AKADEMIA GÓRNICZO-HUTNICZA
Wydział Elektrotechniki, Automatyki, Informatyki i Elektroniki

Projekt
Reserve The Weather

Autorzy: Anna Jochymczyk
Monika Halek
Wiktoria Kowalska
Karol Kubek
Informatyka i Systemy Inteligentne, II rok

Przedmiot: Inżynieria Oprogramowania

1. Udokumentowane wymagania

1.Funkcjonalne

1. Rejestracja

1.1 Utworzenie konta:

System umożliwia klientowi utworzenie nowego konta poprzez wypełnienie formularza rejestracyjnego. Pola formularza to : login, email, hasło, numer telefonu.

1.2 Weryfikacja danych:

System przeprowadza weryfikację wprowadzonych danych, takich jak unikalność loginu, poprawność adresu e-mail, odpowiednia liczba znaków w numerze telefonu. W przypadku niepowodzenia pojawia się odpowiedni komunikat błędu.

1.3 Przekierowanie po rejestracji:

Po pomyślnej rejestracji klient jest automatycznie przekierowywany do swojego konta, wyświetla się jego login.

2. Logowanie

2.1 Formularz logowania:

System wyświetla formularz logowania, gdzie klient wprowadza swój login i hasło.

2.2 Weryfikacja danych logowania:

System weryfikuje wprowadzone dane logowania pod kątem poprawności i istnienia konta. W przypadku nieistniejącego loginu/hasła, błędnych danych pojawia się stosowny komunikat.

2.3 Przekierowanie po zalogowaniu:

Po pomyślnym zalogowaniu klient jest automatycznie przekierowywany do swojego konta, gdzie pojawia się login użytkownika.

3. Rezerwacja

3.1 Wybór terminu:

Klient, będąc zalogowanym, może wybrać dogodny termin rezerwacji na stronie "Moje konto" powinien wybrać opcję “Rezerwuję”, zostanie przekierowany do strony z rezerwacjami. Po wybraniu terminu z kalendarza pojawiają się dostępne zajęcia. Wybór zajęć odbywa się przez opcję “Zapisz”.

3.2 Weryfikacja dostępności miejsc:

Przed potwierdzeniem rezerwacji system zweryfikuje dostępność miejsc na wybrane zajęcia. W przypadku braku miejsc użytkownik nie będzie miał możliwości zapisu.

3.3 Zmiana statusu zajęć:

Po pomyślnej rezerwacji wybrane zajęcia pojawiają się w zakładce “Moje konto” . Liczba dostępnych miejsc zmniejsza się o 1.

4. Anulowanie rezerwacji przez klienta

4.1 Wybór opcji odwołaj:

Klient może anulować swoją rezerwację, wybierając opcję "Odwołaj" na stronie "Dane Rezerwacji".

4.2 Zwolnienie miejsca:

Po anulowaniu rezerwacji miejsce zostanie zwolnione i ponownie staje się dostępne dla innych klientów.

5. Otrzymanie zniżki

5.1Automatyczne przyznanie zniżki na zajęcia:

Zniżka jest przyznawana jeśli są spełnione określone warunki w dniu zajęć (jest zła pogoda - temperatura w dniu zajęć nie mieści się w dozwolonym zakresie przez admina). Zniżka wynosi 20%.

5.2 Weryfikacja warunków zniżki:

System weryfikuje, czy klient spełnia warunki otrzymania zniżki (np. występują złe warunki pogodowe - temperatura nie mieści się w zakresie ustalonym przez admina na dane zajęcia).

6. Usunięcie planowanych zajęć przez Administratora

6.1 Wybór opcji usuń:

Administrator może usunąć planowane zajęcia, wybierając opcję "Usuń" przy konkretnych zajęciach.

6.2 Wpływ usunięcia zajęć na użytkowników:

Po usunięciu zajęć nie będą one już dostępne do wyświetlenia w panelu administratora i u użytkowników, wszystkie rezerwacje na usuniete zajęcia przepadają.

7. Dodanie zajęć przez Administratora

7.1 Wybór opcji dodaj:

Administrator może dodać nowe zajęcia, wybierając opcję "Dodaj" w panelu administratora.

7.2 Wypełnienie formularza:

Administrator wypełnia formularz dodawania zajęć, zawierający informacje takie jak data, godzina, czas trwania, opis, nazwa, miejsca, lokalizacja, cena, minimalna temperatura, maksymalna temperatura (na podstawie wprowadzonych temperatur będzie przeprowadzana weryfikacja czy zniżka jest możliwa).

7.3 Weryfikacja unikalności zajęć:

System weryfikuje, czy dodane zajęcia są unikalne i nie istnieją już w systemie.

7.4 Dodanie zajęć:

Dodane zajęcia mogą zobaczyć użytkownicy w zakładce “Rezerwacje”.

8. Przełożenie zajęć przez Administratora

8.1 Wybór opcji przełóż:

Administrator może przełożyć planowane zajęcia, wybierając opcję "Przełóż" obok wybranych przez niego zajęć.

8.2 Wypełnienie formularza:

Administrator wypełnia formularz przełożenia zajęć, zawierający nową datę.

8.3 Możliwość przełożenia zajęć przez Administratora:

Po pomyślnym przełożeniu zajęć, data planowanych zajęć zostaje zmieniona u użytkowników.

9. Wylogowanie

Po wybraniu opcji wyloguj w zakładce “Moje konto” użytkownik zostaje wylogowany, następuje przekierowanie do strony głównej.

2.Niefunkcjonalne

1.Wydajność:

Strony systemu powinny ładować się szybko.

2.Bezpieczeństwo:

Dane użytkowników, takie jak hasła, powinny być przechowywane bezpiecznie (szyfrowanie haseł).

3.Dostępność:

System powinien być dostępny przez większość czasu, minimalizując okresy przerw serwisowych.

4.Skalowalność:

System powinien być przygotowany do obsługi rosnącej liczby użytkowników, zajęć i rezerwacji.

5.Responsywność:

Interfejs użytkownika powinien być responsywny, dostosowując się do różnych rozmiarów ekranów komputerów.

6.Integracja z OpenWeather API:

System powinien efektywnie komunikować się z OpenWeather API w celu uzyskiwania aktualnych danych pogodowych.

7.Zgodność z Przeglądarkami:

System powinien być zgodny z najnowszymi wersjami popularnych przeglądarek internetowych.

8.Audyt:

System powinien rejestrować istotne operacje, takie jak logowanie, rezerwacje, anulacje i monitorowania aktywności.

9.Dokumentacja:

Pełna dokumentacja systemu, w tym instrukcje użytkowania i konfiguracji, powinna być dostępna dla administratorów i użytkowników.

2. Diagramy UML

Diagram znajduje się w poniższym linku:

reservetheweather.png

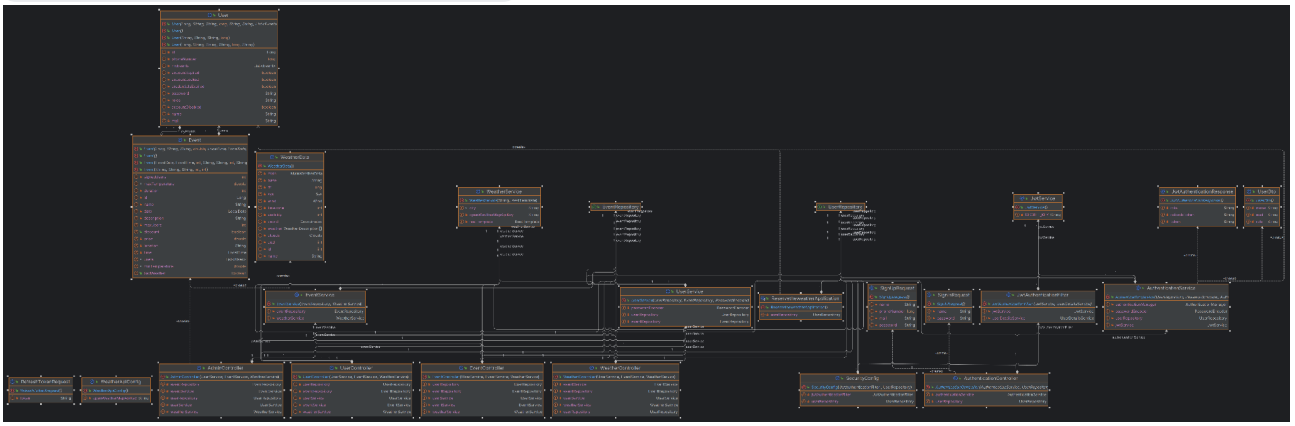
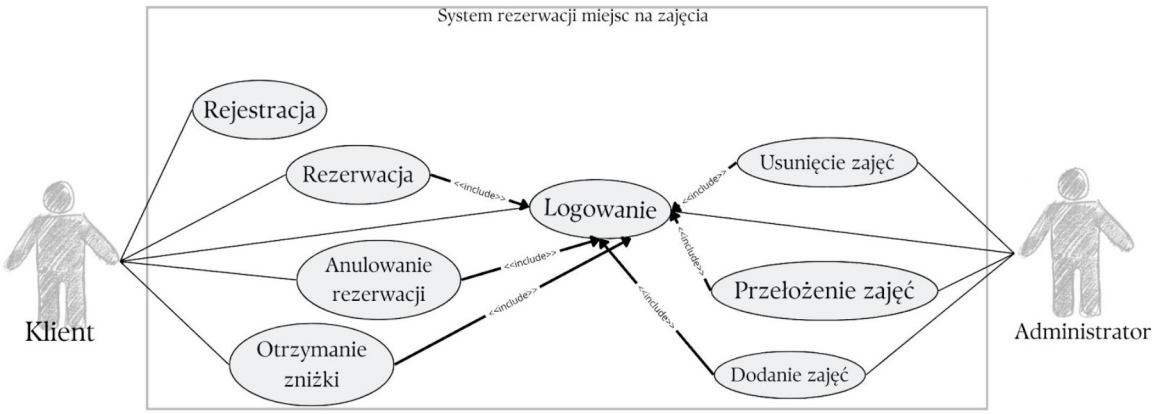


Diagram przypadków użycia



Opis przypadków użycia

Aktorzy:

1.Klient - osoba rezerwująca zajęcia. W przypadku nieodpowiedniej pogody ma prawo do otrzymania zniżki.
2.Administrator - osoba dodająca zajęcia. Ma prawo do odwoływania zaplanowanych wydarzeń, a także zmiany daty planowanych zajęć.

1. **Rejestracja**

Aktorzy: Klient
Zakres: System rezerwacji miejsc na zajęcia ReserveTheWeather
Udziałowcy i ich cele: Klient chce założyć konto
Zdarzenie wyzwalające (trigger): Klient wybiera funkcję Rejestracja
Warunki wstępne: Klient jest niezalogowany, nie posiada konta, znajduje się na stronie z formularzem rejestracji
Warunki końcowe dla sukcesu: Dodanie nowego klienta do bazy kont
Warunki końcowe dla niepowodzenia: Brak zmian w bazie kont, klient w przypadku podania nieprawidłowych danych zostanie poinformowany o przyczynie niepowodzenia.
Scenariusz główny:
1. System wyświetla formularz rejestracji
2. Klient wypełnia dane rejestracyjne(login, e-mail, hasło, numer telefon)
3. System weryfikuje dane. Weryfikacja zakończona sukcesem.
4. Pomyślne zalogowanie prowadzi do przekierowania użytkownika na stronę “Moje konto”
9. Przypadek użycia kończy się.

Scenariusz alternatywny I:
1.System podczas weryfikacji wykrywa nieodpowiednie dane (powtarzający się login, zbyt słabe hasło, nieodpowiednie znaki)

2. System wyświetla formularz z zaznaczonymi błędnymi danymi i instrukcją jak poprawnie należy je wprowadzić
3. Użytkownik poprawia dane
4. System weryfikuje dane. Weryfikacja zakończona sukcesem.

2. **Logowanie**

Aktorzy: Klient
Zakres: System rezerwacji miejsc na zajęcia ReserveTheWeather
Udziałowcy i ich cele: Klient chce się zalogować
Zdarzenie wyzwalające (trigger): Klient wybiera funkcję Logowania
Warunki wstępne: Klient nie może być zalogowany
Warunki końcowe dla sukcesu: Zmiana statusu użytkownika na „zalogowany”
Warunki końcowe dla niepowodzenia: Brak zmian statusu użytkownika, klient zostanie powiadomiony o niepowodzeniu operacji oraz podane zostaną przyczyny (niepoprawne dane do logowania, niedozwolone znaki, brak utworzonego konta)
Scenariusz główny:
1. System wyświetla formularz logowania.
2. Użytkownik wprowadza swój login i hasło.
3. System weryfikuje dane. Weryfikacja zakończona sukcesem.
4. Status użytkownika zostaje zmieniony na „zalogowany”.
Przypadek użycia kończy się.
Scenariusz alternatywny I (nieprawidłowe dane logowania):
1. System weryfikuje dane.
2. Weryfikacja zakończona niepowodzeniem (nieprawidłowe hasło dla danego loginu).
3. System wyświetla informacje o niepoprawnym haśle. Strona ulega odświeżeniu.
Przypadek użycia kończy się.
Scenariusz alternatywny II (brak konta):
1. System weryfikuje dane.
2. Weryfikacja zakończona niepowodzeniem (nie ma danego loginu w bazie).
3. System wyświetla informacje o nieistniejącym loginie. Strona ulega odświeżeniu.
Przypadek użycia kończy się.

3. **Rezerwacja**

Aktorzy: Klient
Zakres: System rezerwacji miejsc na zajęcia ReserveTheWeather
Udziałowcy i ich cele: Klient chce zarezerwować zajęcia
Zdarzenie wyzwalające (trigger): Klient wybiera funkcję „Rezerwuję”
Warunki wstępne: Klient musi być zalogowany i znajdować się w zakładce „Moje konto”, po wybraniu opcji rezerwuję na podstronie “Rezerwacje” może dokonać ostatecznej rezerwacji terminu

Warunki końcowe dla sukcesu: Zmiana statusu wybranych zajęć na „Zarezerwowane”
Warunki końcowe dla niepowodzenia: Brak zmian statusu użytkownika, klient zostanie powiadomiony o niepowodzeniu operacji oraz podana zostanie przyczyna (brak miejsc - inny użytkownik wyprzedził klienta w dokonaniu rezerwacji)
Scenariusz główny:
1. W zakładce moje konto użytkownik wybiera opcję “Rezerwuję”
2. Przekierowany do strony z możliwymi zajęciami klient wybiera dogodny termin, wyświetlają się zajęcia w tym terminie, wybór dokonywany jest przez opcję “Zapisz”.
3. System weryfikuje dane. Weryfikacja zakończona sukcesem.
4. W profilu użytkownika pojawia się rezerwacja ze statusem “Zarezerwowano”.
5. W profilu użytkownika po kliknięciu na status wybranej rezerwacji, użytkownik zostaje przekierowany na stronę z danymi rezerwacji, gdzie może sprawdzić pogodę, aktualną pogodę.
6. Przypadek użycia kończy się.
Scenariusz alternatywny I (Brak miejsc):
1. System weryfikuje czy na wybrane zajęcia nie został przekroczony limit miejsc.
2. Weryfikacja zakończona niepowodzeniem. System wyświetla informację o braku miejsc. Użytkownik zostaje przekierowany do profilu. Przypadek użycia kończy się.

4. **Anulowanie rezerwacji przez klienta**

Aktorzy: Klient
Zakres: System rezerwacji miejsc na zajęcia ReserveTheWeather
Udziałowcy i ich cele: Klient chce anulować rezerwację
Zdarzenie wyzwalające (trigger): Klient wybiera funkcję “Odwołaj” na stronie danej rezerwacji
Warunki wstępne: Klient musi być zalogowany i znajdować się na stronie z danymi rezerwacji.
Warunki końcowe dla sukcesu: Usunięcie danej rezerwacji z konta klienta, zmniejszenie liczby zapisanych na dane zajęcie użytkowników o 1.
Warunki końcowe dla niepowodzenia: Brak zmian statusu rezerwacji danego użytkownika
Scenariusz główny:
1. Na stronie “ Dane rezerwacji” użytkownik wybiera opcję “Odwołaj”
2. System weryfikuje żądanie. Weryfikacja zakończona sukcesem.
3. W profilu użytkownika znika rezerwacja.
4. Byłe zarezerwowane miejsce jest z powrotem dostępne do rezerwacji.
5. Przypadek użycia kończy się.
Scenariusz alternatywny I (użytkownik chce odwołać zajęcia na mniej niż 24 godziny od planowanej daty):
1. System weryfikuje czy do wybranych zajęć jest więcej lub równo 24 godziny.

2. Weryfikacja zakończona niepowodzeniem. System wyświetla informację o braku możliwości odwołania rezerwacji. Użytkownik zostaje przekierowany do profilu. Przypadek użycia kończy się.

5. **Usunięcie planowanych zajęć przez administratora**

Aktorzy: Administrator
Zakres: System rezerwacji miejsc na zajęcia ReserveTheWeather
Udziałowcy i ich cele: Administrator chce usunąć zajęcia
Zdarzenie wyzwalające (trigger): Administrator wybiera opcję "Usuń" przy wybranej przez niego rezerwacji, na której chce dokonać modyfikacji
Warunki wstępne: w systemie są dodane zajęcia, administrator musi być na stronie "Rezerwacje"
Warunki końcowe dla sukcesu: Zmniejszenie o jeden (o wybrane do usunięcia zajęcia) liczby zajęć na stronie rezerwacji.
Warunki końcowe dla niepowodzenia: Brak zmian w liczbie zajęć do rezerwacji.
Scenariusz główny:
1.Administrator wchodzi na stronę rezerwacji i przy wybranych zajęciach wybiera opcję usuń.
2.System weryfikuje żądanie. Weryfikacja zakończona sukcesem.
3.Wybrane przez administratora zajęcia zostają usunięte. Nie są już widoczne u zapisanych na nie użytkowników.
Scenariusz alternatywny I (brak dodanych wcześniej zajęć):
1.System weryfikuje czy są jakiekolwiek dodane zajęcia.
2. Weryfikacja zakończona niepowodzeniem.
3. System wyświetla informacje o braku możliwości usunięcia zajęć. Przypadek użycia kończy się.

6. **Dodanie zajęć przez administratora**

Aktorzy: Administrator
Zakres: System rezerwacji miejsc na zajęcia ReserveTheWeather
Udziałowcy i ich cele: Administrator chce dodać nowe zajęcia
Zdarzenie wyzwalające (trigger): Administrator wybiera opcję "Dodaj" przy wybranej przez niego rezerwacji, na której chce dokonać modyfikacji
Warunki wstępne: administrator musi być na stronie "Rezerwacje", nie może być dodanych zajęć o takich samych danych co planowane do dodania zajęcia
Warunki końcowe dla sukcesu: Zwiększenie o jeden (o wybrane do dodania zajęcia) liczby zajęć na stronie rezerwacji, zajęcia mają być unikalne
Warunki końcowe dla niepowodzenia: Brak zmian w liczbie zajęć do rezerwacji.

Scenariusz główny:

1.Administrator wchodzi na stronę rezerwacji i wybiera opcję dodaj.
2. Administrator wypełnia formularz (pola - nazwa, miejsce, data, godzina, czas trwania, cena, liczba dostępnych miejsc).
3. System weryfikuje żądanie. Weryfikacja zakończona sukcesem.

Scenariusz alternatywny I (zajęcia nie są unikalne):
1.System weryfikuje czy zajęcia są unikalne.
2. Weryfikacja zakończona niepowodzeniem.
3. System wyświetla informacje o braku możliwości dodania nieunikalnych zajęć.
Przypadek użycia kończy się.

7. **Przełożenie zajęć przez administratora**

Aktorzy: Administrator
Zakres: System rezerwacji miejsc na zajęcia ReserveTheWeather
Udziałowcy i ich cele: Administrator chce przełożyć zajęcia
Zdarzenie wyzwalające (trigger): Administrator wybiera opcję “Przełóż” przy wybranej przez niego rezerwacji, na której chce dokonać modyfikacji
Warunki wstępne: w systemie są już dodane zajęcia, administrator musi być w panelu Admina, musi zostać wybrana opcja przełóż.
Warunki końcowe dla sukcesu: w widoku dane rezerwacji u użytkowników i admina daty planowanych zajęć są już zmienione
Warunki końcowe dla niepowodzenia: brak zmian w dacie zajęć
Scenariusz główny:
1.Administrator w panelu wybiera zajęcie którego datę chce zmienić i wybiera opcję “Przełóż”.
2. Administrator wypełnia formularz (pole - data) i wybiera opcję zatwierdź.
3. System weryfikuje żądanie. Weryfikacja zakończona sukcesem.

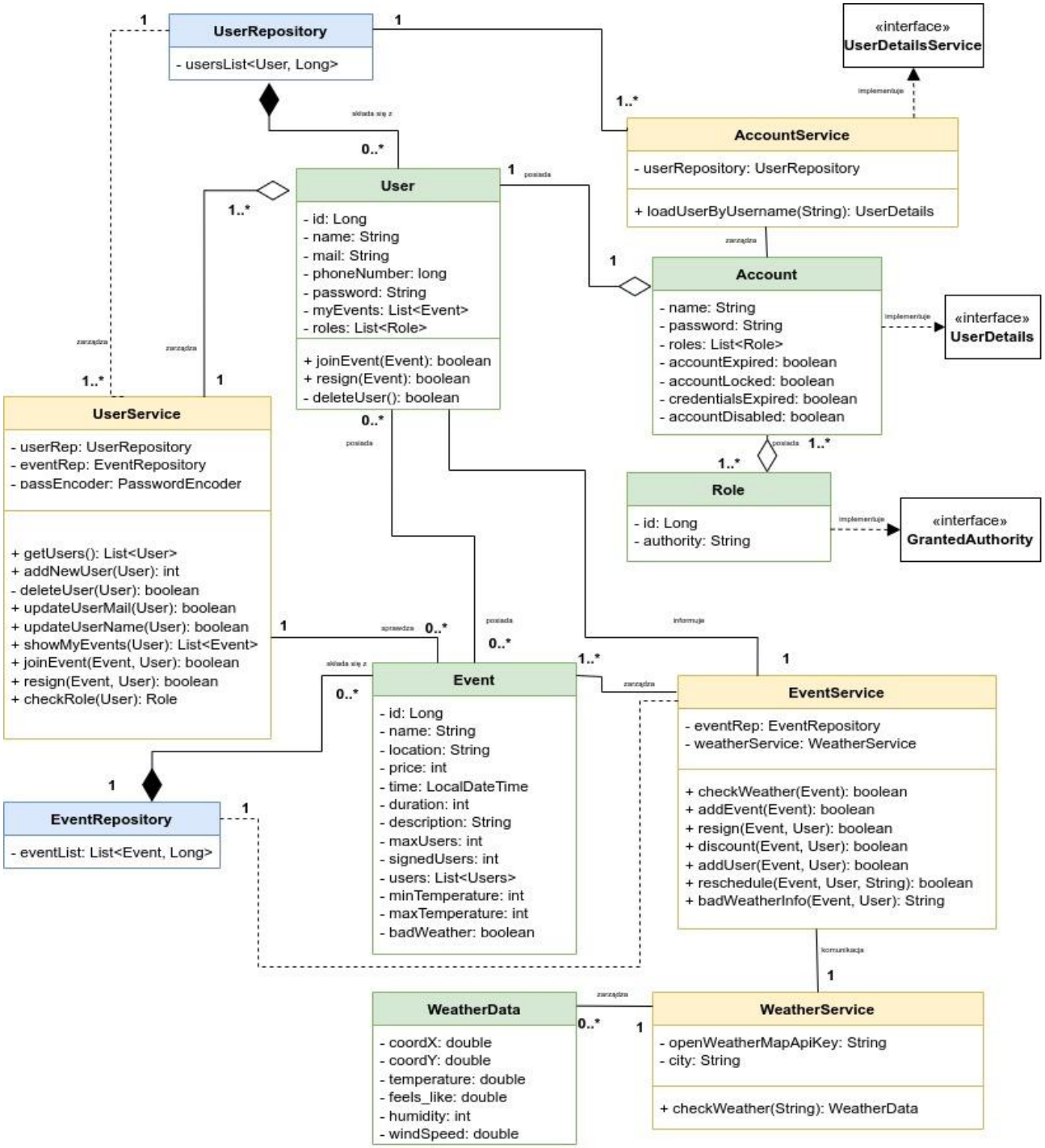
8. **Wylogowanie**

Aktorzy: Klient
Zakres: System rezerwacji miejsc na zajęcia ReserveTheWeather
Udziałowcy i ich cele: Klient chce się wylogować
Zdarzenie wyzwalające (trigger): Klient wybiera opcję “Wyloguj” w zakładce moje konto
Warunki wstępne: użytkownik musi być zalogowany
Warunki końcowe dla sukcesu: użytkownik zostaje wylogowany, przekierowany na stronę Główną, a strony dostępne dla zalogowanych użytkownik nie są dla niego dostępne
Warunki końcowe dla niepowodzenia: strony dostepne dla zalogowanych użytkownik są nadal dostępne poprzez nawigacji z konta
Scenariusz główny:
1. Klient wybiera opcję wyloguj z zakładki “Moje konto”

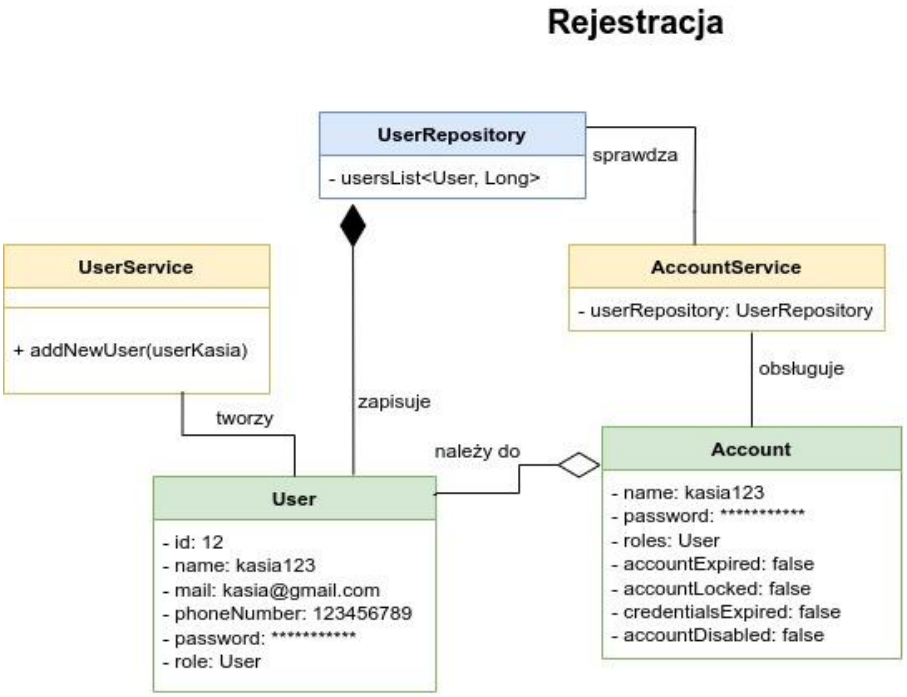
2. Zostaje przekierowany na stronę Główną, nie ma możliwości sprawdzenia rezerwacji, strony dla zalogowanych użytkowników nie są dostępne.

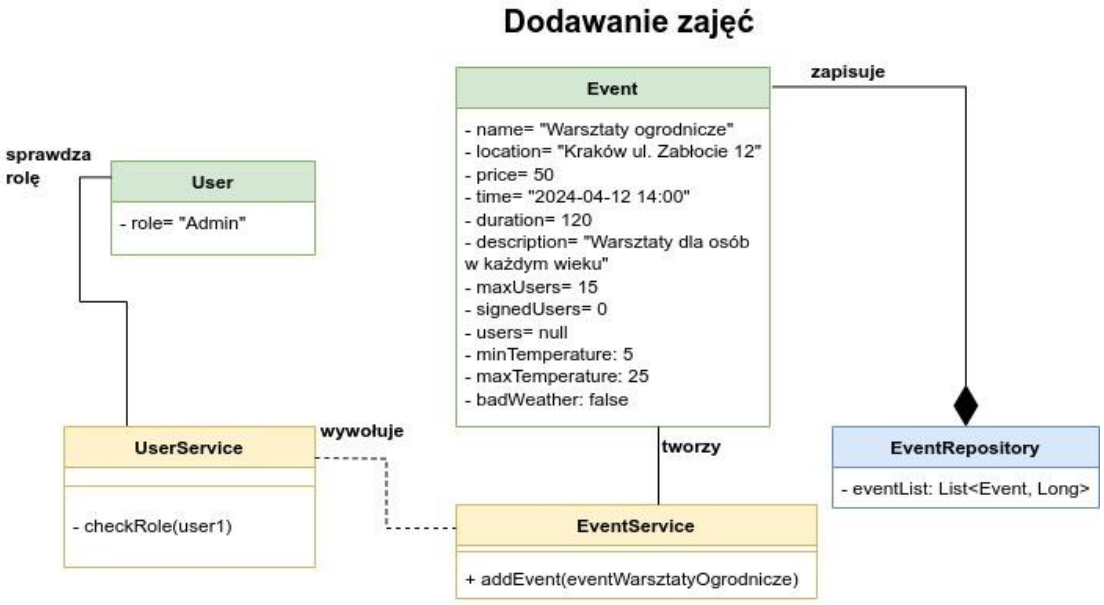
3. System weryfikuje żądanie. Weryfikacja zakończona sukcesem.

Diagram klas

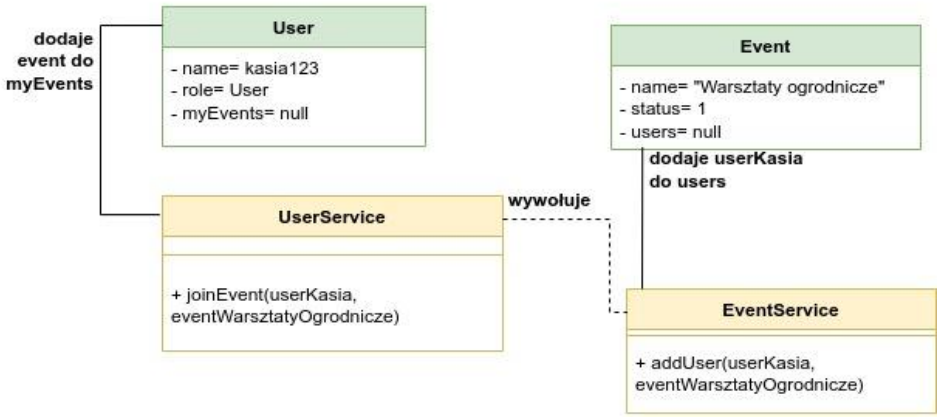


Diagramy obiektów

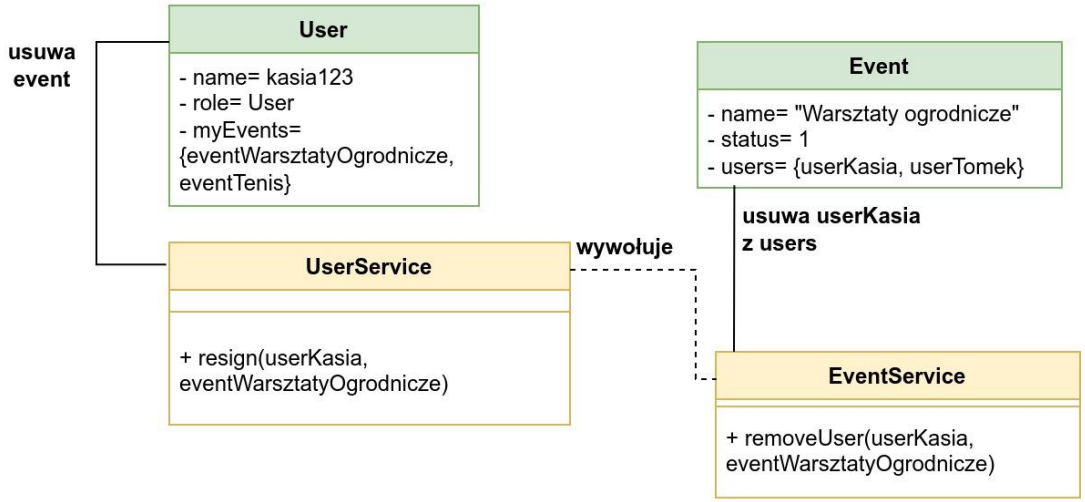




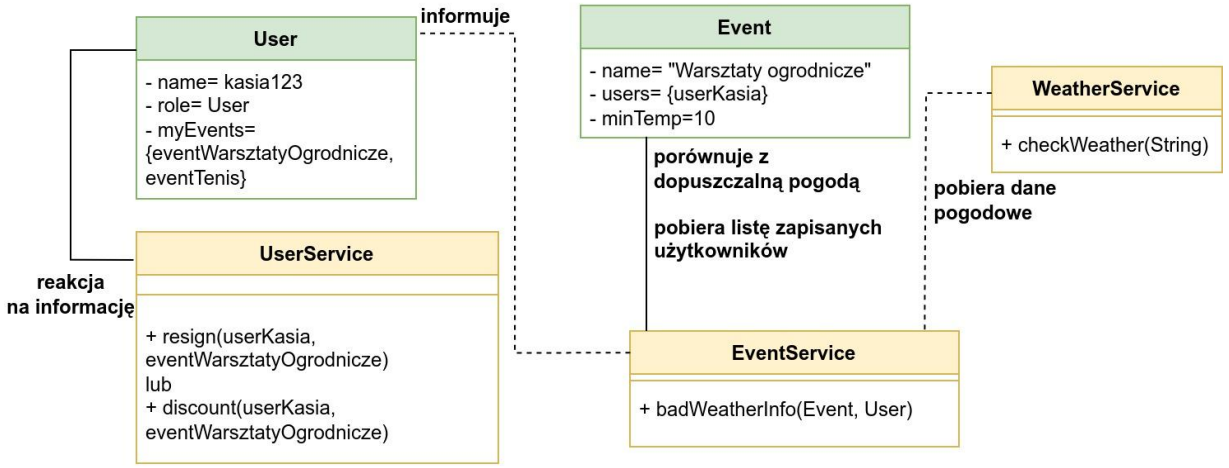
Zapisywanie się na zajęcia



Rezygnacja z zajęć

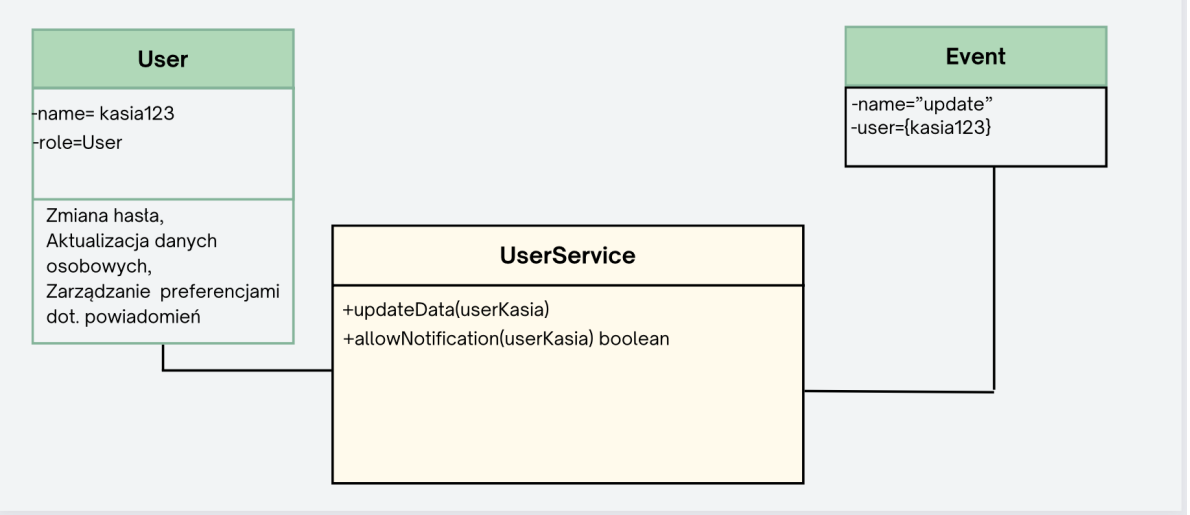


Komunikowanie o złej pogodzie

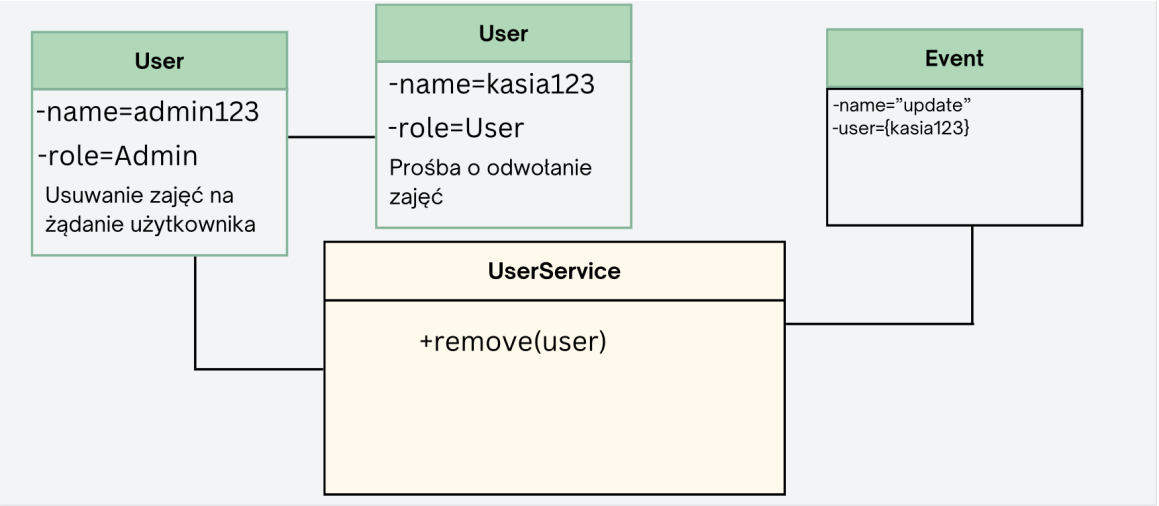


Diagramy aktywności

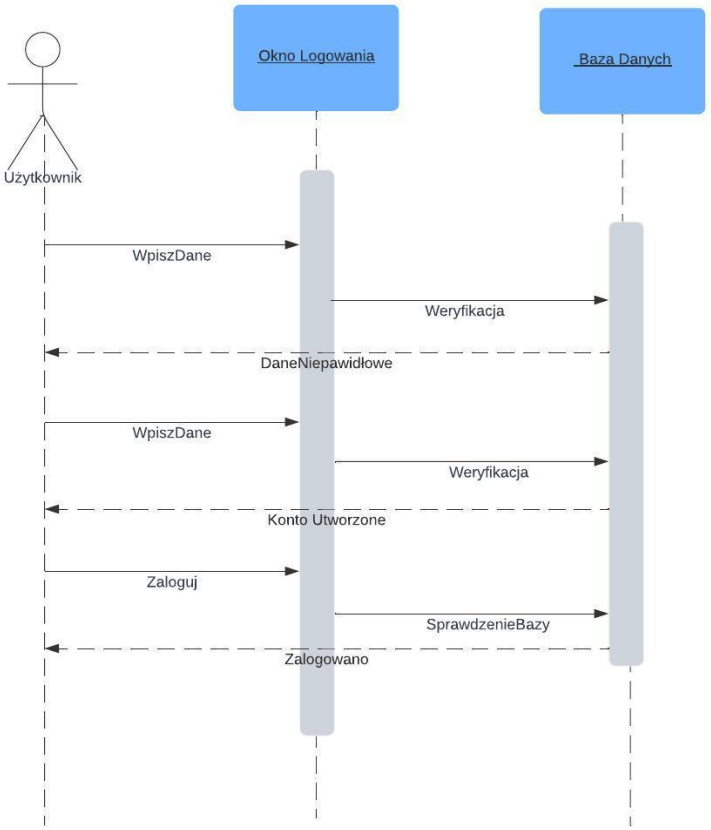
Edycja profilu użytkownika oraz jego preferencji

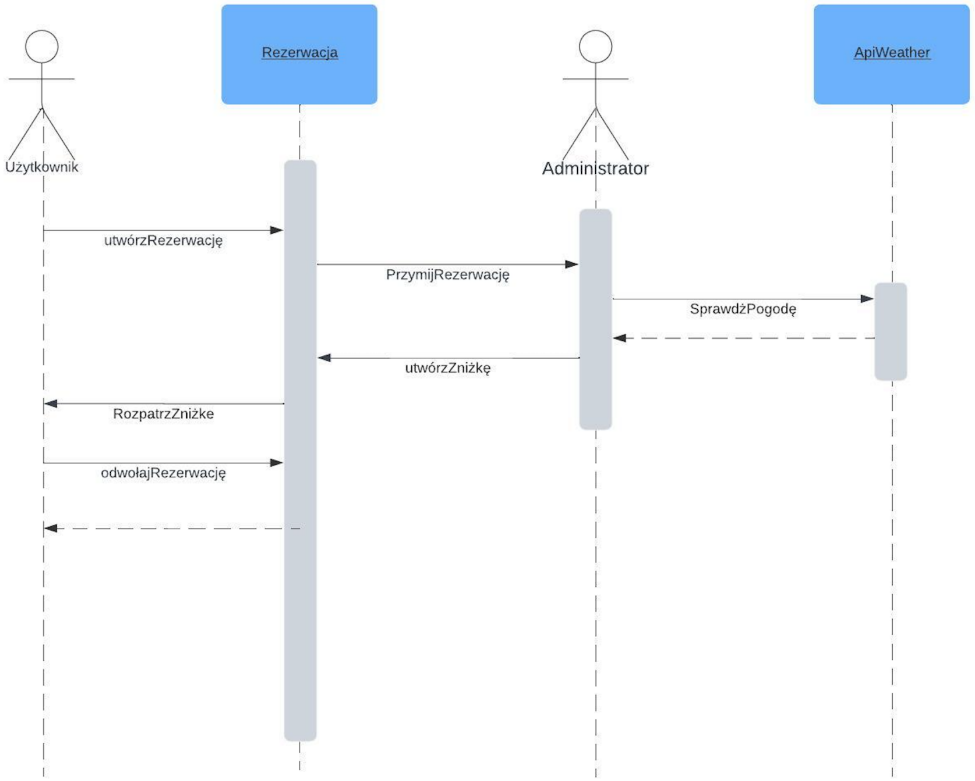


Pozostałe aktywności



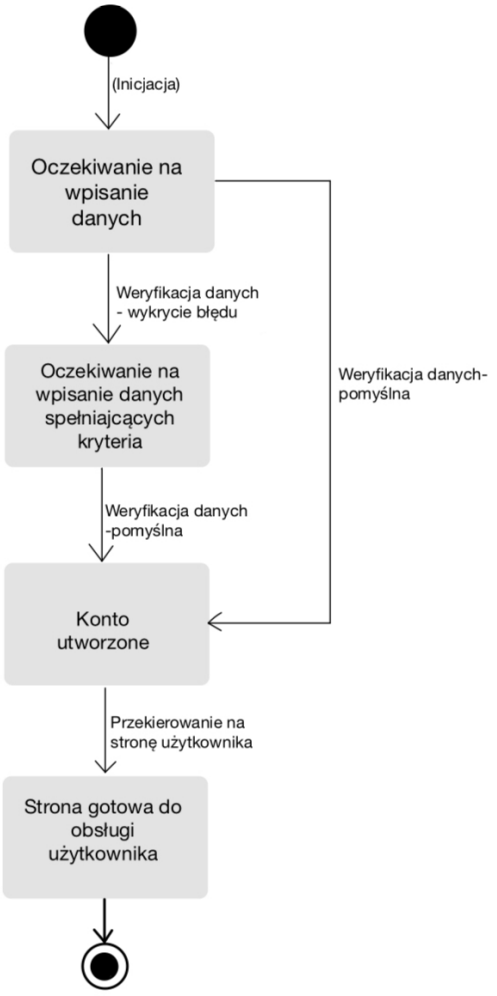
Diagramy sekwencji



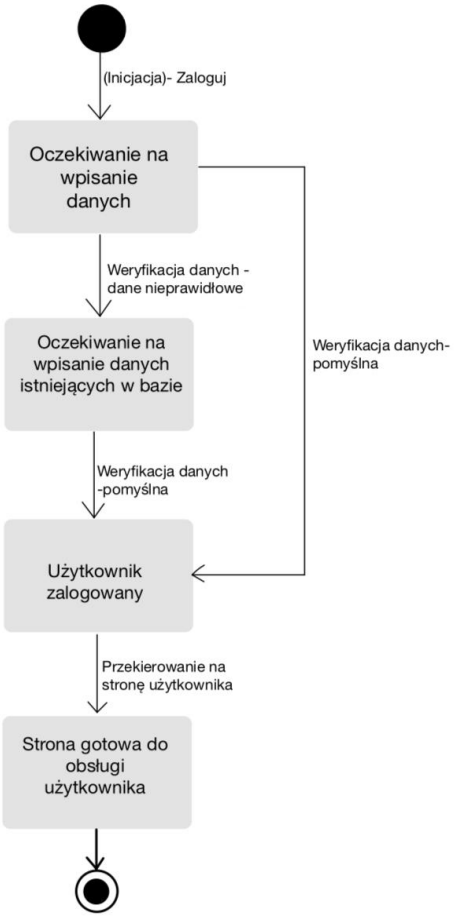


Diagramy stanów

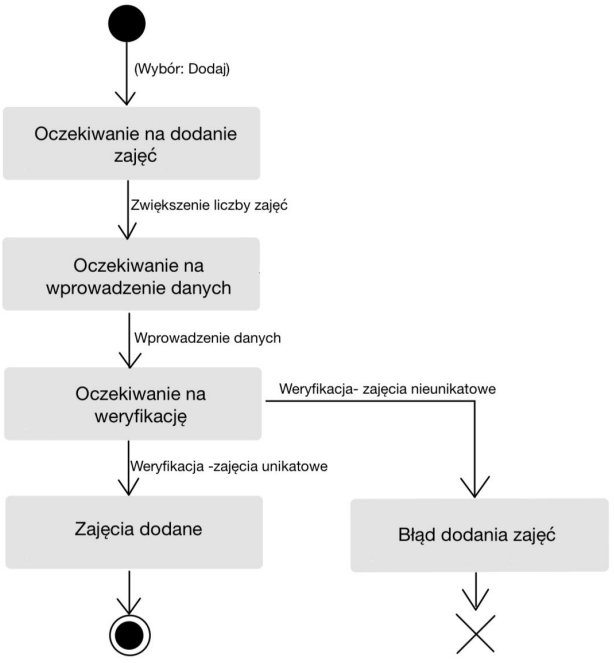
1. Rejestracja



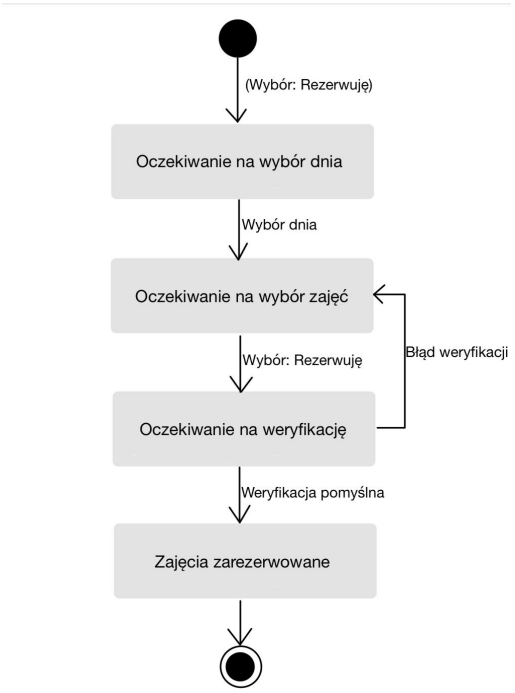
2. Logowanie



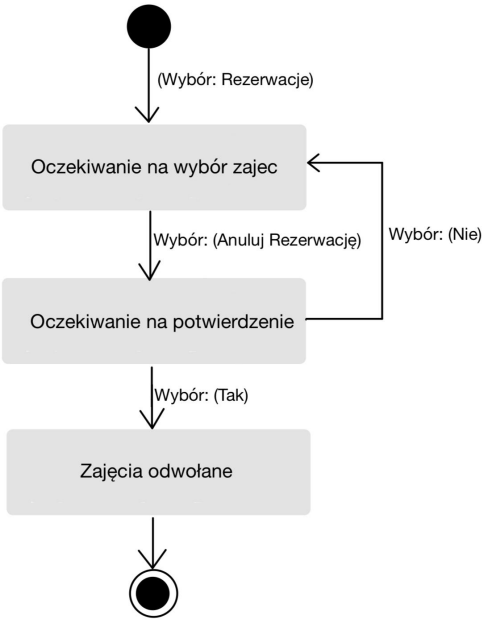
3. Dodawanie zajęć przez organizatora



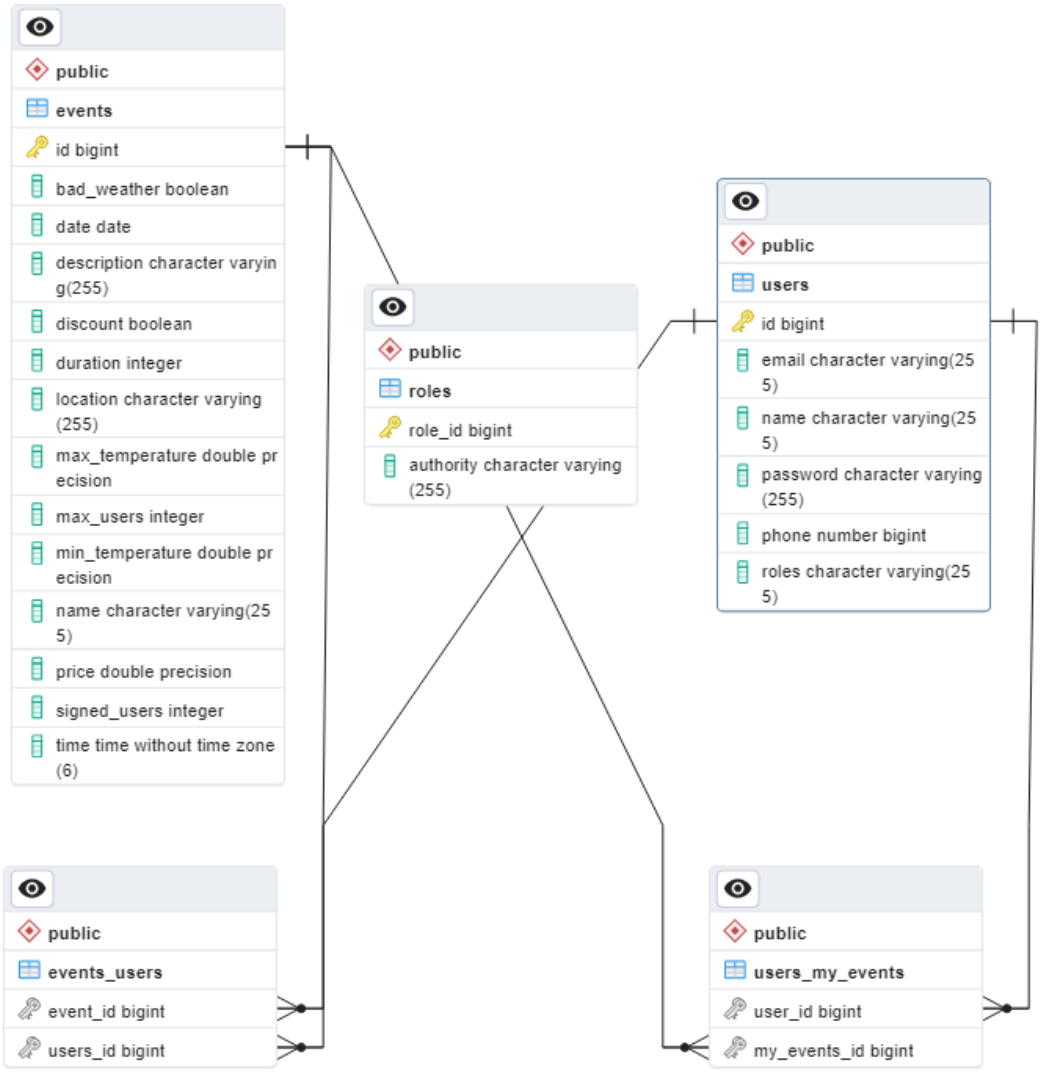
4. Rejestracja na zajęcia

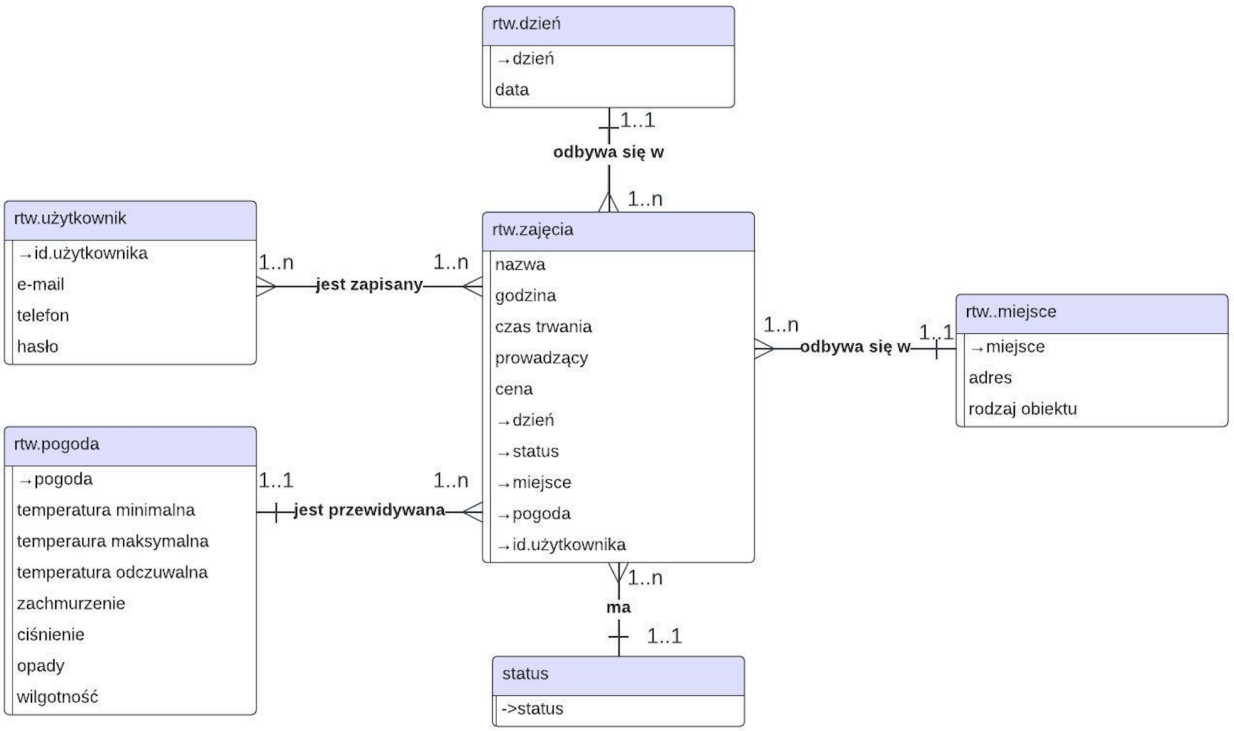


5. Anulowanie rejestracji na zajęcia

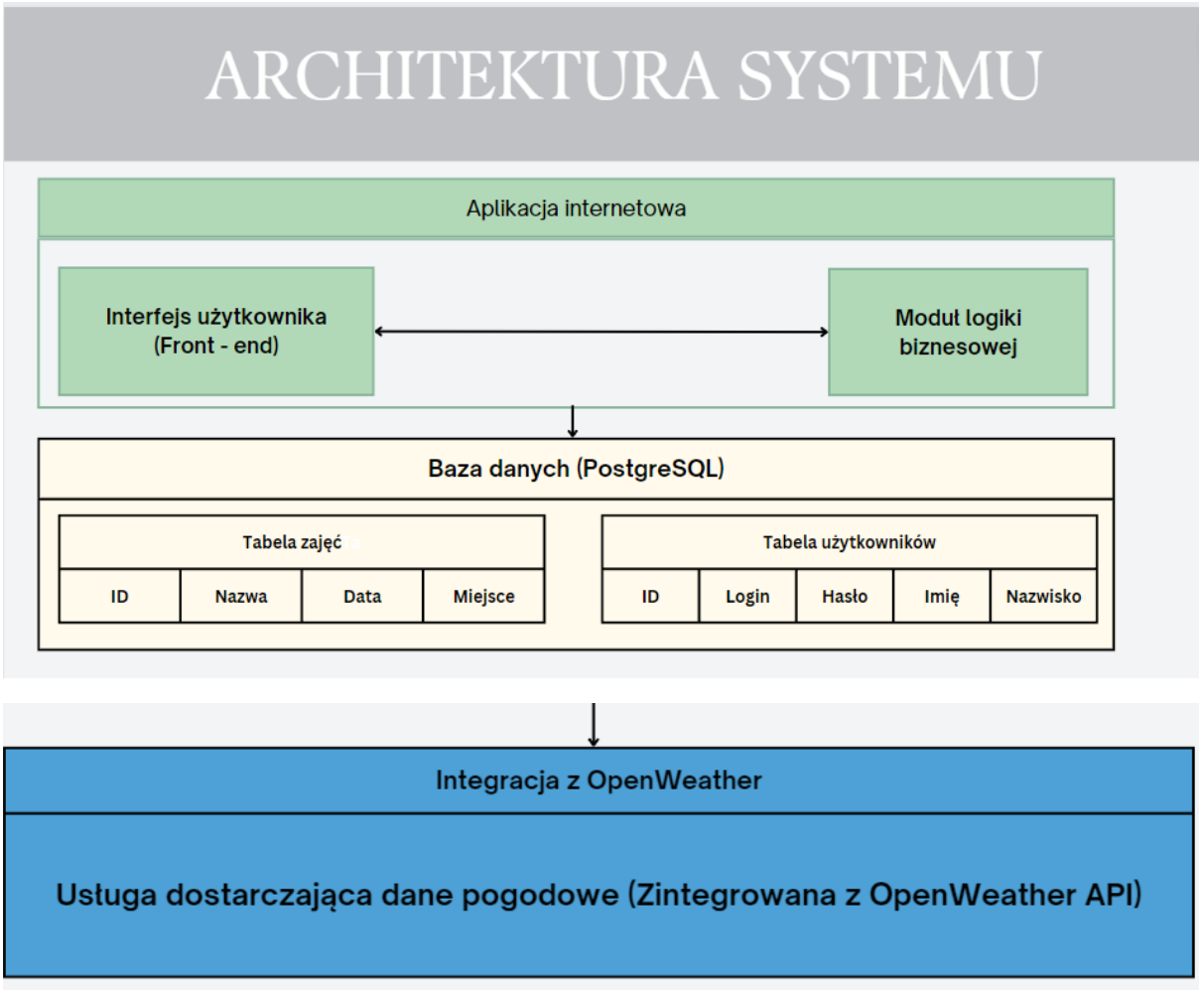


3. Diagramy ER




rtw.dzień
→dzień
data
1..1
odbywa się w
1..n
rtw.użytkownik
→id.użytkownika
e-mail
telefon
hasło
1..n
jest zapisany
1..n
rtw.zajęcia
nazwa
godzina
czas trwania
prowadzący
cena
→dzień
→status
→miejsce
→pogoda
→id.użytkownika
1..n
odbywa się w
1..1
rtw..miejsce
→miejsce
adres
rodzaj obiektu
rtw.pogoda
→pogoda
temperatura minimalna
temperaura maksymalna
temperatura odczuwalna
zachmurzenie
ciśnienie
opady
wilgotność
1..1
jest przewidywana
1..n
1..n
ma
1..1
status
->status

4. Architektura systemu



5. Opis interfejsów

1. Interfejs Baz danych (Java, Spring Boot, Postgres)

Służy do komunikacji z bazą danych w celu pobierania i zapisywania danych związanych z zajęciami oraz użytkownikami systemu. Baza jest połączona zarówno z backendem jak i z frontendem (w JavaScript).

Przykłady użycia:

- Nawiązywanie i zamykanie połączenia
- Zapisywanie rezerwacji
- Usuwanie rezerwacji

- Pobieranie dostępnych zajęć
- Pobieranie zajęć użytkowników
- Usuwanie zajęć (Admin otrzymuje funkcję *delete*, która po wpisaniu id zajęć do usunięcia usuwa zajęcia z bazy)
- Zmiana daty (Admin otrzymuje funkcję *update*, która po wpisaniu id zajęć zmienia datę)

2. Interfejs Administratora (Javascript, React, CSS, Spring Boot, Java, Postgres)

Umożliwia administracji systemem rezerwacji zajęć sportowych. Pozwala na dodawanie nowych zajęć, modyfikację istniejących, oraz zarządzanie innymi aspektami związanymi z dostępnością zajęć.

Przykłady użycia:

- Dodawanie zajęć do systemu
- Modyfikacja istniejących zajęć w systeme (zmiana daty - funkcja *update*)
- Usuwanie zajęć (poprzez funkcję *delete*)
- Pobieranie listy użytkowników zapisanych zajęcia

3. Interfejs Użytkownika (Javascript, React, CSS, Spring Boot, Java, Postgres)

Obsługuje funkcje związane z użytkownikami systemu rezerwacji zajęć sportowych. Pozwala na rejestrację, logowanie, przeglądanie dostępnych zajęć oraz dokonywanie rezerwacji.

Przykłady użycia:

- Rejestracja nowych użytkowników w systemie
- Logowanie użytkownika do systemu
- Wyświetlanie dostępnych zajęć
- Zapisywanie użytkownika na dane zajęcia
- Wycofywanie użytkownika z zajęć
- Wyświetlanie listy utworzonych rezerwacji

4. Interfejs Pogody (Javascript, OpenWeather API)

Pozwala na integrację z serwisem pogodowym w celu uzyskiwania aktualnych informacji o pogodzie. Istotne dla systemu rezerwacji, ponieważ użytkownicy mogą być zainteresowani warunkami atmosferycznymi podczas zajęć na świeżym powietrzu.

Moduł Weryfikacji Warunków Zniżki:

- Opis: Komponent weryfikujący warunki przyznania zniżki na zajęcia.
- Elementy:
    - Integracja z OpenWeather API, logika weryfikacji warunków.

Przykłady użycia:

- Pobieranie danych pogodowych dla danej lokalizacji i daty

6. Lista wykorzystywanych technologii

Poniższe technologie zostały wybrane ze względu na swoją popularność, stabilność, elastyczność i zdolność do efektywnej integracji w celu stworzenia kompleksowej aplikacji webowej. Zastosowanie każdej z nich jest dostosowane do umiejętności zespołu programistycznego.

Java**:* język programowania znany ze swojej przenośności, skalowalności i bezpieczeństwa. ***Spring Boot jest popularnym frameworkiem Java, który ułatwia szybkie tworzenie aplikacji. **Postgres***:* zaawansowany system zarządzania relacyjnymi bazami danych. Na wybór tej technologii miała wpływ jej znajomość u członków zespołu.

CSS, JavaScript, React: podstawowe technologie do budowy interfejsu użytkownika w przeglądarkach internetowych. React jest biblioteką JavaScript do tworzenia dynamicznych interfejsów, co czyni go efektywnym i wygodnym narzędziem. Dzięki CSS możliwe jest idealne dopasowanie wyglądu aplikacji do projektu interfejsu.
Fetch API *:* Umożliwia wysyłanie żądań HTTP do zewnętrznych API, co jest przydatne w przypadku integracji z OpenWeather.
Redux: biblioteka zarządzania stanem w aplikacjach React, pomagająca w efektywnym zarządzaniu danym
React Router: zapewnia nawigację w aplikacji React, co pozwala na przenoszenie się między różnymi widokami bez przeładowywania strony
Node.js*:* platforma do uruchamiania kodu JavaScript po stronie serwera. Ułatwia tworzenie skalowalnych aplikacji.
Next.js*:* framework React dla Node.js, który ułatwia renderowanie stron po stronie serwera, co dodatkowo przyspiesza ładowanie stron, intuicyjny w użyciu.
System kontroli wersji Git*:* umożliwia skuteczne zarządzanie kodem źródłowym, śledzenie zmian i współpracę w zespole.
OpenWeather API*:* wykorzystywane do uzyskiwania aktualnych danych pogodowych, co jest zgodne z funkcjonalnością projektu.
Postman*:* narzędzie do testowania interfejsów API, automatyzacji testów i dokumentowania

API. Ułatwia testowanie i debugowanie żądań HTTP.
Figma: program do projektowania interfejsu aplikacji, pozwala na utworzenie funkcjonalnych prototypów.

7. Projekt testów

7.1 Testy jednostkowe

Narzędzia: JUnit, Mockito
Zakres:

- Symulowanie wszystkich przypadków powodujących zmianę stanu
- Testowanie poszczególnych metod w klasach serwisowych, czy prawidłowo radzą sobie z różnymi scenariuszami
- Mockowanie repozytoriów przy użyciu Mockito
- Weryfikacja, czy metody poprawnie obsługują poszczególne sytuacje

Przypadek testowy	**Oczekiwane zachowanie**	**Testowana metoda**	**Klasa**
Rejestracja użytkownika (przy użyciu dozwolonych znaków, loginu ani adresu e-mail nie ma w bazie)	Zapisanie użytkownika do bazy	addNewUser(User)	UserService
Próba rejestracji użytkownika, gdy login/e-mail jest zajęty	Komunikat informujący, że login lub e-mail jest zajęty		

Rejestracja użytkownika z niepoprawnymi danymi (np. wpisano niedozwolone znaki)	Komunikat informujący o złym formacie danych		
Logowanie użytkownika przy użyciu loginu, którego nie ma w bazie, lub gdy hasło jest niepoprawne	Komunikat o błędzie	getLoggedIn()	UserService
Logowanie użytkownika poprawnymi danymi (rola user/admin)	Komunikat o udanym logowaniu		
Dodanie zajęć przez admina: niepoprawne dane (któreś pole puste/niedozwolone znaki)	Komunikat o błędzie	addEvent(Event)	EventService
Dodanie zajęć przez admina: poprawne dane	Dodanie zajęć do bazy danych		
Usuwanie zajęć przez admina	Usunięcie zajęć z bazy danych	removeEvent(Long)	EventService
Przekładanie zajęć przez admina: podanie poprawnej daty i godziny	Zmiana daty i godziny danego wydarzenia	reschedule(Long, User, String)	EventService
Przekładanie zajęć przez admina: podanie niepoprawnej daty i godziny (np. która już była)	Komunikat o błędzie		
Zapisanie się na zajęcia przez użytkownika (gdy są wolne miejsca).	• Dodanie użytkownika do listy uczestników zajęć • Dodanie zajęć do listy zajęć danego użytkownika	addPerson(Long, User) joinEvent(Long, User)	EventService, UserService
Zapisanie się na zajęcia przez użytkownika, gdy nie ma wolnych miejsc.	Komunikat o braku dostępnych miejsc		
Zapisanie się na zajęcia przez użytkownika, gdy ma już zajęcia w tym terminie.	Komunikat o konieczności wyboru jednych zajęć		
Pobieranie pogody w danym miejscu, testowanie połączenia z OpenWeather API.	Dane pogodowe z wybranego miejsca	checkWeather()	WeatherService
Sprawdzenie, czy uczestnicy wydarzenia są odpowiednio informowani o złej pogodzie (np. poprzez ustawienie miejsca zdarzenia na takie, gdzie zawsze jest nieodpowiednia temperatura).	Otrzymanie odpowiedniego komunikatu	badWeatherInform(User, Event)	EventService

Wybór otrzymania zniżki przez użytkownika po otrzymaniu komunikatu o złej pogodzie.	Zmiana ceny danych zajęć	discount(Event, User)	EventService,
Rezygnacja z zajęć przez użytkownika (przynajmniej 24h przed zajęciami),	• Usunięcie użytkownika z listy uczestników zajęć • Usunięcie zajęć z listy zajęć danego użytkownika	resign(Event, User) resign(Long, User)	EventService, UserService
Rezygnacja z zajęć przez użytkownika na mniej niż 24h przed zajęciami.	Komunikat o błędzie		
Wyświetlenie opisu zajęć.	Opis danych zajęć	showDescription(Event)	EventService
Zmiana adresu e-mail przez użytkownika (niezajęty).	Komunikat z potwierdzeniem zmiany adresu e-mail	updateLogin(Long, String)	UserService
Zmiana adresu e-mail przez użytkownika (zajęty).	Komunikat o błędzie		
Zmiana loginu przez użytkownika (niezajęty).	Potwierdzenie zmiany loginu	updateLogin(Long, String)	UserService
Zmiana loginu przez użytkownika (zajęty).	Komunikat o błędzie		
Usunięcie konta przez użytkownika.	Usunięcie użytkownika z bazy danych	deleteUser(Long)	UserService

7.2 Testy integracyjne

Celem testów integracyjnych będzie sprawdzenie, czy poszczególne elementy systemu dobrze ze sobą współpracują.
Narzędzia: Spring Boot Test, WebMvcTest

Cel	**Sposób realizacji**
Sprawdzenie poprawności integracji z OpenWeather API	·Sprawdzenie, czy informacje są pobierane i poprawnie aktualizowane ·czy w przypadku złej pogody system oferuje zniżkę lub zapisanie na zajęcia w innym terminie.
Testowanie poprawności procesu decyzyjnego użytkownika	·Symulacja scenariusza, gdzie przewidywana jest zła pogoda w czasie zajęć danego użytkownika. ·Sprawdzenie, czy otrzymuje on komunikat.

	·Podjęcie decyzji np. o zniżce. ·Sprawdzenie, czy cena zajęć została zmieniona.
Test przepływu danych przy zapisywaniu użytkownika na zajęcia	·Zalogowanie przez użytkownika. ·Wybranie zajęć, na które chce się zapisać. ·Sprawdzenie, czy w zajęcia zostały dodane do zajęć danego użytkownika, oraz czy liczba uczestników zajęć została zwiększona. ·Zalogowanie przez admina. ·Sprawdzenie, czy użytkownik, który się zapisał poprzednio na zajęcia, znajduje się w liście uczestników tych zajęć.
Test przepływu danych przy zmianie daty zajęć przez admina.	·Zalogowanie przez admina. ·Wybranie zajęcia i przełożenia ich terminu. ·Sprawdzenie, czy termin zajęć został zmieniony. ·Zalogowanie przez uczestnika tych zajęć. ·Sprawdzenie, czy nadal jest na nie zapisany, oraz czy data została zmieniona.
Testowanie kontrolerów	·Sprawdzenie, czy zwracane są odpowiednie statusy i treści odpowiedzi HTTP w różnych przypadkach. ·Symulowanie sytuacji, które powinny spowodować przekierowanie na inne strony, np. rejestracja, logowanie, usunięcie konta oraz klikanie odpowiednich przycisków przez użytkownika

7.3 Testy wydajnościowe

Narzędzia: JMeter

Cel, sposób realizacji: Celem testów wydajnościowych będzie ocena, jak system radzi sobie w przypadkach, gdy wiele użytkowników korzysta z systemu jednocześnie. W tym celu można wykorzystać JMeter, który umożliwi symulację dużego obciążenia. Monitorowanie czasu odpowiedzi i przepustowości pozwoli na ocenę wydajności systemu.

7.4 Testy UI

Narzędzia: Selenium, Cypress

7.4.1 Testy funkcjonalne

- sprawdzenie, czy wszystkie elementy interfejsu zachowują się zgodnie z oczekiwaniami, np. czy formularze działają poprawnie przy rejestracji i dodawaniu zajęć, czy dane wejściowe są walidowane

7.4.2 Testy użyteczności

- sprawdzenie, czy wygląd aplikacji jest czytelny oraz czy aplikacja jest intuicyjna w obsłudze

7.4.3 Testy kompatybilności

- sprawdzenie, czy aplikacja działa tak samo dobrze na różnych przeglądarkach, urządzeniach oraz rozmiarach ekranu

7.4.4 Testy wydajności

- sprawdzenie, ile czasu zajmuje ładowanie poszczególnych elementów interfejsu

8. Analiza ryzyka

Ryzyko	Prawdopodobieństwo	Wpływ na projekt
Brak motywacji	Średnie	Wysoki
Choroba, niedostępność członków zespołu	Średnie	Wysoki
Dezaktualizacja klucza użytkownika/usunięcie konta w OpenWeather	Niskie	Średni
Niedostateczny dostęp do pamięci	Średnie	Wysoki
Zmiana dostępu do danych pogodowych w OpenWeather	Średnie	Niski
Złe oszacowanie czasu pracy nad poszczególnymi etapami projektu	Wysokie	Wysoki
Utrata kodu lub danych	Niskie	Wysoki
Zmiana licencji oprogramowania	Niskie	Średni
Wybór nieodpowiednich technologii	Średnie	Wysoki
Konieczność zmiany technologii	Niskie	Wysoki

Ryzyko	Plan awaryjny/strategia unikania
Brak motywacji	Tworzenie motywującej atmosfery w zespole, współpraca
Choroba, niedostępność członków zespołu	Tworzenie dokumentacji, aby w razie konieczności reszta członków zespołu mogła pomóc w realizacji zadań tych, którzy nie mogą ich kontynuować.
Złe oszacowanie czasu pracy nad poszczególnymi etapami projektu	Staranne planowanie pracy, obmyślanie alternatywnych rozwiązań w przypadku napotykanych trudności.
Utrata kodu	Tworzenie kopii zapasowych, korzystanie z systemu kontroli wersji
Problemy z dostępnością API OpenWeather	Monitorowanie dostępności API, a w razie konieczności, użycie innego narzędzia
Wybór nieodpowiednich technologii	Dokładne rozpatrzenie wad i zalet rozważanych technologii oraz ocenienie, czy ewentualne komplikacje mogą uniemożliwić pracę i zmusić do zmiany używanego narzędzia.

Liczby w kolorowych polach obrazują wagę ewentualnych problemów, ich wpływ na projekt. Pokazują ile jest ryzyk spełniających oba kryteria - prawdopodobieństwo i skutek. Na przykład: 1 w zielonym polu pokazuje, że jedno ryzyko jest średnio prawdopodobne, a jego skutek jest niski, ma niewielki wpływ na pracę.

PRAWDOPODOBIEŃSTWO		SKUTEK		
		Niski	Średni	Wysoki
		1	2	3
Prawie pewne (wysokie)	3			1
Prawdopodobne (średnie)	2	1	1	4
Mało prawdopodobne (niskie)	1		2	1

9. Lista narzędzi planowanych do użycia przy realizacji projektu

1. **Github**
   Cel użycia: Kontrola wersji, zarządzanie repozytorium kodu.
2. **Google docs**
   Cel użycia: Tworzenie, edytowanie i współpraca nad dokumentami online.
3. **Kanban Tool**
   Cel użycia: Zarządzanie zadaniami w metodyce Kanban.