|  |  |  |  |
| --- | --- | --- | --- |
|  | | | |
|  | |  | **Wydział Informatyki i Telekomunikacji** kierunek studiów: Informatyka  specjalność: brak Praca dyplomowa - inżynierska **Aplikacja webowa wspomagająca organizowanie wycieczek grupowych** Joanna Sroka słowa kluczowe:  1 linia  2 linia  3 linia  krótkie streszczenie:  1 linia  2 linia  3 linia  4 linia   |  |  |  |  |  | | --- | --- | --- | --- | --- | | opiekun pracy  dyplomowej | Dr inż. Michał Szczepanik | | | | | *Tytuł/stopień naukowy/imię i nazwisko* | | | | | Ostateczna ocena za pracę dyplomową | | | | | | Przewodniczący Komisji egzaminu dyplomowego | | ..................................................  *Tytuł/stopień naukowy/imię i nazwisko* | ....................... | ....................... | | *ocena* | *podpis* |   *Do celów archiwalnych pracę dyplomową zakwalifikowano do:\**   1. *kategorii A (akta wieczyste)* 2. *kategorii BE 50 (po 50 latach podlegające ekspertyzie)*   *\* niepotrzebne skreślić*   |  | | --- | | pieczątka wydziałowa | |
|  |  |  | Wrocław, 2021 |

Streszczenie

Celem pracy jest wykonanie aplikacji webowej wspomagającej organizowanie wycieczek grupowych. Aplikacja ma ułatwiać komunikację oraz usprawniać koordynację uczestników wycieczki. Udostępniać będzie możliwość wygodnego podziału wspólnie poniesionych kosztów. Aplikacja ma również zapewniać miejsce do dzielenia się dokumentami podróżnymi, takimi jak bilety, paragony czy rezerwacje. Umożliwi również wymianę informacji o ważnych wydarzeniach w czasie wspólnej podróży. Aplikacja zapewni użytkownikom kompleksowe źródło informacji o wycieczce grupowej, dzięki czemu wspólne podróżowanie stanie się łatwiejsze.

Abstract

Treść streszczenia po angielsku

**Spis treści**

[Wstęp 2](#_Toc534292278)

[1. Tytuł rozdziału 2](#_Toc534292279)

[1.1. Tytuł podrozdziału 2](#_Toc534292280)

[1.2. Tytuł podrozdziału 2](#_Toc534292281)

[1.2.1. Tytuł podpodrozdziału 2](#_Toc534292282)

[1.2.2. Tytuł podpodrozdziału 2](#_Toc534292283)

[Zakończenie 2](#_Toc534292284)

[Bibliografia 2](#_Toc534292285)

[Spis rysunków 2](#_Toc534292286)

[Spis tabel 2](#_Toc534292287)

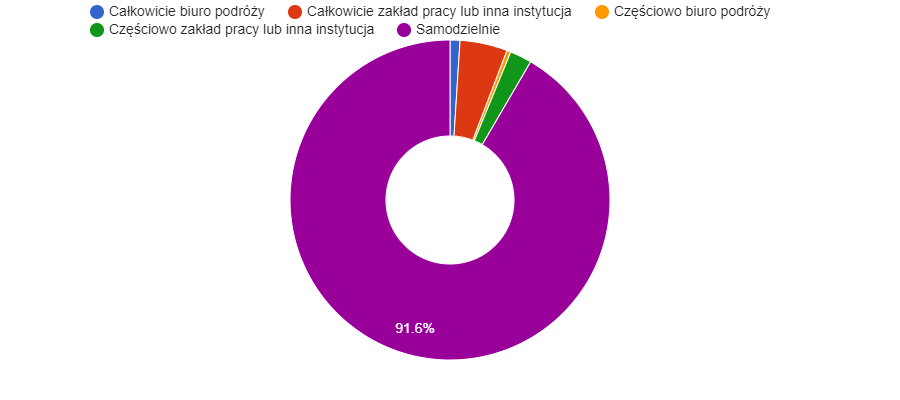
[Załącznik 2](#_Toc534292288)

**Wstęp**

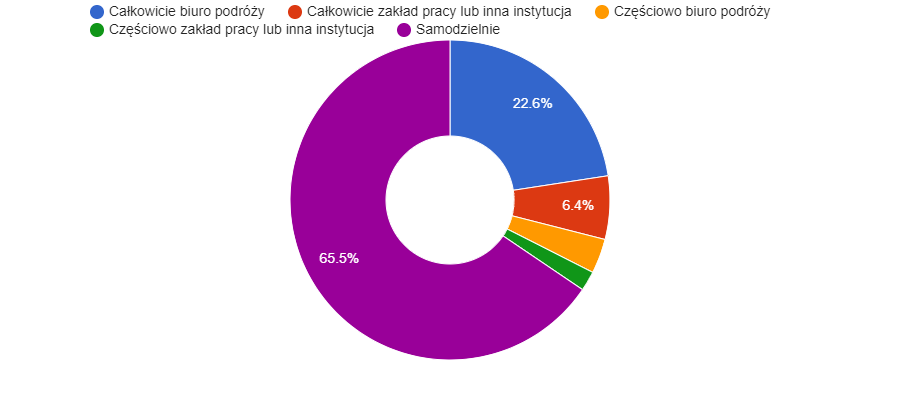
Współcześnie samodzielne organizowanie wycieczek jest najpopularniejszą formą podróżowania. Według Polskiej Organizacji Turystycznej w 2018 roku aż 91,6% podróży krajowych zostało zorganizowanych samodzielnie. Podobnie jest w przypadku podróży zagranicznych, gdzie 65,5% Polaków zdecydowało się na samodzielną organizację wycieczki.

[Dodać do bibliografii 🡪 https://zarabiajnaturystyce.pl/obserwatorium-turystyki/ruch-turystyczny/podroze-polakow/statystyki-i-analizy/sposob-organizacji-krajowych-podrozy-ogolem/]

[https://zarabiajnaturystyce.pl/obserwatorium-turystyki/ruch-turystyczny/podroze-polakow/statystyki-i-analizy/sposob-organizacji-podrozy-zagranicznych/]



[Dodać podpis i źródło 🡪 Sposób organizacji podróży krajowych w 2018 roku (%)]



[Dodać podpis i źródło 🡪 Sposób organizacji podróży zagranicznych w 2018 roku (%)]

Podróżnicy cenią niezależność w decydowanie o sposobie spędzania wolnego czasu w czasie wycieczki, wyboru miejsca noclegu czy też planu zwiedzania. Samodzielna organizacja pozwala na oszczędność pieniędzy oraz dostosowanie planu wycieczki pod własne preferencje.

Podróżnik decydujący się na samodzielną organizację wycieczki musi mierzyć się z wieloma wyzwaniami. Szczególnie problematyczne jest samodzielna organizacja grupowej wycieczki. Jednym z problemów jest zapewnienie sprawnej komunikacji pomiędzy uczestnikami oraz łatwego dostępu do wszystkich ważnych informacji związanych z planowaną podróżą.

Podział wspólnie poniesionych w czasie podróży kosztów również bywa wyzwaniem i wymaga sporego nakładu pracy. Skrupulatne gromadzenie i dokumentowanie paragonów oraz biletów, a także bieżące rozliczanie wydatków z innymi uczestnikami wycieczki bywa uciążliwe i czasochłonne, zwłaszcza w trakcie wakacji, w czasie których odpoczynek staje się priorytetem.

Samodzielna organizacja wycieczek wymaga od podróżników większej dyscypliny i kontroli poniesionych kosztów, zwłaszcza w przypadku wycieczek grupowych. Może to nastręczać wielu trudności związanych z monitorowaniem wydatków, zwłaszcza gdy nie były one równomiernie rozłożone pomiędzy uczestnikami w trakcie wspólnej wycieczki. Analiza wydatków oraz bieżący podział kosztów pomiędzy uczestnikami grupowej wycieczki pozwala usprawnić logistykę wycieczki oraz efektywnie kontrolować środki finansowe w czasie wspólnej podróży. Możliwa jest dzięki temu także prognoza kosztów przyszłych wycieczek oraz podgląd, na co pożytkowany jest budżet.

Kolejnym z istotnych do rozważenia kwestii jest wymiana ważnych dokumentów podróżnych. Uczestnicy w czasie wielodniowych wycieczek mogą posiadać wiele rezerwacji czy biletów i często są one przechowywane w różnych miejscach, co bywa problematyczne w trakcie podróży, a wraz ze wzrostem liczby uczestników wycieczki – coraz trudniejsze.

Łatwy dostęp do Internetu z każdego urządzenia mobilnego oraz stacjonarnego, a także rosnąca popularność samodzielnego organizowanych podróży skłania do korzystania z aplikacji ułatwiających organizację grupowych wycieczek. Powstaje potrzeba posiadania wygodnego narzędzia, które zapewniłoby kompleksowe źródło informacji o wycieczce grupowej, dzięki czemu wspólne podróżowanie stałoby się łatwiejsze.

Celem pracy jest opracowanie aplikacji webowej, która będzie umożliwiać wspomaganie organizowania wycieczek grupowych poprzez możliwość komfortowego podziału wspólnie poniesionych kosztów, zapewnienie łatwo dostępnego miejsca dla wspólnych dokumentów podróżnych, a także ułatwienie wymiany informacji o ważnych wydarzeniach oraz informacjach w czasie wspólnej podróży. Aplikacja ma przede wszystkim usprawnić logistykę wycieczki, uprościć proces rozliczania wydatków podczas podróży wraz z integracją przeliczania walut, a także zapewnienie miejsca do wspólnego dzielenia się szczegółami dotyczącymi planowania wycieczki grupowej.

Aplikacja ma zapewnić kompleksowe źródło informacji o wycieczce grupowej w wygodnej w obsłudze i intuicyjnej formie, co pozwoli cieszyć się wspólnym wyjazdem.

Praca składa się z…

TUTAJ SKRÓT PRACY

We wstępie pracy opisano…

**Analiza rynkowa**

**W analizie rynkowej wyciągnąć wymagania: must-have, nice-to-have oraz te, których nie ma konkurencja.**

Na rynku istnieje wiele rozwiązań ułatwiających organizację wycieczek grupowych. Wiele z nich nie jest skupionych ściśle wokół tematyki podróżowania, jednak posiadają cechy wspomagające samodzielną organizację wyjazdu - takie jak funkcjonalność rozdzielanie należności oraz śledzenia wydatków podczas podróży.

Z dostępnych na rynku rozwiązań zostały wybrane te najbardziej interesujące oraz oferujące największą liczbę możliwości. Niniejsza analiza przedstawia porównanie dostępnych aplikacji webowych, które mogą wspomagać organizację wycieczek grupowych, a także ich wady i zalety oraz podsumowanie.

**Tricount**

Obraz zawierający stół

Opis wygenerowany automatycznie

Obraz zawierający tekst

Opis wygenerowany automatycznie

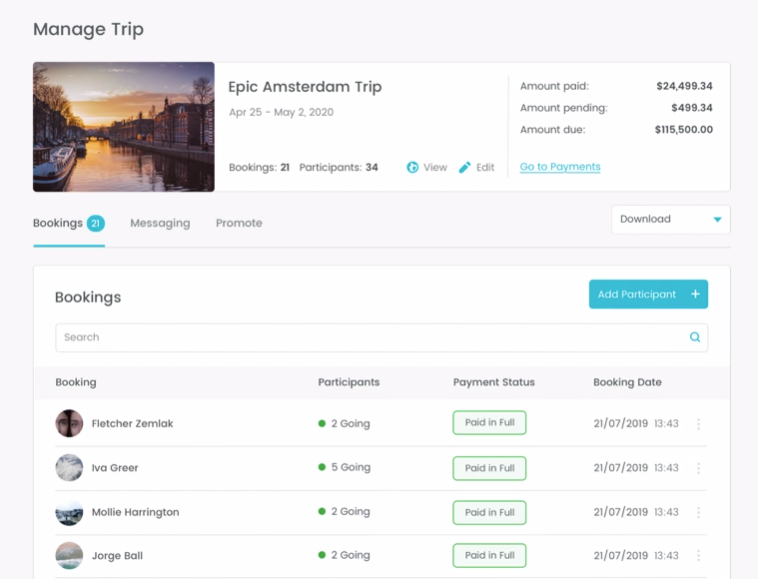
Tricount to aplikacja ułatwiająca organizację wydatków w czasie podróży oraz w życiu codziennym. Aplikacja jest dostępna w wersji webowej oraz na urządzenia mobilne – Android oraz iOS.

Umożliwia wprowadzenie wydatków wraz z określeniem ich kategorii, kwoty oraz osoby płacącej. Ułatwia sprawdzenie salda wydatków oraz usprawnia rozliczanie kosztów – pozwala wykonać obliczenia i ustalić, kto i ile powinien zapłacić. Wspiera przewalutowania, a także możliwość nierównomiernego podziału wydatków oraz eksport wszystkich kosztów do przejrzystego podsumowania w pliku PDF lub CSV.

Aplikacja nie jest jednak kompleksową aplikacją podróżną, ułatwia jedynie rozliczanie wydatków wśród uczestników wyjazdu.

Kolejną wadą aplikacji jest to, że wybraną walutę można ustalić tylko raz dla danej grupy osób. Brak elastycznego wybierania walut dla każdego wydatku jest ogromnym utrudnieniem dla podróżników. Wydłuża to proces rozliczania rachunków w trakcie podróży, ponieważ dla każdej waluty należy utworzyć nową grupę z tymi samymi osobami.

**WeTravel**



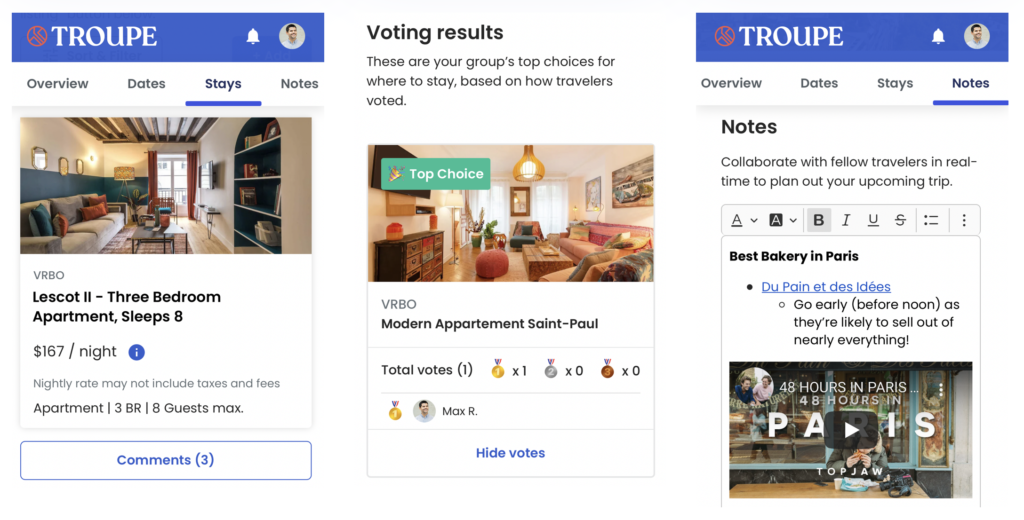
WeTravel jest aplikacją webową wspomagającą organizację komercyjnych wycieczek grupowych. Jest to rozwiązanie usprawniające proces rezerwacji i płatności, skierowane do firm zajmujących się podróżami grupowymi.

Umożliwia łatwe i elastyczne tworzenie stron z grupowymi wycieczkami poprzez korzystanie z gotowych szablonów. Automatyzuje również proces płatności.

WeTravel jest aplikacją płatną, do komercyjnych rozwiązań. Nie ma zastosowania przy organizowaniu prywatnych wycieczek grupowych.

**Troupe**

https://www.maxrosero.com/troupe



Troupe jest aplikacją webową oraz mobilną ułatwiającą planowanie wycieczek grupowych. Aplikacja udostępnia miejsce do przechowywania ważnych terminów związanych ze wspólną wycieczką oraz informacji o miejscu podróży. Posiada funkcjonalność udostępniania istotnych informacji takich jak: szczegółów rezerwacji hotelowych, adresów, numerów lotów oraz odnośników.

Troupe pozwala przechowywać najważniejsze informacje dotyczące wspólnej podróży. Nie posiada jednak funkcjonalności rozliczania wspólnie poniesionych kosztów ani śledzenia grupowych wydatków.

**Podsumowanie analizy rynkowej**

Porównanie konkurencyjnych rozwiązań umożliwiło identyfikację cech aplikacji dostępnych na rynku, które usprawniają organizację grupowych wycieczek. Podczas analizy konkurencyjnych narzędzi zostały zidentyfikowane ich najważniejsze funkcjonalności: możliwość łatwego podziału wspólnie poniesionych kosztów, zapewnienie łatwo dostępnego miejsca dla wspólnych dokumentów podróżnych, a także ułatwienie wymiany informacji o ważnych wydarzeniach oraz informacjach w czasie wspólnej podróży.

Żadne z analizowanych narzędzi nie zapewniło wszystkich wymienionych funkcjonalności. Najlepsze możliwości udostępnia aplikacja Troupe, jednak nie oferuje możliwości podziału wydatków pomiędzy uczestnikami grupowej wycieczki.

Analiza dostępnych rozwiązań umożliwiała określenie obszaru rozwoju aplikacji przeznaczonej dla osób podróżujących w grupie.

**Słownik pojęć**

W słowniku pojęć **NIE tylko** pojęcia domenowe, dorzucić też pojęcia techniczne, skróty, które nie są „oczywiste”, często pojawiają się w pracy, np. retrofit.

|  |  |  |
| --- | --- | --- |
| **Termin** | **Synonimy** | **Definicja terminu** |
| Aktywne konto |  | Konto, które zostało aktywowane przy pomocy linku aktywacyjnego dostarczonego na pocztę e-mail użytkownika. Takie konto ma możliwość logowania do systemu. |
| Autoryzowany użytkownik |  | Użytkownik zalogowany w systemie, posiada aktywne konto. |
| Dokument podróżny | Dokument | Dokument związany z podróżą, potrzebny do podróżowania, taki jak bilet lotniczy czy rezerwacja noclegu w hotelu. |
| Konwersja wydatku |  | Przewalutowanie rachunku na wybraną walutę według aktualnego kursu wymiany. |
| Nieaktywne konto |  | Konto, które nie zostało aktywowane przy pomocy linku aktywacyjnego dostarczonego na pocztę e-mail użytkownika. Takie konto nie ma możliwości logowania do systemu. |
| Nieautoryzowany użytkownik |  | Użytkownik niezalogowany w systemie. Ma możliwość logowania, jeżeli posiada aktywne konto w systemie. |
| Notatka |  | Krótki tekst ułatwiający dzielenie się ważnymi informacjami pomiędzy uczestnikami wycieczki. |
| Podział wydatku |  | Rozdzielenie kwoty rachunku pomiędzy uczestnikami wycieczki równomiernie lub nierównomiernie według procentów, udziałów lub dokładnych kwot. |
| Podział wydatku nierównomiernie |  | Rozdzielenie kwoty rachunku z ustaleniem indywidualnego poziomu spłaty dla każdego uczestnika wycieczki. |
| Podział wydatku równomiernie |  | Rozdzielenie kwoty rachunku po równo pomiędzy uczestnikami wycieczki. |
| Podział wydatku według dokładnych kwot |  | Rozdzielenie kwoty rachunku z ustaleniem dokładnych kwot dla każdego uczestnika wycieczki. |
| Podział wydatku według procentów |  | Rozdzielenie kwoty rachunku z procentowego udziału w wydatku dla każdego uczestnika wycieczki. |
| Podział wydatku według udziałów |  | Rozdzielenie kwoty rachunku z ustaleniem części udziału w wydatku dla każdego uczestnika wycieczki. |
| Właściciel wycieczki | Właściciel | Autoryzowany użytkownik, który dodał wycieczkę do systemu i do niej należy. |
| Wycieczka grupowa | Wycieczka, wyjazd grupowy | Grupa osób biorących udział we wspólnej podróży. |
| Wydatek | Rachunek, opłata | Koszty poniesione w trakcie organizacji oraz w czasie trwania wspólnej podróży, takie jak opłaty za bilety lotnicze, zwiedzanie muzeów, paliwo czy obiady w restauracji. |
| Wydarzenie podróżne | Wydarzenie | Wydarzenie w czasie wspólnej podróży, takie jak wylot z lotniska, zwiedzanie muzeum czy planowany nocleg. Zawiera informacje o dacie, miejscu oraz tematyce wydarzenia. |
| Uczestnik wycieczki | Uczestnik | Autoryzowany użytkownik, który należy do wycieczki. |

**Wymagania funkcjonalne**

1. Zarządzanie wycieczką – przeglądanie, dodawanie oraz edycja wycieczki

Aplikacja ma umożliwiać dokonywanie operacji przeglądania, dodawania oraz edycji wycieczek w ramach których odbywa się wymiana informacji pomiędzy uczestnikami.

1. Zarządzanie uczestnikami wycieczki – przeglądanie, dodawanie oraz usuwanie uczestnika

Aplikacja ma oferować funkcje przeglądania, dodawania oraz usuwania uczestnika z wycieczki, aby ułatwić komunikację oraz usprawnić logistykę wycieczki.

1. Zarządzanie wydatkami – przeglądanie, dodawanie, edycja oraz usuwanie wydatku

Aplikacja ma umożliwiać dokonanie wygodnego podziału wspólnie poniesionych kosztów pomiędzy uczestnikami grupowej wycieczki. System pozwala na dodanie, edycję, przeglądanie oraz usuwanie wydatków. Aplikacja ma oferować możliwość podziału rachunku równomiernie lub nierównomiernie według procentów, udziałów lub dokładnych kwot. Pozwoli to na komfortowe i proste rozdzielenie należności oraz śledzenie skategoryzowanych wydatków dokonanych w czasie wspólnej podróży. Aplikacja wspiera także konwersję kosztów na wybrane waluty według aktualnego kursu wymiany.

1. Zarządzanie dokumentami podróżnymi – przeglądanie, dodawanie, edycja oraz usuwanie dokumentu podróżnego

Aplikacja ma zapewniać miejsce do dzielenia się dokumentami podróżnymi, takimi jak bilety czy rezerwacje. System musi umożliwiać dokonanie operacji przeglądania, dodawania, edycji oraz usuwania dokumentów podróżnych różnego typu należących do wycieczki grupowej.

1. Zarządzanie wydarzeniami podróżnymi – przeglądanie, dodawanie, edycja oraz usuwanie wydarzeń podróżniczych

Aplikacja ma umożliwić wymianę informacji o ważnych wydarzeniach w czasie wspólnej podróży, takich jak rezerwacja lotu, termin zwiedzania muzeum czy planowany nocleg, poprzez przeglądanie, dodawanie, edycję oraz usuwanie wydarzeń. Zapewni to dostęp do najważniejszych informacji – takich jak daty, miejsca oraz tematyki wydarzeń, co pozwoli na efektywne śledzenie nadchodzących planów.

1. Zarządzanie notatkami – przeglądanie, dodawanie, edycja oraz usuwanie notatek

Aplikacja ma usprawnić komunikację poprzez dzielenie się notatkami pomiędzy uczestnikami wycieczki. Aplikacja ma zapewnić miejsce do przeglądania, dodawania, edycji oraz usuwania notatek dotyczących wspólnej podróży.

**Wymagania niefunkcjonalne**

1. Responsywność aplikacji – obsługa urządzeń o różnych rozmiarach ekranu.

Aplikacja musi być responsywna, czyli odpowiednio reagować na różne rozmiary i proporcje ekranów poprzez dostosowanie elementów. Pozwala to na zachowanie czytelności aplikacji oraz uniknięcie ewentualnych nieprawidłowości w wyświetlaniu jej treści. Responsywność pozwala użytkownikom na korzystanie z aplikacji webowej na różnych urządzeniach mobilnych oraz stacjonarnych.

1. Bezpieczeństwo aplikacji.

Aplikacja ma dostarczać rozwiązania, które zapewnią ochronę danych użytkowników poprzez bezpieczny mechanizm uwierzytelniania i autoryzacji, a także ochronę przed typowymi atakami, takimi jak SQL injection lub fałszowanie żądań między witrynami.

1. Wsparcie dla przeglądarek internetowych Microsoft Edge, Google Chrome, Firefox oraz Safari.

Aplikacja działa poprawnie na przeglądarkach internetowych: Microsoft Edge w wersji 95 i nowszej, Chrome w wersji 95 i nowszej, Firefox w wersji 94 i nowszej oraz Safari w wersji 15 i nowszej.

1. Działanie w dwóch językach – polskim oraz angielskim.

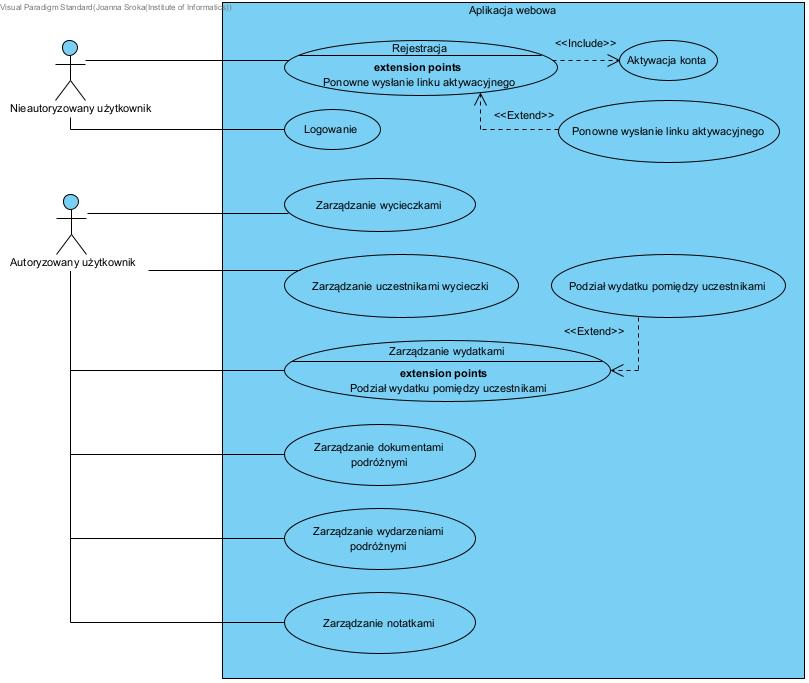
Aplikacja jest dostępna w dwóch wersjach językowych – polskiej oraz angielskiej. Pozwala to aplikacji na dotarcie do większej liczby użytkowników oraz uwzględnia potrzeby wielojęzycznych grup wycieczkowych.

**Przypadki użycia**

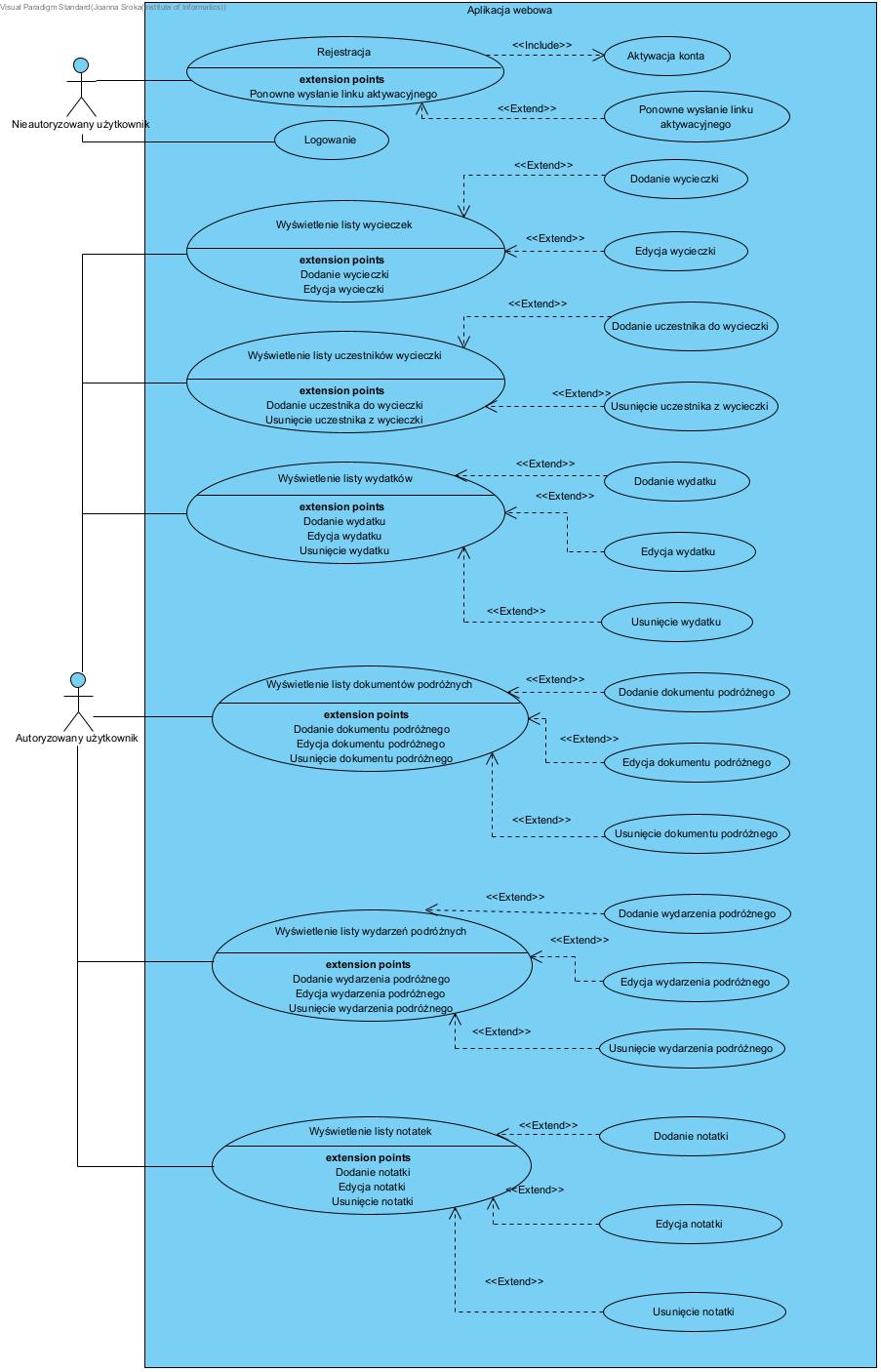
Przypadek użycia, zgodnie ze specyfikacją języka UML [Do biblografii dodać 🡪 https://www.omg.org/spec/UML/2.5/PDF], jest to graficzny sposób uchwycenia wymagań systemu. Reprezentuje on interakcję pomiędzy aktorem a systemem.

**Diagram przypadków użycia**

Przypadki użycia aplikacji zostały zdefiniowane w oparciu o opracowane wymagania funkcjonalne. Rysunek X.Y przedstawia diagram przypadków użycia w języku UML.



LUB TEN:



Dodać podpisy 🡪 Rysunek X.Y Diagram przypadków użycia. Źródło: opracowanie własne

**Opisy streszczające**

|  |  |
| --- | --- |
| **Przypadek użycia** | **Opis** |
| Aktywacja konta | Nieautoryzowany użytkownik dokonuje aktywacji konta w systemie poprzez ustawienie hasła w formularzu aktywacyjnym, do którego dostęp otrzymał na swoją pocztę e-mail. |
| Logowanie | Nieautoryzowany użytkownik dokonuje autoryzacji w systemie poprzez podanie adresu e-mail i hasła w celu nadaniu mu określonych uprawnień. |
| Podział wydatku pomiędzy uczestnikami | Uczestnik wycieczki dokonuje podziału wydatku pomiędzy uczestnikami wycieczki równomiernie lub nierównomiernie według procentów, udziałów lub dokładnych kwot. |
| Ponowne wysłanie linku aktywacyjnego | Nieautoryzowany użytkownik żąda ponownego wysłania linku aktywującego konto na adres e-mail. |
| Rejestracja | Nieautoryzowany użytkownik zakłada konto w systemie poprzez wprowadzenie danych osobowych oraz adresu e-mail. |
| Zarządzanie dokumentami podróżnymi | Dodawanie, przeglądanie, edycja oraz usuwanie dokumentów podróżnych należących do wycieczki przez uczestnika wycieczki. |
| Zarządzanie notatkami | Dodawanie, przeglądanie, edycja oraz usuwanie notatek należących do wycieczki przez uczestnika wycieczki. |
| Zarządzanie uczestnikami wycieczki | Dodawanie, przeglądanie oraz usuwanie uczestnika z wycieczki przez uczestnika wycieczki. |
| Zarządzanie wycieczkami | Dodawanie, przeglądanie oraz edycja wycieczek przez autoryzowanego użytkownika. |
| Zarządzanie wydarzeniami podróżnymi | Dodawanie, przeglądanie, edycja oraz usuwanie wydarzeń podróżnych należących do wycieczki przez uczestnika wycieczki. |
| Zarządzanie wydatkami | Dodawanie, przeglądanie, edycja oraz usuwanie wydatków należących do wycieczki przez uczestnika wycieczki. |

**Scenariusze przypadków użycia**

|  |  |
| --- | --- |
| **Nazwa przypadku użycia** | Logowanie |
| **Aktorzy** | Nieautoryzowany użytkownik |
| **Opis** | Przypadek użycia opisuje funkcjonalność autoryzacji użytkownika w systemie poprzez podanie adresu e-mail oraz hasła. |
| **Warunki wstępne** | Użytkownik nie jest zalogowany w systemie. |
| **Warunki końcowe** | Użytkownik zostaje zautoryzowany w systemie i otrzymuje określone uprawnienia. |
| **Rezultat** | Użytkownik jest zalogowany w systemie. |
| **Punkty rozszerzeń** | - |
| **Scenariusz główny –** logowanie do systemu | 1. Użytkownik chce zalogować się w systemie.  2. System przedstawia okno logowania umożliwiające wprowadzenie danych uwierzytelniających – adresu e-mail oraz hasła.  3. Użytkownik wprowadza dane uwierzytelniające.  4. System stwierdza poprawność wprowadzonych danych uwierzytelniających.  5. Użytkownik zostaje zalogowany do systemu. |
| **Scenariusz wyjątku 1 –** niepoprawne dane uwierzytelniające | 4.1 System stwierdza niepoprawność wprowadzonych danych uwierzytelniających.  4.1.1 System informuje użytkownika o niepoprawności danych uwierzytelniających.  Wejście do kroku 2. scenariusza głównego. |
| **Scenariusz wyjątku 2 –** zablokowanie możliwości logowania | 4.2 System stwierdza, że możliwość logowania dla podanego adresu e-mail jest zablokowana.  4.2.1 System informuje użytkownika o braku możliwości logowania dla podanego adresu e-mail przez okres 1 godziny.  Wejście do kroku 2. scenariusza głównego. |

|  |  |
| --- | --- |
| **Nazwa przypadku użycia** | Rejestracja |
| **Aktorzy** | Nieautoryzowany użytkownik |
| **Opis** | Użytkownik zakłada konto w systemie, aby móc korzystać z funkcjonalności systemu. |
| **Warunki wstępne** | Użytkownik nie jest zalogowany w systemie. |
| **Warunki końcowe** | Użytkownik posiada aktywne konto w systemie. |
| **Rezultat** | Konto dla użytkownika zostaje założone oraz aktywowane. |
| **Punkty rozszerzeń** | Ponowne wysłanie linku aktywacyjnego |
| **Scenariusz główny –** rejestracja w systemie | 1. Użytkownik chce założyć konto w systemie.  2. System przedstawia formularz rejestracji umożliwiający wprowadzanie danych osobowych oraz adresu e-mail, który będzie identyfikował użytkownika.  3. Użytkownik wprowadza dane osobowe oraz adres e-mail.  4. System stwierdza poprawność wprowadzonych danych oraz unikalność adresu e-mail w systemie.  5. System zakłada nieaktywne konto w systemie i wysyła na podany przez użytkownika adres e-mail link aktywacyjny.  6. Należy wywołać przypadek użycia: „Aktywacja konta” w celu aktywacji założonego konta i dokończenia rejestracji. |
| **Scenariusz alternatywny** – ponowne wysłanie linku aktywacyjnego | 2.1 Użytkownik chce otrzymać nowy link aktywacyjny w celu aktywacji konta.  2.1.1 Należy wywołać przypadek użycia „Ponowne wysłanie linku aktywacyjnego”. |
| **Scenariusz wyjątku** – podany e-mail istnieje w bazie systemu | 4.1 System stwierdza, że podany przez użytkownika e-mail już istnieje w systemie.  4.1.1 System informuje użytkownika o niemożliwości założenia konta, ponieważ w systemie istnieje konto z podanym e-mailem.  Wejście do kroku 2. scenariusza głównego. |

|  |  |
| --- | --- |
| **Nazwa przypadku użycia** | Aktywacja konta |
| **Aktorzy** | Nieautoryzowany użytkownik |
| **Opis** | Przypadek użycia opisuje funkcjonalność aktywacji konta w systemie poprzez ustawienie hasła w formularzu aktywacyjnym. |
| **Warunki wstępne** | Użytkownik posiada nieaktywne konto w systemie.  Może być wywołany z przypadku użycia: „Rejestracja”. |
| **Warunki końcowe** | Zmiany dotyczące konta użytkownika, jego aktywacja oraz ustawione hasło, są zatwierdzone w systemie. |
| **Rezultat** | Rejestracja konta użytkownika zostaje zakończona, a jego konto zostaje aktywowane. |
| **Punkty rozszerzeń** | - |
| **Scenariusz główny –** aktywacja konta | 1. Użytkownik chce aktywować konto, aby zakończyć rejestrację i móc zalogować się do systemu.  2. Użytkownik wybiera link aktywacyjny, do którego dostęp otrzymał w wiadomości e-mail.  3. System stwierdza ważność linku aktywacyjnego użytkownika.  4. System przedstawia formularz aktywacji konta umożliwiający wprowadzenie hasła oraz zatwierdzenie aktywacji.  5. Użytkownik wprowadza hasło.  6. Użytkownik zatwierdza aktywację konta.  7. System zapisuje zmiany – aktywuje konto użytkownika oraz zapisuje wprowadzone przez użytkownika hasło. |
| **Scenariusz wyjątku –** niepoprawny link aktywacyjny | 3.1 System stwierdza nieważność linku aktywacyjnego użytkownika.  3.1.1 System informuje użytkownika o nieważnym linku aktywacyjnym oraz o możliwości wysłania nowego linku na adres e-mail. |

|  |  |
| --- | --- |
| **Nazwa przypadku użycia** | Ponowne wysłanie linku aktywacyjnego |
| **Aktorzy** | Nieautoryzowany użytkownik |
| **Opis** | Przypadek użycia opisuje funkcjonalność ponownego wysłania linku aktywacyjnego. |
| **Warunki wstępne** | Użytkownik posiada nieaktywne konto w systemie.  Może być wywołany z przypadku użycia: „Rejestracja”. |
| **Warunki końcowe** | Nowy link aktywacyjny zostaje wysłany na adres e-mail nieaktywnego użytkownika. |
| **Rezultat** | Użytkownik otrzymuje nowy link aktywacyjny do aktywacji konta w systemie. |
| **Punkty rozszerzeń** | - |
| **Scenariusz główny –** ponowne wysłanie linku aktywacyjnego | 1. Użytkownik chce otrzymać nowy link aktywacyjny, aby aktywować konto i zakończyć rejestrację.  2. System wyświetla okno umożliwiające wprowadzenie adresu e-mail, na który ma zostać wysłany link nowy link aktywacyjny.  3. Użytkownik wprowadza adres e-mail.  4. Użytkownik zatwierdza wysłanie nowego linku aktywacyjnego.  5. System stwierdza poprawność wprowadzonego adresu e-mail.  6. System zapisuje nowy link aktywacyjny dla użytkownika oraz wysyła go na podany adres e-mail. |
| **Scenariusz wyjątku –** niepoprawny adresu email użytkownika | 4.1 System stwierdza, że nie istnieje nieaktywne konto dla podanego adresu e-mail.  4.1.1 System informuje użytkownika o niepoprawnym adresie e-mail.  Wejście do kroku 2. scenariusza głównego. |

|  |  |
| --- | --- |
| **Nazwa przypadku użycia** | Zarządzanie wycieczkami |
| **Aktorzy** | Autoryzowany użytkownik |
| **Opis** | Przypadek użycia opisuje funkcjonalność zarządzania wycieczkami, tj. przeglądania, dodawania lub edycji wycieczki. |
| **Warunki wstępne** | Użytkownik jest zalogowany do systemu. |
| **Warunki końcowe** | Zmiany dotyczące wycieczek są zatwierdzone w systemie. |
| **Rezultat** | Wprowadzenie do systemu zmian dotyczących wycieczek użytkownika. |
| **Punkty rozszerzeń** | - |
| **Scenariusz główny –** przeglądanie istniejących wycieczek | 1. Użytkownik chce przeglądać wycieczki, do których należy.  2. System wyświetla listę zawierającą wycieczki użytkownika.  3. Użytkownik przegląda dane dotyczące wycieczek. |
| **Scenariusz alternatywny 1 –** dodanie nowej wycieczki | 3.1 Użytkownik dodaje nową wycieczkę.  3.1.1 Użytkownik wybiera opcję reprezentującą dodanie nowej wycieczki.  3.1.2 System wyświetla formularz umożliwiający wprowadzenie danych wycieczki.  3.1.3 Użytkownik wprowadza dane wycieczki.  3.1.4 Użytkownik zatwierdza dodanie nowej wycieczki.  3.1.5 System stwierdza poprawność wprowadzonych danych nowej wycieczki.  3.1.6 System zapisuje nową wycieczkę.  Wejście do kroku 2. scenariusza głównego. |
| **Scenariusz alternatywny 2 –** edycja istniejącej wycieczki | 3.2 Użytkownik edytuje istniejącą wycieczkę.  3.2.1 Użytkownik wybiera wycieczkę z listy wycieczek.  3.2.2 System umożliwia edycję danych wycieczki.  3.2.3 Użytkownik edytuje dane wycieczki.  3.2.4 System stwierdza poprawność wprowadzonych danych edytowanej wycieczki.  3.2.5 System zapisuje zmiany.  Wejście do kroku 2. scenariusza głównego. |
| **Scenariusz wyjątku 1 –** niepoprawne dane nowej wycieczki | 3.1.5.1 System stwierdza niepoprawność danych nowej wycieczki.  3.1.5.1.1 System informuje użytkownika o niepoprawności danych wraz z dokładną informacją o przyczynie niepowodzenia operacji dodania nowej wycieczki.  Wejście do kroku 3.1.2. scenariusza alternatywnego 1. |
| **Scenariusz wyjątku 2 –** niepoprawna modyfikacja istniejącej wycieczki | 3.2.4.1 System stwierdza niepoprawność danych edytowanej wycieczki.  3.2.4.1.1 System informuje użytkownika o niepoprawności modyfikacji wraz z dokładną informacją o przyczynie niepowodzenia modyfikacji istniejącej wycieczki.  Wejście do kroku 3.2.2. scenariusza alternatywnego 2. |

|  |  |
| --- | --- |
| **Nazwa przypadku użycia** | Zarządzanie uczestnikami wycieczki |
| **Aktorzy** | Autoryzowany użytkownik |
| **Opis** | Przypadek użycia opisuje funkcjonalność zarządzania uczestnikami wycieczki, tj. przeglądania, dodawania lub usuwania uczestnika z wycieczki. |
| **Warunki wstępne** | Użytkownik jest zalogowany do systemu.  Użytkownik należy do wycieczki. |
| **Warunki końcowe** | Zmiany dotyczące uczestników wycieczki są zatwierdzone w systemie. |
| **Rezultat** | Wprowadzenie do systemu zmian dotyczących uczestników wycieczki. |
| **Punkty rozszerzeń** | - |
| **Scenariusz główny –** przeglądanie uczestników wycieczki | 1. Użytkownik chce przeglądać uczestników wycieczki, do której należy.  2. System wyświetla listę uczestników wycieczki.  3. Użytkownik przegląda dane dotyczące uczestników wycieczki. |
| **Scenariusz alternatywny 1 –** dodanie uczestnika do wycieczki | 3.1 Użytkownik dodaje uczestnika do wycieczki.  3.1.1 Użytkownik wybiera opcję reprezentującą dodanie uczestnika do wycieczki.  3.1.2 System wyświetla okno umożliwiające wprowadzenie adresu e-mail uczestnika do dodania.  3.1.3 Użytkownik wprowadza adres e-mail użytkownika, który ma zostać uczestnikiem wycieczki.  3.1.4 System stwierdza poprawność wprowadzonych danych nowego uczestnika.  3.1.5 System zapisuje zmiany.  Wejście do kroku 2. scenariusza głównego. |
| **Scenariusz alternatywny 2 –** usunięcie uczestnika wycieczki | 3.2 Użytkownik usuwa uczestnika z wycieczki.  3.2.1 Użytkownik wybiera uczestnika z listy uczestników wycieczki.  3.2.2 Użytkownik wybiera opcję reprezentującą usunięcie wybranego uczestnika wycieczki.  3.2.3 System przedstawia komunikat wymagający potwierdzenia usunięcia uczestnika w celu uniknięcia przypadkowej utraty danych. 3.2.4 Po potwierdzeniu usunięcia uczestnika, system stwierdza poprawność operacji usunięcia i usuwa uczestnika z wycieczki.  Wejście do kroku 2. scenariusza głównego. |
| **Scenariusz wyjątku 1 –** niepoprawne danych nowego uczestnika wycieczki | 3.1.4.1 System stwierdza niepoprawność danych nowego uczestnika wycieczki – użytkownik nie istnieje w systemie lub już należy do danej wycieczki.  3.1.4.1.1 System informuje użytkownika o niepoprawności danych wraz z dokładną informacją o przyczynie niepowodzenia operacji dodania nowego uczestnika do wycieczki.  Wejście do kroku 3.1.2. scenariusza alternatywnego 1. |
| **Scenariusz wyjątku 2 –** niepoprawna operacja usuwania uczestnika wycieczki | 3.2.4.1 System stwierdza niepoprawność operacji usunięcia uczestnika wycieczki – uczestnik posiada nierozliczone wydatki w ramach wycieczki.  3.2.4.1.1 System informuje użytkownika o niepoprawności operacji usuwania wraz z dokładną informacją o przyczynie niepowodzenia.  Wejście do kroku 2. scenariusza głównego. |

|  |  |
| --- | --- |
| **Nazwa przypadku użycia** | Zarządzanie wydatkami |
| **Aktorzy** | Autoryzowany użytkownik |
| **Opis** | Przypadek użycia opisuje funkcjonalność zarządzania wydatkami, tj. przeglądania, dodawania, edycji lub usuwania wydatku. |
| **Warunki wstępne** | Użytkownik jest zalogowany do systemu.  Użytkownik należy do wycieczki. |
| **Warunki końcowe** | Zmiany dotyczące wydatków są zatwierdzone w systemie. |
| **Rezultat** | Wprowadzenie do systemu zmian dotyczących wydatków należących do wycieczki. |
| **Punkty rozszerzeń** | Podział wydatku pomiędzy uczestnikami |
| **Scenariusz główny –** przeglądanie istniejących wydatków | 1. Użytkownik chce przeglądać wydatki należące do wycieczki.  2. System wyświetla listę wydatków.  3. Użytkownik przegląda dane dotyczące wydatków. |
| **Scenariusz alternatywny 1 –** dodanie nowego wydatku | 3.1 Użytkownik dodaje nowy wydatek.  3.1.1 Użytkownik wybiera opcję reprezentującą dodanie nowego wydatku.  3.1.2 System wyświetla formularz umożliwiający wprowadzenie danych wydatku.  3.1.3 Użytkownik wprowadza dane wydatku.  3.1.4 Użytkownik zatwierdza dodanie nowego wydatku.  3.1.5 System stwierdza poprawność wprowadzonych danych.  3.1.6 System zapisuje nowy wydatek.  Wejście do kroku 2. scenariusza głównego. |
| **Scenariusz alternatywny 2 –** edycja istniejącego wydatku | 3.2 Użytkownik edytuje istniejący wydatek.  3.2.1 Użytkownik wybiera wydatek z listy wydatków.  3.2.2 System umożliwia edycję danych wydatku.  3.2.3 Użytkownik edytuje dane wydatku.  3.2.4 System stwierdza poprawność wprowadzonych danych.  3.2.5 System zapisuje zmiany.  Wejście do kroku 2. scenariusza głównego. |
| **Scenariusz alternatywny 3 -** usunięcie wydatku | 3.3 Użytkownik usuwa wydatek.  3.3.1 Użytkownik wybiera wydatek z listy wydatków wycieczki.  3.3.2 Użytkownik wybiera opcję reprezentującą usunięcie wybranego wydatku.  3.3.3 System przedstawia komunikat wymagający potwierdzenia usunięcia wydatku w celu uniknięcia przypadkowej utraty danych. 3.3.4 Po potwierdzeniu usunięcia wydatku, system stwierdza poprawność operacji usunięcia i usuwa wydatek z wycieczki.  Wejście do kroku 2. scenariusza głównego. |
| **Scenariusz alternatywny** **4** – podział wydatku pomiędzy uczestnikami | 3.4 Użytkownik chce podzielić wydatek pomiędzy uczestnikami wycieczki.  3.4.1 Należy wywołać przypadek użycia „Podział wydatku pomiędzy uczestnikami”. |
| **Scenariusz wyjątku 1 –** niepoprawne dane nowego wydatku | 3.1.5.1 System stwierdza niepoprawność danych nowego wydatku.  3.1.5.1.1 System informuje użytkownika o niepoprawności danych wraz z dokładną informacją o przyczynie niepowodzenia operacji dodania nowego wydatku.  Wejście do kroku 3.1.2. scenariusza alternatywnego 1. |
| **Scenariusz wyjątku 2 –** niepoprawna modyfikacja istniejącego wydatku | 3.2.4.1 System stwierdza niepoprawność danych edytowanego wydatku.  3.2.4.1.1 System informuje użytkownika o niepoprawności modyfikacji wraz z dokładną informacją o przyczynie niepowodzenia modyfikacji istniejącego wydatku.  Wejście do kroku 3.2.2. scenariusza alternatywnego 2. |
| **Scenariusz wyjątku 3 -** niepoprawna operacja usuwania wydatku | 3.3.4.1 System stwierdza niepoprawność operacji usunięcia wydatku – wydatek nie został oznaczony jako rozliczony.  3.3.4.1.1 System informuje użytkownika o niepoprawności operacji usuwania wraz z dokładną informacją o przyczynie niepowodzenia.  Wejście do kroku 2. scenariusza głównego. |

|  |  |
| --- | --- |
| **Nazwa przypadku użycia** | Podział wydatku pomiędzy uczestnikami |
| **Aktorzy** | Autoryzowany użytkownik |
| **Opis** | Przypadek użycia opisuje funkcjonalność podziału wydatku pomiędzy uczestnikami wycieczki równomiernie lub nierównomiernie: według procentów, udziałów lub dokładnych kwot. |
| **Warunki wstępne** | Użytkownik jest zalogowany do systemu.  Użytkownik należy do wycieczki. |
| **Warunki końcowe** | Zmiany dotyczące wydatku są zatwierdzone w systemie. |
| **Rezultat** | Wprowadzenie do systemu zmian dotyczących podziału wydatku pomiędzy uczestnikami wycieczki. |
| **Punkty rozszerzeń** | - |
| **Scenariusz główny –** równomierny podział wydatku | 1. Użytkownik chce podzielić wydatek pomiędzy uczestnikami wycieczki.  2. System wyświetla listę wszystkich uczestników wycieczki.  3. Użytkownik wybiera uczestników, których dotyczy wydatek.  4. System wyświetla okno umożliwiające dokonanie podziału wydatku.  5. Użytkownik wybiera opcję reprezentującą równomierny podział wydatku pomiędzy wybranymi uczestnikami.  6. System stwierdza poprawność wprowadzonych danych.  7. System zapisuje zmiany dotyczące podziału wydatku. |
| **Scenariusz alternatywny 1 –** podział wydatku według procentów | 5.1 Użytkownik chce podzielić wydatek według procentów.  5.1.1 Użytkownik wybiera opcję reprezentującą podział wydatku według procentów.  5.1.2 System wyświetla okno umożliwiające wpisanie procentowego udziału w rachunku dla każdego uczestnika.  Wejście do kroku 6. scenariusza głównego. |
| **Scenariusz alternatywny 2 –** podział wydatku według udziałów | 5.2 Użytkownik chce podzielić wydatek według udziałów.  5.2.1 Użytkownik wybiera opcję reprezentującą podział wydatku według udziałów.  5.2.2 System wyświetla okno umożliwiające wpisanie udziału w rachunku dla każdego uczestnika.  Wejście do kroku 6. scenariusza głównego. |
| **Scenariusz alternatywny 3 –** podział wydatku według dokładnych kwot | 5.3 Użytkownik chce podzielić wydatek według dokładnych kwot.  5.3.1 Użytkownik wybiera opcję reprezentującą podział wydatku według dokładnych kwot.  5.3.2 System wyświetla okno umożliwiające wpisanie dokładnych kwot dla każdego uczestnika.  Wejście do kroku 6. scenariusza głównego. |
| **Scenariusz wyjątku –** niepoprawny podział wydatku | 6.1 System stwierdza niepoprawny podział wydatku pomiędzy wybranymi uczestnikami wycieczki.  6.1.1 System informuje użytkownika o niepoprawności operacji podziału wydatku wraz z dokładną informacją o przyczynie niepowodzenia.  Wejście do kroku 4. scenariusza głównego. |

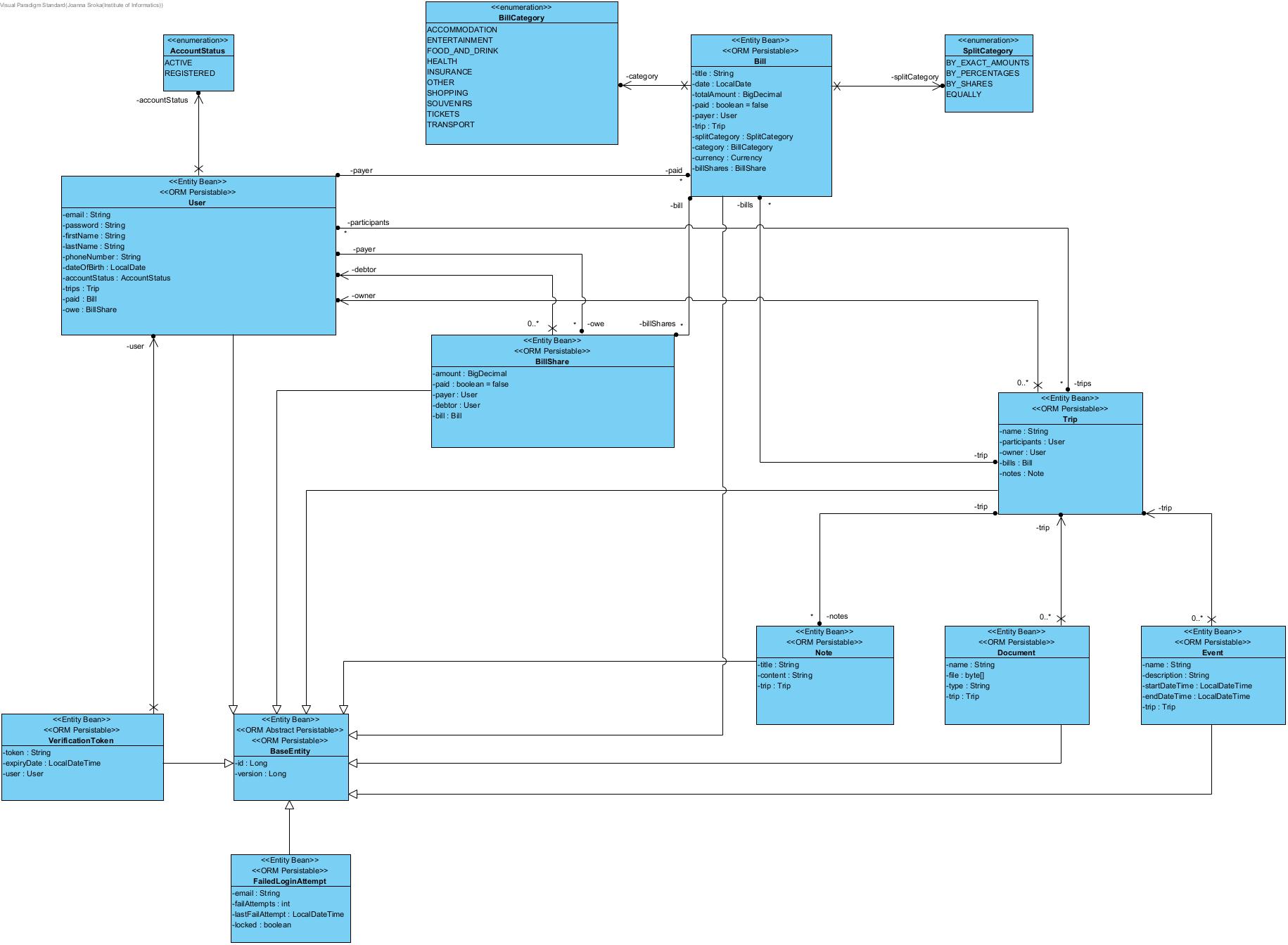
|  |  |
| --- | --- |
| **Nazwa przypadku użycia** | Zarządzanie dokumentami podróżnymi |
| **Aktorzy** | Autoryzowany użytkownik |
| **Opis** | Przypadek użycia opisuje funkcjonalność zarządzania dokumentami podróżnymi, tj. przeglądania, dodawania, edycji lub usuwania dokumentu. |
| **Warunki wstępne** | Użytkownik jest zalogowany do systemu.  Użytkownik należy do wycieczki. |
| **Warunki końcowe** | Zmiany dotyczące dokumentów podróżnych są zatwierdzone w systemie. |
| **Rezultat** | Wprowadzenie do systemu zmian dotyczących dokumentów podróżnych należących do wycieczki. |
| **Punkty rozszerzeń** | - |
| **Scenariusz główny –** przeglądanie istniejących dokumentów podróżnych | 1. Użytkownik chce przeglądać dokumenty podróżne należące do wycieczki.  2. System wyświetla listę zawierającą dokumenty podróżne.  3. Użytkownik przegląda dane dotyczące dokumentów podróżnych. |
| **Scenariusz alternatywny 1 –** dodanie nowego dokumentu podróżnego | 3.1 Użytkownik dodaje nowy dokument podróżny.  3.1.1 Użytkownik wybiera opcję reprezentującą dodanie nowego dokumentu.  3.1.2 System wyświetla formularz umożliwiający dodanie dokumentu.  3.1.3 Użytkownik zatwierdza dodanie nowego dokumentu.  3.1.4 System zapisuje nowy dokument podróżny.  Wejście do kroku 2. scenariusza głównego. |
| **Scenariusz alternatywny 2 –** edycja istniejącego dokumentu podróżnego | 3.2 Użytkownik edytuje istniejący dokument podróżny.  3.2.1 Użytkownik wybiera dokument z listy dokumentów podróżnych.  3.2.2 System umożliwia edycję danych dokumentu.  3.2.3 Użytkownik edytuje dane dokumentu.  3.2.4 System zapisuje zmiany.  Wejście do kroku 2. scenariusza głównego. |
| **Scenariusz alternatywny 3 –** usunięcie dokumentu podróżnego | 3.2 Użytkownik usuwa dokument podróżny.  3.2.1 Użytkownik wybiera dokument z listy dokumentów podróżnych.  3.2.2 Użytkownik wybiera opcję reprezentującą usunięcie wybranego dokumentu.  3.2.3 System przedstawia komunikat wymagający potwierdzenia usunięcia dokumentu w celu uniknięcia przypadkowej utraty danych. 3.2.4 Po potwierdzeniu usunięcia dokumentu podróżnego, system usuwa dokument.  Wejście do kroku 2. scenariusza głównego. |

|  |  |
| --- | --- |
| **Nazwa przypadku użycia** | Zarządzanie wydarzeniami podróżnymi |
| **Aktorzy** | Autoryzowany użytkownik |
| **Opis** | Przypadek użycia opisuje funkcjonalność zarządzania wydarzeniami podróżnymi, tj. przeglądania, dodawania, edycji lub usuwania wydarzenia. |
| **Warunki wstępne** | Użytkownik jest zalogowany do systemu.  Użytkownik należy do wycieczki. |
| **Warunki końcowe** | Zmiany dotyczące wydarzeń podróżnych są zatwierdzone w systemie. |
| **Rezultat** | Wprowadzenie do systemu zmian dotyczących wydarzeń podróżnych należących do wycieczki. |
| **Punkty rozszerzeń** | - |
| **Scenariusz główny –** przeglądanie istniejących dokumentów podróżnych | 1. Użytkownik chce przeglądać wydarzenia podróżne należące do wycieczki.  2. System wyświetla listę zawierającą wydarzenia podróżne.  3. Użytkownik przegląda dane dotyczące wydarzeń podróżnych. |
| **Scenariusz alternatywny 1 –** dodanie nowego wydarzenia podróżnego | 3.1 Użytkownik dodaje nowe wydarzenie podróżne.  3.1.1 Użytkownik wybiera opcję reprezentującą dodanie nowego wydarzenia.  3.1.2 System wyświetla formularz umożliwiający wprowadzenie danych wydarzenia.  3.1.3 Użytkownik zatwierdza dodanie nowego wydarzenia.  3.1.4 System stwierdza poprawność wprowadzonych danych nowego wydarzenia.  3.1.5 System zapisuje nowe wydarzenie podróżne.  Wejście do kroku 2. scenariusza głównego. |
| **Scenariusz alternatywny 2 –** edycja istniejącego wydarzenia podróżnego | 3.2 Użytkownik edytuje istniejące wydarzenie podróżne.  3.2.1 Użytkownik wybiera wydarzenie z listy wydarzeń podróżnych.  3.2.2 System umożliwia edycję danych wydarzenia.  3.2.3 Użytkownik edytuje dane wydarzenia.  3.2.4 System stwierdza poprawność operacji edycji wydarzenia.  3.2.5 System zapisuje zmiany.  Wejście do kroku 2. scenariusza głównego. |
| **Scenariusz alternatywny 3 –** usunięcie wydarzenia podróżnego | 3.2 Użytkownik usuwa wydarzenie podróżne.  3.2.1 Użytkownik wybiera wydarzenie z listy wydarzeń podróżnych.  3.2.2 Użytkownik wybiera opcję reprezentującą usunięcie wybranego wydarzenia.  3.2.3 System przedstawia komunikat wymagający potwierdzenia usunięcia wydarzenia w celu uniknięcia przypadkowej utraty danych. 3.2.4 Po potwierdzeniu usunięcia wydarzenia podróżnego, system usuwa wydarzenie.  Wejście do kroku 2. scenariusza głównego. |
| **Scenariusz wyjątku 1 –** niepoprawne dane nowego wydarzenia podróżnego | 3.1.4.1 System stwierdza niepoprawność danych nowego wydarzenia.  3.1.4.1.1 System informuje użytkownika o niepoprawności danych wraz z dokładną informacją o przyczynie niepowodzenia operacji dodania nowego wydarzenia.  Wejście do kroku 3.1.2. scenariusza alternatywnego 1. |
| **Scenariusz wyjątku 2 –** niepoprawna modyfikacja istniejącego wydarzenia podróżnego | 3.2.4.1 System stwierdza niepoprawność danych edytowanego wydarzenia.  3.2.4.1.1 System informuje użytkownika o niepoprawności modyfikacji wraz z dokładną informacją o przyczynie niepowodzenia modyfikacji istniejącego wydarzenia.  Wejście do kroku 3.2.2. scenariusza alternatywnego 2. |

|  |  |
| --- | --- |
| **Nazwa przypadku użycia** | Zarządzanie notatkami |
| **Aktorzy** | Autoryzowany użytkownik |
| **Opis** | Przypadek użycia opisuje funkcjonalność zarządzania notatkami, tj. przeglądania, dodawania, edycji lub usuwania notatek. |
| **Warunki wstępne** | Użytkownik jest zalogowany do systemu.  Użytkownik należy do wycieczki. |
| **Warunki końcowe** | Zmiany dotyczące notatek są zatwierdzone w systemie. |
| **Rezultat** | Wprowadzenie do systemu zmian dotyczących notatek należących do wycieczki. |
| **Punkty rozszerzeń** | - |
| **Scenariusz główny –** przeglądanie istniejących dokumentów podróżnych | 1. Użytkownik chce przeglądać notatki należące do wycieczki.  2. System wyświetla listę notatek.  3. Użytkownik przegląda listę notatek. |
| **Scenariusz alternatywny 1 –** dodanie nowej notatki | 3.1 Użytkownik dodaje nową notatkę.  3.1.1 Użytkownik wybiera opcję reprezentującą dodanie nowej notatki.  3.1.2 System wyświetla formularz umożliwiający dodanie notatki.  3.1.3 Użytkownik zatwierdza dodanie nowej notatki.  3.1.4 System zapisuje nową notatkę.  Wejście do kroku 2. scenariusza głównego. |
| **Scenariusz alternatywny 2 –** edycja istniejącej notatki | 3.2 Użytkownik edytuje istniejącą notatkę.  3.2.1 Użytkownik wybiera notatkę z listy notatek.  3.2.2 System umożliwia edycję notatki.  3.2.3 Użytkownik edytuje dane notatki.  3.2.4 System zapisuje zmiany.  Wejście do kroku 2. scenariusza głównego. |
| **Scenariusz alternatywny 3 –** usunięcie notatki | 3.2 Użytkownik usuwa notatkę.  3.2.1 Użytkownik wybiera notatkę z listy notatek.  3.2.2 Użytkownik wybiera opcję reprezentującą usunięcie wybranej notatki.  3.2.3 System przedstawia komunikat wymagający potwierdzenia usunięcia notatki w celu uniknięcia przypadkowej utraty danych. 3.2.4 Po potwierdzeniu usunięcia notatki, system usuwa notatkę.  Wejście do kroku 2. scenariusza głównego. |

**Model domenowy**

Na rysunku X.X znajduje się model domenowy, który reprezentuje obiekty wycinka rzeczywistości modelowane w aplikacji. Główną rolę pełni w niej klasa wycieczki (Trip), która reprezentuje wycieczkę grupową i wiążę asocjacjami klasy użytkownika (User), wydatku (Bill), dokumentu (Document), wydarzenia (Event) oraz notatki (Note). Enumeracja BillCategory reprezentuje rodzaje wydatków dostępne w aplikacji, a enumeracja SplitCategory określa typy podziału wydatku pomiędzy uczestnikami. Z klasą wydatku jest również powiązana klasa BillShare opisująca udział uczestnika w wydatku – kwotę oraz stan płatności. Dla każdego użytkownika przypisany jest jego status (AccountStatus), który określa stan konta w systemie – zarejestrowany albo aktywny. System przechowuje także informacje o nieudanych próbach logowania użytkowników (FailedLoginAttempt).

****[Podpis]

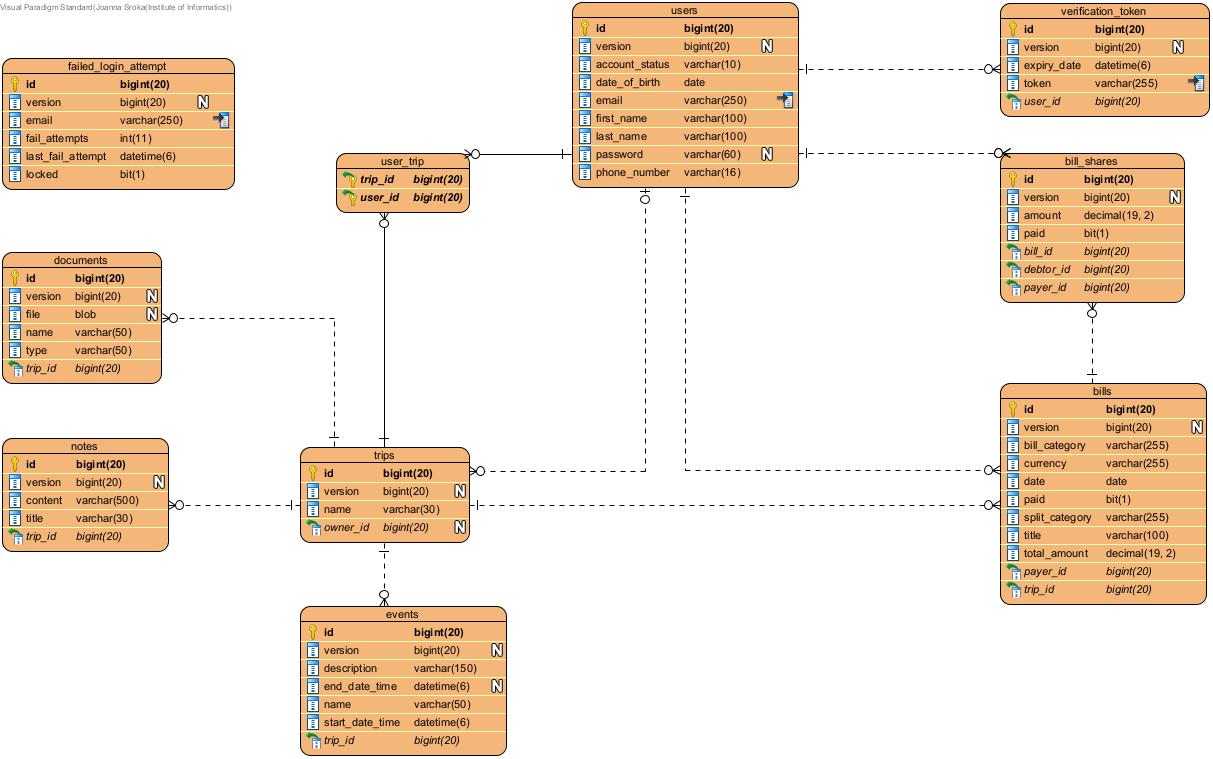
**Coś o relacyjnej bazie danych? Dlaczego MariaDB?**

**Baza danych**

Wybranym systemem zarządzania relacyjną bazą danych została MariaDB. Jest to narzędzie oparte o licencje wolnego i otwartego oprogramowania

**Schemat bazy danych**

Model bazy danych aplikacji został opracowany w oparciu o model domenowy i przedstawiony na rysunku X.X



**Trips – users 🡨 relacja wiele do wielu??? Pomiędzy ownerem a wycieczką, usunąć z diagramu!**

Wszystkie encje występujące w modelu bazy danych posiadają identyfikator będący kluczem głównym oraz wersję rekordu, która zapewnia wykorzystanie optymistycznego mechanizmu blokowania dostarczonego przez Java Persistence API. Optymistyczny mechanizm blokowania dostarcza funkcjonalność prawidłowego zarządzania równoczesnym dostępem do bazy danych. Oznacza to, że wiele operacji wykonywanych na tych samych danych jednocześnie nie kolidują ze sobą. [źródło: <https://www.baeldung.com/jpa-optimistic-locking> <-- zmienić na dokumentacje?]

W modelu bazy danych pojawiają się następujące encje:

* User

Encja reprezentująca zarejestrowanego użytkownika aplikacji. Przechowuje dane osobowe użytkownika, takie jak: imię, nazwisko, data urodzenia, numer telefonu. Obiekt użytkownika w bazie posiada również dane niezbędne do autoryzacji w systemie: adres e-mail oraz hasło, które zostało zaszyfrowane przy użyciu funkcji haszującej. Encja przechowuje informacje o statusie konta w systemie.

* VerificationToken

Encja reprezentująca token użytkownika wykorzystywany do aktywacji konta w systemie. Zawiera także informacje o dacie wygaśnięcia tokenu.

* UserTrip

Encja przechowująca informacje o uczestnikach wycieczek.

* Trip

Encja reprezentująca wycieczkę. Przechowuje informacje o nazwie wycieczki. Poprzez relację jeden do wielu z encją UserTrip przechowywana jest informacja o uczestnikach wycieczki.

* Bill

Encja reprezentująca wydatek. Zwiera takie atrybuty jak tytuł, całkowita kwota, waluta, data, kategoria wydatku oraz typ jego podziału, a także informacje, czy rachunek został rozliczony. Jest w relacji jeden do wielu z encją BillShare przechowując informację o podziale rachunku pomiędzy uczestnikami.

* BillShare

Encja reprezentująca udział uczestnika wycieczki w wydatku. Zawiera informacje o kwocie udziału w wydatku oraz o statusie płatności.

* Event

Encja reprezentująca wydarzenie podróżnicze w ramach wycieczki. Przechowuje informacje o nazwie, opisie dacie początkowej i nieobowiązkowo dacie końcowej wydarzenia. Jest w relacji jeden do wiele z encją wycieczki.

* Document

Encja reprezentująca dokument podróżny w ramach wycieczki. Przechowuje dane dokumentu oraz jego nazwę i typ. Jest w relacji jeden do wiele z encją wycieczki.

* Note

Encja reprezentująca notatkę w ramach wycieczki. Zawiera takie atrybuty jak tytuł oraz treść notatki. Jest w relacji jeden do wiele z encją wycieczki.

* FailedLoginAttempt

Encja opisująca nieudane próby autoryzacji w systemie. Przechowuje dane o adresie e-mail, z którego próbowano logować się do aplikacji, ostatniej nieudanej próbie, a także liczbie nieudanych prób, które nastąpiły w ciągu godziny. Zawiera także informacje o blokadzie adresu e-mail, aby uniemożliwić ponowne próby autoryzacji, jeżeli przekroczono limit prób.

**Architektura logiczna**

Do zaimplementowania aplikacji wykorzystano wzorzec Model-View-Controller (MVC). Został wybrany z uwagi na wsparcie wykorzystanego framework’u Spring w implementacji wzorca oraz przez wzgląd na jasny podział warstw prezentacji oraz logiki biznesowej.

Wzorzec MVC wyróżnia i oddziela od siebie różne aspekty aplikacji webowej – logikę biznesową, interfejs użytkownika oraz logikę wejściową. Jest to model aplikacji, który składa się z trzech warstw: modelu (ang. model), widoku (ang. view) oraz kontrolera (ang. controller).

Model jest warstwą odpowiedzialną za reprezentację logiki biznesowej aplikacji, zawiera całą wiedzę domenową. Jest to opis wykorzystywanych encji i powiązań pomiędzy nimi, a także używanych serwisów i modeli (DTO?) używanych w pozostałych częściach aplikacji. Nie jest w posiadaniu wiedzy o interfejsie użytkownika, dlatego może być wykorzystany przez wiele różnych interfejsów.

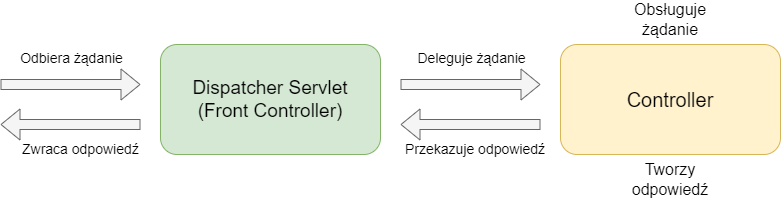
Widok jest warstwą odpowiedzialną za interfejs użytkownika, zawiera widoki aplikacji. Jest warstwą, z którą użytkownik ma bezpośredni kontakt i jest odpowiedzialna za to, co widzi użytkownik.

Kontroler jest warstwą odpowiedzialna za przetwarzanie żądań użytkowników aplikacji oraz przechowywanie kontrolerów sterujących działaniem aplikacji. Tu jest umieszczona logika działania, czyli opis powiązań pomiędzy zdarzeniami zachodzącymi w systemie.

**Spring MVC / implementacja?**

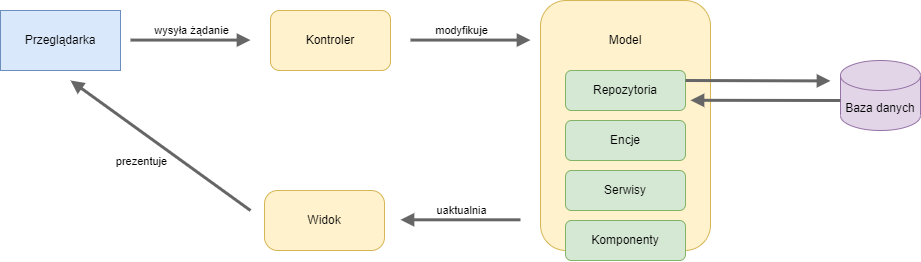
Do implementacji wzorca wykorzystano framework Spring MVC, który jest zgodny ze wzorcem projektowym Model-View-Controller, a także dostarcza komponenty, które usprawniają pracę nad aplikacją webową. Spring MVC dostarcza gotowych rozwiązań pozwalających na implementację MVC w aplikacji przy pomocy DispatcherServlet’u. DispatcherServlet odbiera przychodzące żądania i kieruje je do odpowiedniego kontrolera. DispatcherServlet pełni rolę Front Controller’a w aplikacji.

Proces przetwarzania żądań w frameworku Spring MVC przez DispatcherServlet jest zilustrowany na rysunku X.X.



Rysunek X.X Źródło: opracowanie własne

Diagram na rysunku X.X przedstawia architekturę logiczną aplikacji.



Rysunek X.X opracowanie własne

**Implementacja**

Niniejszy rozdział przedstawia implementację systemu wspomagającego organizację wycieczek grupowych. Rozdział rozpoczęto od przeglądu i analizy dostępnych na rynku narzędzi, a także od uzasadnienia wyboru wykorzystanych technologii w projekcie. Poruszono tutaj kwestie wyboru środowiska deweloperskiego, które posłużyło do wytworzenia aplikacji, framework’u wykorzystanego w projekcie oraz narzędzi informatycznych ułatwiających implementację. Omówiono także zaimplementowane funkcjonalności, wykorzystane wzorce projektowe oraz opis interfejsu użytkownika. Rozdział zawiera opis testów integracyjnych, jednostkowych oraz manualnych, które potwierdzają poprawne działanie systemu. Na zakończenie zaprezentowano potencjalne obszary rozwoju aplikacji.

**Wykorzystane technologie**

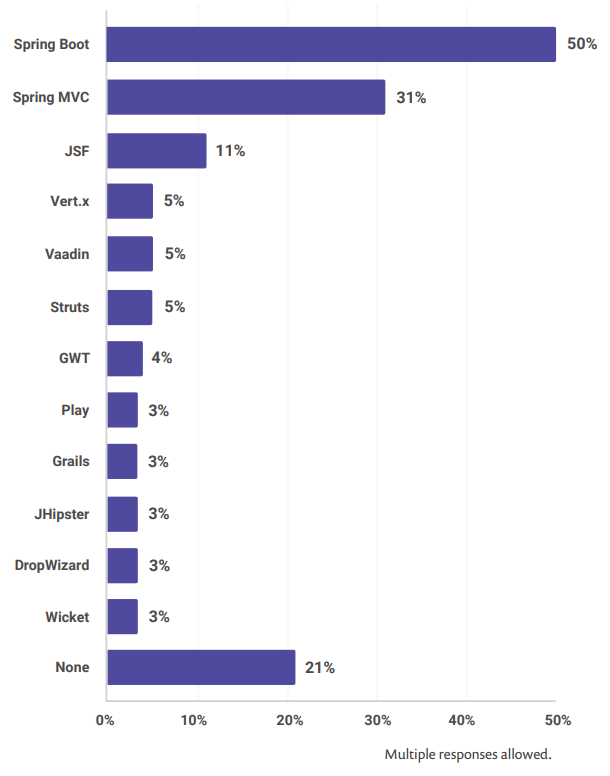
**Wstęp**

W niniejszym rozdziale zaprezentowano uzasadnienie wyboru technologii użytych w trakcie pracy nad aplikacją.

**Framework Spring**

Kluczowym aspektem popularności framework’ów jest łatwość ich wykorzystania w czasie pracy. Korzystanie z framework’u pozwala skupić się na głównym zadaniu związanym z dostarczeniem systemu, definiuje strukturę aplikacji oraz pomaga korzystać z wzorców projektowych.

W raporcie dotyczącym wirtualnej maszyny Javy z 2020 roku [źródło: JVM Ecosystem Report 2020 🡪 do bibliografii https://snyk.io/blog/jvm-ecosystem-report-2020/] framework’iem o największym udziale rynkowym jest Spring. Połowa badanych osób korzysta ze Spring Boot’a w czasie pracy, a prawie jedna trzecia używa Spring MVC.



^ dodać podpis!

Spring cieszy się zaufaniem i popularnością wśród programistów na całym świecie, przez co posiada ogromne wsparcie organizacji i społeczności programistów. Jest to bardzo obszerny i elastyczny framework, który dostarcza wielu narzędzi pozwalających na dostarczenie funkcjonalnej, szybkiej i bezpiecznej aplikacji webowej.

Podstawą działania Spring’a jest wykorzystanie wzorców projektowych odwrócenia sterowania (ang. Inversion of Control, IoC) oraz wstrzykiwania zależności (ang. Dependency Injection, DI). [dokumentacja Spring: <https://docs.spring.io/spring-framework/docs/3.2.x/spring-framework-reference/html/overview.html> ]

Odwrócenie sterowania opiera się na przeniesieniu odpowiedzialności za sterowania obiektami, tworzenie i zarządzanie zależnościami pomiędzy komponentami aplikacji oraz przejęcie kontroli nad przepływem programu do framework’a Spring. Odwrócenie sterowania można osiągnąć za pomocą mechanizmu wstrzykiwania zależności.

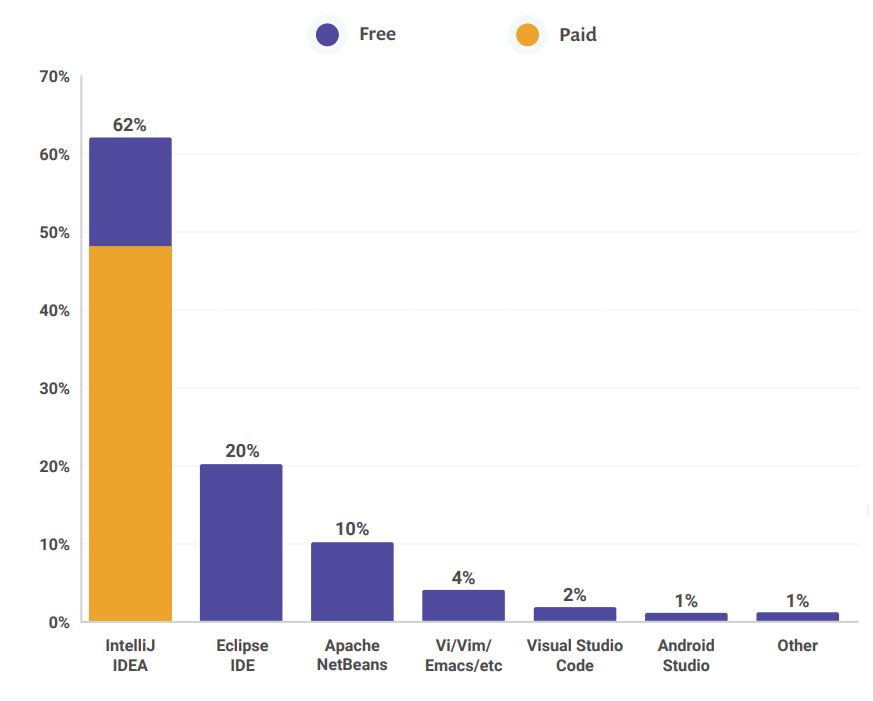
Wstrzykiwanie zależności zawęża termin odwrócenia sterowania i niweluje potrzebę tradycyjnego sposobu na tworzenie obiektu – wzorzec ten polega na tym, że klasa nie inicjalizuje swoich zależności sama, lecz przyjmuje je z zewnątrz poprzez „wstrzykiwanie”. Sprawia to także, że obiekty w aplikacji są ze sobą luźniej powiązane (ang. loose coupling) poprzez wykorzystanie zasady Odwrócenia zależności (ang. Dependency inversion principle), przez co pozwala zmniejszyć zależności pomiędzy komponentami.

Framework Spring ułatwia także wykorzystanie wzorca Model-Widok-Kontroler w aplikacji oraz umożliwia pracę z różnymi bazami danych.

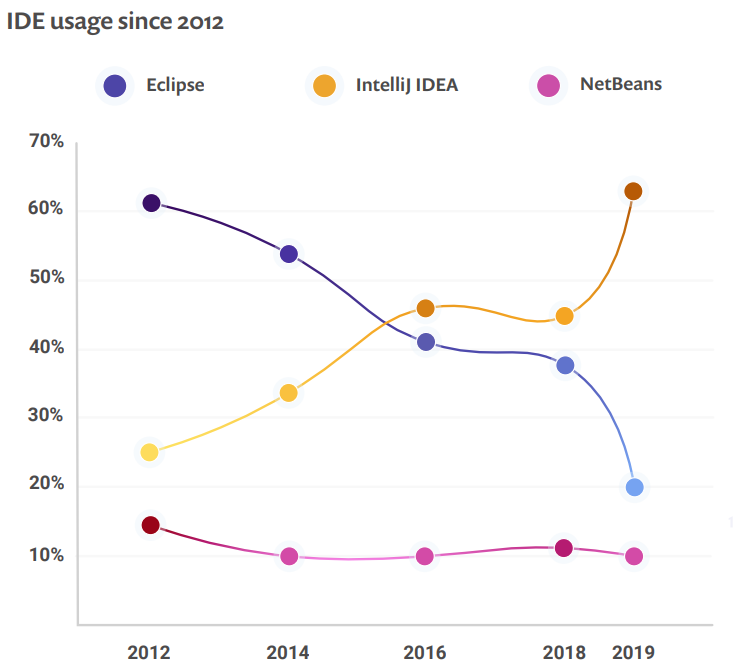
Wymienione cechy sprawiają, że Spring jest wygodnym oraz elastycznym narzędziem do pracy nad aplikacją webową. Posiada on obszerną dokumentację, a przez swoją popularność także szerokie wsparcie ze strony społeczności programistów. Wybranie framework’u Spring pozwoliło na dostarczenie funkcjonalnej aplikacji webowej.

**Środowisko deweloperskie IntelliJ IDEA**

IntelliJ IDEA to najczęściej używane środowisko do implementowania aplikacji w Javie. Według raportu zamieszczonego na rysunku X.X [źródło: JVM Ecosystem Report 2020 🡪 do bibliografii https://snyk.io/blog/jvm-ecosystem-report-2020/] przeprowadzonego w 2020 roku, korzysta z niego 62% programistów, a jego popularność z roku na rok rośnie w porównaniu do innych dostępnych środowisk na rynku, co jest widoczne na rysunku X.Y.



Rysunek X.X Raport dotyczący popularności środowisk deweloperskich wśród programstów Javy



Rysunek X.Y przedstawiający popularność w latach

Jako środowisko deweloperskie został wybrany IntelliJ IDEA. Jest to narzędzie, które posiada wersję darmową (Community) oraz płatną (Ultimate), która jest dostępna bez dodatkowych opłat dla studentów.

IntelliJ IDEA w wersji Ultimate jest skierowane do pracy nad aplikacjami w frameworku Spring, zapewniając zestaw wbudowanych narzędzi programistycznych, które ułatwiają m.in. testowanie aplikacji Spring, instalację potrzebnych bibliotek, wykonywanie żądań HTTP oraz uruchamianie zapytań do bazy danych bezpośrednio w środowisku deweloperskim. Wykorzystanie rozbudowanego środowiska deweloperskiego zdecydowanie ułatwia pracę nad aplikacją, a także wiąże się ze wzrostem produktywności programisty.

**System kontroli wersji**

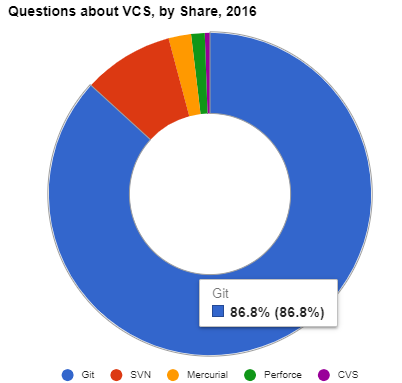
Kontrola wersji to system, który archiwizuje i śledzi wszystkie zmiany następujące w projekcie. Daje to możliwość przywracania wybranych plików lub całego projektu do poprzedniego stanu, podglądu wprowadzonych modyfikacji oraz osoby, która jest jej autorem, a także porównania ze sobą zmian w historii projektu. System kontroli wersji pozwala także na odzyskanie utraconych plików. Celem systemów kontroli wersji jest wspomaganie zespołów programistycznych w pracy nad wspólnym projektem w bardziej efektywny sposób. Wszystko to sprawia, że jest to standardowe narzędzie wykorzystywane do pracy nad projektem informatycznym.

[Źródło: Pro Git book, written by Scott Chacon and Ben Straub]

[Źródło: Dokumentacja Git’a]

Na rynku istnieje wiele systemów kontroli wersji (ang. Version Control System, VCS). Niektóre z nich są scentralizowane (przykładem jest SVN), gdzie programiści korzystają z jednego współdzielonego repozytorium. Inne systemy są zdecentralizowane (przykładem jest Git), gdzie każda z osób pracująca nad projektem posiada pełną kopię repozytorium przechowywaną na swoim komputerze.

Według ankiety przeprowadzonej w 2016 roku przez portal Stack Overflow, który jest skierowany do programistów, (Rysunek X.X) najpopularniejszym systemem kontroli wersji okazał się Git. Z ponad 20 tysięcy tematów dotyczących systemów kontroli wersji, około 87% z nich poświecono Git’owi, co świadczy o jego dużej popularności oraz zapewnia duże wsparcia ze strony programistów.



Udział systemów kontroli wersji na serwisie Stack Overflow w 2016. Źródło: https://rhodecode.com/insights/version-control-systems-2016

Dodatkowo środowisko IntelliJ IDEA dostarcza wtyczkę wspomagającą pracę z tym systemem kontroli wersji poprzez wizualizację historii wprowadzonych zmian widoczną na rysunku X.X, a także udostępnienie graficznego interfejsu do wersjonowania kodu. Usprawnia to pracę z kontrolą wersji dostarczając niezbędne wsparcie.

Obraz zawierający tekst

Opis wygenerowany automatycznie

Rysunek X.X Źródło: opracowanie własne

Wymienione cechy oraz wcześniejsze doświadczenia autora sprawiły, że jako system kontroli wersji wykorzystany w trakcie pracy nad projektem wybrano Git. Do utworzenia repozytorium wykorzystano serwis GitHub, co pozwoliło na pracę nad aplikacją z dowolnego urządzenia, a także chroniło projekt przed utratą w przypadku awarii lokalnego komputera.

**System zarządzania bazą danych MariaDB**

**Framework frontendowy Vue.js**

Vue.js jest frameworkiem frontend’owym używanym do budowania interfejsu użytkownika.

**Gradle**

**Hibernate**

???

**MariaDB**

**???**

**Biblioteki zewnętrzne**

???

**Wykorzystane wzorce projektowe**

Wzorzec projektowy jest sprawdzonym, ogólnym rozwiązaniem często napotykanych problemów w trakcie projektowania oprogramowania. Nie dostarcza implementacji rozwiązania, a jedynie jego opis, dzięki czemu jest uniwersalny oraz można go wykorzystać w wielu różnych okolicznościach. Ponadto pozwalają na komunikację z innymi programistami przy użyciu ogólnie znanych i zrozumiałych nazw wykorzystanych wzorców.

Źródło jakaś mądra książka

W trakcie pracy wykorzystano wzorce projektowe, aby usprawnić pracę nad tworzoną aplikacją, ułatwić implementację funkcjonalności poprzez zapewnienie sprawdzonych rozwiązań, a także zwiększyć czytelność kodu.

**Strategia**

Strategia jest behawioralnym wzorcem projektowym, który pozwala na zmianę zachowania poprzez zdefiniowanie różnych wymiennych algorytmów. Daje to możliwość stosowania każdego z nich zależnie od sytuacji.

Wzorzec ten został wykorzystany do implementacji funkcjonalności podziału wydatku równomiernie lub nierównomiernie według procentów, udziałów lub dokładnych kwot, co jest widoczne na rysunku X.X. W zależności od wybranego typu podziału zostaje wykonana odpowiednia metoda dzieląca rachunek pomiędzy uczestnikami.

Obraz zawierający tekst, monitor, ekran, zrzut ekranu

Opis wygenerowany automatycznie

Rysunek X.X Źródło: opracowanie własne

Wzorzec strategia dał możliwość wyboru jednego z wariantów zachowania obiektu zależnie od wyboru typu podziału rachunku. Pozwolił także odizolować logikę biznesową od szczegółów implementacyjnych algorytmów odpowiedzialnych za podział wydatku pomiędzy uczestnikami poprzez wykorzystanie interfejsu.

**Budowniczy**

Budowniczy to kreacyjny wzorzec projektowy, który umożliwia tworzenie złożonych obiektów poprzez podzielenie tego procesu na mniejsze etapy. Daje to możliwość tworzenia różnych reprezentacji tego samego obiektu.

Obraz zawierający tekst

Opis wygenerowany automatycznie

Rysunek X.X Opracowanie własne

Wzorzec budowniczy został wykorzystany w trakcie testowania działania aplikacji do tworzenia obiektów występujących w projekcie. Na rysunku X.X znajduje się diagram klas wykorzystujących wzorzec budowniczy do tworzenia instancji obiektu użytkownika oraz obiektów transferu danych użytkownika (ang. Data Transfer Object, DTO), a także jej wykorzystanie w testach na rysunku X.Y.

Obraz zawierający tekst

Opis wygenerowany automatycznie

Rysunek X.Y Opracowanie własne

**Odwrócenie sterowania**

Niby napisałam wyżej jak opisywałam Springa.

**Zaimplementowane funkcjonalności**

Autoryzacja i autentykacja

Wsparcie dla dwóch języków w aplikacji – polski oraz angielski

Zarządzanie wycieczkami

Zarządzanie uczestnikami wycieczki

Zarządzanie rachunkami

Podział wydatków

Wsparcie dla przewalutowania

Zarządzanie dokumentami

Zarządzanie wydarzeniami

Zarządzanie notatkami

Zabezpieczenia aplikacji

Baza danych – hashowanie hasła

Token autoryzacji w żądaniu

Blokowanie prób po 5 nieudanych na godzinę

**Testy**

**Wstęp**

Testowanie aplikacji zostało podzielone na wykonanie testów jednostkowych, integracyjnych oraz manualnych.

**Testy jednostkowe**

**Testy integracyjne**

**Testy manualne**

**Analiza użyteczności**

**Potencjalne obszary rozwoju**

???

**Możliwości rozwoju i dalsze kierunki prac**

???

**Zakończenie**

**Bibliografia**

|  |  |
| --- | --- |
| [1] | J. Nowak, „The title of the article,” *Journal Name,* pp. 20-24, 2016. |
| [2] | P. Babington, The title of the book, Wydawca, 2008. |
| [3] | P. Eston, „The title of the chapter,” w *The title of the book*, City, Springer, 2009,  pp. 122-144. |
| [4] | „Nazwa strony,” 2017. [Online]. Available: www.strona.pl. |

**Spis rysunków**

**Spis tabel**

**Załącznik**