacje o użytkownikach, między innymi jego identyfikator, adres e-mail i nazwę do wyświetlania. Kluczem głównym jest pole Id.

Groups

Tabela zawiera identyfikator grupy, jej nazwę oraz sekretne hasło. Kluczem głównym jest pole GroupId.

Expenses

Tabela zawiera identyfikator wydatku, jego opis, datę i kwotę. Odnosi się także przez klucz obcy do tabeli Groups, aby połączyć wydatek z grupą, oraz przez tabelę złączeniową *ApplicationUserExpenses* do tabeli *AspNetUsers*, by zidentyfikować uczestników wydatku. Pole *UserPayingId* przechowuje identyfikator użytkownika, płacącego za dany wydatek.

Transfers

Tabela zawiera identyfikator przelewu, jego tytuł, datę i kwotę. Odnosi się także przez klucz obcy do tabeli *Groups*, aby połączyć przelew z grupą, oraz do tabeli *AspNetUsers*, by zidentyfikować nadawcę przelewu (pole *ApplicationUserId*) oraz odbiorcę (*ReceiverId*).

Integracja aplikacji mobilnych

Jak wspomniano w rozdziale „Wymagania niefunkcjonalne”, aplikacja ma łatwo integrować się z aplikacjami mobilnymi. Baza danych jest wspólna dla aplikacji internetowej i mobilnej. Pierwsza będzie łączyć się z bazą danych bezpośrednio, jednocześnie wystawiając API¹ z dostępem przez protokół HTTP, z którego będzie korzystać aplikacja mobilna. Ze względu na standardowy format danych (JSON² i XML³), w którym przesyłane są odpowiedzi na żądania do protokołu HTTP, Wspólna Kasa może mieć aplikację na każdą platformę mobilną.

Programistyczny interfejs aplikacji oferuje następujące operacje:

- Pobranie listy obiektów,
- Pobranie konkretnego obiektu,
- Utworzenie nowego obiektu,
- Aktualizacja obiektu,
- Usunięcie obiektu.

Operacje dostępne dla klas typu: Group, Expense, Transfer.

Szczegółowa dokumentacja do API ze względu na swój czysto techniczny charakter została wyodrębniona do osobnego dokumentu.

¹ Interfejs programistyczny aplikacji (ang. *Application Programming Interface*)

² ang. *JavaScript Object Notation*

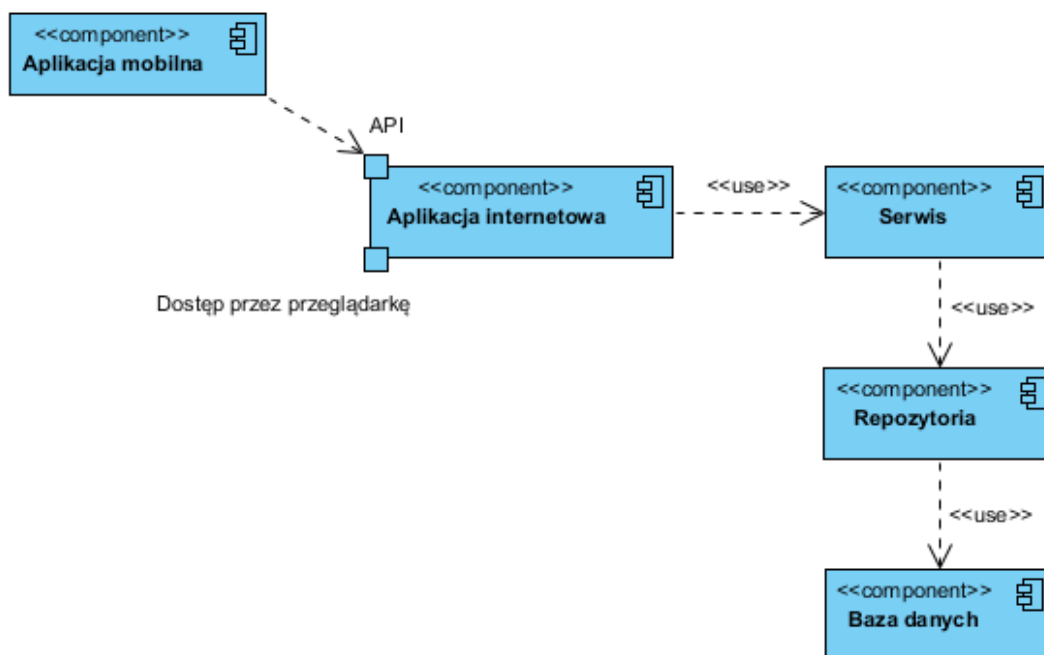
³ ang. *Extensible Markup Language*

Architektura aplikacji internetowej

Aplikacja internetowa została oparta o wzorec architektoniczny MVC⁴. Dzięki temu odpowiedzialności obsługi wyświetlania danych, przetwarzania żądań i przechowywania danych zostały rozdzielone.

Schemat komponentów

Schemat 2: Schemat komponentów

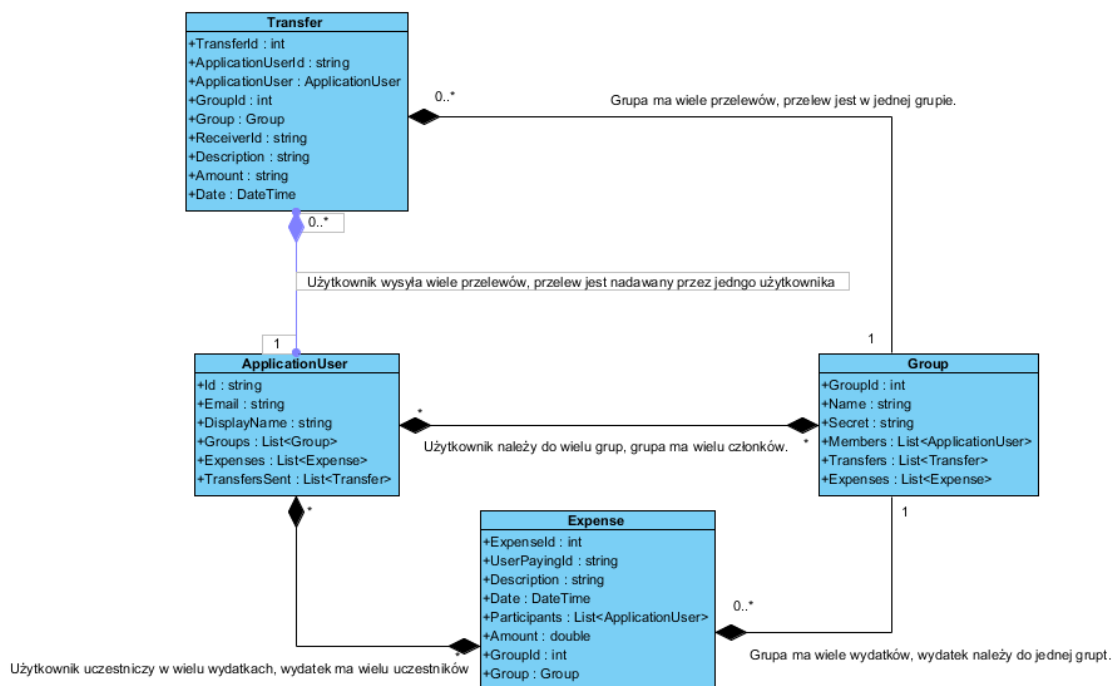


Aplikacja internetowa, odpowiedzialna za przetwarzanie żądań pochodzących z API oraz z przeglądarki, odwołuje się do odpowiedniego serwisu, który jest odpowiedzialny za logikę systemu. Jeżeli wymagany jest dostęp do danych, serwis zleca to odpowiednim repozytoriom, będącym abstrakcją źródła danych.

⁴ Model-View-Controller (zob. <http://pl.wikipedia.org/wiki/Model-View-Controller>)

Klasy opisujące model danych

Schemat 3: Klasy odpowiadające schematowi bazy danych

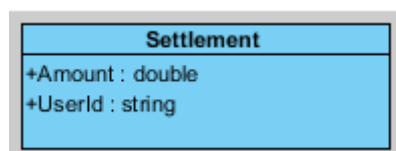


Klasy przedstawione na powyższym schemacie, odpowiadają tabelom bazy danych. Ponieważ model obiektowy pozwala na odzwierciedlenie relacji wiele do wielu, nie ma tutaj klas pośredniczących.

Dzięki użyciu techniki ORM⁵ stosując narzędzie *Entity Framework Code First*⁶ schemat bazy danych został automatycznie wygenerowany na podstawie powyższego modelu.

Dodatkowe klasy

Schemat 4: Klasa Settlement



Na kolejnym poziomie aplikacji została dodana klasa **Settlement**, opisująca bilans wobec innego użytkownika. Kolekcja obiektów typu **Settlement**, jest używany do przedstawienia bilansu ze wszystkimi użytkownikami, lub użytkownikami wybranej grupy.

Warstwa prezentacji

Warstwa prezentacji, napisana w technologii MVC (Model-View-Controller), wybiera potrzebne pola z klas modelowych i przechowuje je w klasach typu model. Zadaniem kontrolera jest przetworzenie żądania i wyświetlenie odpowiedniego widoku na podstawie powiązanego modelu.

⁵ Object-relational mapping (zob. http://en.wikipedia.org/wiki/Object-relational_mapping)

⁶ Zob. <https://www.asp.net/mvc/overview/getting-started/getting-started-with-ef-using-mvc/creating-an-entity-framework-data-model-for-an-asp-net-mvc-application>

Architektura sprzętowa

Aplikacja do działania potrzebuje serwer z systemem Windows i usługami IIS⁷, natomiast baza danych wymaga oprogramowania Microsoft SQL Server. Serwery, na których Wspólna Kasa działa, muszą mieć dostęp do Internetu.

Powyższe warunki zostały spełnione dzięki usłudze Windows Azure, która pozwala m.in. na hosting aplikacji internetowych i powiązanych baz danych.

⁷ Internet Information Services (zob. http://pl.wikipedia.org/wiki/Internet_Information_Services)

Interfejs graficzny użytkownika

Poniżej zostaną zaprezentowane zrzuty ekranu najważniejszych fragmentów aplikacji wraz z krótkimi opisami.

Zrzut ekranowy 1: Kokpit

The dashboard is titled 'Wspólna Kasa' and includes links for 'Ustawienia profilu' and 'Wyloguj'. It is divided into three main sections: 'Grupy' (Groups), 'Ludzie' (People), and 'Aktualności' (Transactions).

Grupy (Groups):

- Roentgena:** Asia Borowska (52.00 zł), Diana Szumilas (16.67 zł)
- K&D Team**
- Zakopcowe love**

Ludzie (People):

- Asia Borowska (52.00 zł)
- Diana Szumilas (147.17 zł)
- Razem: 199.17 zł**

Aktualności (Transactions):

Grupa	Osoba	Opis	Data	Kwota
Roentgena	Krzysztof Kogut	UPC Internet	5/15/2015	90.00 zł
K&D Team	Krzysztof Kogut	Prezent dla Martyny Schulz	5/13/2015	24.00 zł
Roentgena	Krzysztof Kogut	Śniadaniowe (sok, chleb, ser, kielbasa krakowska)	5/12/2015	24.00 zł
K&D Team	Krzysztof Kogut	Piwko piweczko	5/11/2015	20.00 zł
K&D Team	Krzysztof Kogut	Piec na szewskiej	5/11/2015	42.00 zł
K&D Team	Diana Szumilas	Rzeszów Trip	5/11/2015	136.00 zł
Roentgena	Krzysztof Kogut	Carrefour	5/10/2015	30.00 zł
K&D Team	Diana Szumilas	Piwo i napitki	5/8/2015	15.00 zł
Roentgena	Diana Szumilas	Warzywniak + pierdółki	5/4/2015	70.00 zł
K&D Team	Krzysztof Kogut	Majówka	5/3/2015	100.00 zł

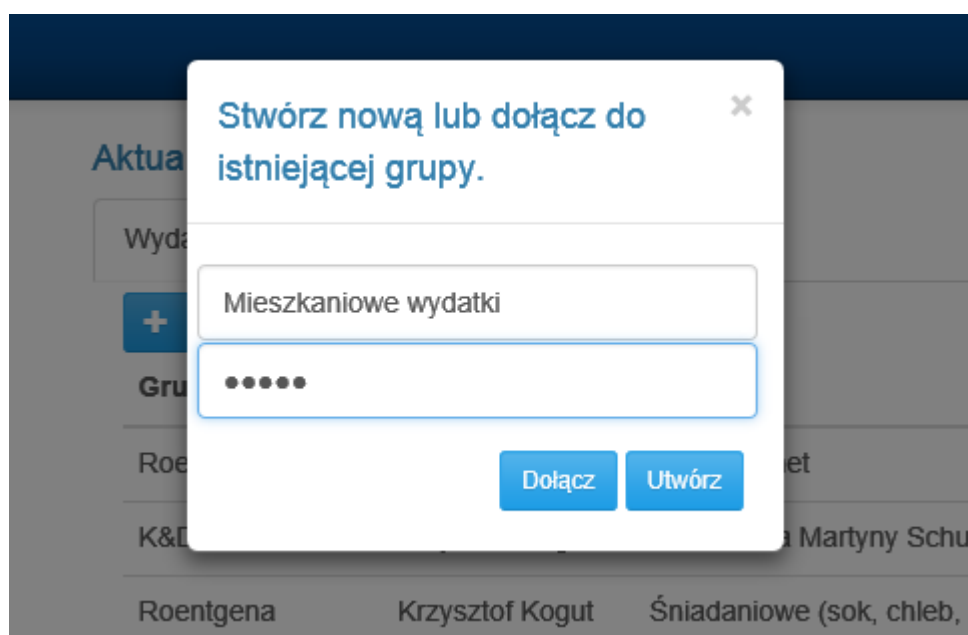
Z poziomu kokpitu można zobaczyć informacje takie jak:

- Przynależność do grup,
- Podsumowania wydatków wewnątrz grup i pomiędzy znajomymi,
- Zestawienie wydatków,
- Zestawienie przelewów.

Można także wykonać następujące akcje:

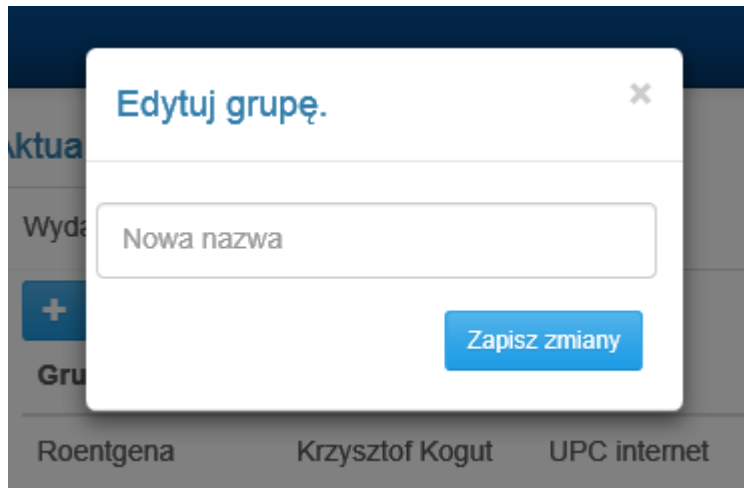
- Dodanie nowej lub dołączenie do istniejącej grupy.

Zrzut ekranowy 2: Dodanie nowej lub dołączenie do istniejącej grupy



- Zmiana nazwy grupy.

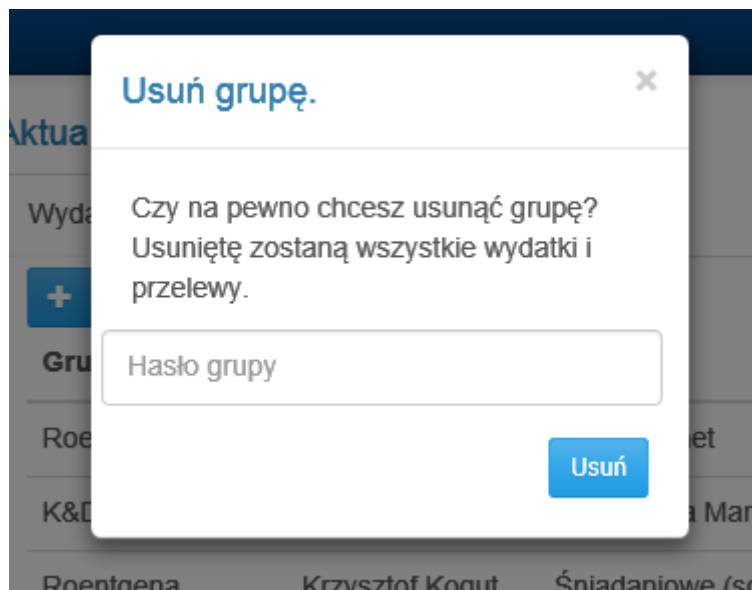
Zrzut ekranowy 3: Zmiana nazwa grupy



The screenshot shows a modal dialog box titled "Edytuj grupę." with a close button (X) in the top right corner. Inside the dialog, there is a text input field labeled "Nowa nazwa" and a blue button labeled "Zapisz zmiany". The background is a blurred view of a web application interface with a sidebar and a main content area.

- Usunięcie grupy.

Zrzut ekranowy 4: Usunięcie grupy



The screenshot shows a modal dialog box titled "Usuń grupę." with a close button (X) in the top right corner. Inside the dialog, there is a confirmation message: "Czy na pewno chcesz usunąć grupę? Usuniętę zostaną wszystkie wydatki i przelewy." Below the message is a text input field labeled "Hasło grupy" and a blue button labeled "Usuń". The background is a blurred view of a web application interface with a sidebar and a main content area.

- Dodanie, edytowanie i usunięcie wydatku.

Zrzut ekranowy 5: Dodanie wydatku

Dodaj wydatek.

Wydatek: Roentgena

Grupa: Asia Borowska, Krzysztof Kogut, Diana Szumilas

Opis: Wspólne kosmetyki

Data: 2015-05-15

Kwota: 45,78

Dodaj

- Dodanie przelewu.

Zrzut ekranowy 6: Dodanie przelewu

Wprowadź przelew.

Opis: Zwrot za kwiecień

Data: 2015-05-03

Kwota: 32,5

Dodaj

- Filtrowanie list wydatków i przelewów po grupie.

Przypadki użycia i scenariusze testowe

Podejście do testów w projekcie

Projekt, zgodnie z dokumentem pt. „Wizja” był prowadzony wg. zasad TDD⁸ i ciągłej integracji. Wszystkie metody serwisowe, czyli te, w których znajduje się logika biznesowa zostały opisane testami jednostkowymi.

Dodatkowo, zostały opisane przypadki użycia, służące jako scenariusze testowe. Odtwarzając je, można całościowo zweryfikować poprawność działania aplikacji.

Przypadki użycia

Zarejestrowanie użytkownika

1. Należy kliknąć w głównym menu na przycisk „Zarejestruj”.
2. Należy wypełnić wszystkie pola formularza i kliknąć na przycisk „Zarejestruj”.
3. Jesteśmy zarejestrowani i zalogowani, co można zobaczyć po zmianie górnego menu i pojawieniu się przycisku „Wyloguj”.

Wspólna Kasa

Zarejestruj Zaloguj

Wspólna Kasa

Rozliczaj wspólne wydatki ze znajomymi. Szybko i łatwo. Bez żadnych nieporozumień.

Przejdź do kokpitu

Wspólna Kasa

Zarejestruj się.

krzysiek.kogut@mail.com

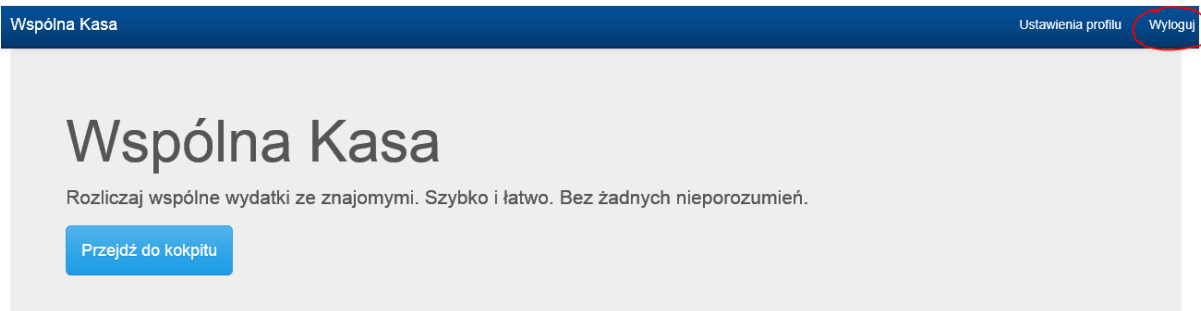
Krzysztof Kogut

••••••••

••••••••|

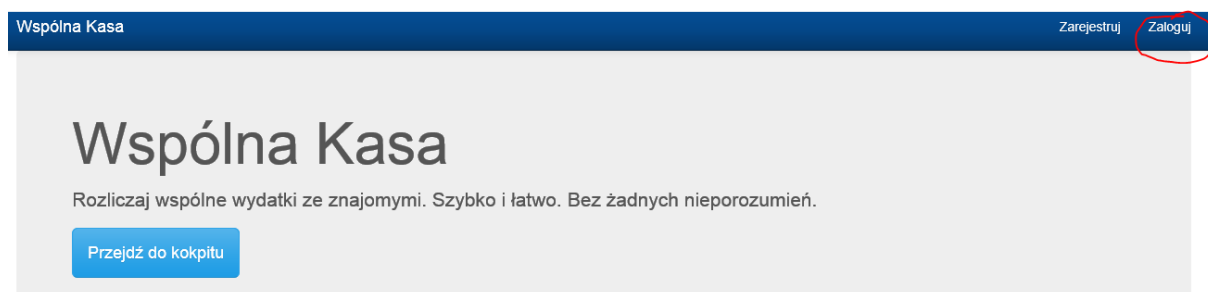
Zarejestruj

⁸ Test Driven Development (ang. programowanie sterowane testami).




Logowanie

1. Klikamy w głównym menu na przycisk „Zaloguj”.
2. Podajemy adres e-mail wybrany podczas rejestracji i hasło lub klikamy na przycisk „Zaloguj przez Facebook”.
3. Jesteśmy zalogowani, co można zobaczyć po zmianie górnego menu i pojawieniu się przycisku „Wyloguj” (jak w podrozdziale „Rejestrowanie użytkownika”).



Zaloguj się.

 Zaloguj przez Facebook

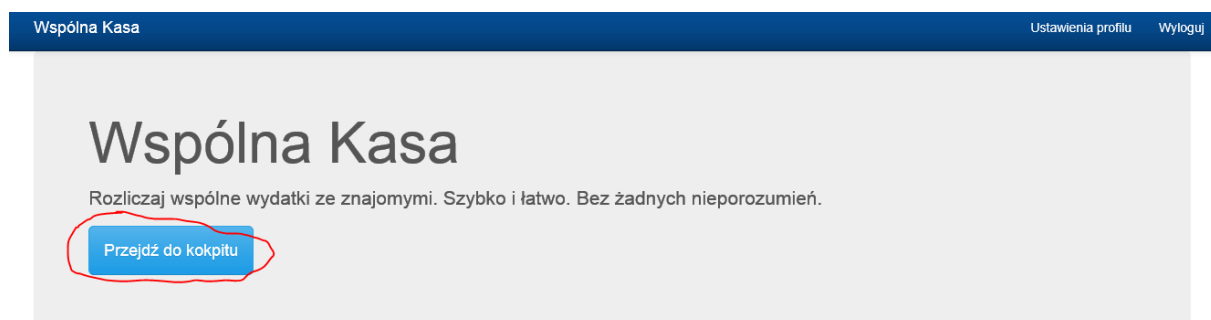
Zaloguj

☐ Nie wylogowuj mnie.

[Zapomniałeś hasła?](#)

Tworzenie grupy

Przypadki użycia opisane w tym i kolejnych podrozdziałach zaczynają się od wejścia do „kokpitu” (patrz rysunek niżej), lecz nie będą zawierały zrzutów ekranowych, gdyż są one umieszczone w rozdziale „Interfejs graficzny użytkownika”.



1. Nad listą grup, po lewej stronie klikamy na przycisk ze znakiem „+”.
2. Wypełniamy pola i klikamy na przycisk „Utwórz”.
3. Grupa powinna pojawić się na liście.

Zmiana nazwy grupy

1. Klikamy na nazwę grupy.
2. Aktywuje się przycisk z ikoną ołówka (środkowy), klikamy na niego.
3. Wpisujemy nową nazwę.
4. Klikamy przycisk „Zapisz zmiany”.
5. Grupa ze zmienioną nazwą powinna pojawić się na liście.

Usunięcie grupy

1. Klikamy na nazwę grupy.
2. Aktywuje się przycisk z ikoną ołówka „x”, klikamy na niego.
3. Wpisujemy hasło grupy i klikamy przycisk „Usuń”.
4. Grupy nie powinna być widoczna na liście.

Dodanie wydatku

1. Klikamy na przycisk „+” w sekcji „Aktualności”.
2. Wybieramy grupę, na liście osób wybrane osoby z grupy (używając przycisku Ctrl), podajemy opis, datę, kwotę,
3. Wciskamy przycisk „Dodaj”.
4. Wydatek powinien pojawić się na liście.

Edytowanie wydatku

1. Klikamy na przycisk z ikoną ołówka (środkowy) w sekcji „Aktualności”.
2. Zmieniamy wybrane dane.
3. Wciskamy przycisk „Zmień”.
4. Wydatek powinien zostać zaktualizowany.

Usunięcie wydatku

1. Klikamy na przycisk z ikoną „x” w sekcji „Aktualności”.
2. Wciskamy przycisk „Usuń”.
3. Wydatek powinien zostać usunięty z listy.

Dodanie przelewu

Przelewu można dokonać wewnątrz grupy do osoby, której zalegamy pieniądze.

1. Na rozwijanej liście grup znajdujemy osobę z ujemną kwotą (wyświetla się na czerwono).
2. Klikamy na tę osobę.
3. Wpisujemy kwotę (domyślnie pełen dług), opis i datę.
4. Klikamy „Dodaj”.
5. Przelew powinien pojawić się na liście przelewów w sekcji „Aktualności”.

Opis implementacji

Użyte technologie

W projekcie zostały użyte następujące technologie:

Technologia	Cel
ASP.NET MVC	Platforma do budowy aplikacji internetowych opartych na wzorcu MVC.
ASP.NET WebAPI	Środowisko umożliwiające utworzenie interfejsu programistycznego aplikacji.
Entity Framework Code First	Biblioteka służąca do mapowania obiektowo relacyjnego i generowania schematu bazy danych.
SendGrid	Usługa internetowa pozwalająca na wysyłanie wiadomości e-mail.
Bootstrap	Framework CSS, zawierający zestaw narzędzi ułatwiających tworzenie interfejsu graficznego stron internetowych.
jQuery, jQuery UI	Biblioteka programistyczna JavaScript ułatwiająca manipulację DOM.
C#	Język zastosowany do zaprogramowania serwerowej części aplikacji.

Schemat przepływu

Kontrolery

Każde żądanie, które przychodzi do aplikacji, czy to przez przeglądarkę, czy przez API, jest przetwarzane przez odpowiedni kontroler (kontroler MVC lub kontroler WebAPI). W aplikacji mamy poniższe kontrolery:

MVC

- HomeController – do wyświetlania strony głównej.
- AccountController – do przetwarzania żądań związanych z rejestracją i logowaniem.
- ManageController – do przetwarzania żądań związanych z zarządzaniem kontem.
- DashboardController – do przetwarzania żądań dotyczących głównych funkcjonalności aplikacji.

Kontrolery MVC są odpowiedzialne za wyświetlanie odpowiednich widoków i tworzenie odpowiednich modeli, na podstawie danych obliczonych w serwisach.

WebAPI

- AccountController – do przetwarzania żądań związanych z rejestracją i logowaniem.
- ExpensesController – operacje CRUD⁹ na obiektach dotyczących wydatków.
- GroupsController – operacje CRUD na obiektach dotyczących grup.
- TransfersController – operacje CRUD na obiektach dotyczących przelewów.

Kontrolery WebAPI delegują jedynie żądania do bazy danych.

⁹ Create, Read, Update, Delete (ang. utwórz, czytaj, aktualizuj, usuń).

Serwisy

Wszystkie operacje wymagające dostępu do danych są delegowane do repozytoriów, będących abstrakcją źródła danych.

- GroupService – odpowiedzialny za logikę związaną z grupami. Pozwala na dodanie grupy, dołączenie do grupy, pobranie jednej lub wszystkich dla danego użytkownika, edycję czy usunięcie.
- TransactionService – odpowiedzialny za logikę związaną z transakcjami, pod pojęciem których rozumie się przelewy i wydatki. Oprócz ich pobierania, edycji i usuwania, serwis pozwala na wyliczanie bilansów dla użytkownika ogólnie i wewnątrz danej grupy.

Kod źródłowy obliczający bilans dla użytkownika na podstawie pobranych przelewów i wydatków

```
private IEnumerable<Settlement> GroupTransfersAndExpensesToSummary(
    string userId, IEnumerable<Transfer> transfers, IEnumerable<Expense> expenses)
{
    var transfersSent = transfers
        .Where(t => t.ApplicationUserId == userId)
        .GroupBy(t => t.ReceiverId)
        .Select(tg => new Settlement { Amount = tg.Sum(s => s.Amount), UserId = tg.Key });

    var transfersReceived = transfers
        .Where(t => t.ReceiverId == userId)
        .GroupBy(t => t.ApplicationUserId)
        .Select(tg => new Settlement { Amount = -tg.Sum(s => s.Amount), UserId = tg.Key });

    var expensesPaid = expenses
        .Where(x => x.UserPayingId == userId);
    List<Settlement> expensesPaidSettlements = new List<Settlement>();
    foreach (var exp in expensesPaid)
    {
        var amount = exp.Amount / exp.Participants.Count;
        foreach (var set in exp.Participants.Where(x => x.Id != userId))
        {
            expensesPaidSettlements.Add(new Settlement { Amount = amount, UserId = set.Id });
        }
    }

    var expensesParticipated = expenses
        .Where(x => x.Participants.Select(y => y.Id).Contains(userId))
        .Except(expensesPaid)
        .Select(x => new Settlement { UserId = x.UserPayingId, Amount = -x.Amount / x.Participants.Count });

    return
        transfersSent
        .Concat(transfersReceived)
        .Concat(expensesPaidSettlements)
        .Concat(expensesParticipated)
        .GroupBy(x => x.UserId)
        .Select(x => new Settlement { Amount = x.Sum(y => y.Amount), UserId = x.Key });
}
```

Podsumowanie

Spełnienie wymagań niefunkcjonalnych

Łatwość obsługi

Dzięki użyciu najnowszych technologii i podążania za standardami tworzenia nowoczesnych aplikacji internetowych Wspólna Kasa jest intuicyjna w użyciu. Stosowanie spójnych oznaczeń oraz dodanie tzw. tooltipów, czyli podpowiedzi po najechaniu na przycisk, pozwala szybko odnaleźć się w aplikacji.

Możliwość integracji z aplikacjami mobilnymi

Dzięki stworzeniu API, Wspólna Kasa pozwala na dostęp za pomocą innych aplikacji, w tym aplikacji mobilnych.

Uwierzytelnienie za pomocą Facebooka

Logowanie za pomocą Facebooka zostało pokazane w przypadkach użycia.

Możliwe zastosowania

Wspólna Kasa ma zastosowanie w codziennym życiu, na przykład do rozliczania mieszkaniowych wydatków, a także okazynie: podczas wyjazdów, składek na prezent czy imprezę. Dzięki możliwości tworzenia grup, łatwo rozgraniczyć różne typy wydatków.

Dalszy rozwój aplikacji

Wspólna Kasa dzięki stworzonemu API może dotrzeć do szerszego grona użytkowników dzięki aplikacjom mobilnym.

Funkcjonalnie, warto zastanowić się nad możliwością dodawania wydatków, których koszt nie jest dzielony proporcjonalnie na użytkowników, a w ustalony przez dodającego sposób. Przykładem takiego wydatku jest wyjście do restauracji, gdzie jedna osoba płaci za wszystkich, a każdy zamówił coś w innej cenie.

Niefunkcjonalnie, wprowadzenie innych języków (w szczególności angielskiego) może rozszerzyć grono odbiorców o cały świat.

Literatura

1. Strona domowa technologii ASP.NET MVC: <http://www.asp.net/mvc>
2. Wzorzec architektoniczny Model-View-Controller: <http://pl.wikipedia.org/wiki/Model-View-Controller>
3. Mapowanie obiektowo-relacyjne: http://en.wikipedia.org/wiki/Object-relational_mapping
4. Entity Framework Code First: <https://www.asp.net/mvc/overview/getting-started/getting-started-with-ef-using-mvc/creating-an-entity-framework-data-model-for-an-asp-net-mvc-application>
5. Kent Beck. „TDD. Sztuka wytwarzania dobrego kodu.”, wyd. Helion, 2003 r.
6. Robert C. Martin. „Czysty kod. Podręcznik dobrego programisty.”, wyd. Helion, 2010 r.