PartyKLINer

Grupa I

Semestr 2021Z

Spis treści

1 Opis systemu

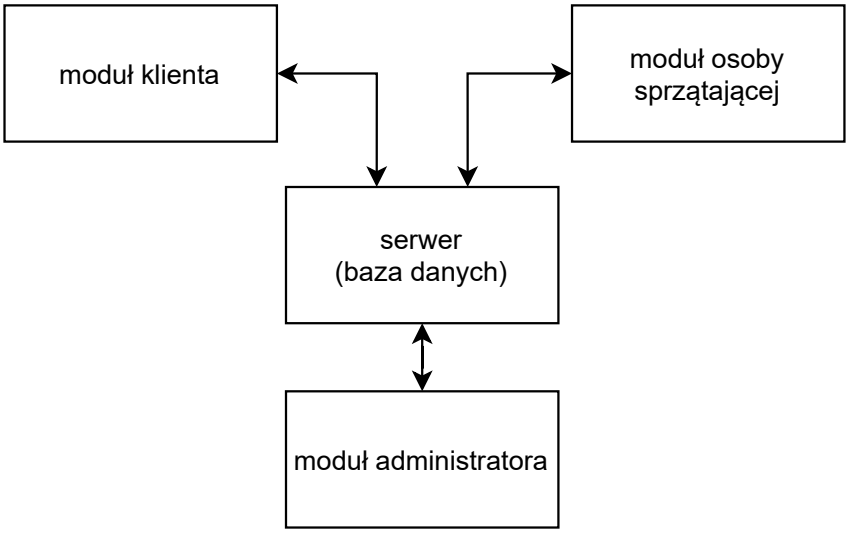
1.1 Cele i koncepcja

Głównym zadaniem projektowanego systemu jest możliwość wygodnego zamawiania przez klientów usług sprzątających przez Internet. Docelowo użytkownik będzie mógł, po uprzednim podaniu daty, lokalizacji oraz kilku innych informacji, zamówić usługę sprzątająca, bez obowiązku wcześniejszego wyszukiwania takich firm lub osób. Z kolei firmy sprzątające będą miały możliwość zarejestrowania się do systemu i w ten sposób pozyskiwanie zleceń. Aplikacja dopuszcza też możliwość rejestracji pojedynczych osób zainteresowanych świadczeniem usług sprzątających, co takim osobom może pozwolić na znalezienie pracy bez potrzeby zatrudniania się w zewnętrznej firmie

Aplikacja minimalizuje wysiłek wymagany do zamówienia usług sprzątających to jest np. szukanie takich firm/osób, weryfikowanie dostępności takich firm/osób i tym podobnym. Z kolei firmy/osoby sprzątające dzięki aplikacji nie muszą inwestować w rzeczy takie jak reklama, czy wewnętrzny system trzymania zleceń, gdyż cała funkcjonalność znajduje się po stronie systemu.

Organizator systemu dopasowuję zlecenia użytkowników do dostępnych terminów firm/osób sprzątających i pilnują by wszystkie dane w systemie były aktualne. Są oni też osobami odpowiedzialnymi za anulowanie akceptacji zleceń po ówczesnym kontakcie i podaniu powodu przez firmę/osobę sprzątającą. Dodatkowo do ich zadań należy moderacja serwisu (pełnią rolę administratora).

1.2 Moduły systemu



Rysunek 1: Diagram przypadków użycia dla projektowanego systemu.

Podział ten wyłania się naturalnie z celów i koncepcji systemu opisanych wyżej. Dostęp każdego z tych modułów jest możliwy po ówczesnym zarejestrowaniu i zalogowaniu, przy czym o ile do modułu klienta oraz sprzątacza może zarejestrować się każdy, tak do modułu organizatora mają dostęp tylko uprawnione osoby. Każdych z tych modułów jest realizowany w postaci strony internetowej (aplikacji webowej), komunikującej się z serwerem (API bazy danych). Dokładne funkcjonalności systemu zostaną opisane w następnym rozdziale w postaci historii użytkowników

1.3 Historyjki użytkownika

Dla projektowane systemu wyróżniono 3 aktorów: klienta, osobę sprzątająca oraz organizatora. Dla nich przygotowane zostały historyjki użytkownika, które zostały uszeregowane według swojej ważności za pomocą kryterium MoSCoW.

Tabela 1: Historyjki klienta w projektowanym systemie

Jako...	**Chcę...**	**Aby...**	MoSCoW
Klient	się zalogować/zarejestrować	móc korzystać z serwisu	M
Klient	zmienić hasło	móc zadbać o bezpieczeństwo konta	S
Klient	edytować profil	można było szybciej wypełniać zgłoszenia potrzeby na sprzątanie	W
Klient	usunąć konto	moje dane mogły zostać trwale usunięte	S
Klient	zgłosić potrzebę na sprzątanie	moje mieszkanie/dom mogło zostało posprzątane	M
Klient	sprawdzić status mojego zgłoszenia	móc dowiedzieć się czy przydzielono mi już ekipę sprzątającą	W
Klient	anulować zgłoszenie	móc zrezygnować z usługi gdy stwierdzę że nie jest mi jednak potrzebna	S
Klient	wyświetlić historię moich zamówień	móc ocenić zrealizowaną usługę	C
Klient	wystawić opinię po sprzątaniu	móc rekomendować dobrych sprzątających lub odradzać złych	M

Tabela 2: Historyjki organizatora w projektowanym systemie

Jako...	**Chcę...**	**Aby...**	MoSCoW
Organizator	się zalogować	móc zarządzać serwisem	M
Organizator	połączyć sprzątacza ze zleceniem	sprzątacz miał możliwość wykonania zlecenia	M
Organizator	banować użytkowników	powstrzymać nieodpowiednie zachowania w serwisie	M
Organizator	definiować prowizję	serwis zarabiał na zleceniach	M
Organizator	usuwać oferty	móc czyścić serwis z nieodpowiednich lub zbędnych treści	S
Organizator	anulować przydział do zgłoszenia	móc zmienić decyzję dotyczącą połączenia sprzątacza ze zleceniem	W
Organizator	sprawdzić status akceptacji sprzątających	wiedzieć czy sprzątacz przyjął zlecenie	S
Organizator	przeglądać dane użytkowników	posiadać informacje niezbędne do podejmowania decyzji w serwisie	C

Tabela 3: Historyjki osoby sprzątającej w projektowanym systemie

Jako...	**Chcę...**	**Aby...**	MoSCoW
Osoba sprzątająca	się zalogować lub zarejestrować	móc korzystać z serwisu	M
Osoba sprzątająca	zmienić hasło	móc zadbać o bezpieczeństwo konta	C
Osoba sprzątająca	edytować profil	móc otrzymywać jak najlepiej dopasowane do mnie zlecenia	M
Osoba sprzątająca	usunąć konto	moje dane mogły zostać trwale usunięte	C
Osoba sprzątająca	ustawić filtry interesujących mnie zgłoszeń	móc sprawniej wyszukiwać tylko te zgłoszenia, których chciałbym się podjąć	M
Osoba sprzątająca	edytować grafik dostępności	móc deklarować godziny dostępności do sprzątania	S
Osoba sprzątająca	zaakceptować/odrzucić zlecenie	móc podjąć się/nie podejmować się sprzątania w ramach danego zlecenia	M
Osoba sprzątająca	sprawdzić szczegóły dotyczące zlecenia	móc łatwiej zdecydować, czy chcę podjąć się jego realizacji	S
Osoba sprzątająca	skontaktować się z organizatorem	móc informować o sytuacjach wyjątkowych, np. o niedyspozycji	C
Osoba sprzątająca	wyświetlić historię moich zleceń	móc rzetelniej ocenić współpracę z danym klientem	W
Osoba sprzątająca	potwierdzić wykonanie zlecenia	poinformować organizatora i klienta o zakończeniu sprzątania mieszkania/domu	S
Osoba sprzątająca	wystawić opinię klientowi po zakończeniu zlecenia	ułatwić kolejnym sprzątającym podjęcie decyzji czy chcą podejmować się współpracy z danym klientem	S

1.4 Kluczowe wymagania systemu

1.4.1 Klient

Jako klient chcę zgłosić potrzebę na sprzątanie aby moje mieszkanie/dom mogło zostać posprzątane.

Wymagania funkcjonalne

- Czy mogę dodać adres nieruchomości?
- Czy mogę opisać nieruchomość?
- Czy mogę określić stopień spodziewanego bałaganu?
- Czy mogę określić próg ocen od którego sprzątający mogą być mi przydzieleni?
- Czy mogę określić zakres kwot które jestem w stanie zapłacić za sprzątanie?
- Czy mogę podać informacje kontaktowe?
- Czy mogę określić datę i godziny w których zamawiam usługę?

Wymagania niefunkcjonalne

- Czy adres i informacje kontaktowe są autouzupełniane z profilu?
- Czy wpisany adres jest weryfikowany pod kątem poprawności?
- Czy blokowane jest posiadanie więcej niż 5 aktywnych ogłoszeń w jednym czasie?

Jako klient chcę wystawić opinię po sprzątaniu aby móc rekomendować dobrych sprzątających lub odradzać złych.

Wymagania funkcjonalne

- Czy mogę ocenić ekipę sprzątającą w skali od 1 do 5 gwiazdek?
- Czy mogę dodać komentarz?
- Czy mogę załączyć zdjęcia przed i po?

Wymagania niefunkcjonalne

- Czy komentarz i zdjęcia są nieobowiązkowe?
- Czy przypomnienie o możliwości oceny jest wysyłane na maila?
- Czy zablokowana jest dwukrotna ocena tego samego zamówienia?

1.4.2 Organizator

Jako organizator chcę banować użytkowników aby powstrzymać nieodpowiednie zachowania w serwisie.

Wymagania funkcjonalne

- Czy mogę zbanować dowolnego użytkownika?
- Czy mogę podać powód bana?
- Czy mogę określić czas trwania bana?

Wymagania niefunkcjonalne

- Czy dane zbanowanego użytkownika są usuwane?
- Czy muszę dodatkowo potwierdzić decyzję o banie?
- Czy użytkownik zostaje poinformowany o banie?
- Czy zbanowany użytkownik może korzystać z serwisu?

Jako organizator chcę połączyć sprzątacza ze zleceniem aby sprzątacz miał możliwość wykonania zlecenia.

Wymagania funkcjonalne

- Czy mogę dodać sprzątacza niespełniającego kryteriów oferty?
- Czy mogę dodać zbanowanego sprzątacza?
- Czy mogę dodać wielu sprzątaczy?
- Czy mogę dodać sprzątacza do oferty w toku?

Wymagania niefunkcjonalne

- Czy sprzątacz otrzymuje powiadomienie że został dodany?
- Czy klient musi zaakceptować sprzątacza?
- Czy sprzątacz musi zaakceptować ofertę?

1.4.3 Osoba sprzątająca

Jako osoba sprzątająca chcę sprawdzić szczegóły dotyczące zlecenia aby móc łatwiej zdecydować, czy chcę podjąć się jego realizacji.

Wymagania funkcjonalne

- Czy mogę sprawdzić odległość od mojego adresu do adresu podanego w zleceniu?
- Czy mogę sprawdzić stopień spodziewanego bałaganu?
- Czy mogę określić wielkość powierzchni domu/mieszkania, które należy wysprzątać w ramach zlecenia?
- Czy mogę sprawdzić ile klient oferuje za wypełnienie zlecenia?
- Czy mogę sprawdzić średnią ocenę klienta który stworzył zlecenie?
- Czy mogę sprawdzić informacje kontaktowe klienta?
- Czy mogę sprawdzić dokładną datę wykonania usługi?

Wymagania niefunkcjonalne

- Czy odległość od mojego adresu do adresu podanego w zleceniu jest wyliczana automatycznie korzystając z adresu podanego w moim profilu?

Jako klient chcę ustawić filtry interesujących mnie zgłoszeń aby móc sprawniej wyszukiwać tylko te zgłoszenia, których chciałbym się podjąć.

Wymagania funkcjonalne

- Czy mogę ustawić minimalną średnią ocenę klienta?

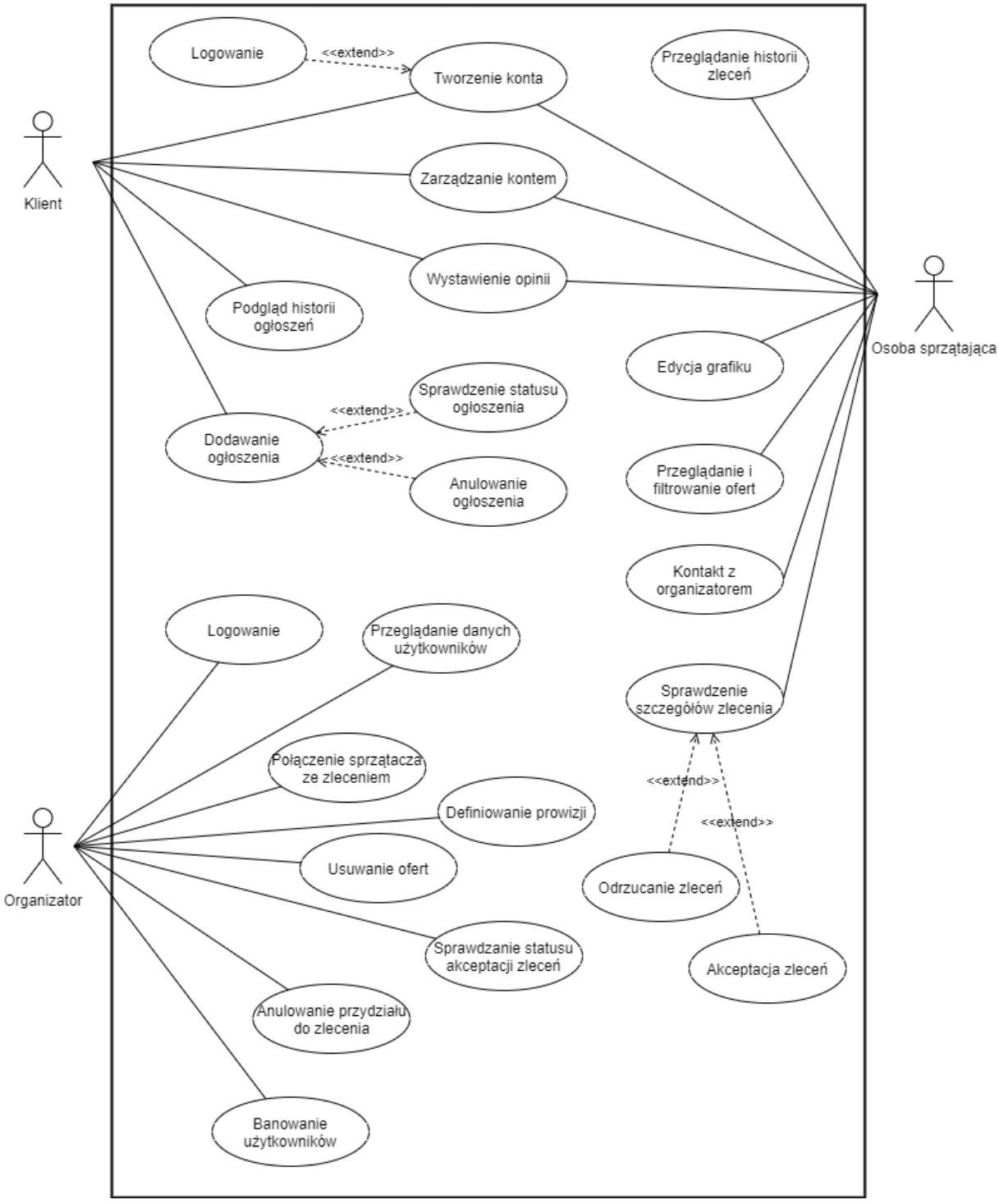
- Czy mogę ustawić minimalną kwotę za wykonanie zlecenia?
- Czy mogę ustawić maksymalną odległość od mojego adresu podanego w profilu do mieszkania/domu do wysprzątania w ramach zlecenia?
- Czy mogę filtrować zlecenia według spodziewanego poziomu bałaganu?
- Czy mogę filtrować zlecenia według powierzchni domu/mieszkania, które należy wysprzątać w ramach zlecenia?
- Czy mogę filtrować zlecenia po dacie podanej w zleceniu?

Wymagania niefunkcjonalne

- Czy filtr daty pozwala mi na wygodne wybranie dat z wyskakującego okienka z kalendarzem?
- Czy odległość od mojego adresu do adresu podanego w zleceniu jest wyliczana automatycznie korzystając z adresu podanego w moim profilu?

1.5 Diagram przypadków użycia

Całość przedstawionych powyżej historyjek użytkownika zawiera (oraz grupuje) poniższy diagram przypadków użycia.

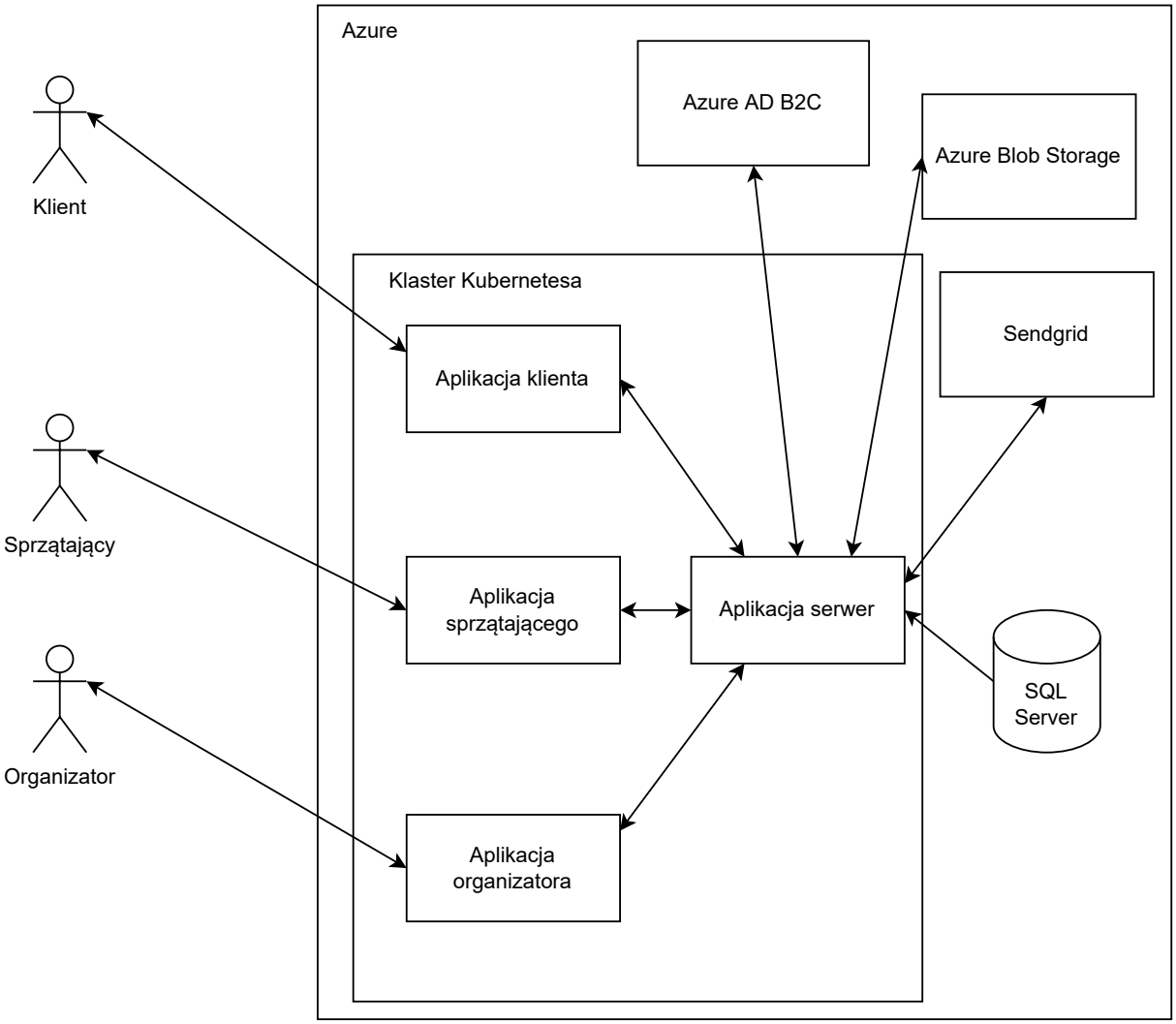


Rysunek 2: Diagram przypadków użycia dla projektowanego systemu.

2 Architektura

2.1 Założenia technologiczne

System od strony technologicznej składa się z 4 modułów, pierwszy to aplikacja backendowa (nazywana serwerem), pozostałe to aplikacje frontendowe (moduł klienta, sprzątającego i administratora). Aplikacja backendowa produkcyjne deployowana będzie jako kontener Dockerowy na klastrze Kubernetesa (Azure). Rozwiązanie to pozwala na zapewnienie auto-skalowania i aktualizacji bez down-time. Podobnie apliakcje frontendowe wdrażane będą jako kontenery Dockerowe zawierające serwer HTTP nginx, hostujący kod aplikacji. Jedynie serwer komunikuje się bezpośrednio z bazą danych (SQL Server na Azure), w której przechowywane są wszelkie informacje odnośnie stanu systemu. Autentykacja użytkowników opiera się o usługę Azure AD B2C. W przypadku lokalnych testów aplikacji, poszczególne serwisy mogą zostać uruchomione poza kontenerami i z lokalną instancją SQL Server. Wysyłka maili obsługiwana jest przez usługę Sendgrid, a obsługa plików, jeśli zajdzie taka konieczność, powinna być obsłużona przez Azure Blob Storage. Infrastruktura chmurowa nie powinna być tworzona ręcznie ale z użyciem narzędzi IaC, takich jak Terraform.



Rysunek 3: Architektura systemu

2.1.1 Moduły klienckie

Moduły klienckie, jak zostało wspomniane, są aplikacjami frontendowymi. Wszystkie trzy zostaną napisane w Typescripcie (v 4.5) i bibliotece React, a interfejs graficzny zostanie zaprojektowany w HTML5 i CSS3. Oczywiście, niezbędny jest menadżera pakietów i tutaj użyty zostanie NPM i, dodatkowo, jako framework do testów wskazane jest skorzystanie z Jest-a. Dzięki wykorzystaniu zewnętrznego rozwiązania zapewniającego autentykację, jakim jest Azure AD B2C, można wykorzystać dostępne biblioteki do integracji z tą usługą, dla React-a jest to biblioteka @azure/msal-react. Wymieniono tutaj jedynie kluczowe, z punktu widzenia architektury systemu, technologie i biblioteki, lista ta powinna być odpowiednio w miarę potrzeb rozszerzana.

2.1.2 Moduł serwera

Moduł serwera jest aplikacją backendową i zostanie napisany w C# (.NET 6). Wykorzystane tutaj zostaną: framework ASP.NET Core, menadżer pakietów nuget i framework xUnit do testów jednostkowych i integracyjnych. Dla zapewnienia niezbędnej dla aplikacji komunikacji z bazą danych użyte zostanie Entity Framework (ORM). Dodatkowo potrzebne będą biblioteki do integracji z Azure AD B2C, Blob Storage i Sendgrid - wybrano te oferowane przez Microsoft.

2.2 Funkcjonalność

System powinien zapewnić kompletną obsługę procesu składania, przydzielania i oceny realizacji zamówienia zgodnie z niniejszą specyfikacją. Na każdym etapie uprawienia użytkowników powinny być skrzętnie weryfikowane, nie dopuszcza się sytuacji modyfikacji lub odczytu danych przez osobę do tego niepowołaną. Wszelkie dane osobowe powinny być bezwzględnie chronione. Dodatkowo, system powinien być odporny na popularne ataki takie jak SQL Injection i CSRF.

2.3 Użyteczność

Interfejs graficzny użytkownika w modułach klienta i sprzątającego powinien być możliwe prosty i intuicyjny. Korzystanie z systemu nie powinno wymagać żadnych wcześniejszych umiejętności. W przypadku interfejsu organizatora najważniejsze jest zapewnienie interfejsu pozwalającego na możliwe szybkie i sprawne wykonywanie działań (użytkownik tego modułu może potrzebować przeszkolenia).

2.4 Niezawodność

System powinien być dostępny 24/7, w przypadku awarii któregoś z modułów, powinien on być natychmiast zastąpiony swoją repliką. Aktualizacje również nie powinny powodować niedostępności systemu.

2.5 Wydajność

System powinien być w stanie dostosować się do zmiennego natężenia ruchu i posiadać funkcjonalność auto-skalowania. Moduł, który jest bliski przeciążenia powinien zostać zduplikowany, a ruch powinien być rozkładany na wszystkie dostępne instancje.

2.6 Wsparcie

System powinien być monitorowalny, administrator systemu powinien mieć dostęp do statystyk odnośnie natężenia ruchu i logów z ewentualnych błędów.

3 Struktura systemu

3.1 Struktury

We wszystkich wyróżnionych modułach systemu pojawiają się identyczne struktury przechowujące dane, ich opis znajduje się poniżej.

Tabela 4: Struktury w systemie PartyKLINer

Struktura	**Pole**	**Opis**
PersonalInfo	Name	Obowiązkowe imie osoby sprzątającej
	Surname	Obowiązkowe nazwisko osoby sprzątającej
	Email	Obowiązkowy adres email osoby sprzątającej
TimeSpan	Start	Obowiązkowy początek okresu czasu wyznaczanego przez TimeSpan
	End	Obowiązkowy koniec okresu czasu wyznaczanego przez TimeSpan
Address	Street	Obowiązkowa nazwa ulicy adresu
	BuildingNo	Obowiązkowy numer budynku adresu
	FlatNo	Nieobowiązkowy numer mieszkania adresu (może być pusty)
	City	Obowiązkowe miasto adresu
	PostalCode	Obowiązkowy kod pocztowy miasta adresu
	Country	Obowiązkowy kraj adresu

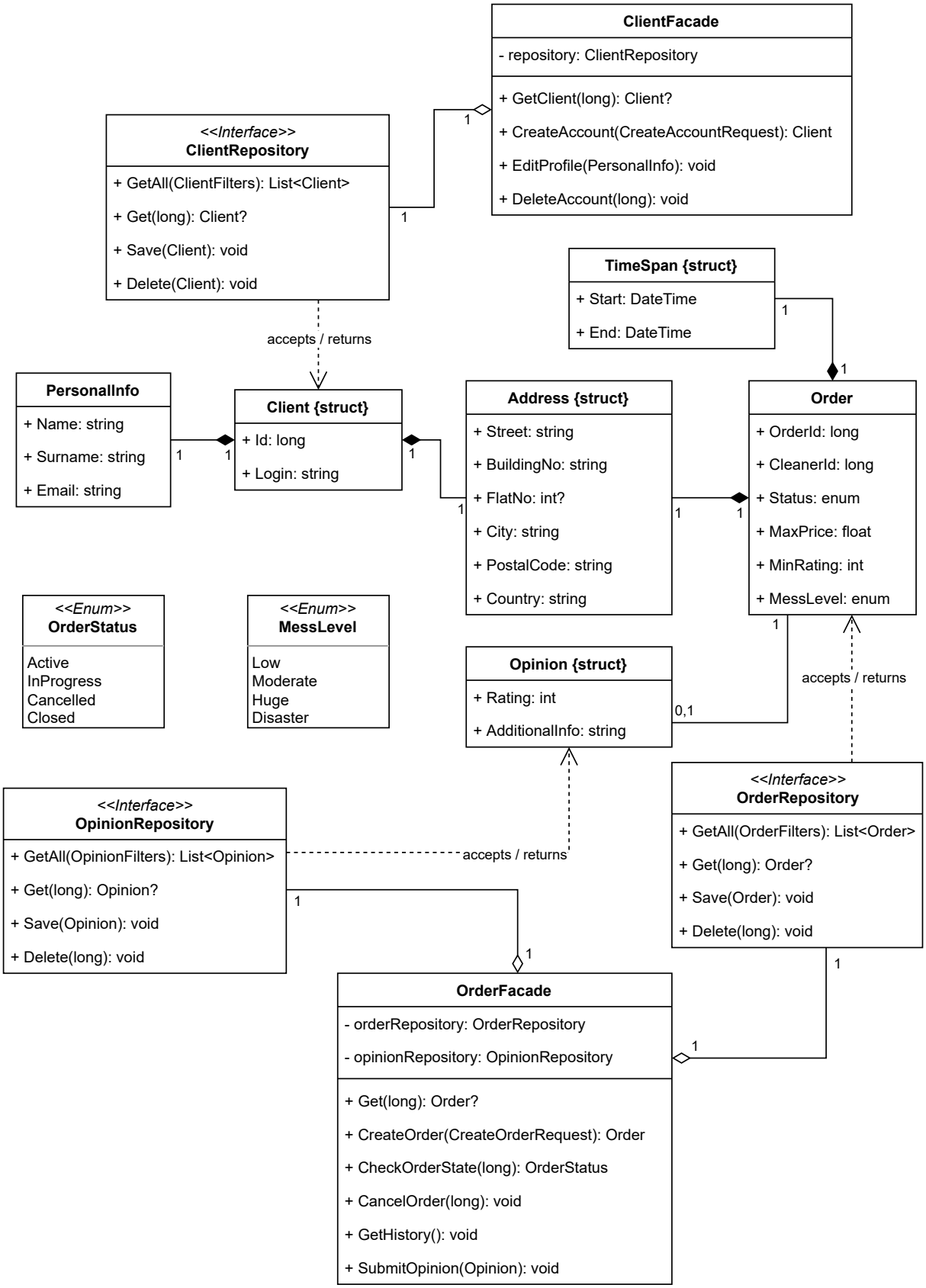
3.2 Typy wyliczeniowe

W każdym module pojawiają dwa typy wyliczeniowe: wyrażający status zamówienia (*OrderStatus*) i opisujący poziom bałaganu do posprzątania (*MessLevel*). Interpretacja drugiego z nich jest samotłumacząca się, natomiast wartości *OrderState* i ich znaczenia są następujące:

- *Active* - zamówienie złożone
- *InProgress* - zamówienie z przypisanym wykonawcą
- *Cancelled* - zamówienie anulowane
- *Closed* - zamówienie zakończone

3.3 Moduł klienta

Klient w systemie reprezentuje osobę, która potrzebuje aby jego mieszkanie było posprzątane. Może on zgłosić zapotrzebowanie na posprzątanie swojej nieruchomości i następnie sprawdzać jego status. Dopuszczalne jest także anulowanie zamówienia, a po zrealizowanym sprzątaniu, klient może wystawić opinie.



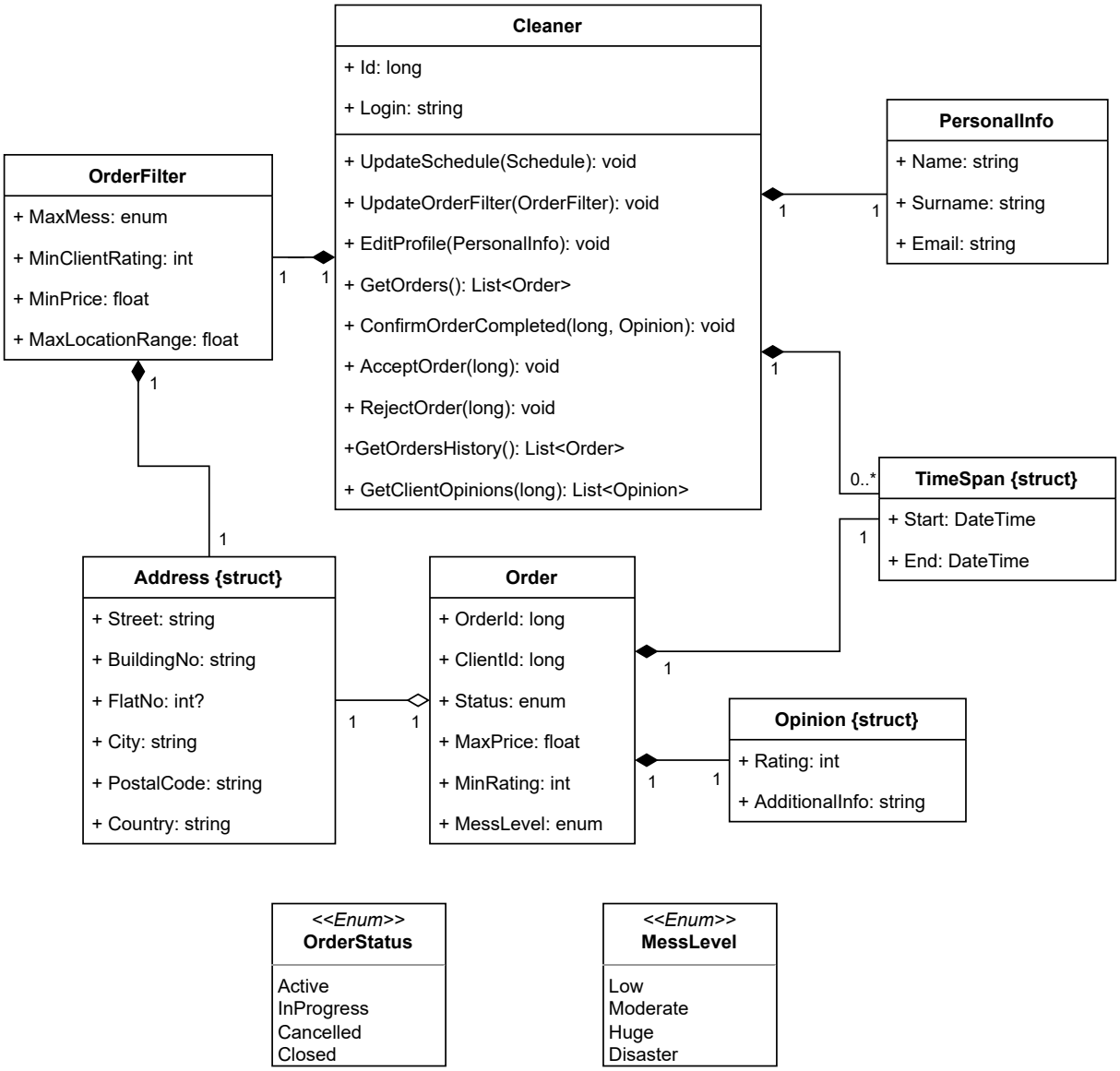
Rysunek 4: Klasy w module klienta.

Tabela 5: Opis klas w module klienta.

Klasa	**Pole/metoda**	**Opis**
ClientFacade	GetClient(long clientId)	Pozwala pobrać klienta po jego id
	CreateAccount(CreateAccountRequest)	Tworzy konto klienta i zwraca obiekt je reprezentujący
	EditProfile(PersonalInfo)	Pozwala zedytować konto klienta
	DeleteAccount(long clientId)	Usuwa konto klienta
OrderFacade	Get(long orderId)	Pozwala pobrać obiekt zamówienia
	CreateOrder(CreateOrderRequest)	Tworzy obiekt zamówienia i zwraca go
	CheckOrderState(long orderId)	Zwraca status zamówienia po jego id
	CancelOrder(long orderId)	Pozwala klientowi anulować zamówienie po danym id
	GetHistory(long clientId)	Pozwala pobrać historie zamówień klienta o podanym id
	SubmitOpinion(Opinion, long orderId)	Pozwala dodać opienię do zamówienia o podanym id
Client	Id	Nadawne przez system id klienta
	Login	Obowiązkowy login - podawany przy logowaniu
Opinion	Rating	Obowiązkowa ocena w sakli od 1 do 5
	AdditionalInfo	Komenatrz do oceny (może być pusty)
Order	Id	Nadawne przez system id zamówienia
	ClientId	Obowiązkowe id klienta, który stworzył zamówienie
	Status	Obecny stan zamówienia
	MaxPrice	Maksymalna cena jaką klient jest w stanie zapłacić
	MinRating	Minimalna ocena sprzątającego, który może podjąć się realizacji
	MessLevel	Orientacyjny poziom bałaganu do posprzątania

3.4 Moduł osoby sprzątającej

Osoba sprzątająca reprezentuje w systemie osobę, która chce podjąć się sprzątania domu lub mieszkania w celach zarobkowych. Może ona ustawić filtr jakie ogłoszenia chce, aby były jej wyświetlane, akceptować je i odrzucać. W celu ułatwienia podjęcia decyzji może sprawdzić dodatkowe informacje na temat ogłoszenia, np. sprawdzić średnią ocenę klienta lub odległość od własnego adresu do adresu ogłoszenia. Po wykonaniu zlecenia może oznaczyć ten fakt w aplikacji wystawiając równocześnie opinię.



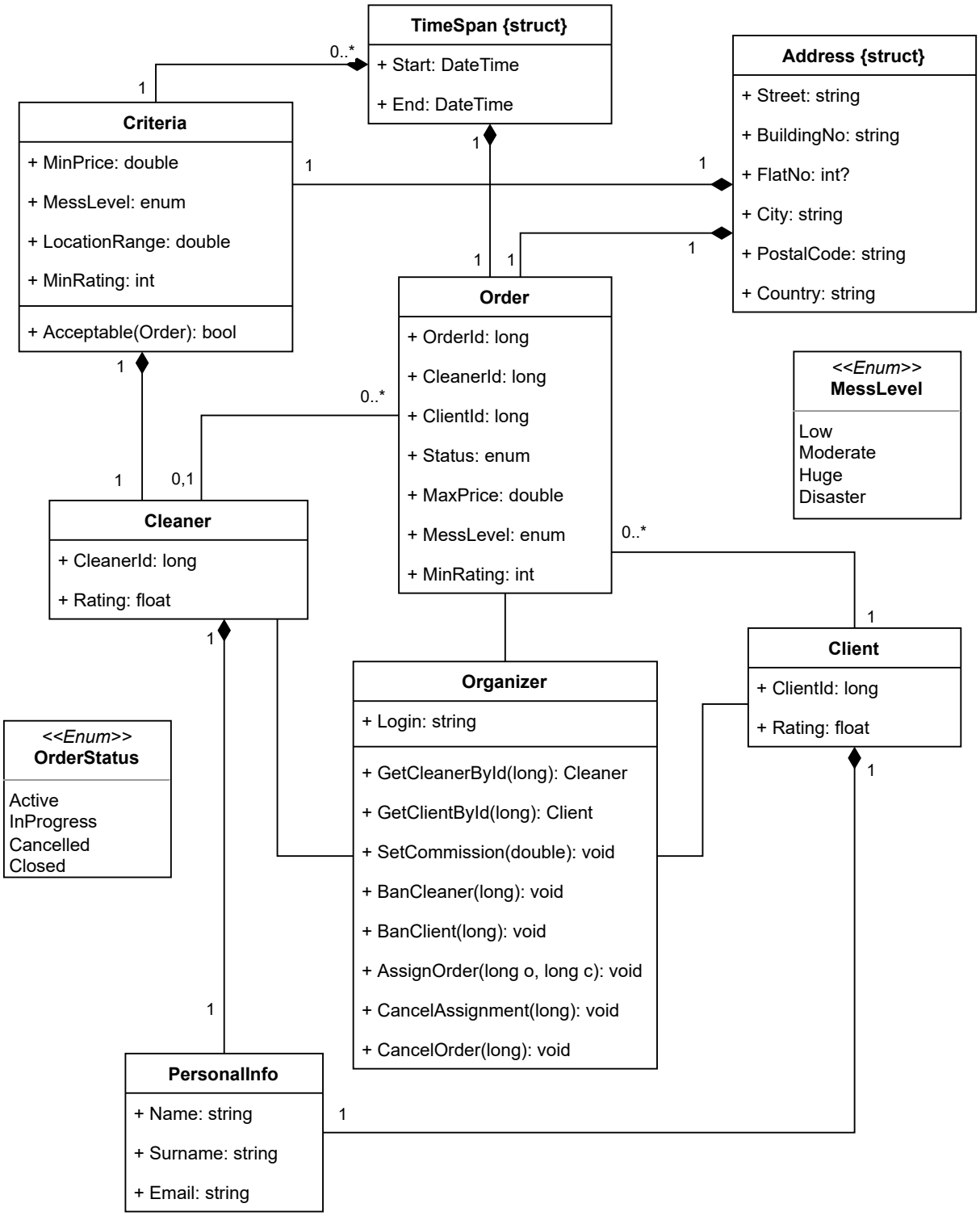
Rysunek 5: Opis klas w module osoby sprzątającej.

Tabela 6: Opis klas w module osoby sprzątającej.

Klasa	**Pole/metoda**	**Opis**
Cleaner	Login	Unikalny login osoby sprzątającej
	Id	Nadawane przez system id osoby sprzątającej
	UpdateSchedule(Schedule)	Aktualizuje grafik dostępności
	UpdateOrderFilter (OrderFilter)	Ustawia filtr ogłoszeń, które będą przesyłane do osoby sprzątającej na podany poprzez argument
	EditProfile(PersonalInfo)	Pozwala ustawić i edytować dane osobowe w profilu na podane poprzez argument
	GetOrders()	Pozwala przejrzeć wszystkie ogłoszenia, które spełniają ustawiony filtr ogłoszeń
	ConfirmOrderCompleted (long, Opinion)	Pozwala oznaczyć ogłoszenie jako zakończone wraz z przesłaniem opinii na temat ogłoszenia
	AcceptOrder(long)	Pozwala osobie sprzątającej zaakceptować przydział do zlecenia
	RejectOrder(long)	Pozwala osobie sprzątającej odrzucić przydział do zlecenia
	GetOrdersHistory()	Zwraca listę ogłoszeń, zrealizowanych przez osobę sprzątającą
	GetClientOpinions(long)	Pozwala osobie sprzątającej sprawdzić opinie wystawione do ogłoszeń danego klienta
Opinion	Rating	Obowiązkowa ocena w skali od 1 do 5
	AdditionalInfo	Komentarz do oceny (może być pusty)
Order	OrderId	Nadawane przez system id zamówienia
	ClientId	Obowiązkowe id klienta, który stworzył zamówienie
	Status	Obecny status zamówienia
	MaxPrice	Maksymalna cena jaką klient jest w stanie zapłacić za wykonanie zlecenia
	MinRating	Minimalna akceptowalna ocena osoby sprzątającej
	MessLevel	Określony przez klienta przybliżony poziom bałaganu
OrderFilter	MaxMess	Filtr maksymalnego poziomu nieczystości które osoba sprzątająca chce sprzątać
	MinClientRating	Filtr minimalnej oceny klienta zakładającego ogłoszenie, które osoba sprzątająca chce przyjmować
	MinPrice	Filtr minimalnej ceny za którą osoba sprzątająca chce się podjąć ogłoszeń
	MaxLocationRange	Filtr maksymalnej odległości od adresu podanego w profilu osoby sprzątającej do adresu ogłoszenia

3.5 Moduł organizatora

Organizator reprezentuje w systemie właściciela aplikacji. Zajmuje się jej administracją - banuje szkodliwych użytkowników, a także przypisuje i anuluje przypisanie ogłoszenia do osoby sprzątającej. Ustala również procent prowizji potrącanej od ceny każdego ogłoszenia na rzecz właściciela aplikacji.



Rysunek 6: Klasy w module organizatora.

Tabela 7: Opis klas w module organizatora.

Klasa	**Pole/metoda**	**Opis**
Organizer	Login	Unikalny login identyfikujący pracownika
	GetCleanerById(long)	Pobiera dane o osobie sprzątającej
	GetClientById(long)	Pobiera dane o kliencie
	SetCommission(double)	Ustawia/aktualizuje wartość naliczanej prowizji
	BanCleaner(long)	Blokuje osobę sprzątającą
	BanClient(long)	Blokuje klienta
	AssignOrder(long o, long c)	Przypisuje osobę sprzątającą c do zlecenia o
	CancelAssignment(long)	Anuluje przypisanie osoby sprzątającej (po zaakceptowaniu przez nią zlecenia)
	CancelOrder(long)	Anuluje zamówienie
Client	ClientId	Unikalne ID klienta
	Rating	Średnia ocena klienta
Cleaner	CleanerId	Unikalne ID osoby sprzątającej
	Rating	Średnia ocena osoby sprzątającej
Order	OrderId	Unikalne ID zamówienia
	ClientId	ID składającego zamówienie klienta
	CleanerId	ID osoby sprzątającej; -1 gdy nie została przydzielona
	Status	Status zamówienia
	MaxPrice	Maksymalna cena, jaką klient jest w stanie zapłacić
	MinRating	Minimalna średnia ocena sprzątających
	MessLevel	Szacunkowy stan bałaganu w miejscu do posprzątania
Criteria	MinPrice	Minimalna cena za zlecenie
	MessLevel	Maksymalny akceptowalny poziom bałaganu
	LocationRange	Maksymalna odległość od zamówienia do adresu podanego w kryteriach
	MinRating	Minimalna ocena klienta, od którego zamówienia mogą być akceptowane
	Acceptable(Order)	Sprawdza czy dane zamówienie spełnia kryteria

4 Stany systemu

4.1 Klient

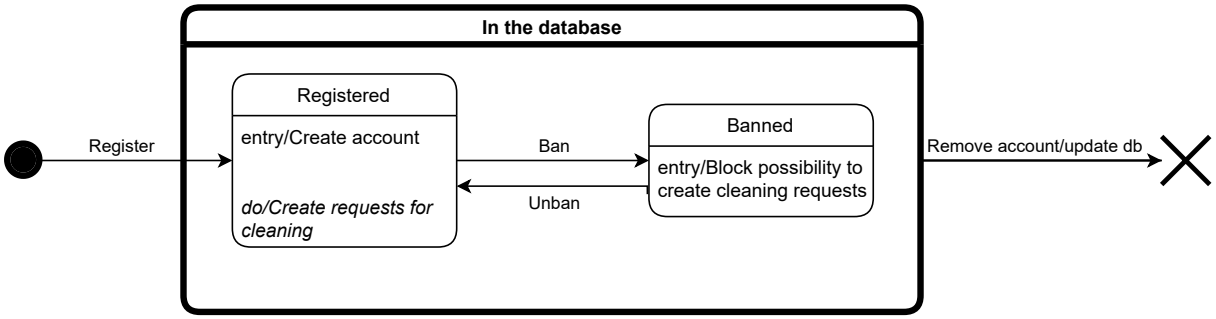
Obiekt klient ma pole "Zablokowany" które może przyjmować jedną z dwóch wartości:

- prawda
- fałsz

Klient chcąc rozpocząć korzystanie z systemu musi najpierw dokonać rejestracji wypełniając takie pola jak imię, nazwisko, adres e-mail oraz login. Po utworzeniu konta klient w przechodzi stan zarejestrowany, którym to ma możliwość tworzenia zleceń na sprzątanie oraz normalnego używania swojego konta, to jest: oceniania sprzątających, przeglądania historii zleceń itp.

W momencie dokonywania przez klienta zabronionych czynności takich jak na przykład tworzenie fałszywych zleceń, nagminne anulowanie zleceń, wykorzystywanie serwisu do celów innych niż przeznaczono, za interwencją administratora następuje zablokowanie konta użytkownika, a co za tym idzie zablokowanie możliwości tworzenia zleceń na sprzątanie. W stanie tym akcje użytkownika są ograniczone. Jedyną możliwością przywrócenia konta do stanu umożliwiającego ponowne tworzenie zleceń jest ponowna interwencja administratora.

Oba te stany zakładają istnienie konta klienta w systemie, natomiast w momencie otrzymania informacji od klienta o prośbie usunięcia jego konta następuje usunięcie go z systemu, a co za tym idzie wszystkich informacji z nim związanych.



Rysunek 7: Diagram stanu klienta.

4.2 Osoba sprzątająca

Obiekt osoby sprzątającej ma pole "Stan" które może przyjmować jedną z trzech wartości:

- Zarejestrowany
- Aktywny
- Zablokowany

Przy czym dwa pierwsze stany logicznie przynależą do stanu "Niezablokowany"

Osoba sprząta chcąc zaoferować swoje usługi w systemie musi dokonać rejestracji w trakcie której musi podać takie informacje jak imię, nazwisko, adres e-mail oraz login. Po utworzenia konta znajduję się ono w stanie zarejestrowanym, tzn istnieje w systemie natomiast nie jest ono wykorzystywanie w procesie dopasowywania zleceń do osób sprzątających.

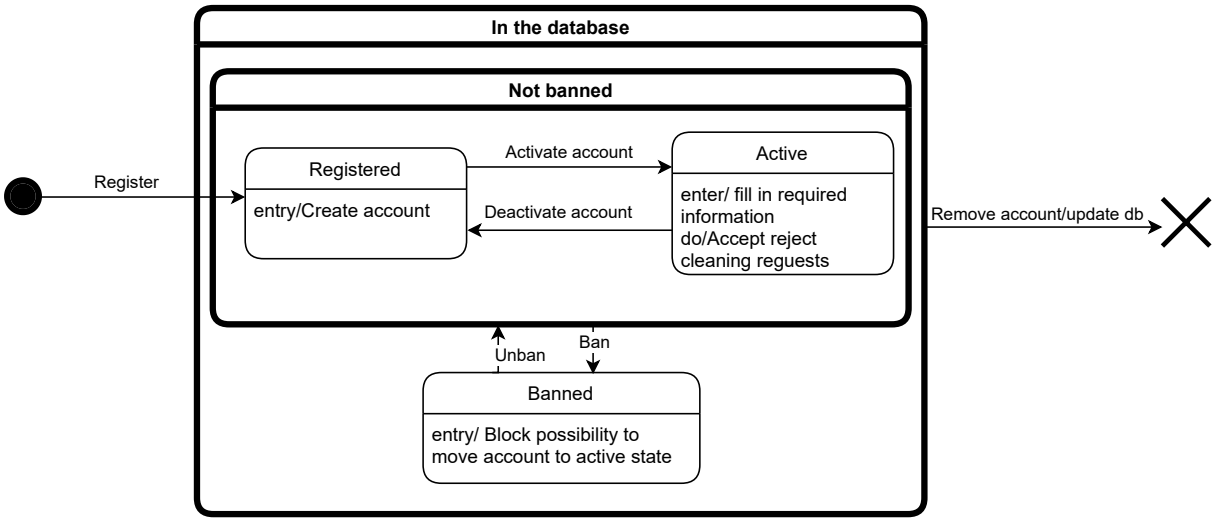
Aby mieć możliwość dostawania zleceń i akceptowania lub odrzucania wymagane jest aby osoba sprzątająca wypełniła dodatkowe informacje potrzebne w procesie parowania zleceń do osób sprzątających. Te informacje to:

- grafik dostępności
- lokalizacja w której mają być przydzielane zlecenia wraz z maksymalną odległością dostępności
- maksymalny bałagan jaki osoba sprzątająca jest w stanie sprzątnąć
- minimalna cena
- minimalna ocena klienta, który wystawia zlecenie

Na podstawie tych kryteriów dokonywane jest dopasowywanie zlecenia do osoby sprzątającej. Po wypełnieniu tych informacji konto osoby sprzątającej przechodzi w stan aktywny, a co za tym idzie jest wykorzystywanie w procesie parowania zleceń z osobami sprzątającymi. Oczywiście w przypadku kiedy osoba sprzątająca chce na jakiś czas zaprzestać wykonywania usług może ona przełączyć konto w stan Zarejestrowany a co za tym idzie zatrzymać otrzymywanie zleceń.

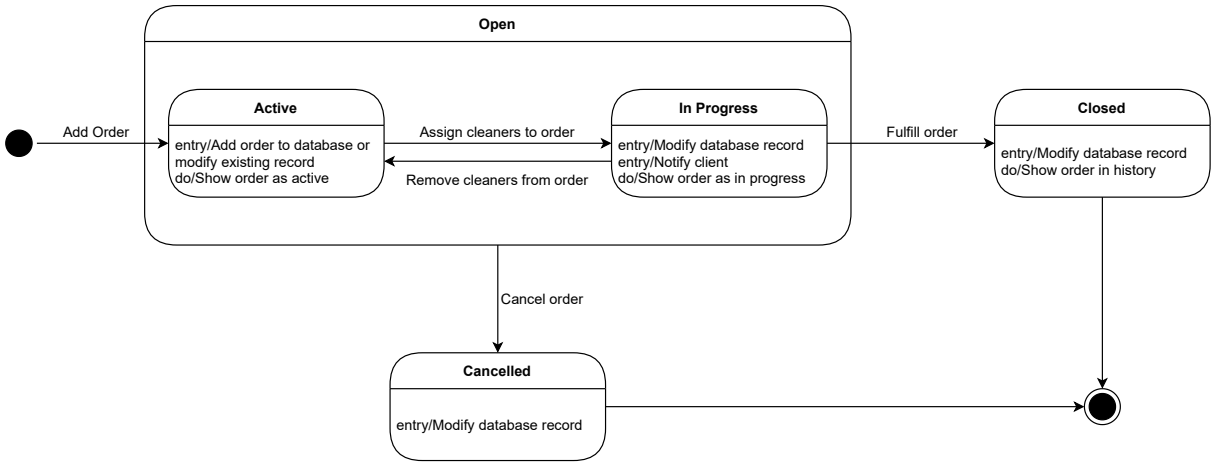
Tak jak w przypadku klienta także osoba sprzątająca podlega moderacji ze strony administratora i w przypadku dokonania zabronionych czynności konto osoby sprzątającej zostaje zablokowane i przechodzi w stan zablokowany z którego nie może, bez ponownej interwencji administratora wyjść. Niezależnie w jakim stanie była osoba sprzątająca przed zablokowaniem, w stanie zablokowanym nie bierze ona udziału w procesie parowania zleceń i nie ma możliwości tego zmienić. Tylko administrator może przenieść osobę sprzątającą do stanu sprzed zablokowania.

Wszystkie te stany zakładają istnienie konta osoby sprzątającej w systemie, natomiast w momencie otrzymania prośby o usunięcia konta następuje usunięcie go z systemu, a co za tym idzie wszystkich informacji z nim związanych.



Rysunek 8: Diagram stanu osoby sprzątającej.

4.3 Zamówienie



Rysunek 9: Diagram stanu zamówienia.

Zamówienie w naszym systemie może znajdować się w czterech stanach:

- Active
- In Progress
- Closed
- Cancelled

Przy czym stany Active i In Progress należą do jednego nadstanu Open. Stan zamówienia jest przedstawiany w systemie w postaci pola Status klasy Order, które przyjmuje wartość odpowiednią dla obecnego stanu.

Po utworzeniu zamówienia przechodzi ono od razu do stanu Active, co oznacza, że zamówienie zostało złożone przez klienta, lecz nie ma jeszcze przypisanego żadnej osoby sprzątającej.

Gdy zamówienie jest w stanie Active mamy możliwość przypisania mu odpowiedniej osoby sprzątającej. Wówczas zamówienie przechodzi w stan In Progress, co oznacza, że może ono zostać wypełnione. Nasz system pozwala na usunięcie osoby sprzątającej z zamówienia, co skutkuje powrotem do stanu Active.

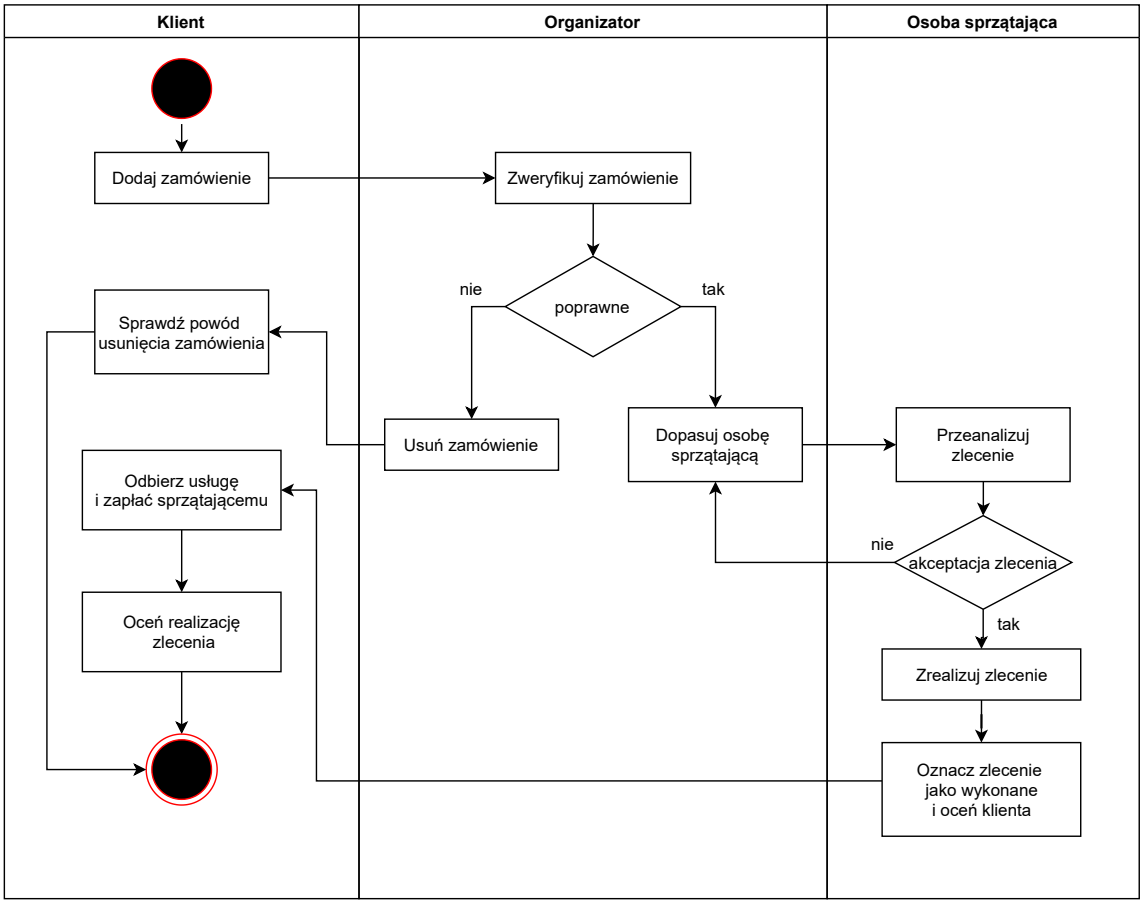
Gdy zamówienie zostanie wypełnione przez osobę sprzątającą oraz zweryfikowane, przechodzi ono na stałe do stanu Closed.

W naszym systemie dajemy klientowi możliwość podjęcia decyzji o anulowaniu zamówienia, lecz ta decyzja musi być zatwierdzona, aby uniknąć sytuacji, w której klient anuluje zamówienie w momencie kiedy jest już ono częściowo wykonane. Zamówienie po anulowaniu przechodzi na stałe do stanu Cancelled.

5 Aktywności systemu

5.1 Proces realizacji zamówienia

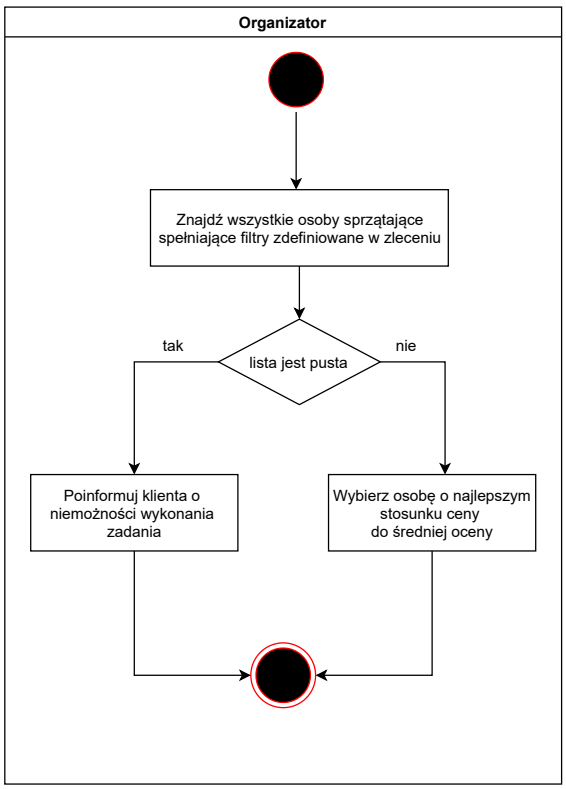
Kluczowym elementem, na którym opiera się opisywany system, jest proces realizacji zamówienia. Proces ten rozpoczyna klient, wypełniając formularz zamówienia i wysyłając go. Każde zamówienie musi zostać zweryfikowane przez organizatora pod kątem zgodności z regulaminem serwisu. Jeśli weryfikacja zostanie pomyślnie zakończona to dopasowywana jest osoba sprzątająca, która może przyjąć lub odrzucić zamówienie. Zauważmy, że może nie udać się znaleźć odpowiedniej osoby w wyznaczonym czasie, klient zostanie wtedy poinformowany o braku możliwości realizacji zamówienia. Po znalezieniu odpowiedniej osoby powinna ona przystąpić do realizacji. Po realizacji obie strony mogą się wzajemnie ocenić. Klient ocenia jakość usługi, a sprzątający zgodność stanu faktycznego z opisem zlecenia. Możliwa jest także sytuacja, w której zamówienie nie przejdzie weryfikacji, klient ma wtedy możliwość poznania przyczyny. W każdym momencie procesowania zamówienia klient może sprawdzić jego status a także je wycofać, co nie zostało, dla czytelności, uwzględnione na diagramie. Wszelkie sytuacje niestandardowe będą załatwiane poprzez bezpośredni kontakt mailowy z organizatorem.



Rysunek 10: Procesowanie zamówienia w systemie.

5.2 Przypisanie sprzątającego

Aby zamówienie mogło zostać zrealizowane, organizator musi wyznaczyć odpowiednią osobę sprzątającą, która spełnia kryteria określone przez klienta. Jeśli w danym momencie niewielu sprzątających zadeklarowało dostępność, lub kryteria ustawiona przez klienta są zbyt wysokie to może nie udać się znaleźć odpowiedniej osoby. W takiej sytuacji organizator może wstrzymać się z aktualizacją statusu zlecenia lub, jeśli czas realizacji upływa niebawem, zakomunikować klientowi brak możliwości sprzątania. W przeciwnym przypadku, gdy zostanie znaleziona odpowiednia osoba, organizator wysyła do niej ofertę realizacji zlecenia i czeka na odpowiedź. Możliwe, że w niektórych przypadkach więcej niż jeden kandydat będzie pasował do zamówienia, wówczas organizator podejmuje decyzje kierując się ceną i średnią z ocen.



Rysunek 11: Przypisywanie sprzątającego do zamówienia.

6 Komunikacja w systemie

6.1 Wprowadzenie

Aplikacja dzieli się na 4 fizyczne moduły: klienta, osoby sprzątającej, organizatora oraz serwer (bazy danych). Serwer zajmuję się cała wymianą informacji pomiędzy resztą modułów, tzn. wszystkie zapytania z innych modułów są wysyłane w stronę serwera, który to za pomocą bazy danych przekazuje informacje resztę modułów.

Komunikacja jest realizowana za pomocą REST API oraz zapytań HTTPS i w ogólności wygląda następująco:

- Moduł X wysyła zapytanie do serwera z prośbą o wykonanie danej akcji.
- Serwer w razie potrzeby zapisuje zmiany w bazie danych
- Moduł Y, który periodycznie nasłuchuje na interesujące go zmiany otrzymuje informacje o tej zmianie, jeśli jest ona dla niego istotna i również może wysłać zapytanie na serwer, by zareagować na zmiany modułu X.

Dodatkowo w celach autoryzacji przychodzących zapytań, wymagane będzie zawarcie w wiadomości api-key które posłuży do weryfikacji tożsamości, oraz weryfikacji czy zapytanie jest dostępne dla konkretnego modułu, co oznacza że np. klient nie będzie w stanie realizować zapytań, przypisanych organizatorowi lub osobie sprzątającej.

6.2 Możliwe sytuacje wyjątkowe - awarie

Wszystkie moduły aplikacji wymieniają między sobą informacje wyłącznie z wykorzystaniem pośrednika w postaci serwera, dlatego ewentualna awaria któregoś z modułów jest niezauważalna z punktu widzenia pozostałych modułów. W przypadku wykrycia takiej awarii użytkownik zostaje poinformowany o zaistniałym problemie, a stosowna informacja zostaje wysłana do osób odpowiedzialnych za konserwację aplikacji, których zadaniem jest podjęcie stosownej akcji.

Inaczej dzieje się natomiast, gdy awarii ulegnie serwer. W takiej sytuacji każdy niepomyślny request użytkownika wymagający wymiany informacji z serwerem zostaje po upłynięciu niewielkiego odstępu czasu powtórzony. Jeśli po 5 próbach serwer w dalszym ciągu nie odpowiada, użytkownikowi wyświetlany zostaje stosowny komunikat z prośbą o ponowienie operacji w późniejszym czasie. W tym czasie osoby odpowiedzialne za konserwacje aplikacji zajmują się naprawieniem usterki.

Dodatkowo należy zwrócić uwagę na to, że wszystkie zapytania do serwera, a co za tym idzie zapytania do bazy, wykorzystują Entity Framework Core. Dzięki takiemu podejściu są one przetwarzane w sposób transakcyjny, stąd niemożliwa jest sytuacja, aby dane zwrócone przez serwer były niepoprawne lub nieaktywne.

6.3 Schematy komunikacji

6.3.1 Rejestracja, zmiana hasła, banowanie

Moduł klienta zarówno jak moduł osoby sprzątającej, zanim zaoferują pełną funkcjonalność użytkownikowi wymagają rejestracji. Jest to jedno z niewielu zapytań które nie będzie wymagało zawarcia api-key w zapytaniu. Użytkownik wysyła zapytanie POST, zawierając w nim formularz rejestracyjny, który następnie jest walidowany po stronie serwera. W przypadku poprawnych danych serwer dodaje użytkowania do bazy danych i informuje go o pomyślnej rejestracji. Jeśli nadesłane dane są niepoprawne, serwer nie rejestruje użytkownika i odsyła mu informacje o błędzie. Docelowo realizacja tej funckcjonalności nastąpi za pomocą Azure Active Directory (więcej w rozdziale Dokumentacja API).

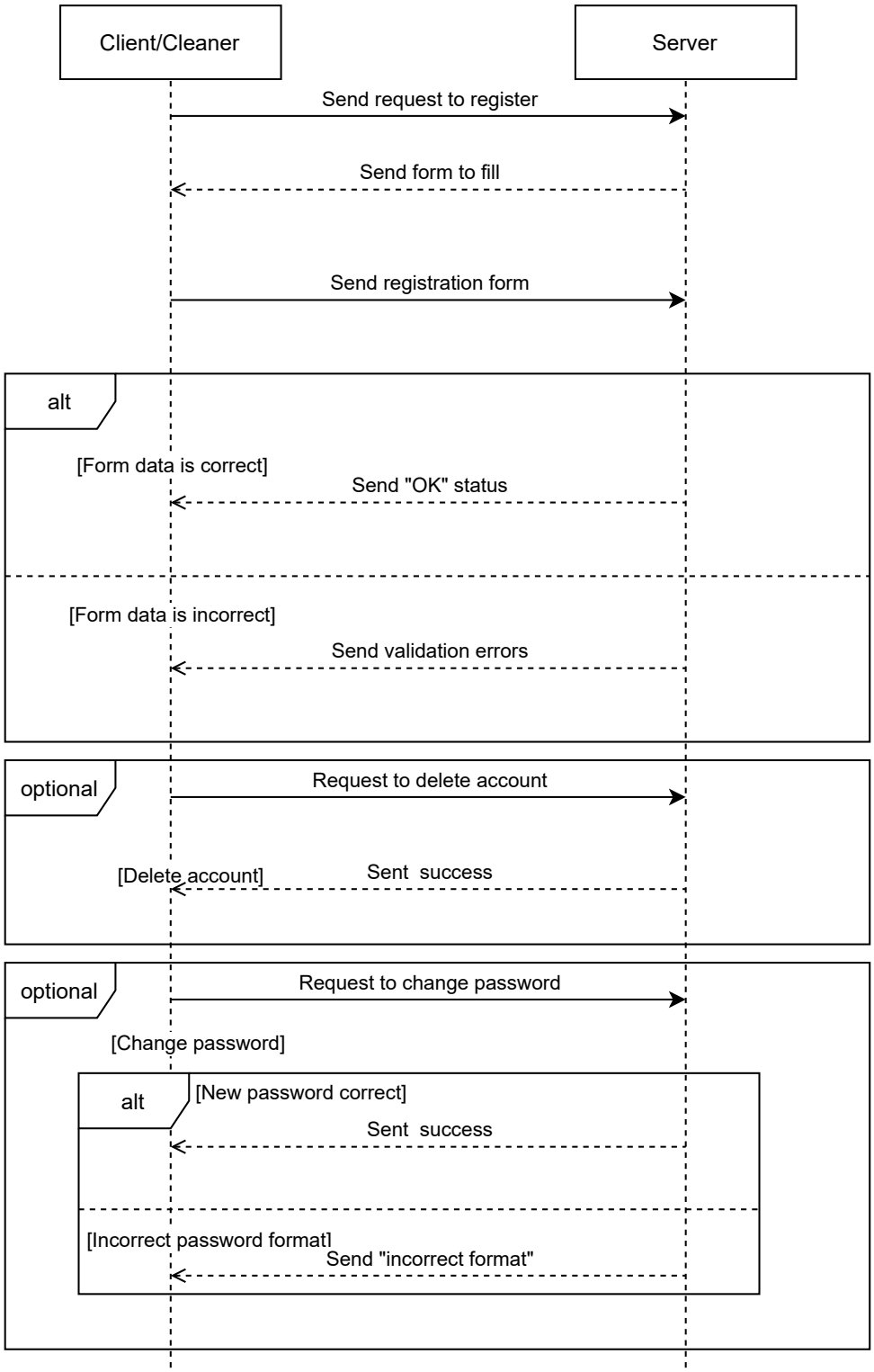
Dodatkowo, użytkownik za pomocą zapytania POST może zmienić swoje hasło dostępu do konta, oraz za pomocą zapytania DELETE usunąć swoje konto. Obie te rzeczy będą realizowane za pomocą Azure AD B2C, a funkcjonalność po stronie aplikacji ograniczy się do przekierowania do odpowiedniej strony lub

wysłania konkretnego zapytanie do platformy Azure AD B2C. Zgodnie z obecnym standardem użytkownik chcący zmienić hasło zostanie przekierowany na odpowiednią stronę, na której będzie wymagane ponowne zalogowanie, a następnie pojawi się możliwość zmiany hasła.

Również banowanie użytkowników będzie realizowane za pomocą platformy Azure AD B2C. Organizator z poziomu swojego modułu może wysłać zapytanie na serwer z informacją o tym jakiego użytkownika zbanować, natomiast serwer wyśle zapytanie do servera Azure i ustawi użytkownika jako zbanowanego. Następnie taki użytkownik po zalogowaniu będzie miał zablokowaną możliwość dodawania nowych zleceń - czyli w praktyce jego interakcje z systemem zostaną ograniczone do monitorowania obecnych zleceń (które także mogą być wycofane przez organizatora jeśli to one były przyczyną bana).

6.3.2 Logowanie

Logowanie we wszystkich modułach odbywa się z użyciem zewnętrznego API, które to odpowiada za uwierzytelnienie i autoryzacje użytkownika. Posłuży do tego pojedyncze zapytanie POST z danymi logowania. W odpowiedzi API zwraca token, który następnie dołączany będzie do wszystkich innych zapytań w stronę serwera. Posłuży on do kontroli czy dana osoba ma uprawnienia do wysłanego zapytania, oraz do odpowiedniego wykonania zapytania w bazie.

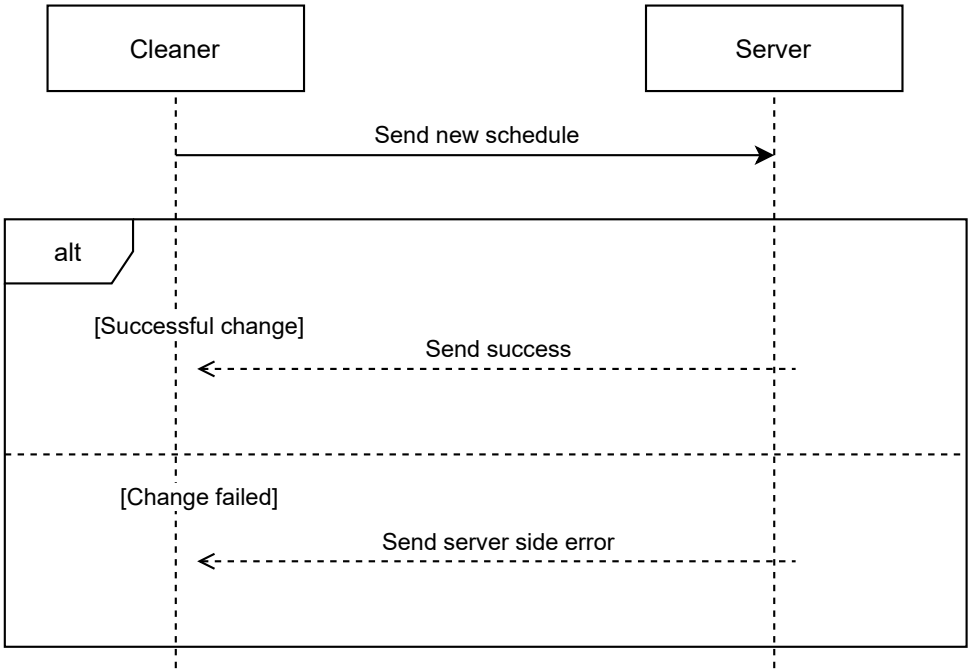


Rysunek 12: Komunikacja w procesie rejestracji klienta i osoby sprzątającej.

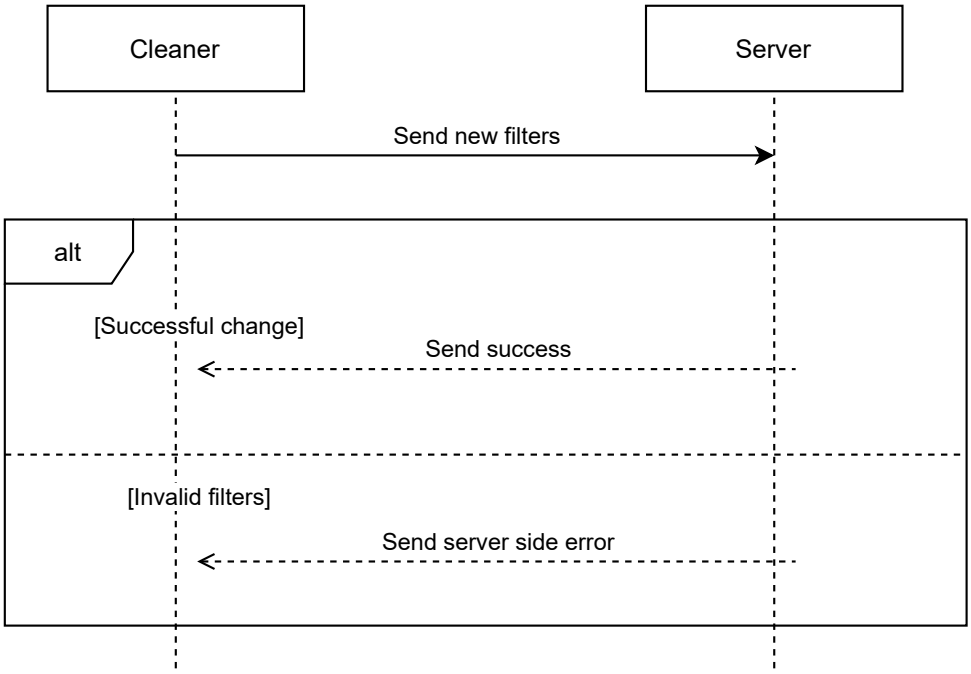
6.3.3 Uzupełnianie grafiku, zmiana filtrów i inne proste akcje w systemie

Osoba sprzątająca za pomocą pojedynczego zapytania POST może zmienić grafik, używany przy dopasowywaniu dla niej ofert sprzątania. Podobnie sprawa wygląda z aktualizacji filtrów używanych do dopasowywania zamówień. Obie te akcji sprowadzają się do wysłania jednego zapytania POST. Takich akcji w systemie jest dużo, natomiast wszystkie diagramy komunikacji takich aktywności są bliźniacze, dlatego reszta tych zapytań opisana została dokładnie w rozdziale "Opis API" wraz z dokładnym opisem,

co takie zapytania zawierają i jakich odpowiedzi oczekują.



Rysunek 13: Komunikacja w procesie uzupełniania grafiku.

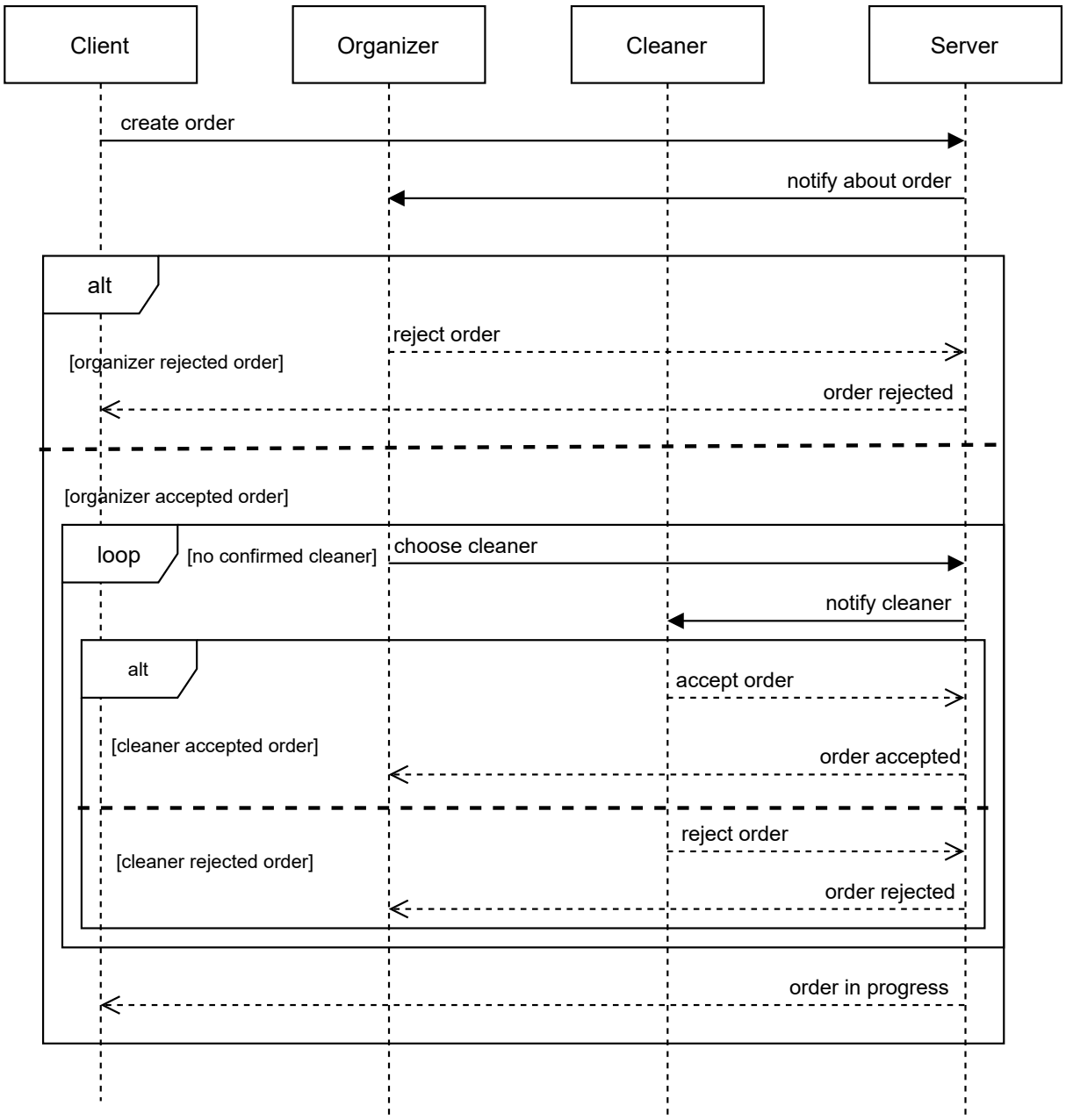


Rysunek 14: Komunikacja w procesie aktualizacji filtrów ofert.

6.3.4 Tworzenie i obsługa zamówienia

Klient może stworzyć zamówienie przesyłając zapytanie POST do serwera, który dodaje zamówienie do bazy danych. Gdy zamówienie znajdzie się w bazie danych, do jego weryfikacji może przystąpić organizator. W przypadku gdy organizator chce odrzucić zamówienie wysyła zapytanie POST i serwer zmienia stan zamówienia na Cancelled. W przypadku akceptacji zamówienia organizator przystępuje do szuka-

nia osoby sprzątającej. Gdy organizator wybierze odpowiednią osobę, wysyła zapytanie POST, które przypisze ją do zamówienia. Moduł osoby sprzątającej zauważając, że użytkownik został przypisany do zamówienia pyta go czy chce się podjąć zamówienia. Akceptacja jest realizowana wysłaniem zapytania POST, które zmienia stan zamówienia na In Progress, zaś odmowa przebiega w postaci zapytania POST, które usuwa osobę sprzątającą z zamówienia. Organizator szuka osoby sprzątającej tak długo, aż któryś użytkownik zaakceptuje zamówienie.



Rysunek 15: Komunikacja w procesie tworzenia i obsługi zamówienia.

7 Opis API

Dokumentacja API modułu bazy przygotowana została przy użyciu specyfikacji OpenAPI 2.0.0.

7.1 Rejestracja, logowanie, banowanie i obsługa kont klienckich

Dokumentacja w sposób celowy nie zawiera opisu autoryzacji, przechowywania informacji o użytkowniku, zmiany hasła, banowania użytkowników. Wszystkie te elementy będą zrealizowane za pomocą Azure Active Directory B2C (zwanego dalej AAD) i zawartego tam mechanizmowi opcjonalnych claims'ów. Po zalogowaniu się do portalu AAD do zapytań wysyłanych w kierunku serwera dołączany będzie token otrzymany z AAD, który będzie zawierał potrzebne informacje na temat użytkownika i który (w uproszczeniu, gdyż AAD dołącza również pola na swoje własne potrzeby) będzie miał następującą postać:

```
Token:
    type: "object"
    properties:
      oid:
        type: "string"
        description: "unique guid, used to distinct registered users"
      userType:
        $ref: "#/definitions/UserType"
      email:
        type: "string"
      name:
        type: "string"
      surname:
        type: "string"
      address:
        $ref: "#/definitions/Address"
      isBanned:
        type: "boolean"
```

Dokładny opis klas zawartych w tokenie został podany niżej razem z opisem całego api. Z racji na to, że usługa Azure Active Directory B2C jest cały czas rozwijana, modyfikowana i ulepszana opis api nie zawiera dokładnego opisu wiadomości wysyłanych z naszej aplikacji do AAD. Dodatkowo wiadomości te będą wysyłane za pomocą zewnętrznego api (opisanego dokładniej w rozdziale Technologia), zatem z punktu widzenia rozwijanego oprogramowania dokładny opis tych wiadomości oraz struktur nie jest istotny.

Jak widać wszystkie "stałe" informacje będą zawarte w tokenie pobranym z AAD, natomiast informacje które często mogą ulegać zmianie jak grafik, filtry itp. zostały opisane niżej w dokładnej dokumentacji API. również usuwanie klienta, ze względu na ograniczenia AAD, będzie realizowany w standardowy sposób poprzez wysłanie zapytania na moduł serwera.

7.2 Dokładna dokumentacja API

Dla wszystkich poniższych requestów zakładamy że mają one postać: [nazwa aplikacji]/api/[reszta url]

7.2.1 Obsługa zamówień

```
  /orders:
    get:
      summary: "Get all orders (for organizer)"
      produces:
```

```
      - "application/json"
    responses:
      "200":
        description: "OK"
        schema:
          type: "array"
          items:
            $ref: "#/definitions/Order"
      "403":
        description: "Not Authorised"
  post:
    summary: "Add order"
    consumes:
      - "application/json"
    parameters:
      - name: "body"
        in: body
        schema:
          $ref: "#/definitions/NewOrder"
    responses:
      "200":
        description: "OK"
      "403":
        description: "Not Authorised"
      "400":
        description: "Bad Request"
/orders/{orderId}:
  get:
    summary: "Get order info"
    produces:
      - "application/json"
    responses:
      "200":
        description: "OK"
        schema:
          $ref: "#/definitions/Order"
      "403":
        description: "Not Authorised"
  post:
    summary: "Modify order info"
    consumes:
      - "application/json"
    parameters:
      - name: "body"
        in: body
        schema:
          $ref: "#/definitions/Order"
    responses:
      "200":
        description: "OK"
      "403":
```

```
        description: "Not Authorised"
      "400":
        description: "Bad Request"
  delete:
    summary: "Removes offer"
    consumes:
      - "application/json"
    responses:
      "200":
        description: "OK"
      "403":
        description: "Not Authorised"
/orders/cleaner:
  get:
    summary: "Get assigned orders"
    produces:
      - "application/json"
    responses:
      "200":
        description: "OK"
        schema:
          type: "array"
          items:
            $ref: "#/definitions/Order"
      "403":
        description: "Not Authorised"
/orders/client:
  get:
    summary: "Get created orders"
    produces:
      - "application/json"
    responses:
      "200":
        description: "OK"
        schema:
          type: "array"
          items:
            $ref: "#/definitions/Order"
      "403":
        description: "Not Authorised"
```

7.2.2 Operacje konta osoby sprzątającej

```
/cleaner/{cleanerId}:
  get:
    summary: "retrieves cleaner information including filters and
      schedule"
    consumes:
      - "application/json"
    responses:
      "200":
```

```
        description: "OK"
        schema:
          $ref: "#/definitions/CleanerInfo"
      "403":
        description: "Not Authorised"
  post:
    summary: "updates cleaner information"
    consumes:
      - "application/json"
    parameters:
      - name: "body"
        in: body
        schema:
          $ref: "#/definitions/CleanerInfo"
    responses:
      "200":
        description: "OK"
      "403":
        description: "Not Authorised"
      "400":
        description: "Bad Request"
```

7.2.3 Operacje dla kont użytkowników(klienta i osoby sprzątającej)

```
/client/{id}:
  delete:
    summary: "Deletes account from the system"
    produces:
    - "application/json"
    responses:
      "200":
        description: "OK"
      "403":
        description: "Not Authorised"
/user/{id}/rate:
  post:
    summary: "rates client/cleaner by opposite side (in connection to
      execution of an order)"
    produces:
    - "application/json"
    parameters:
      - name: "body"
        in: body
        schema:
          $ref: "#/definitions/AddRating"
    responses:
      "200":
        description: "OK"
      "403":
        description: "Not Authorised"
```

```
      "400":
        description: "Bad Request"
  /user/{id}/ban:
    post:
      summary: "ban client/cleaner"
      produces:
      - "application/json"
      responses:
        "200":
          description: "OK"
        "403":
          description: "Not Authorised"
```

7.2.4 Ustawianie Prowizji

```
  /Configuration/Commission:
    get:
    summary: "Get commission value"
      produces:
        - "application/json"
      responses:
        "200":
          description: "OK"
          schema:
            $ref: "#/definitions/SetCommission"
    post:
      summary: "sets commision for the orders"
      produces:
      - "application/json"
      parameters:
        - name: "body"
          in: body
          schema:
            $ref: "#/definitions/SetCommission"
      responses:
        "200":
          description: "OK"
        "403":
          description: "Not Authorised"
        "400":
          description: "Bad Request"
```

7.2.5 Definicje Klas

```
definitions:
  MessLevel:
    type: "string"
    enum:
      - "Low"
```

```
      - "Moderate"
      - "Huge"
      - "Disaster"
OrderStatus:
  type: "string"
  enum:
    - "Active"
    - "InProgress"
    - "Cancelled"
    - "Closed"
Order:
  type: "object"
  properties:
    id:
      type: "integer"
      format: "int64"
    clientId:
      type: "string"
    cleanerId:
      type: "string"
    status:
      $ref: "#/definitions/OrderStatus"
    maxPrice:
      type: "number"
    minRating:
      type: "integer"
    date:
      type: "DateTimeOffset"
    messLevel:
      $ref: "#/definitions/MessLevel"
    address:
      $ref: "#/definitions/Address"
NewOrder:
  type: "object"
  properties:
    clientId:
      type: "string"
    maxPrice:
      type: "number"
    minRating:
      type: "integer"
    messLevel:
      $ref: "#/definitions/MessLevel"
    date:
      type: "DateTimeOffset"
    address:
      $ref: "#/definitions/Address"
UserType:
  type: "string"
  enum:
    - "Client"
```

```
    - "Cleaner"
    - "Administrator"
Address:
  type: "object"
  properties:
    street:
      type: "string"
    buildingNo:
      type: "string"
    flatNo:
      type: "integer"
      nullable: true
    city:
      type: "string"
    postalCode:
      type: "string"
    country:
      type: "string"
Token:
  type: "object"
  description: "fields that are included in the b2c claims that will be
      neccessary to process requests"
  properties:
    oid:
      type: "string"
      description: "unique guid, used to disctinct registered users"
    userType:
      $ref: "#/definitions/UserType"
    email:
      type: "string"
    name:
      type: "string"
    surname:
      type: "string"
    address:
      $ref: "#/definitions/Address"
    isBanned:
      type: "boolean"
DayOfWeek:
  type: "string"
  enum:
    - "Monday"
    - "Tuesday"
    - "Wednesday"
    - "Thursday"
    - "Friday"
    - "Saturday"
    - "Sunday"
ScheduleEntry:
  type: "object"
  description: "single entry of Schedule"
```

```
    properties:
      dayOfWeek:
        $ref: "#/definitions/DayOfWeek"
      start:
        type: "string"
        description: "Time in format HH:MM from which avaliability starts"
      end:
        type: "string"
        description: "Time in format HH:MM on which avaliablity ends"
  CleanerInfo:
    type: "object"
    description: "cleaner information necessary to match orders"
    properties:
      scheduleEntries:
        type: "array"
        items:
          $ref: "#/definitions/ScheduleEntry"
      maxMess:
        $ref: "#/definitions/MessLevel"
      minClientRating:
        type: "integer"
      minPrice:
        type: "number"
        format: "float"
      maxLocationRange:
        type: "number"
        format: "float"
  SetCommission:
    type: "object"
    properties:
      newProvision:
        description: "new commision in fraction"
        type: "number"
        format: "decimal"
        maximum: 1.0
        minimum: 0.0
  AddRating:
    type: "object"
    properties:
      rating:
        type: "integer"
        maximum: 10
        minimum: 1
      comment:
        type: "string"
```

8 Scenariusze testowe

Poniżej przedstawione zostały scenariusze testowe dla większości skonstruowanych *User stories*. Zagadnienia związane z logowaniem, rejestracją itp. są zapewniane przez zewnętrznego dostawcę, więc nie znalazły odzwierciedlenia w zaplanowanych scenariuszach testowych.

8.1 Klient

Poniższe czynności są częścią zabezpieczonego widoku, do których dostęp zyskuje się poprzez rejestrację i logowanie. Przy wystąpieniu ewentualnych błędów klientowi wyświetla się informatywny komunikat i prośbę o próbę ponownego wykonania danej akcji.

8.1.1 Złożenie zamówienia

Po wypełnieniu formularza zamówienia i jego walidacji do serwera przesyłane są dane nowego zamówienia. Klientowi wyświetla się stosowną informację zwrotną.

8.1.2 Sprawdzenie statusu zamówienia

Jeśli klient złożył zamówienie i jest ono aktywne, to może on sprawdzić jego status. Po odpytaniu serwera wyświetla się jedna z możliwych wartości.

8.1.3 Anulowanie zamówienia

Jeśli klient ma aktywne zamówienie i nie zostało ono jeszcze zaakceptowane przez osobę sprzątającą, to może je anulować, co skutkuje odpowiednim zapytaniem do modułu serwera.

8.1.4 Przeglądanie historii zamówień

Klient może wybrać widok historii zamówień, w wyniku czego aplikacja odpytuje serwer o jego zamówienia i wyświetla listę lub informuje o braku jakichkolwiek zleceń w systemie.

8.1.5 Wystawianie opinii

Jeśli jedno z zamówień klienta zostało zrealizowane, to może on wystawić do niego opinię, co również skutkuje wysłaniem zapytania o aktualizację danych sprzątającego w bazie.

8.2 Organizator

8.2.1 Łączenie zamówień

W momencie gdy w systemie istnieje co najmniej 1 klient z aktywnym zamówieniem, które nie ma jeszcze przydzielonego wykonawcy oraz dostępna jest co najmniej jedna osoba sprzątająca, to organizator może podjąć próbę przydzielenia wykonawcy do danego zamówienia. Po rozpoczęciu takiego procesu moduł organizatora odpytuje serwer o listę osób sprzątających pasujących do kryteriów zamówienia. Jeśli zwrócona lista będzie niepusta, to nastąpi próba przypisania wykonawcy, poprzez wysłanie odpowiedniego komunikatu do serwera. W przeciwnym przypadku organizator przerywa proces lub rozpoczyna go ponownie.

8.2.2 Sprawdzenie stanu akceptacji

Jeśli w systemie jest zlecenie, dla którego podjęto próbę dopasowania osoby sprzątającej, niezależnie czy zakończyła się, czy jeszcze nie, organizator może sprawdzić status akceptacji przydziału. Dzieje się to poprzez odpytanie modułu serwera.

8.2.3 Anulowanie przydziału

Jeśli osoba sprzątająca zaakceptowała zlecenie, ale nie zostało ono jeszcze zrealizowane, to organizator ma możliwość dla takiego zlecenia anulować przydział, wysyłając do modułu serwera zapytanie skutkujące usunięciem przydziału w bazie danych.

8.2.4 Ustalanie prowizji

Po zalogowaniu do systemu, organizator ma możliwość zmiany prowizji dla nowych zleceń. Odbywa się to poprzez wysłanie do serwera zapytania z nową wartością prowizji.

8.2.5 Banowanie użytkowników

Jeśli w systemie są zarejestrowani jacyś klienci lub osoby sprzątające, organizator może wskazać, aby ktoś z nich został zbanowany. Wówczas wysyłany jest do serwera request z informacją o tym kogo należy zablokować.

8.2.6 Przeglądanie danych użytkowników

Po zalogowaniu do systemu organizator może wybrać opcję przeglądania danych użytkowników danego typu. Odpytuje w tym celu serwer o listę z danymi klientów lub osób sprzątających. Wyświetla mu się lista obiektów lub informacja o braku zarejestrowanych użytkowników.

8.3 Osoba sprzątająca

8.3.1 Ustawienie preferencji

Zalogowana osoba sprzątająca ma możliwość ustawienia preferencji zleceń, jakie chciałaby otrzymywać. Po wprowadzeniu nowych danych do odpowiedniego formularza są one przesyłane w zapytaniu do serwera, który uaktualnia wpis w bazie.

8.3.2 Edycja grafiku

Po zalogowaniu osoba sprzątająca ma również dostęp do swojego grafiku. Jeśli zmodyfikuje go i wybierze opcję zapisania, do modułu serwera przesłane zostanie zapytanie z danymi o nowym grafiku. Serwer porównuje je z wcześniejszym grafikiem i aktualizuje wpisy w bazie danych.

8.3.3 Akceptacja/odrzucenie zlecenia

Jeśli organizator przyporządkuje osobie sprzątającej zlecenie, dostanie ona informację o nowym zleceniu oczekującym na zaakceptowanie. Zarówno akceptacja jak i odrzucenie powodują wysłanie do modułu serwera odpowiedniej informacji.

8.3.4 Sprawdzenie szczegółów zlecenia

Jeśli spełnione są warunki opisane powyżej, to osoba sprzątająca może dokonać sprawdzenia szczegółowego opisu nowo otrzymanego zlecenia. Zapytanie o przesłanie szczegółowych danych o zamówieniu wysyłane są do serwera, a sformatowana zawartość odpowiedzi jest przedstawiana potencjalnemu zleceniobiorcy.

8.3.5 Kontakt z organizatorem

W każdym momencie po zalogowaniu dostępna jest dla sprzątającego możliwość sprawdzenia kontaktu do organizatora. Moduł osoby sprzątającej odpytuje moduł serwera o aktualne dane organizatora i wyświetla je użytkownikowi.

8.3.6 Przeglądanie historii zleceń

Zalogowana osoba sprzątająca może sprawdzić historię swoich zleceń. Po przesłaniu odpowiedniego zapytania do modułu serwera dostaje ona listę zaakceptowanych w przeszłości zleceń.

8.3.7 Potwierdzenie wykonania zlecenia

Po zaakceptowaniu zlecenia (i jeśli przydział nie został anulowany) osoba sprzątająca może oznaczyć zlecenie jako wykonane. Skutkuje to wysłaniem zapytania o aktualizację statusu zamówienia do modułu serwera.

8.3.8 Wystawienie opinii

Po zakończeniu danego zlecenia osoba sprzątająca może również wystawić opinię dla przyjętego ogłoszenia. Dane takiej opinii przesyłane są do serwera, który dokonuje stosownych aktualizacji w bazie danych.