Politechnika Śląska

Wydział Automatyki, Elektroniki i Informatyki

Technologie aplikacji internetowych
Projekt
PairPay

Natalia Stręk,
Jakub Kula,
Paweł Wójtowicz

Gliwice 2024

1 Przedstawienie pomysłu na projektu

Aplikacja do zarządzania wydatkami została zaprojektowana z myślą o osobach regularnie dzielących się kosztami, na przykład podczas wspólnych podróży, zakupów czy spotkań towarzyskich. Jej głównym celem jest uproszczenie procesu rejestrowania, monitorowania i rozliczania wydatków w grupach, co umożliwia efektywne zarządzanie finansami oraz minimalizuje ryzyko nieporozumień związanych z podziałem kosztów.

Użytkownicy mogą zarejestrować się i zalogować do systemu, uzyskując tym samym dostęp do chronionych funkcji aplikacji. Aplikacja pozwala na tworzenie nowych rachunków, do których można dodawać wydatki oraz uczestników, a następnie automatycznie dzieli koszty na podstawie wprowadzonych danych. Dzięki funkcji skanowania paragonów użytkownicy mogą łatwo importować informacje o wydatkach, eliminując konieczność ręcznego wprowadzania szczegółów.

Aplikacja oferuje także możliwość edytowania i usuwania rachunków oraz przeglądania historii wydatków, co zapewnia użytkownikom pełną kontrolę nad ich finansami. Dodatkowo, opcje filtrowania i sortowania rachunków, a także etykietowania wydatków, zwiększają użyteczność aplikacji, umożliwiając łatwe odnalezienie oraz zarządzanie danymi.

Wszystkie te funkcje sprawiają, że aplikacja jest nieocenionym narzędziem dla osób pragnących efektywnie zarządzać swoimi wydatkami, oszczędzając przy tym czas i minimalizując ryzyko błędów przy podziale kosztów.

2 User Story

2.1 Para dzieląca się wydatkami

Para, która mieszka razem, często robi wspólne zakupy. Zazwyczaj jedna osoba wybiera się na większe zakupy spożywcze, a druga później zwraca swoją część kosztów. Niestety, czasami pojawiają się problemy z ustaleniem, ile dokładnie każdy powinien zapłacić, bo paragonów przybywa, a nie zawsze jest czas, by od razu je podsumować. Dzięki aplikacji pariPay można na bieżąco dodawać wydatki do wspólnego rachunku i nie martwić się o szczegóły.

2.2 Rozliczenie rachunku w restauracji

Grupa znajomych wybrała się na kolację do ulubionej restauracji. Po posiłku jedna osoba zdecydowała się opłacić cały rachunek. Zamiast przeliczać, kto co zamówił i ile powinien zapłacić, skorzystano z aplikacji pariPay. Wszystkie pozycje z paragonu zostały wprowadzone, przypisane do odpowiednich osób, a jednym kliknięciem wysłano znajomym wiadomość o rozliczeniu. Każdy może teraz bez problemu opłacić swoją część rachunku bez nieporozumień i przeliczania.

2.3 Składka na imprezę z różnym rozliczaniem kosztów

Podczas organizacji domowej imprezy grupa znajomych ustaliła, że złożą się na jedzenie i przekąski, a jedna osoba zadba o zakup napojów alkoholowych. Po imprezie pojawiła się potrzeba rozliczenia kosztów, jednak nie wszyscy spożywali alkohol, więc podział nie mógł być równy. W takim przypadku skorzystano z aplikacji pariPay, która pozwala na wprowadzenie wszystkich wydatków do wspólnego rachunku i precyzyjne przypisanie kosztów – np. pełne

rozliczenie napojów alkoholowych tylko do tych, którzy je spożywali. Dzięki temu rachunek został podzielony sprawiedliwie, uwzględniając rzeczywiste wydatki każdej osoby.

3 Specyfikacja wymagań aplikacji

Nr	Nazwa	Opis	Kryterium odbioru
F_1	Logowanie i rejestracja	Użytkownik może zakładać konto oraz logować się na nie.	System musi umożliwiać rejestrację, logowanie, oraz uzyskanie dostępu do chronionych funkcji aplikacji. Każda interakcja z chronionymi zasobami wymaga sprawdzenia tożsamości. Użytkownik musi także móc się wylogować, co kończy sesję i blokuje dostęp do zasobów.
F_2	Tworzenie nowego rachunku	Użytkownik może tworzyć nowy rachunek, dodawać przedmioty oraz uczestników do podziału kosztów.	Użytkownik musi móc wprowadzić nazwę rachunku, dodać uczestników i szczegóły, a następnie zapisać rachunek. Rachunek musi być widoczny dla wszystkich dodanych uczestników. System musi poprawnie zapisywać dane rachunku oraz umożliwiać ich przeglądanie i edycję.
F_3	Dodanie nowych wydatków	Aplikacja umożliwia dodawanie nowych wydatków do istniejących rachunków.	Użytkownik musi móc wprowadzić szczegóły wydatku, takie jak kwota, opis oraz przypisanie do uczestników. System musi aktualizować saldo rachunku i widok wydatków, a także potwierdzić dodanie wydatku.
F_4	Automatyczny podział kosztów	Aplikacja automatycznie dzieli koszty między użytkowników.	System musi poprawnie obliczać i wyświetlać udział każdej osoby w rachunku, bazując na wydatkach oraz przypisanych uczestnikach. Użytkownik może wybrać metodę podziału (np. równy lub według udziałów) i otrzymać podsumowanie. System aktualizuje informacje po dodaniu nowych wydatków.
F_5	Automatyczne odczytywanie paragonów	Aplikacja umożliwia wgranie zdjęcia paragonu, z którego odczytywane są informacje o kosztach.	System musi poprawnie rozpoznać i wydobyć kluczowe dane z paragonu (datę, kwotę, szczegóły wydatków). Użytkownik może przesłać obraz paragonu, a system zapisze te informacje w rachunku, potwierdzając ich dodanie.
F_6	Tworzenie grup znajomych	Możliwość zapisywania grup osób, z którymi użytkownik często współdzieli rachunki.	Użytkownik może tworzyć grupy, nadając im nazwy i dodając członków z listy znajomych. System umożliwia edytowanie, usuwanie grup, oraz wyświetla wszystkie grupy użytkownika.
F_7	Edytowanie i usuwanie rachunku	Aplikacja umożliwia edytowanie oraz usuwanie rachunków.	Użytkownik może edytować szczegóły rachunku (nazwa, uczestnicy, wydatki). System zapisuje zmiany i aktualizuje informacje, a także umożliwia usunięcie rachunku z odpowiednim potwierdzeniem i powiadomieniem uczestników.
F_8	Historia rachunków	Aplikacja zapisuje poprzednie rachunki, umożliwiając przeglądanie historii.	Użytkownik ma możliwość wyświetlania listy wcześniejszych rachunków, wraz z informacjami o nazwie, dacie oraz uczestnikach, a także przeglądania szczegółów każdego rachunku.
F_9	Filtrowanie i sortowanie rachunków	Aplikacja umożliwia filtrowanie i sortowanie rachunków według różnych kryteriów.	Użytkownik może stosować różne filtry, takie jak data, uczestnicy, kwota. System aktualizuje widok rachunków zgodnie z wybranymi kryteriami, umożliwiając sortowanie według wybranych parametrów.
F_10	Etykietowanie rachunków	Aplikacja umożliwia przypisanie etykiet do wydatków, określających ich typ (np. rozrywka, transport, dom).	System musi umożliwiać łatwe przypisywanie i przeglądanie etykiet, co ułatwia klasyfikację wydatków w rachunkach.
F_11	Panel administracji	Administrator może monitorować różne statystyki za pośrednictwem aplikacji typu ilość użytkowników, grup.	System powinien umożliwiać wgląd tylko autoryzowanym użytkownikom i w prawdiłowy sposób wyświetlać informację. System umożliwia na
F_12	Obsługa płatności	Obsługa rozliczeń rachunków w grupach użytkowników	

Tabela 1: Opis funkcjonalności aplikacji oraz kryteriów odbioru

Kod	Nazwa	Opis
NF_1	Wydajność	Aplikacja powinna ładować strony i przeprowadzać operacje w sposób płynny, zapewniając komfortowe doświadczenie użytkownika.
NF_2	Bezpieczeństwo	• Bezpieczne przechowywanie wrażliwych danych użytkowników poprzez szyfrowanie. • Ochrona przed nieuprawnionym dostępem (ochrona przed atakami typu SQL injection, autentykacja i autoryzacja) • Ochrona zasobów poprzez ścieżki kontrolowane (panel administracji) oraz autoryzację za pomocą tokenów
NF_3	Użyteczność	Interfejs użytkownika powinien być prosty i intuicyjny, zawierając jasne instrukcje dotyczące dodawania rachunków, etykietowania i podziału kosztów, aby zminimalizować krzywą uczenia się dla nowych użytkowników.
NF_4	Dokumentacja	Aplikacja powinna posiadać szczegółową dokumentację zarówno dla użytkowników, jak i administratorów, opisującą wszystkie funkcje oraz procedury użytkowania.
NF_5	Kompatybilność	Aplikacja musi działać poprawnie na większości popularnych przeglądarek internetowych, a także na urządzeniach o różnej rozdzielczości ekranu.
NF_6	Intuicyjność obsługi aplikacji	Aplikacja powinna mieć prosty i intuicyjny interfejs, umożliwiający łatwe korzystanie użytkownikom w różnym wieku.

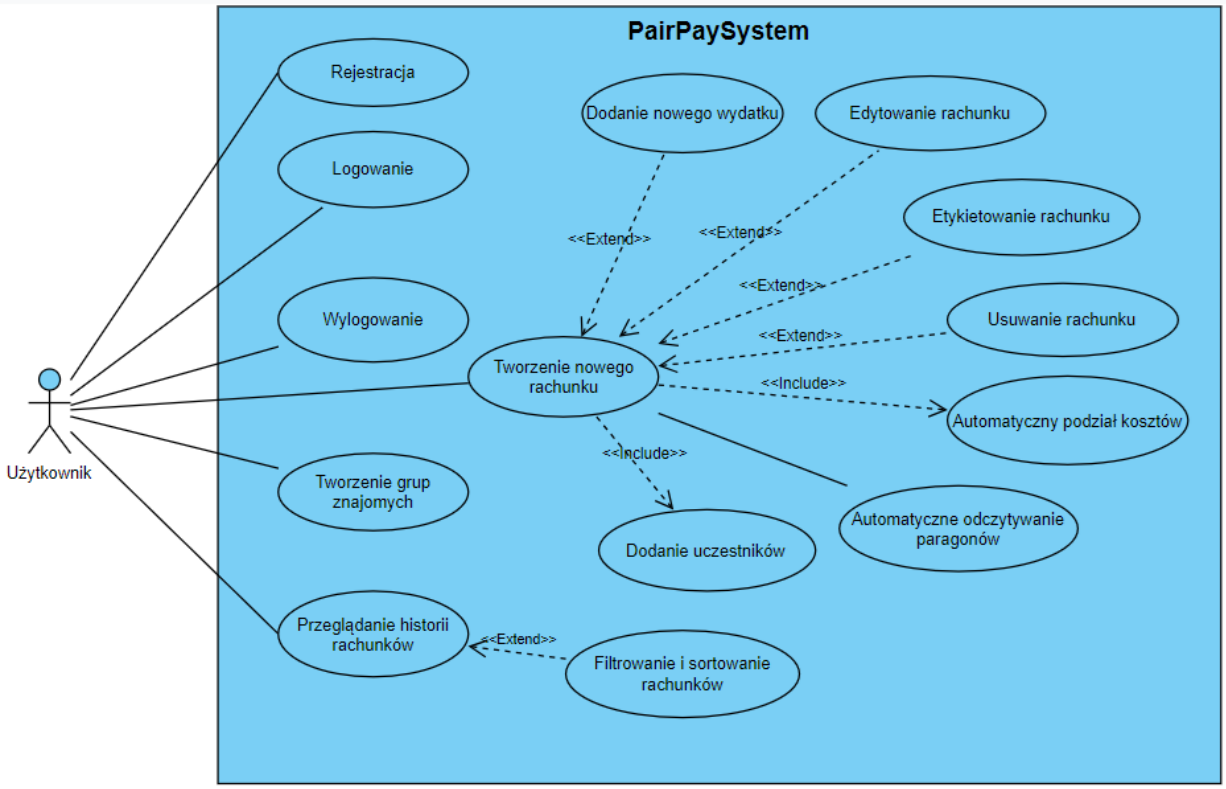
Tabela 2: Opis wymagań niefunkcjonalnych aplikacji

4 Skala MoSCow

Funkcjonalność	**Skala MoSCoW**
F_1	Must
F_2	Must
F_3	Must
F_4	Should
F_5	Could
F_6	Should
F_7	Should
F_8	Should
F_9	Could
F_10	Could
F_11	Could
F_12	Won't
NF_1	Must
NF_2	Must
NF_3	Should
NF_4	Could
NF_5	Must
NF_6	Should

Tabela 3: Klasyfikacja funkcjonalności według skali MoSCoW

5 Diagram przypadków użycia



6 Komponenty systemu

6.1 Backend - REST API

- Technologie: Python z frameworkiem Flask
- Uzasadnienie: Flask jest oparty o język Python, co umożliwia korzystanie z jego licznych zalet. To lekki framework, który umożliwia szybkie tworzenie aplikacji backendowych. Daje pełną elastyczność w projektowaniu różnych rozwiązań. Jest idealny do tworzenia REST API i umożliwia łatwą integrację między frontendem a backendem. Kolejnym ważym argumentem jest jego wsparcie dla OAuth, co pozwala zapewnić bezpieczne logowanie i autoryzację użytkowników. Kolejnym ważnym aspektem jest to, że Flask świetnie współdziała z takimi bibliotekami jak SQLAlchemy czy Tortoise ORM, co znacznie upraszcza operacje na bazie danych i tworzenie powiązań między tabelami. Ogromną zaletą Flaska jest również wsparcie społeczności i rozbudowana dokumentacja, która umożliwi szybkie znalezienie rozwiązań na różne problemy.

6.2 Frontend

- Technologie: TypeScript z frameworkiem React
- Uzasadnienie: React to popularna biblioteka do budowy interfejsów użytkownika, która umożliwia tworzenie dynamicznych i responsywnych aplikacji webowych. Dzięki zastosowaniu komponentów, zarządzanie stanem aplikacji staje się prostsze, a ponowne wykorzystanie kodu znacząco przyspiesza proces developmentu. Wirtualny DOM, oferowany przez React, zwiększa wydajność aplikacji, umożliwiając szybkie aktualizacje interfejsu użytkownika bez nadmiernego obciążania przeglądarki. Dodatkowo, dostępność licznych bibliotek z gotowymi komponentami UI, takich jak NextUI czy Shadcn, umożliwia szybkie i efektywne tworzenie elementów interfejsu, co przyspiesza rozwój aplikacji.

6.3 Baza danych

- Technologie: PostgreSQL
- Uzasadnienie: PostgreSQL jest relacyjną bazą danych, która jest wydajna oraz dobrze skalowalna, potrafi obsłużyć duże ilości danych i skomplikowane zapytania. Posiada również wsparcie dla JSON, co pozwala na przechowywanie danych nieustrukturyzowanych. Kolejnym ważnym aspektem jest aktywna społeczność, co powoduje, że jest wiele zasobów oraz dokumentacji w przypadku pojawienia się jakiegoś problemu. Dodatkowo, PostgreSQL oferuje zaawansowane funkcje bezpieczeństwa, takie jak wielopoziomowe uwierzytelnianie, kontrola dostępu oraz szyfrowanie danych. Te mechanizmy zabezpieczeń chronią przed nieautoryzowanym dostępem i utratą danych, co jest kluczowe w kontekście aplikacji, które przechowują wrażliwe informacje finansowe.

6.4 Development

- Technologie: Git (GitHub) oraz CI/CD z Dockerem (GitHub Actions) ?? do ci/cd mozna napisac ze umozliwia na automatyczne puszczanie testow jednostkowych czy e2e, na sprawdzanie kodu zastosowanie jakis formaterow kodu, budowanie obrazow dockerowych itp... ??TODO docker-compose i zastanowic sie czy to dac

- Uzasadnienie: Umożliwiaja monitorowanie zmian w kodzie, co pozwala na łatwe śledzenie postępów w projekcie oraz ułatwia pracę zespołową poprzez wykorzystanie branchy do izolacji pracy nad różnymi funkcjonalnościami aplikacji, co zmniejsza ryzyko konfliktów w kodzie. GitHub Actions, jako narzędzie wbudowane w GitHub, automatyzuje procesy CI/CD bezpośrednio w repozytoriach i oferuje elastyczność w definiowaniu potoków CI/CD w plikach YAML.

6.5 Diagram komponentów

6.6 Diagram sekwencji

7 Bezpiczeństwo

7.1 Zabezpieczenia logowania i autoryzacji Zarządzanie sesjami

Mechanizmy uwierzytelniania: Rozważ użycie protokołów takich jak OAuth 2.0

7.2 Zabezpieczenia bazy danych

- Wrażliwe dane zostaną zaszyfrowane za pomocą bcrypt (szyfrowanie haseł) i AES (szyfrowanie innych danych wrażliwych),
- Zabezpieczenie przed SQL injection poprzez wykorzystanie ORM zamiast surowych zapytań oraz walidacja danych wejściowych.

7.3 Zarządzanie dostępem

Stworzenie ról typu admin, użytkownik które posiadają odpowiednie uprawnienia.