# Wstęp

Niniejsza praca omawia temat automatycznych systemów informatycznych wspomagających korporacje taksówkarskie w procesie zamówienia taksówki, przekazywania informacji o zamówieniu do taksówkarza oraz wspomagania go podczas przewożenia klientów.

Zagadnienie to zostało wybrane ze względu na to, że omawiane systemy prezentują wysoki stopień złożoności nie tylko ze względu na swoją architekturę, ale także wykorzystują zaawansowane algorytmy w celu optymalizacji uzyskiwanych wyników, które mają swoje bezpośrednie odzwierciedlenie w przychodach generowanych dla firm, w których są stosowane.

Celem pracy jest wykonanie aplikacji pt. „Automatyczna dyspozytornia taksówek”, która będzie implementować wszystkie funkcjonalności dostępne w istniejących systemach komercyjnych.

W pierwszym rozdziale zawarto omówienie istniejących systemów stosowanych w polskich korporacjach taksówkowych, w różnych miastach na terenie całego kraju. Drugi rozdział zawiera analizę algorytmów, które zostały wybrane do realizacji dwóch zadań: wyznaczenia trasy dojazdu dla taksówkarza oraz ograniczenia dostępu do poszczególnych modułów aplikacji dla niektórych użytkowników. Wymienia ponadto wszystkie funkcje zaimplementowanego systemu, opisując także pokrótce proces zamówienia taksówki z jego wykorzystaniem. Ostatnim rozdziałem niniejszej pracy jest dokumentacja projektowa zawierająca diagramy przypadków użycia oraz schemat bazy danych wykorzystanej przy realizacji aplikacji.

# Opis zagadnienia

## Dyspozytornia jako system informatyczny

System informatyczny obsługujący dyspozytornię taksówek można podzielić na 2 główne moduły:

* moduł dyspozytora
* moduł taksówkarza

W celu usprawnienia przyjmowania zgłoszeń system może obejmować także moduł klienta, czyli osoby zamawiającej taksówkę. Poniżej opisane zostaną główne zadania, które, zdaniem autora niniejszej pracy, powinny realizować poszczególne moduły oraz zagrożenia związane z obsługą ich przez człowieka.

### Moduł dyspozytora

Dyspozytor pracujący w tradycyjnej centrali korporacji taksówkarskiej jest odpowiedzialny za przyjmowanie zgłoszeń (najczęściej za pośrednictwem telefonu), a następnie za wprowadzanie ich do systemu, który następnie przekazuje informację o nowym zgłoszeniu do kierowców. Przy takim podejściu, łatwo zauważyć zagrożenia, które mogą się pojawić:

* zbyt mała (lub zbyt duża) liczba pracowników dyspozytorni w stosunku do ilości zgłoszeń

Ze względu na to, że każde zgłoszenie musi zostać osobiście odebrane przez dyspozytora, nie ma możliwości jednoczesnego przyjęcia większej liczby zgłoszeń niż wynosi liczba aktualnie pracujących dyspozytorów. Z tego powodu, w okresach dużego zapotrzebowania na taksówkę, klienci mogą być zmuszeni do oczekiwania na przyjęcie zgłoszenia. Z kolei, jeżeli liczba dyspozytorów będzie zbyt duża, to wprawdzie zgłoszenia będą mogły być natychmiast przyjmowane, ale część osób nie będzie ich w ogóle przyjmować, przez co ich zatrudnianie przestaje być opłacalne.

* możliwość popełnienia błędu przez dyspozytora

Niezależnie od sposobu wprowadzania zgłoszenia do systemu, dyspozytor może popełnić błąd na tym etapie polegający na podaniu niewłaściwego adresu wynikający z niezrozumienia wypowiedzi klienta (np. Praska zamiast Bracka, Sienna zamiast Siewna) lub przypadkowego wprowadzenia błędnych danych (tzw. „czeski błąd”, przykładowo numer domu 121 zamiast 212)

* nieoptymalny wybór zgłoszenia przez taksówkarza

Taksówkarz podejmuje zgłoszenie, jeżeli subiektywnie uważa, że jest blisko celu i kurs jest dla niego opłacalny. Jednak najczęściej nie ma wiedzy o położeniu innych taksówkarzy, którzy mogą w rzeczywistości znajdować się bliżej adresu zgłoszenia. Może to generować zwiększone koszty dla kierowców, związane z niepotrzebnym zużyciem paliwa na przejechanie dodatkowej drogi.

Ze względu na powyższe, należy dążyć do tego, aby maksymalnie zautomatyzować proces zamawiania taksówki i rozdzielania zgłoszeń do kierowców. Przyjmowanie zgłoszeń można zrealizować poprzez wdrożenie zaawansowanego systemu rozpoznawania mowy, dzięki któremu można pozostawić tradycyjną drogę zamawiania taksówki. Systemy rozpoznawania mowy nie są tematem niniejszej pracy, dlatego wszelkie zagadnienia z nimi związane zostaną pominięte. Inną metodą automatyzacji jest przygotowanie aplikacji, w której klient wprowadzi adres oraz pozostałe dane zamówienia. W ten sposób cała odpowiedzialność za poprawność danych zostaje przeniesiona na klienta, który w trakcie zamawiania jest w stanie na bieżąco je kontrolować. Z kolei przydział zleceń odpowiednim kierowcom można zrealizować przy użyciu algorytmów wyznaczających najkrótszą trasę lub najkrótszy czas przejazdu między obecnym położeniem poszczególnych taksówkarzy a adresem zamówienia. W ten sposób rola dyspozytora zostaje ograniczona do przeglądania bieżących zgłoszeń i położenia klientów i taksówek.

### Moduł taksówkarza

Podstawowe zadania, przed którymi stoi kierowca taksówki, to po pierwsze, decyzja, czy podjąć nowe zgłoszenie (tylko, jeżeli dyspozytor nie przydziela zleceń odgórnie), a po drugie, jaką trasę dojazdu wybrać celem optymalizacji długości i czasu. Obecnie coraz częściej trasa jest wyznaczana przez zainstalowaną w pojeździe nawigację satelitarną, jednak należy brać pod uwagę, że nie wszystkie systemy nawigacji reagują na zmieniającą się sytuację drogową (korki, remonty itp.), ponadto utrzymywanie aktualności mapy często wiąże się z dodatkowym kosztem. Obydwa te zadania mogą zostać zrealizowane przez automatyczny system, który wyśle zlecenie odpowiedniemu kierowcy oraz wyznaczy trasę przejazdu uwzględniając wszystkie podane wcześniej uwarunkowania. Moduł systemu przeznaczony dla taksówkarza wyświetli jedynie otrzymane informacje. Ponadto powinien on przesyłać do systemu centrali bieżące położenie taksówki oraz umożliwiać logowanie się do systemu w celu zgłoszenia gotowości na zlecenia.

### Moduł klienta

Standardowa procedura zamówienia taksówki może być kłopotliwa dla klienta. Oprócz konieczności znajomości numeru, pod który należy zadzwonić (z czym wiąże się także koszt połączenia), należy również podać swoją dokładną lokalizację, co niekiedy nie jest łatwe do określenia, szczególnie dla osób nieznających okolicy. W przypadku Krakowa można wyobrazić sobie sytuację, że klient zamawia taksówkę na przystanek Rondo Mogilskie, lecz nie zdaje sobie sprawy, że jest kilka przystanków o tej samej nazwie w dość dużej odległości od siebie, a on sam znajduje się na poziomie przeznaczonym dla tramwajów. Ponadto, dyspozytor może jedynie w przybliżeniu podać czas oczekiwania na taksówkę. W celu uniknięcia takich komplikacji, można udostępnić aplikację dla klienta, która będzie dodatkowym modułem zintegrowanego systemu. Najlepszym rozwiązaniem jest aplikacja na telefon komórkowy, posiadający wbudowany moduł GPS. Dzięki temu, osoba zamawiająca zyskuje możliwość dokonania zamówienia, bez konieczności znajomości swojej lokalizacji. Taka aplikacja powinna również na bieżąco informować o przewidywanym czasie przyjazdu taksówki. Przydatną funkcjonalnością byłaby także możliwość sprawdzenia kosztu przejazdu pomiędzy dwoma podanymi miejscami.

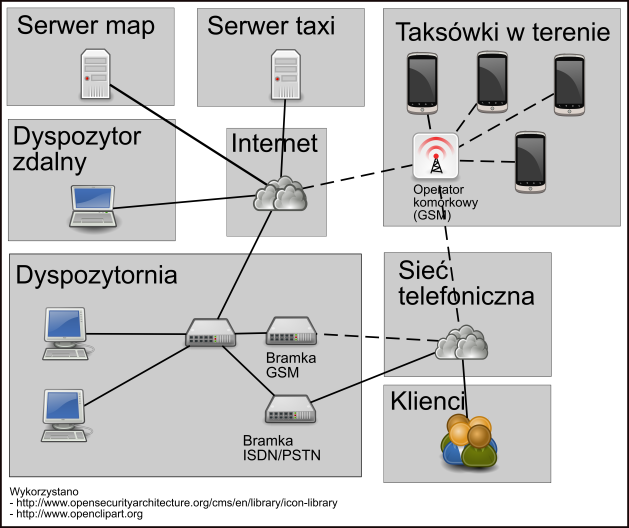
## Istniejące rozwiązania

W tym podrozdziale przedstawione zostaną dwa komercyjne rozwiązania, które są używane przez korporacje taksówkarskie w dużych miastach Polski.

### esysTaxi: Internetowy System Obsługi Taxi

System esysTaxi został stworzony przez firmę ESYSCODER Dariusz Bączkowski z Koszalina. Aktualnie system wykorzystywany jest przez korporacje: Radio Taxi Mega (Kraków), Green Taxi (Olsztyn), Taxi MPT Koszalin (Koszalin, Słupsk, Kołobrzeg, Szczecinek), MERC TAXI (Grudziądz). System bazuje na komunikacji poprzez Internet wykorzystując połączenie udostępniane przez operatorów sieci komórkowych. System wykorzystuje szereg technologii, od centrali telefonii VoIP, poprzez GPS (mapy Targeo lub OpenStreetMap), telefony z systemem Android w roli terminali, do komunikacji zwrotnej z klientem przez SMS.

Poniższy diagram (Rys. 1) obrazuje strukturę systemu esysTaxi:

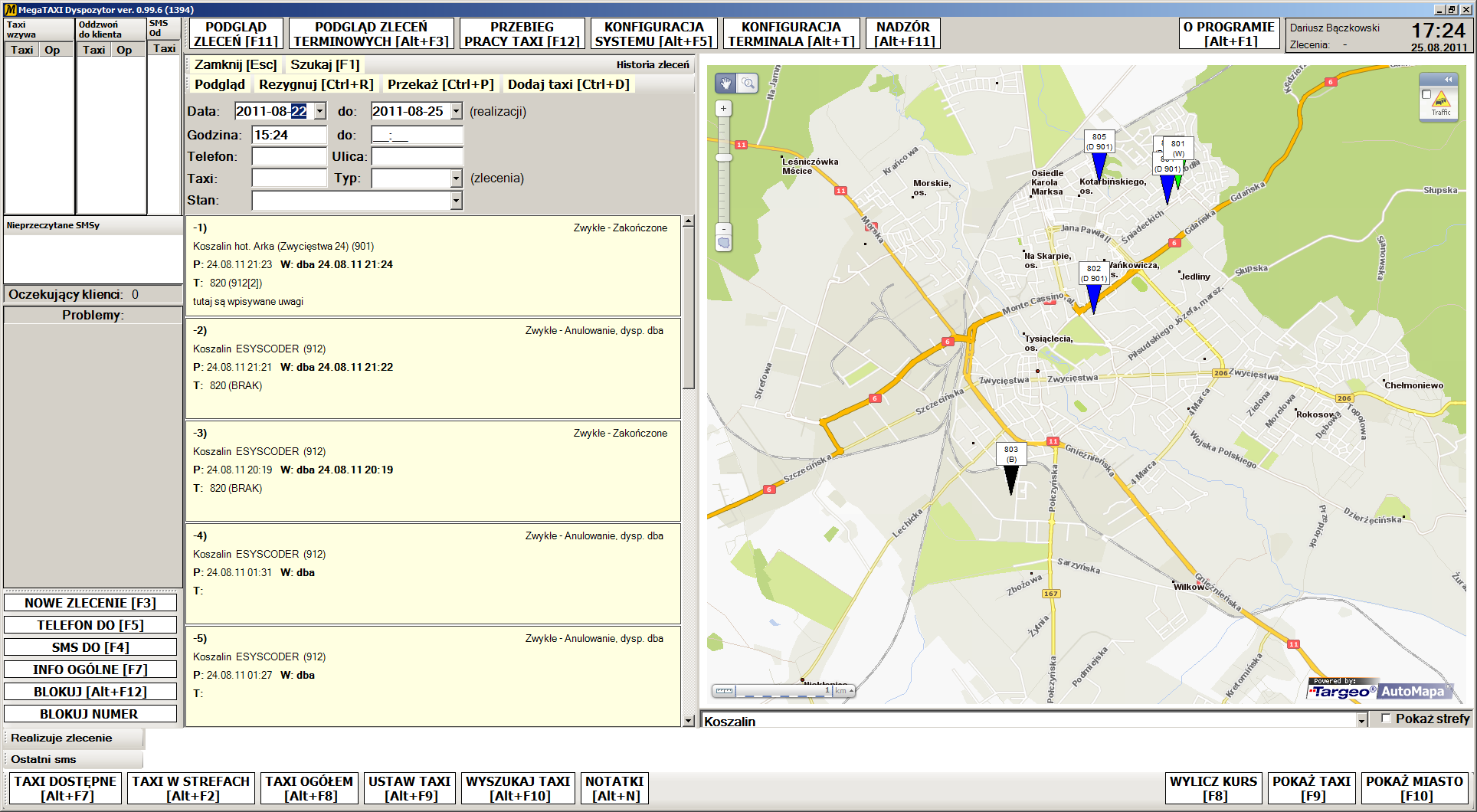


Rys. 1. Schemat struktury systemu esysTaxi [1]

Jak widać na powyższym schemacie, system pozwala na lokalną (w dyspozytorni) oraz zdalną (np. z domu) pracę dyspozytora. Komunikacja pomiędzy dyspozytornią a serwerami odbywa się poprzez sieć Internet. Autor na stronie internetowej opisującej system pisze, że serwer taxi oraz serwer map znajdują się w centrum przetwarzania danych we Francji. Jego zdaniem pozwala to na „uzyskanie bardzo dobrego łącza oraz gwarancji na sprzęt” [1]. Terminalami dla poszczególnych taksówkarzy są telefony z systemem Android, przesyłające do centrali dane o aktualnym stanie i położeniu taksówki oraz odbierające powiadomienia o zleceniu. Autor podaje na swojej stronie listę poszczególnych telefonów i tabletów, na których z powodzeniem wdrożono aplikację używaną przez taksówkarzy. Informuje także, że możliwe jest dostosowanie aplikacji do konkretnego modelu telefonu i wersji systemu Android, w przypadku wystąpienia komplikacji.

Oprócz komercyjnego systemu dla korporacji taksówkarskiej, dostępna jest także bezpłatna aplikacja na telefon komórkowy z systemem Android, przeznaczona dla klienta zamawiającego taksówkę. Aplikacja ta jest obecnie dostępna jako dedykowana dla klientów poszczególnych korporacji wykorzystujących system esysTaxi. Aplikacja pozwala na zamówienie taksówki bezpośrednio przez połączenie telefoniczne, jak również wpisując adres lub używając geolokalizacji GPS. Można także wykorzystać wcześniej zamawiany adres.

Stanowisko dyspozytora obejmuje komputer oraz telefon VoIP. Po odebraniu telefonu, system wyświetla dyspozytorowi listę poprzednich zleceń klienta w celu szybszej możliwości wskazania adresu. Podanie nowego adresu wymusza ręczne wprowadzenie go przez dyspozytora. Dalsza część obejmująca wybranie odpowiedniej taksówki i przesłanie zgłoszenia do kierowcy, jest wykonywana automatycznie przez system. Ponadto, dyspozytor ma na ekranie swojego stanowiska podgląd mapy z zaznaczonymi pozycjami taksówek. Poniżej, na Rys. 2, przedstawiono wygląd przykładowego okna, jakie podczas pracy może zobaczyć dyspozytor.



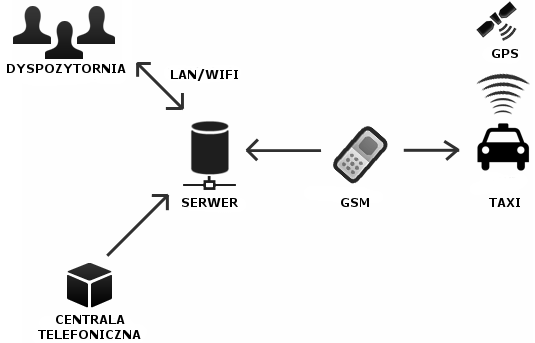
Rys. 2. Przykładowe okno systemu esysTaxi – podgląd zleceń dla dyspozytora [1]

### Tiskel DS

Tiskel DS jest systemem stworzonym przez firmę Tiskel. Autorzy na swojej stronie przedstawiają referencje z korporacji EkoTaxi (Warszawa) oraz Radio Taxi Serc (Wrocław). Ponadto, wśród swoich klientów wymieniają także m.in Tele-Taxi (Ruda Śląska), Express Taxi (Bydgoszcz), Radio Taxi Rzeszów. W odróżnieniu od wcześniej opisywanego esysTaxi, Tiskel DS umożliwia instalację systemu na lokalnym serwerze pracującym w dyspozytorni (wersja oprogramowania Tiskel DS Pro). Oprócz tego, możliwy jest zakup systemu zainstalowanego na wydzielonym serwerze (wersja Tiskel DS Chmura). Autorzy nie podają szczegółów technicznych serwera aplikacji.

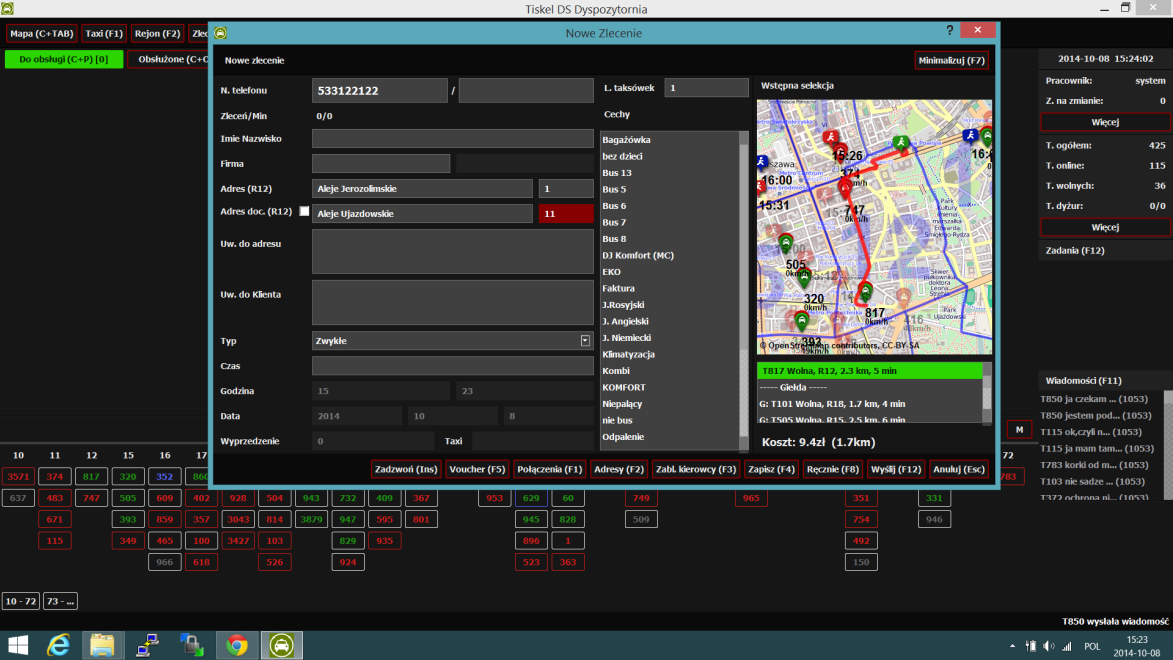
Diagram na kolejnej stronie obrazuje strukturę systemu Tiskel DS. Ten system także wykorzystuje geolokalizację GPS do ustalania położenia taksówki, ponadto komunikacja pomiędzy dyspozytornią a kierowcą również odbywa się poprzez sieć Internet, udostępnianą przez operatora komórkowego. Obydwa systemy funkcję terminala kierowcy realizują w postaci aplikacji na telefon z systemem Android. Oprogramowaniem dostarczającym mapy do systemu jest projekt OpenStreetMap. Ponadto firma Tiskel na swojej stronie pisze o wykorzystywaniu centrali telefonicznych firmy Platan, jako „szczególnie zalecanych do współpracy z systemem Tiskel DS” [2].

Działanie systemu jest niemal bliźniacze w porównaniu do esysTaxi. Tutaj także dzwoniący klient jest rozpoznawany po numerze telefonu, a dyspozytorowi zostaje wyświetlony podgląd poprzednich zleceń. Dyspozytor wprowadza do systemu szczegóły zgłoszenia, a system przekazuje je do odpowiedniego taksówkarza. Autorzy na swojej stronie opisują szczegółowo politykę przydzielania zleceń podkreślając jednocześnie, że jest to jedynie przykład konfiguracji systemu i może zostać dostosowany do wymagań klienta. Opisany schemat przedstawiono na Rys. 3.



Rys 3. Schemat działania systemu Tiskel DS [2]

Moduł dyspozytorni, podobnie jak w systemie esysTaxi, pozwala na przeglądanie bieżącej sytuacji na mapie. Ponadto dyspozytor może czasowo zablokować wybraną taksówkę, aby nie były jej przydzielane kolejne zlecenia. Przykładowy wygląd okna dyspozytora przedstawia Rys. 4.



Rys. 4. Przykładowe okno dyspozytora – wprowadzanie nowego zlecenia [2]

Podobnie jak wcześniej, także w tym przypadku dostępna jest aplikacja na telefon komórkowy przeznaczona dla klienta – CAB4YOU. Jednak w odróżnieniu od systemu esysTaxi, aplikacja ta integruje w sobie wszystkie miasta, w których działają korporacje będące użytkownikami systemu Tiskel DS.

# Opis wykonanego rozwiązania

W tym rozdziale przedstawiono główne algorytmy wykorzystane przy implementacji systemu oraz opisano technologie wykorzystane do jego wykonania. Pokazano także zakres funkcjonalności, dostępny w każdym z modułów aplikacji: module admistracyjnym, module taksówkarza oraz module klienta.

## Algorytm grafowy Floyda-Warshalla

Mapa miasta, składająca się przede wszystkich z ulic i skrzyżowań, została zaimplementowana w postaci grafu skierowanego, w którym ulice stanowią krawędzie, a skrzyżowania – wierzchołki grafu (przy czym dla ulic jednokierunkowych, ścieżka od jednego wierzchołka do drugiego jest możliwa do przejścia tylko zgodnie z kierunkiem ruchu na danej ulicy). Rys. 5. przedstawia przykładowy fragment mapy, na którym naniesiono strukturę grafu.



Rys. 5. Fragment mapy z naniesioną strukturą grafu

([3], opracowanie własne)

Z tego względu, należało podjąć decyzję polegającą na wybraniu algorytmu, który jak najlepiej poradzi sobie z wyznaczaniem najkrótszych ścieżek pomiędzy dwoma dowolnymi wierzchołkami.

Charakterystyczną cechą implementowanego systemu jest bardzo mała zmienność – ulice nie zmieniają przecież swojej długości, a nowe połączenia drogowe nie pojawiają się ani nie znikają zbyt często. Dzięki temu nie ma potrzeby każdorazowego wyznaczania ścieżki pomiędzy dwoma takimi samymi punktami na mapie, jeżeli nie zaszły żadne zmiany. Można więc przyjąć, że trasy łączące dowolne dwa skrzyżowania wystarczy wyznaczyć tylko raz i ewentualnie przeliczać je na nowo dopiero w momencie, w którym zaszła zmiana.

Wiedząc o powyższych cechach systemu, wybrano algorytm Floyda-Warshalla. Pozwala on na wyznaczenie ścieżek pomiędzy wszystkimi parami wierzchołków grafu (o ile takie połączenie istnieje). Zapamiętanie wyznaczonych ścieżek pozwala na natychmiastowe podanie drogi łączącej dwa wierzchołki. Algorytm ten został szczegółowo zaprezentowany w publikacji „Grafy i rekurencje” J. Mareckiego [4], poniżej zostanie jedynie pokrótce przedstawiony w postaci pseudo-kodu.

Punktem startowym algorytmu jest macierz *H*, dla której na przecięciu wiersza *i* oraz kolumny *j* znajduje się wartość *k* oznaczająca długość krawędzi łączącej wierzchołki *i* oraz *j*. Jeżeli taka krawędź nie istnieje, to wartość w tym miejscu wynosi *∞*. Algorytm wyznacza w kolejnych krokach macierze *D(i)* dla *i = 0..n*, zawierające minimalne odległości pomiędzy wierzchołkami grafu:

n:=Rozmiar(H)

D(0) := H

for k := 1 to n do

for i := 1 to n do

for j := 1 to n do

d(k)ij := min {d(k-1)ij, d(k-1)ik + d(k-1)kj}

return D(n)

Listing 1. Algorytm Floyda-Warshalla w postaci pseudokodu [4]

Jednakże, oprócz poznania najkrótszej odległości pomiędzy dwoma punktami, niezbędne jest także pamiętanie poszczególnych wierzchołków na ścieżce. W tym celu tworzona jest macierz poprzedników *S*, która początkowo, na przecięciu wiersza *i* oraz kolumny *j* przyjmuje wartość *i*, jeżeli istnieje krawędź łącząca wierzchołki *i* oraz *j* lub *nil*, jeżeli taka krawędź nie istnieje. Następnie, jeżeli funkcja *min* powyższego algorytmu zwróci drugą wartość, to na przecięciu wiersza *i* oraz kolumny *j* zostanie wstawiona wartość *k*. Gdy algorytm zakończy swoje działanie, ścieżka pomiędzy wierzchołkami *a* i *b* będzie mogła zostać odtworzona za pomocą poniższego sposobu:

if sab != nil then

Pokaż(a,b)

print(b)

else nie istnieje ścieżka z a do b

Listing 3. Metoda odtworzenia ścieżki z punktu *a* do *b* w formie pseudokodu [4]

Funkcja rekurencyjna *Pokaż(x,y)* została zdefiniowana jak niżej:

Pokaż(i,j)

k := sij

if i = k then print(k)

else Pokaż(i,k)

Pokaż(k+1,j)

Listing 3. Funkcja *Pokaż(x,y)* w postaci pseudokodu [4]

Funkcja ta wypisuje kolejne wierzchołki znalezionej ścieżki. Cały algorytm można zmodyfikować w ten sposób, aby po wyznaczeniu macierzy *D* – odległości i macierzy *S* – poprzedników, zapamiętywać ścieżki wyznaczone metodą jak na listingu 2. Dzięki temu, oprócz początkowego wyliczenia, całe działanie algorytmu ograniczy się jedynie do zwracania wcześniej wyznaczonych ścieżek. Taki sposób działania pozwala na natychmiastowe podawanie trasy przejazdu.

Jednakże takie podejście również rodzi kilka problemów, zaczynając od wysokiej złożoności obliczeniowej algorytmu, która ze względu na potrójną pętlę for sięga rzędu *O(n3)*, poprzez konieczność przechowywania macierzy o rozmiarze *n x n* na potrzeby wyznaczania odległości i ścieżki, aż do przymusu pełnego przeliczania algorytmu po dokonaniu jakichkolwiek zmian w postaci grafu. Na etapie realizacji niniejszej pracy stwierdzono jednak, że problemy te nie są aż tak istotną przeszkodą, aby rezygnować z tego algorytmu na rzecz innego, który miałby każdorazowo od nowa wyznaczać trasę.

## Mechanizm zarządzania uprawnieniami użytkowników

Oprócz algorytmu grafowego, który jest „sercem” aplikacji, należało także zaimplementować kilka mechanizmów, które ułatwią użytkowanie programu. Jednym z nich jest kontrola uprawnień użytkowników, polegająca na ograniczeniu dostępu do poszczególnych funkcjonalności.

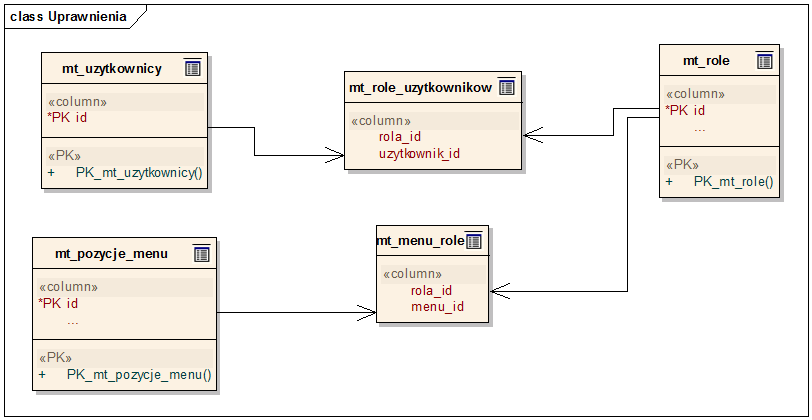
Podstawową formą ograniczenia dostępu jest konieczność zalogowania się, aby móc korzystać z aplikacji. Dzięki temu, nie będzie możliwości uzyskania nieautoryzowanego dostępu do żadnej z funkcjonalności systemu.

Oprócz tego, w zakresie zalogowanych użytkowników, można podzielić ich na grupy dysponujące różnym zestawem uprawnień. Można także pozostawić pewien ograniczony zakres funkcjonalności dostępny dla niezalogowanych użytkowników. Ten właśnie model został zaimplementowany w aplikacji będącej przedmiotem niniejszej pracy.

Użytkownik niezalogowany może korzystać jedynie z modułu klienta – aplikacji desktopowej służącej do zamówienia taksówki. Po zalogowaniu się, użytkownik będący klientem może uzyskać dostęp do bardziej szczegółowych opcji zamówienia, może także przeglądać dane swojego konta. Z kolei, jeżeli użytkownik jest taksówkarzem, zostaje mu udostępniony oddzielny panel zawierający niezbędne informacje.

Część aplikacji dostępna przez przeglądarkę internetową jest implementacją modułu dyspozytora. Zalogować się do niej mogą jedynie użytkownicy będący dyspozytorami bądź użytkownik – szef korporacji. Pozostali użytkownicy, pomimo posiadania konta w systemie, nie mogą uzyskać dostępu do tej części aplikacji. Również w tej części systemu zakres dostępnych do wykonania operacji zależy od uprawnień użytkownika.

Poszczególne uprawnienia zostały opisane w pojedynczej tabeli bazy danych. Kolejna tabela zawiera przyporządkowanie uprawnień do roli. „Rola” w tym kontekście jest zestawem uprawnień. Osobna tabela przechowuje przypisanie ról do użytkowników. Schemat powiązań tabel w bazie danych obrazujący powyższy opis został przedstawiony na Rys. 6.



Rys. 6. Schemat bazy danych dotyczący ról i uprawnień użytkowników (opracowanie własne)

Przyjęto, że poszczególni użytkownicy, posiadający dane role, mają w ich zakresie takie same uprawnienia. Podczas realizacji opisywanego systemu nie pojawiła się konieczność przypisywania użytkownikom indywidualnych uprawnień do jakiejkolwiek części systemu.

## Techniczny opis implementacji

Wykonany system został w głównej części zaimplementowany w języku programowania Java. Został on wybrany ze względu na dobrą jego znajomość przez autora niniejszej pracy, a także z powodu dostępności darmowych narzędzi pozwalających na tworzenie oprogramowania w tym języku (m.in. Eclipse IDE [5]). W celu przyswojenia kwestii przydatnych w niektórych szczegółach implementacji, przydatna okazała się pozycja „Thinking in Java” autorstwa B. Eckela [6].

Do wykonania modułu administracyjnego, będącego aplikacją internetową, wykorzystano darmowy serwer aplikacji Apache Tomcat 7 [7]. Poszczególne strony wykonano przy użyciu technologii Java Server Pages [8], używając także elementów języków HTML i CSS w celu utworzenia pożądanego wyglądu.

Wszelkie dane potrzebne do działania systemu zostały zachowane w darmowej bazie danych PostgreSQL [9].

Ponadto, w celu bezpiecznego przechowywania całego wytworzonego kodu źródłowego, a także tekstu niniejszej pracy, wykorzystano system kontroli wersji Git, a repozytorium umieszczono na bezpłatnej platformie GitHub, pod adresem podanym w pozycji [10].

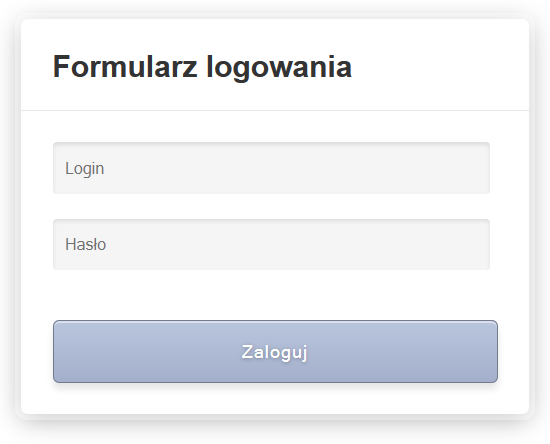
## Zakres dostępnych funkcjonalności

### Moduł administracyjny

Moduł administracyjny jest aplikacją internetową, dostępną przez przeglądarkę. Został on przeznaczony dla dwóch grup użytkowników biznesowych: szefa korporacji oraz dyspozytora.

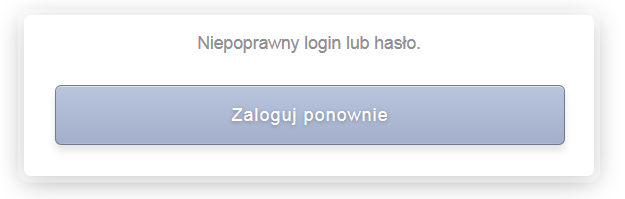
#### Logowanie i ekran główny

Dostęp do modułu administracyjnego został zabezpieczony stroną logowania, której wygląd zaprezentowano na Rys. 7.



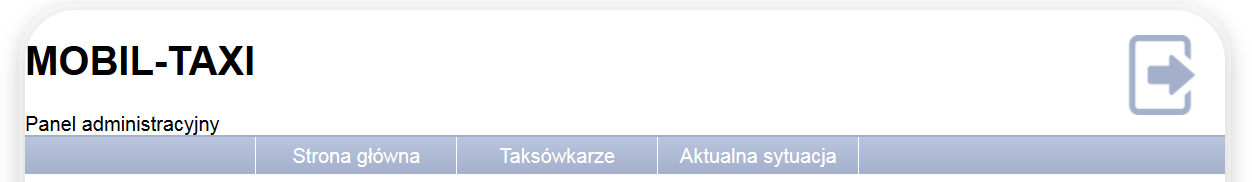
Rys. 7. Ekran logowania do modułu administracyjnego

Użytkownik musi podać login oraz hasło, aby uzyskać dostęp do modułu adminstracyjnego. Niepoprawny login lub hasło spowodują wyświetlenie strony błędu logowania, z możliwością powrotu do strony logowania. Ekran błędu logowania zaprezentowano na Rys. 8.



Rys. 8. Ekran błędu logowania

Poprawne zalogowanie powoduje przejście do strony głównej aplikacji. Jej wygląd został zaprezentowany na Rys. 9.



Rys. 9. Strona główna

W górnej części okna znajduje się menu pozwalające na dostęp do poszczególnych funkcji. Zaprezentowane na Rys. 9 menu jest dostępne dla użytkownika posiadającego rolę Szef korporacji. Ma on możliwość przeglądania oraz pełnej edycji danych taksówkarzy, wraz z danymi wykonanych przez nich kursów. Ma również możliwość wglądu w aktualną sytuację – ta pozycja jest dostępna także dla użytkownika posiadającego rolę Dyspozytor.

Ostatni przycisk widoczny po prawej stronie pozwala na wylogowanie się z aplikacji.

#### Dane taksówkarzy

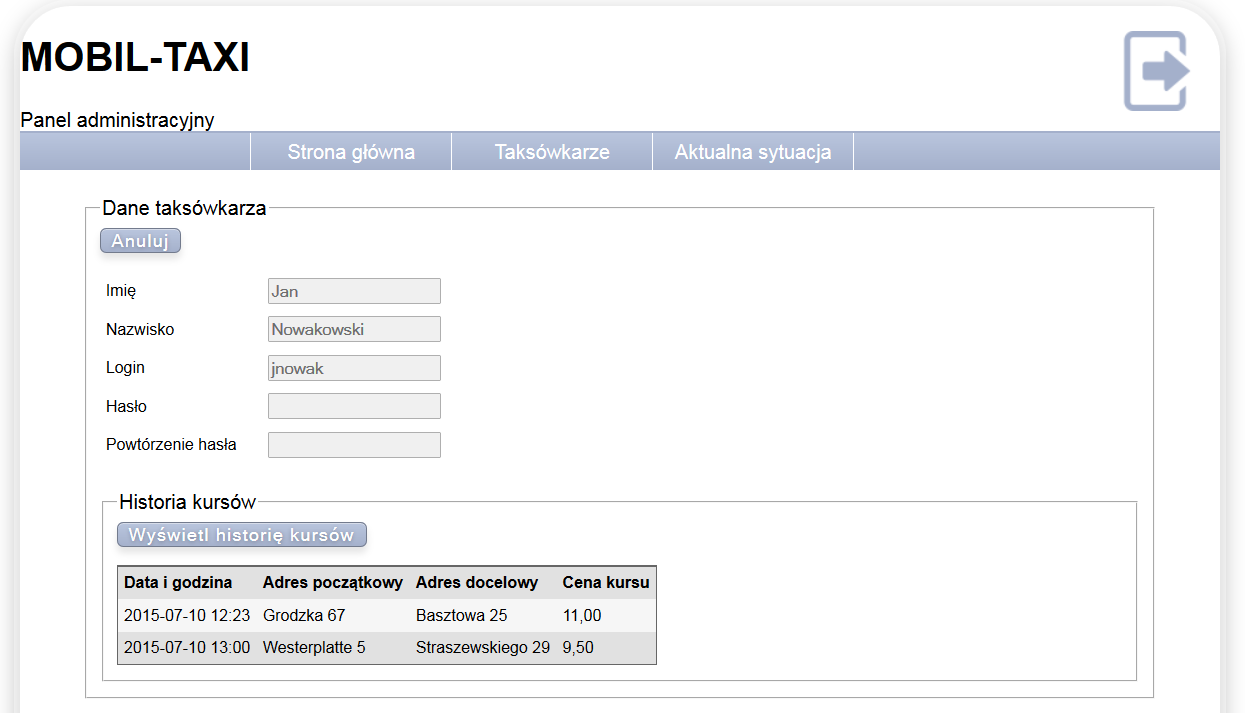
Wskazanie pozycji menu „Taksówkarze” powoduje przejście do strony zawierającej listę wszystkich aktualnie zatrudnionych taksówkarzy (Rys. 10).



Rys. 10. Lista taksówkarzy

Lista taksówkarzy zawiera imię, nazwisko oraz login taksówkarza. Ponadto, możliwe jest przeglądanie i edycja danych taksówkarza. Istnieje także możliwość zwolnienia taksówkarza oraz zatrudnienia nowego taksówarza. Przycisk „Odśwież” powoduje ponowne pobrane informacji o taksówkarzach z bazy danych.

Zatrudnienie nowego taksówkarza oraz przeglądanie i edycja danych odbywają się na oknie danych pojedynczego taksówkarza, zaprezentowanym na Rys. 11.



Rys. 11. Ekran danych taksówkarza – tryb przeglądania

Na oknie danych taksówkarza widoczne są imię, nazwisko i login taksówkarza – login jest wykorzystywany do zalogowania się do modułu taksówkarza. W trybie edycji można ponadto zmienić hasło taksówkarza. W trybie zatrudnienia nowego taksówkarza podanie hasła jest obowiązkowe. Nowo zatrudniony taksówkarz może korzystać z aplikacji bezpośrednio po dodaniu, korzystając z podanego na tym oknie loginu i hasła.

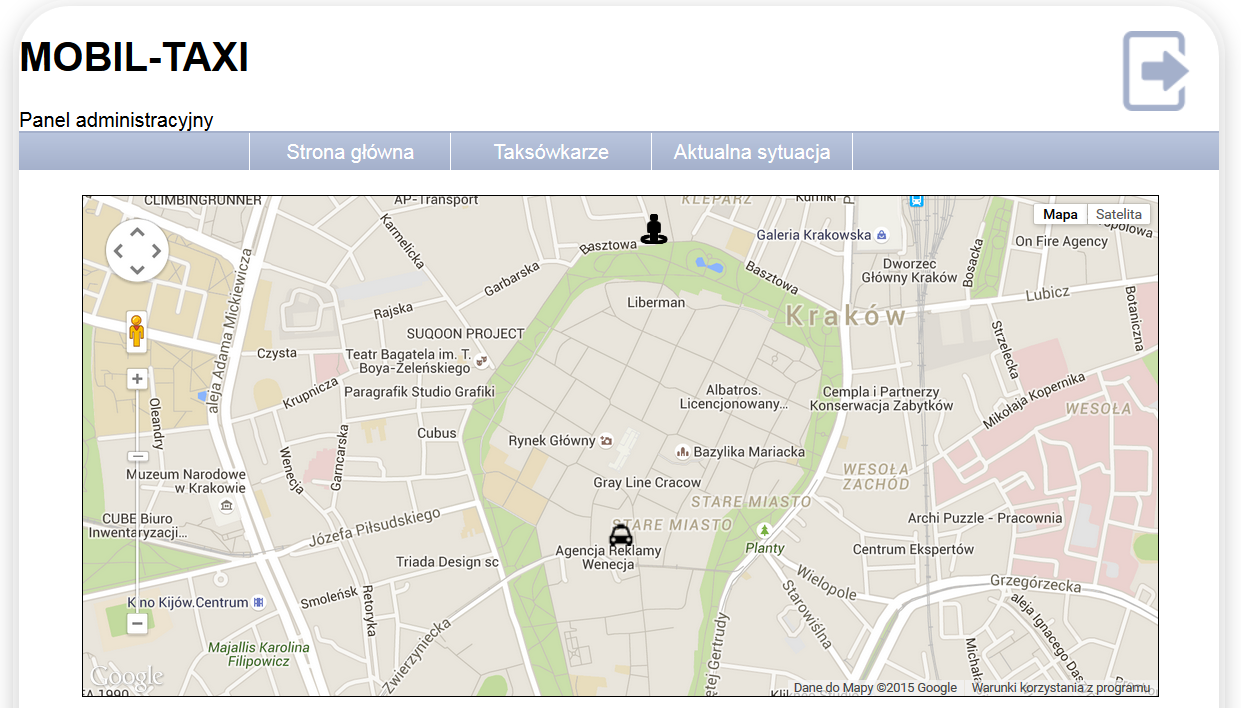
Okno danych taksówkarza pozwala także na przeglądanie danych wykonanych kursów. Tabela z wykonanymi kursami zawiera dokładną datę rozpoczęcia kursu (od momentu zabrania klienta), adres początkowy i docelowy oraz cenę kursu.

#### Przeglądanie aktualnej sytuacji

Wskazanie pozycji menu „Aktualna sytuacja” powoduje wyświetlenie strony prezentującej mapę, na której zaznaczono bieżącą pozycję wszystkich aktywnych (zalogowanych) taksówkarzy oraz oczekujących klientów (Rys. 12).

Wskazanie kursorem myszy ikony taksówkarza wyświetla informacje o tym, czy dany kierowca obecnie jedzie do klienta lub wykonuje kurs z klientem oraz dokąd obecnie jedzie. Wyświetlane są ponadto podstawowe dane taksówkarza – jego imię i nazwisko.

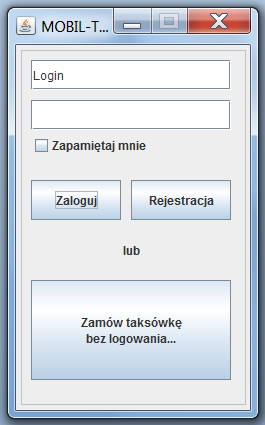
Z kolei wskazanie ikony klienta powoduje wyświetlenie informacji, czy dany klient jest zarejestrowany i zalogowany lub anonimowy, dokąd chce jechać oraz jak długo już oczekuje na taksówkę.



Rys. 12. Ekran przeglądania aktualnej sytuacji

### Moduł taksówkarza / klienta

Moduł taksówkarza / klienta jest aplikacją kliencką uruchamianą na stacji roboczej taksówkarza lub klienta. Po uruchomieniu, aplikacja działa domyślnie w trybie klienta. Przełączenie jej w tryb taksówkarza odbywa się po zalogowaniu się do aplikacji, podając login i hasło użytkownika posiadającego rolę Taksówkarz.



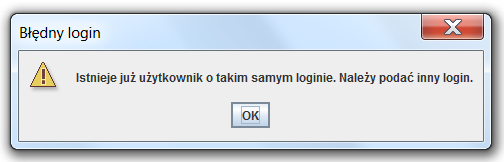
Rys. 13. Ekran startowy modułu taksówkarza / klienta

Aplikacja kliencka wymaga do poprawnego działania połączenia z siecią Internet. Jest ono niezbędne, ponieważ wszystkie dane potrzebne do jej poprawnego działania pobierane są z serwera centralnego (odpowiedzialnego za wyświetlanie zawartości modułu administracyjnego).

#### Rejestracja w systemie

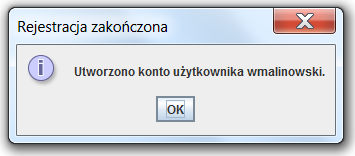
Użytkownik może zarejestrować swoje konto w systemie. Zarejestrowanie konta powoduje utworzenie w systemie nowego konta użytkownika z rolą Klient – osoba zamawiająca taksówkę. Po rejestracji użytkownik może zalogować się do aplikacji. Zamawianie taksówki jako zalogowany użytkownik pozwala na dostęp do większej liczby opcji przy składaniu zamówienia. Zostaną one opisane w dalszych punktach.

Rejestracja konta w systemie wymaga podania loginu oraz hasła, a następnie kliknięcia „Rejestruj”. System sprawdza unikalność podanego loginu – w systemie może istnieć tylko jeden użytkownik o danym identyfikatorze. W przypadku podania loginu istniejącego użytkownika, zostanie wyświetlony komunikat jak na Rys. 14.



Rys. 14. Komunikat błędu rejestracji wyświetlany, gdy użytkownik poda istniejący login

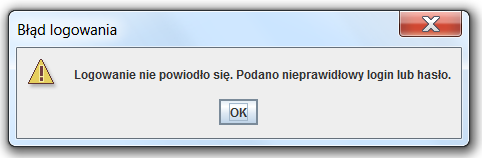
Po poprawnej rejestracji zostaje wyświetlony komunikat informujący o założeniu nowego konta w systemie (Rys. 15) i następuje przejście do ekranu zamawiania taksówki jako użytkownik zalogowany na utworzone przed chwilą konto.



Rys. 15. Komunikat informujący o poprawnej rejestracji

#### Logowanie do aplikacji

Uruchomienie modułu powoduje wyświetlenie ekranu startowego (Rys. 7). Na tym ekranie zarejestrowany użytkownik (klient lub taksówkarz) może zalogować się do aplikacji, podając swój indywidualny login i hasło. Każdorazowo po podaniu danych dostępowych, aplikacja łączy się z serwerem sprawdzając ich poprawność. W przypadku stwierdzenia braku podanego użytkownika, niepoprawnego hasła bądź niewłaściwej roli, zostaje wyświetlona informacja o błędzie logowania (Rys. 16).

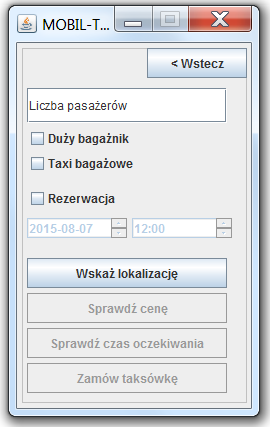


Rys. 16. Komunikat błędu logowania

#### Zamówienie taksówki

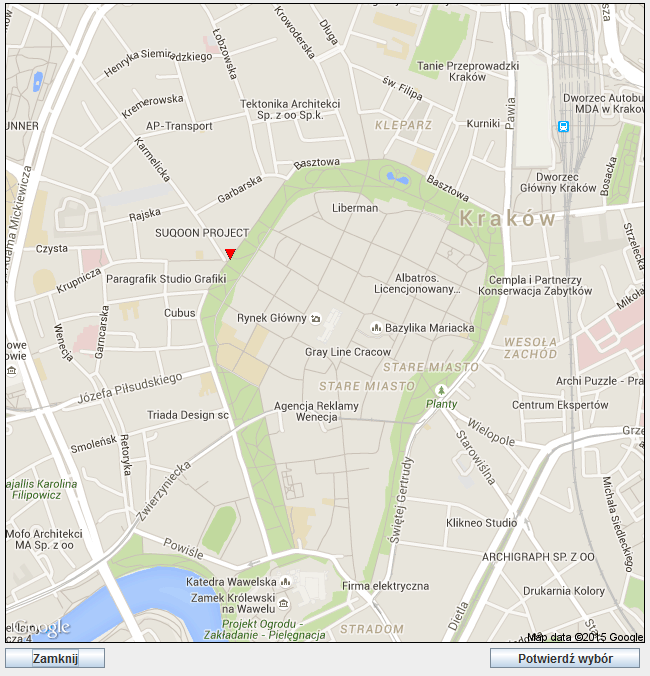
Użytkownik może zamówić taksówkę zarówno posiadając konto w systemie i będąc zalogowanym, jak i anonimowo, bez podawania żadnych swoich danych, klikając „Zamów taksówkę bez logowania” na oknie startowym aplikacji. Jednakże, zalogowanie się do aplikacji pozwala na dokonanie rezerwacji taksówki na określony dzień i godzinę.

Okno zamawiania taksówki widoczne jest na Rys. 17. Użytkownik ma możliwość podania liczby pasażerów, wskazania, czy zamawiana taksówka powinna mieć duży bagażnik lub czy powinno to być taxi bagażowe. Dla zalogowanych użytkowników przewidziano także możliwość zarezerwowania taksówki na dany dzień i godzinę, z minimalnym wyprzedzeniej jednej godzininy oraz maksymalnym wyprzedzeniem jednego miesiąca.



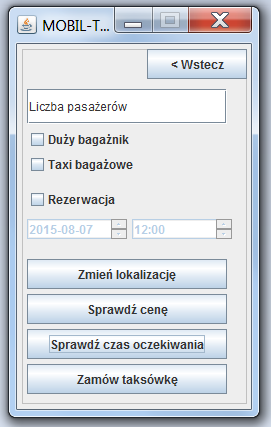
Rys. 17. Okno zamawiania taksówki – widok zalogowanego klienta

Przycisk „Wskaż lokalizację” powoduje otworzenie okna z podglądem mapy, na której użytkownik wskazuje miejsce, gdzie będzie oczekiwał na przyjazd taksówki. Wygląd okna został zaprezentowany na Rys. 18.



Rys. 18. Okno wyboru lokalizacji. Czerwony trójkąt wskazuje miejsce wybrane przez użytkownika.

Kliknięcie w dowolny punkt na mapie powoduje oznaczenie tego miejsca czerwonym trójkątem. Użytkownik zatwierdza wskazanie lokalizacji przyciskiem „Potwierdź wybór”, po czym, okno zostaje zamknięte i następuje powrót na okno zamawiania taksówki. Po wskazaniu lokalizacji na oknie tym zostają udostępnione dodatkowe funkcjonalności, jak zostało przedstawione na Rys. 19.



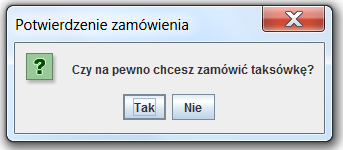
Rys. 19. Okno zamawiania taksówki

Przycisk „Zmień lokalizację” otwiera takie samo okno, jak wcześniej przycisk „Wskaż lokalizację”, z możliwością zmiany wcześniej wybranej lokalizacji. Po otworzeniu okna, jest na nim zaznaczona aktualnie wybrana pozycja. Zmiana wybranej lokalizacji odbywa się poprzez wskazanie nowego miejsca oczekiwania i zatwierdzenie przyciskiem „Potwierdź wybór”. Wyjście z tego okna przyciskiem „Zamknij” nie spowoduje zapisania wykonanej zmiany.

Przycisk „Sprawdź cenę” służy do sprawdzenia ceny przejazdu taksówką. Jego kliknięcie również otwiera okno z mapą, na którym, po wskazaniu miejsca docelowego, zostaje wyświetlona przewidywana cena przejazdu.

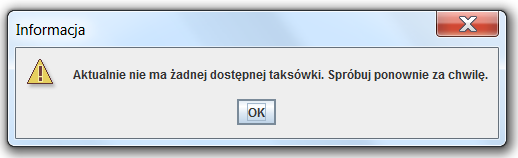
Przycisk „Sprawdź czas oczekiwania” pozwala na wyświetlenie przewidywanego czasu oczekiwania na taksówkę.

Przycisk „Zamów taksówkę” służy do potwierdzenia zamówienia taksówki. Jego kliknięcie powoduje wyświetlenie najpierw komunikatu potwierdzenia (Rys. 20), a następnie faktyczne złożenie zamówienia.

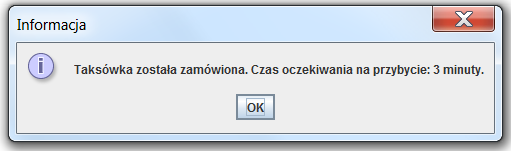


Rys. 20. Komunikat potwierdzenia zamówienia

System sprawdza dostępność taksówki oraz czas jej przejazdu do klienta. Jeżeli żadna taksówka nie jest aktualnie dostępna, zostaje wyświetlony komunikat jak na Rys. 21. W przeciwnym razie, użytkownik otrzymuje informację o potwierdzeniu złożenia zamówienia, z przewidywanym czasem przyjazdu taksówki (Rys. 22).

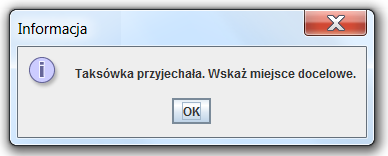


Rys. 21. Komunikat informującu o braku dostępnej taksówki.



Rys. 22. Komunikat informujący o pozytywnym zamówieniu taksówki wraz z przewidywanym czasem oczekiwania na przybycie taksówki

O przybyciu taksówki użytkownik zostaje poinformowany oddzielnym komunikatem, który został zaprezentowany na Rys. 23.



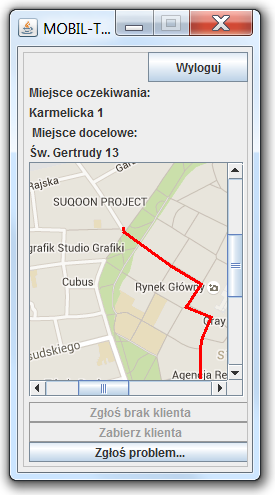
Rys. 23. Komunikat informujący o przybyciu taksówki

Następnie użytkownik wskazuje miejsce docelowe (w takim samym oknie jak na Rys. 18) i zostaje poinformowany o koszcie i czasie przejazdu.

W tym miejscu proces zamawiania taksówki przy pomocy aplikacji zostaje zakończony. Dalsza rejestracja kursu odbywa się w widoku zalogowanego taksówkarza (opisanym w kolejnym punkcie) oraz na oknie przeglądania aktualnej sytuacji w panelu administracyjnym (Rys. 12).

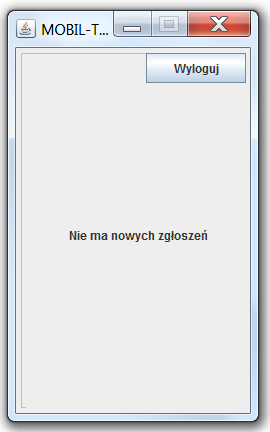
#### Okno zamówienia – widok taksówkarza

Po zalogowaniu się do aplikacji taksówkarz zostaje przeniesiony do okna, na którym wyświetlane jest bieżące zamówienie. Oprócz adresu miejsca, w którym oczekuje klient, wyświetlana jest także mapa z zaznaczoną trasą przejazdu, prowadzącą od aktualnej pozycji taksówkarza, do miejsca oczekiwania klienta. Wygląd tego okna zaprezentowano na Rys. 24.



Rys. 24. Okno taksówkarza – widok aktualnego zgłoszenia

Może także wystąpić sytuacja, w której po zalogowaniu się do aplikacji, żadne nowe zgłoszenie nie będzie oczekiwać na taksówkarza. W takiej sytuacji, okno będzie miało postać jak na Rys. 25.

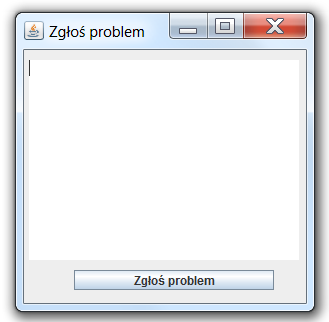


Rys. 25. Okno taksówkarza – brak zgłoszeń

Po dotarciu na miejsce oczekiwania klienta, taksówkarz może zasygnalizować brak klienta. Dokonuje tego za pośrednictwem przycisku „Zgłoś brak klienta”. Przycisk uaktywnia się po 5 minutach od przybycia taksówkarza. W takiej sytuacji kurs zostaje anulowany.

Po zabraniu klienta, taksówkarz wybiera przycisk „Zabierz klienta”, czym rozpoczyna właściwy kurs. Po kliknięciu tego przycisku, zmienia on swój opis na „Zakończ kurs”. Po dotarciu do miejsca docelowego taksówkarz wybiera przycisk „Zakończ kurs”, co powoduje zapisanie odbytego kursu w danych taksówkarza oraz klienta (o ile ten był zalogowany), w centralnej bazie danych.

Od momentu przyjęcia zgłoszenia do zakończenia kursu, taksówkarz może zgłosić nieprzewidziany problem, który uniemożliwia mu podjęcie klienta z miejsca oczekiwania bądź dokończenie kursu z klientem. Sluży do tego przycisk „Zgłoś problem”. Jego wybranie powoduje otworzenie okna, na którym taksówkarz może podać opis problemu (np. awaria samochodu). Wygląd tego okna jest widoczny na Rys. 26.



Rys. 26. Okno zgłoszenia problemu

Zgłoszenie problemu powoduje wylogowanie taksówkarza z aplikacji, aby nie mógł on przyjmować kolejnych zamówień do czasu rozwiązania problemu. Zgłoszenie problemu powoduje przerwanie kursu. Klient powinien ponownie zamówić taksówkę, aby dotrzeć do miejsca docelowego.

# Dokumentacja projektowa

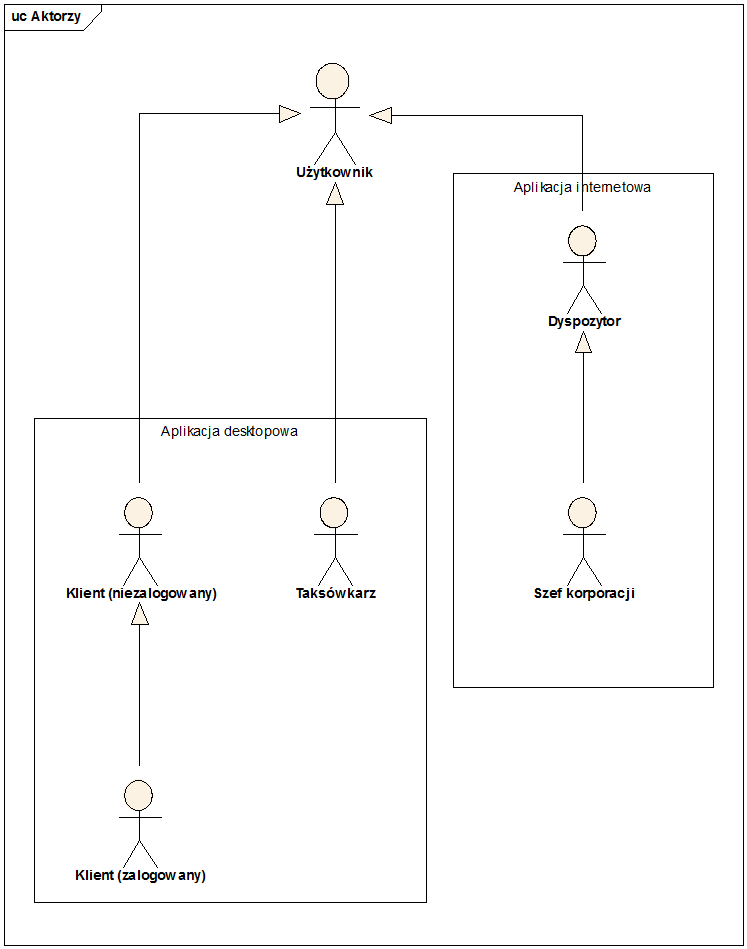
Niniejszy rozdział zawiera dokumentację projektową aplikacji będącej przedmiotem niniejszej pracy. Dokumentacja zawiera pełen zakres funkcjonalności przedstawiony w postaci przypadków użycia oraz schemat bazy danych.

Do przygotowania diagramów wykorzystano program Enterprise Architect [11], którego 30-dniowa wersja demonstracyjna jest dostępna bezpłatnie.

## Model przypadków użycia

### Aktorzy

Diagram przedstawiony na Rys. 27 zawiera wszystkich aktorów, czyli rodzajów użytkowników występujących w aplikacji, zarówno internetowej, jak i desktopowej.



Rys. 27. Aktorzy

#### Dyspozytor

Użytkownik aplikacji internetowej mający uprawnienia jedynie do przeglądania bieżącej sytuacji. Może wyświetlić mapę pokazującą aktualne położenie zalogowanych taksówkarzy oraz oczekujących klientów.

#### Klient (niezalogowany)

Użytkownik aplikacji desktopowej, mogący zamówić taksówkę.

#### Klient (zalogowany)

Użytkownik aplikacji desktopowej, mogący zamówić taksówkę lub zarezerwować taksówkę na przyszły termin. Ponadto, może przeglądać swoje wcześniejsze zamówienia.

#### Szef korporacji

Użytkownik aplikacji internetowej, mający uprawnienia do zatrudniania i zwalniania taksówkarzy. Ponadto może również przeglądać bieżącą sytuację na mapie.

#### Taksówkarz

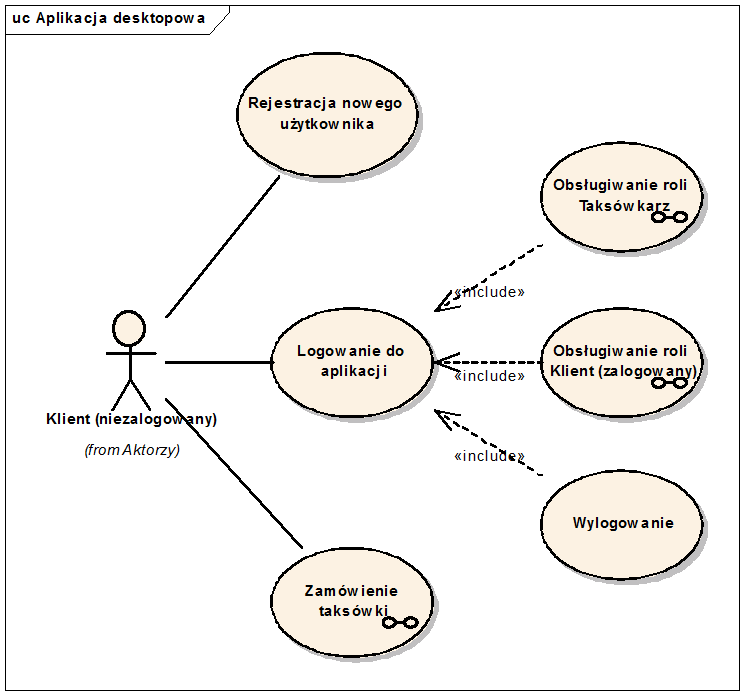
Użytkownik aplikacji desktopowej, kierowca taksówki. Rola Taksówkarz zezwala na przyjmowanie zamówień i wykonywanie kursów oraz ewentualne zgłaszanie zaistniałych problemów.

#### Użytkownik

Abstrakcyjny użytkownik aplikacji internetowej lub desktopowej.

### Aplikacja desktopowa

Diagram na Rys. 28 zawiera przypadki użycia występujące w aplikacji desktopowej – module taksówkarza lub klienta, w zależności od zalogowanego użytkownika.



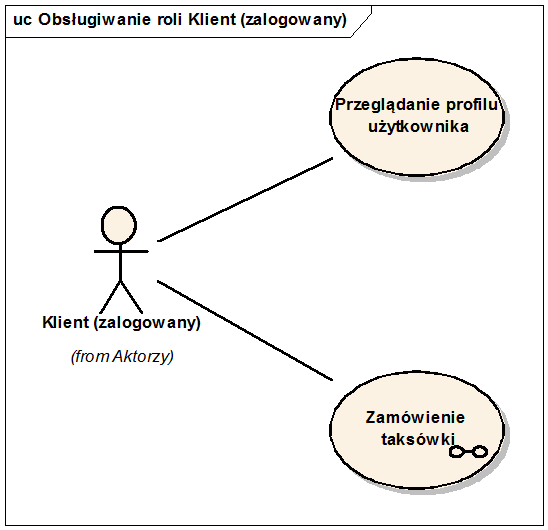
Rys.28. Aplikacja desktopowa

#### Logowanie do aplikacji

Umożliwia zalogowanie się do aplikacji desktopowej. Zalogowanie się wymaga podania loginu i hasła użytkownika. Taksówkarz otrzymuje login i hasło po zatrudnieniu go przez szefa korporacji. Klient tworzy swoje konto w procesie rejestracji.

#### Obsługiwanie roli Klient (zalogowany)

Złożony, abstrakcyjny przypadek użycia, zawierający przypadki użycia dostępne po zalogowaniu się do aplikacji desktopowej, jako Klient.



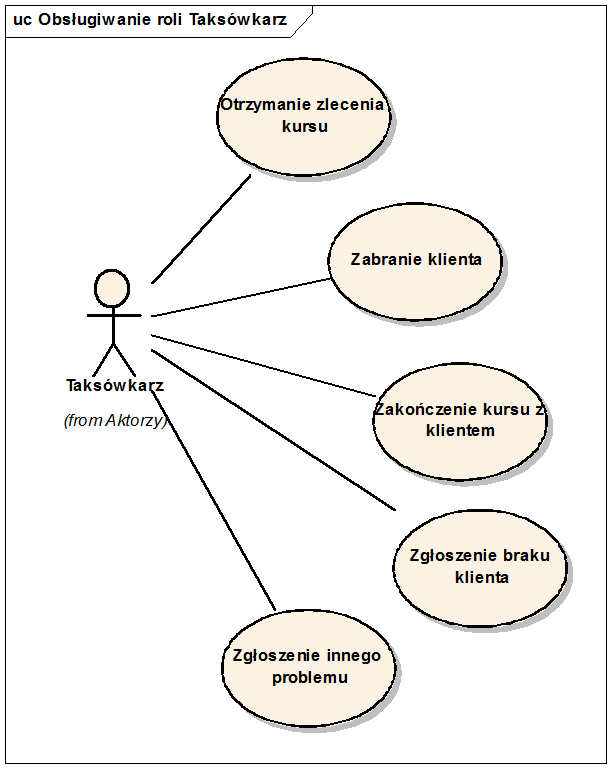
Rys. 29. Obsługiwanie roli Klient (zalogowany)

##### Przeglądanie profilu użytkownika

Pozwala na przeglądanie profilu użytkownika. Wyświetlane są wszystkie dotychczasowe kursy, które użytkownik do tej pory zamówił, będąc zalogowanym do aplikacji.

#### Obsługiwanie roli Taksówkarz

Złożony, abstrakcyjny przypadek użycia, zawierający przypadki użycia dostępne po zalogowaniu się do aplikacji desktopowej jako Taksówkarz.



Rys. 30. Obsługiwanie roli Taksówkarz

##### Otrzymanie zlecenia kursu

Bezpośrednio po zalogowaniu się do aplikacji, lub po zakończeniu poprzedniego kursu, taksówkarz może otrzymać zlecenie kolejnego kursu. Zlecenie jest wyświetlane w głównym oknie aplikacji.

##### Zabranie klienta

Taksówkarz, po dotarciu na miejsce rozpoczęcia kursu, zabiera klienta i zgłasza tę czynność w aplikacji.

##### Zakończenie kursu z klientem

Taksówkarz zgłasza zakończenie kursu po dotarciu z klientem do miejsca zakończenia kursu.

##### Zgłoszenie braku klienta

Taksówkarz, po dotarciu na miejsce rozpoczęcia kursu, jeżeli klient nie oczekuje na niego, może zgłosić brak klienta. Zgłoszenie braku klienta jest możliwe po 5 minutach oczekiwania. Wykonanie zgłoszenia powoduje anulowanie kursu.

##### Zgłoszenie innego problemu

Taksówkarz może w dowolnym momencie zgłosić problem, który nie pozwala mu na wykonanie kursu. Problemem może być np. awaria samochodu. Po zgłoszeniu problemu bieżący kurs jest anulowany, a taksówkarz jest wylogowywany z aplikacji, aby kolejne zgłoszenia nie zostały mu przydzielane. Taksówkarz powinien ponownie zalogować się dopiero, gdy będzie w stanie wykonać kolejny kurs.

#### Rejestracja nowego użytkownika

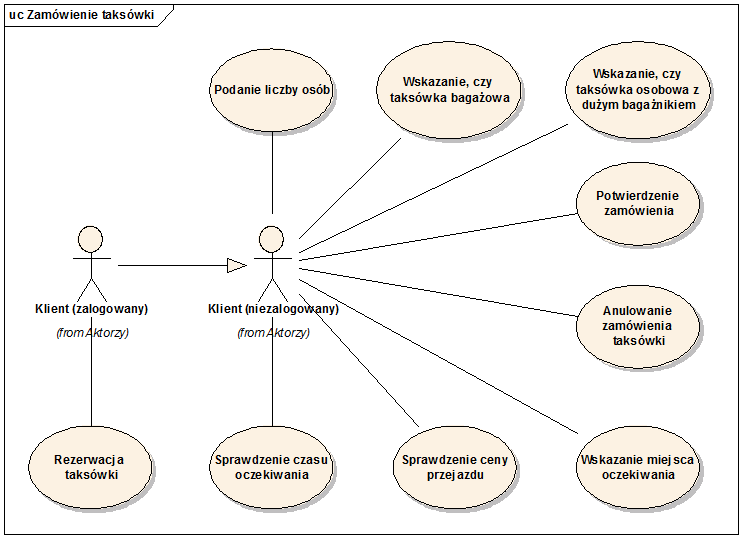
Przypadek użycia umożliwiający zarejestrowanie nowego użytkownika jako klienta. Rejestracja wymaga podania loginu i hasła, które będą później wykorzystywane podczas logowania. Po udanej rejestracji użytkownik jest od razu logowany.

#### Wylogowanie

Umożliwia wylogowanie się z aplikacji. Wylogowanie się taksówkarza oznacza zakończenie pracy przez niego.

#### Zamówienie taksówki

Złożony przypadek użycia, zawierający wszystkie opcje dostępne podczas zamawiania taksówki.



Rys. 31. Zamówienie taksówki

##### Anulowanie zamówienia taksówki

Umożliwia anulowanie zamówienia po jego złożeniu. Anulowanie zamówienia jest możliwe do momentu przyjazdu taksówkarza na miejsce oczekiwania klienta.

##### Podanie liczby osób

Umożliwia podanie liczby osób, które pojadą danym kursem. Nie jest to obowiązkowe, ale w przypadku dużej liczby osób pozwala na dobranie większej taksówki.

##### Potwierdzenie zamówienia

Pozwala na potwierdzenie wykonania zamówienia. Po potwierdzeniu, zamówienie jest wysyłane do aplikacji, a następnie przydzielane jednemu z dostępnych taksówkarzy.

##### Rezerwacja taksówki

Umożliwia wykonanie rezerwacji taksówki na konkretny dzień i godzinę. Rezerwacja jest możliwa w przedziale od jednej godziny do jednego miesiąca.

##### Sprawdzenie ceny przejazdu

Pozwala na sprawdzenie ceny przejazdu pomiędzy miejscem oczekiwania a miejscem docelowym.

##### Sprawdzenie czasu oczekiwania

Umożliwia sprawdzenie, jak długo klient będzie czekał na przybycie taksówki we wskazane miejsce oczekiwania.

##### Wskazanie miejsca oczekiwania

Pozwala na wskazanie miejsca oczekiwania, gdzie powinna podjechać taksówka. Po wskazaniu, użytkownik ma możliwość zmiany miejsca oczekiwania. Gdy zamówienie zostanie złożone, miejsce oczekiwania nie może zostać zmienione.

##### Wskazanie, czy taksówka bagażowa

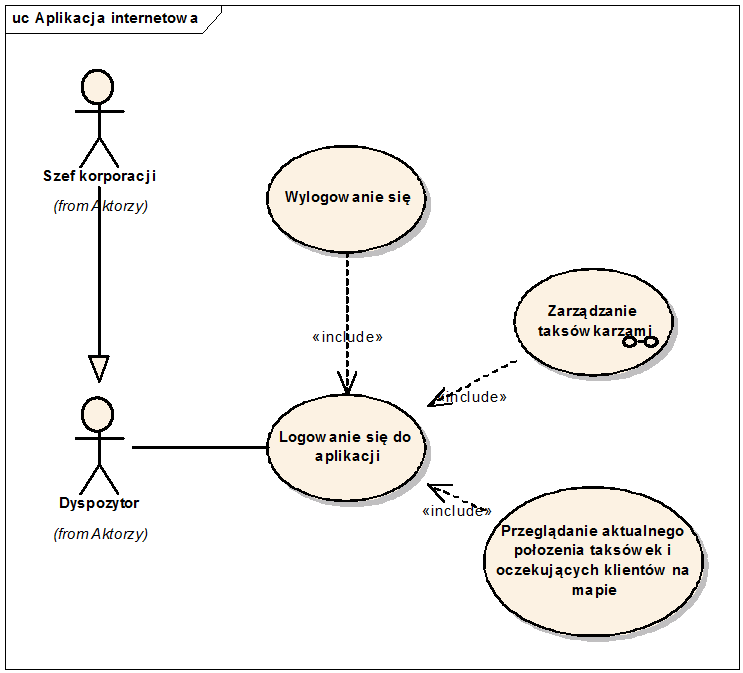
Umożliwia wskazanie, czy powinna podjechać taksówka bagażowa.

##### Wskazanie, czy taksówka osobowa z dużym bagażnikiem

Pozwala na wskazanie, czy klient zamawia taksówkę z dużym bagażnikiem (kombi).

### Aplikacja internetowa

Diagram na Rys. 6 przedstawia przypadki użycia dostępne w aplikacji internetowej, będącej modułem administracyjnym systemu.



Rys. 32. Aplikacja internetowa

#### Logowanie się do aplikacji

Umożliwia zalogowanie się do aplikacji. Zalogowanie się wymaga podania loginu i hasła.

#### Przeglądanie aktualnego położenia taksówek i oczekujących klientów na mapie

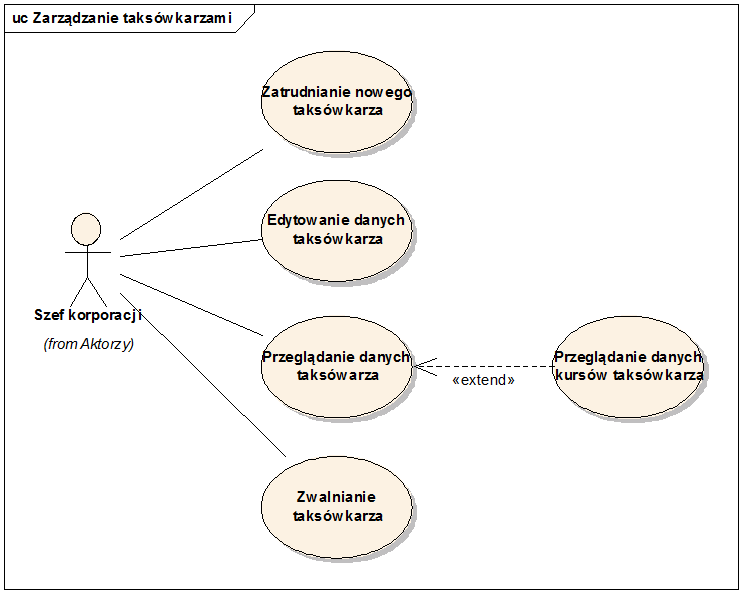
Umożliwia przeglądanie na mapie bieżącej sytuacji. Mapa przedstawia aktualne położenie zalogowanych taksówkarzy oraz oczekujących klientów.

#### Wylogowanie się

Pozwala na wylogowanie się z aplikacji internetowej.

#### Zarządzanie taksówkarzami

Złożony, abstrakcyjny przypadek użycia, zawierający przypadki użycia związane z zarządzaniem taksówkarzami. Przypadki użycia przedstawiono na Rys. 33 i opisano w dalszych punktach.



Rys. 33. Zarządzanie taksówkarzami

##### Edytowanie danych taksówkarza

Umożliwia modyfikację danych taksówkarza – jego danych osobowych oraz danych dostępowych do aplikacji: loginu i hasła.

##### Przeglądanie danych kursów taksówkarza

Pozwala na przeglądanie kursów wykonanych przez taksówkarza.

##### Przeglądanie danych taksówkarza

Pozwala na przeglądanie danych taksówkarza.

##### Zatrudnianie nowego taksówkarza

Umożliwia zatrudnienie nowego taksówkarza. Wraz z dodaniem nowego taksówkarza zostaje utworzone nowe konto użytkownika z rolą Taksówkarz, dzięki któremu kierowca może zalogować się do aplikacji desktopowej i realizować zamówienia.

##### Zwalnianie taksówkarza

Umożliwia zwolnienie taksówkarza. Po zwolnieniu, taksówkarz nie jest widoczny na liście taksówkarzy, nie ma on też możliwości logowania się do aplikacji desktopowej.

## Model fizyczny bazy danych

Na Rys. 34 przedstawiono schemat bazy danych zaprojektowanej do realizacji projektu stanowiącego przedmiot niniejszej pracy. W kolejnych punktach opisano szczegółową strukturę każdej z tabel bazy danych.



Rys. 34. Model fizyczny bazy danych

### mt\_aktualne\_tokeny

Tabela zawiera tokeny wykorzystywane do autoryzacji poszczególnych aplikacji klienckich podczas ich komunikacji z serwerem.

|  |  |  |
| --- | --- | --- |
| Nazwa kolumny | Typ danych | Opis |
| id | integer not null | Identyfikator tokena |
| token | varchar(50) not null | Wartość tokena |

### mt\_dane\_skrzyzowania

Tabela przechowuje dane skrzyżowań – wierzchołków grafu.

|  |  |  |
| --- | --- | --- |
| Nazwa kolumny | Typ danych | Opis |
| id | integer not null | Identyfikator skrzyżowania |
| polozenie\_x | numeric(10,6) not null | Współrzędna X położenia – długość geograficzna |
| polozenie\_y | numeric(10,6) not null | Współrzędna Y położenia – szerokość geograficzna |
| opis | varchar(200) | Opis skrzyżowania, nazwy ulic wychodzących ze skrzyżowania |
| sasiednie\_ids | varchar(50) | Identyfikatory sąsiednich skrzyżowań |

Ograniczenia:

| Nazwa ograniczenia | Typ | Kolumny | Opis |
| --- | --- | --- | --- |
| PK\_mt\_dane\_skrzyzowania | «PK» | id | Klucz główny tabeli |

Relacje:

| Kolumny | Asocjacja |
| --- | --- |
| (skrzyzowanie\_start = id) | 0..\* mt\_dane\_ulice.mt\_ulice\_skrzyzowania\_fk1  **1** mt\_dane\_skrzyzowania**.** PK\_mt\_dane\_skrzyzowania |
| (skrzyzowanie\_stop = id) | 0..\* mt\_dane\_ulice.mt\_ulice\_skrzyzowania\_fk2  **1** mt\_dane\_skrzyzowania**.** PK\_mt\_dane\_skrzyzowania |
| (skrzyzowanie\_2 = id) | 0..\* mt\_sasiednie\_skrzyzowania.mt\_ss\_fk2  **1** mt\_dane\_skrzyzowania**.** PK\_mt\_dane\_skrzyzowania |
| (skrzyzowanie\_1 = id) | 0..\* mt\_sasiednie\_skrzyzowania.mt\_ss\_fk1  **1** mt\_dane\_skrzyzowania**.** PK\_mt\_dane\_skrzyzowania |

### mt\_dane\_ulice

Tabela przechowująca dane ulic – krawędzi grafu. Każdy odcinek ulicy pomiędzy dwoma skrzyżowaniami jest oddzielnym rekordem w tej tabeli.

|  |  |  |
| --- | --- | --- |
| Nazwa kolumny | Typ danych | Opis |
| id | integer not null | Identyfikator ulicy |
| nazwa | varchar(100) | Nazwa ulicy |
| skrzyzowanie\_start | integer not null | Identyfikator skrzyżowania początkowego |
| skrzyzowanie\_stop | integer not null | Identyfikator skrzyżowania końcowego |
| jednokierunkowa | boolean not null default false | Czy ulica jest jednokierunkowa |
| dlugosc | numeric(6,3) not null | Długość ulicy (w kilometrach) |

Ograniczenia:

| Nazwa ograniczenia | Typ | Kolumny | Opis |
| --- | --- | --- | --- |
| mt\_ulice\_skrzyzowania\_fk1 | «FK» | skrzyzowanie\_start | Klucz obcy do tabeli mt\_dane\_skrzyżowania |
| mt\_ulice\_skrzyzowania\_fk2 | «FK» | skrzyzowanie\_stop | Klucz obcy do tabeli mt\_dane\_skrzyżowania |
| PK\_mt\_dane\_ulice | «PK» | id | Klucz główny tabeli |

Relacje:

| Kolumny | Asocjacja |
| --- | --- |
| (skrzyzowanie\_start = id) | 0..\* mt\_dane\_ulice.mt\_ulice\_skrzyzowania\_fk1  **1** mt\_dane\_skrzyzowania**.** PK\_mt\_dane\_skrzyzowania |
| (skrzyzowanie\_stop = id) | 0..\* mt\_dane\_ulice.mt\_ulice\_skrzyzowania\_fk2  **1** mt\_dane\_skrzyzowania**.** PK\_mt\_dane\_skrzyzowania |

### mt\_klienci

Tabela zawiera dane zarejestrowanych klientów.

|  |  |  |
| --- | --- | --- |
| Nazwa kolumny | Typ danych | Opis |
| id | integer not null | Identyfikator klienta |
| uzytkownik\_id | integer not null | Identyfikator użytkownika |

Ograniczenia:

| Nazwa ograniczenia | Typ | Kolumny | Opis |
| --- | --- | --- | --- |
| mt\_klienci\_uzytkownicy\_fk | «FK» | uzytkownik\_id | Klucz obcy do tabeli mt\_uzytkownicy |
| PK\_mt\_klienci | «PK» | id | Klucz główny tabeli |

Relacje:

| Kolumny | Asocjacja |
| --- | --- |
| (klient\_id = id) | 0..\* mt\_zamowienia.mt\_zamowienia\_klienci\_FK  **1** mt\_klienci**.** PK\_mt\_klienci |
| (uzytkownik\_id = id) | 0..\* mt\_klienci.mt\_klienci\_uzytkownicy\_fk  **1** mt\_uzytkownicy**.** PK\_mt\_uzytkownicy |

### mt\_menu\_role

Tabela asocjacyjna, przechowująca przypisanie pozycji menu do konkretnych ról.

|  |  |  |
| --- | --- | --- |
| Nazwa kolumny | Typ danych | Opis |
| rola\_id | integer not null | Identyfikator roli |
| menu\_id | integer not null | Identyfikator pozycji menu |

Ograniczenia:

| Nazwa ograniczenia | Typ | Kolumny | Opis |
| --- | --- | --- | --- |
| mt\_menu\_role\_menu\_fk | «FK» | menu\_id | Klucz obcy do tabeli mt\_pozycje\_menu |
| mt\_menu\_role\_role\_fk | «FK» | rola\_id | Klucz obcy do tabeli mt\_role |

Relacje:

|  |  |
| --- | --- |
| Kolumny | Asocjacja |
| (rola\_id = id) | 0..\* mt\_menu\_role.mt\_menu\_role\_role\_fk  **1** mt\_role**.** PK\_mt\_role |
| (menu\_id = id) | 0..\* mt\_menu\_role.mt\_menu\_role\_menu\_fk  **1** mt\_pozycje\_menu**.** PK\_mt\_pozycje\_menu |

### mt\_pozycje\_menu

Tabela przechowuje dane pozycji menu, wyświetlanych w aplikacji internetowej.

|  |  |  |
| --- | --- | --- |
| Nazwa kolumny | Typ danych | Opis |
| id | integer not null | Identyfikator pozycji menu |
| tekst | varchar(100) not null | Tekst pozycji menu |
| url | varchar(150) not null | Adres URL wywoływany po kliknięciu danej pozycji menu |

Ograniczenia:

| Nazwa ograniczenia | Typ | Kolumny | Opis |
| --- | --- | --- | --- |
| PK\_mt\_pozycje\_menu | «PK» | id | Klucz główny tabeli |

Relacje:

| Kolumny | Asocjacja |
| --- | --- |
| (menu\_id = id) | 0..\* mt\_menu\_role.mt\_menu\_role\_menu\_fk  **1** mt\_pozycje\_menu**.** PK\_mt\_pozycje\_menu |

### mt\_role

Tabela zawiera dane ról wykorzystywanych w aplikacji.

|  |  |  |
| --- | --- | --- |
| Nazwa kolumny | Typ danych | Opis |
| id | integer not null | Identyfikator roli |
| nazwa\_roli | varchar(50) not null | Nazwa roli |
| opis | varchar(200) | Opis roli |

Ograniczenia:

| Nazwa ograniczenia | Typ | Kolumny | Opis |
| --- | --- | --- | --- |
| PK\_mt\_role | «PK» | id | Klucz główny tabeli |

Relacje:

| Kolumny | Asocjacja |
| --- | --- |
| (rola\_id = id) | 0..\* mt\_menu\_role.mt\_menu\_role\_role\_fk  **1** mt\_role**.** PK\_mt\_role |
| (rola\_id = id) | 0..\* mt\_role\_uzytkownikow.mt\_role\_uzy\_role\_fk  **1** mt\_role**.** PK\_mt\_role |

### mt\_role\_uzytkownikow

Tabela asocjacyjna, przechowująca przypisanie ról konkretnym użytkownikom.

|  |  |  |
| --- | --- | --- |
| Nazwa kolumny | Typ danych | Opis |
| rola\_id | integer not null | Identyfikator roli |
| uzytkownik\_id | integer not null | Identyfikator użytkownika |

Ograniczenia:

| Nazwa ograniczenia | Typ | Kolumny | Opis |
| --- | --- | --- | --- |
| mt\_role\_uzy\_role\_fk | «FK» | rola\_id | Klucz obcy do tabeli mt\_role |
| mt\_role\_uzy\_uzy\_fk | «FK» | uzytkownik\_id | Klucz obcy do tabeli mt\_uzytkownicy |

Relacje:

|  |  |
| --- | --- |
| Kolumny | Asocjacja |
| (rola\_id = id) | 0..\* mt\_role\_uzytkownikow.mt\_role\_uzy\_role\_fk  **1** mt\_role**.** PK\_mt\_role |
| (uzytkownik\_id = id) | 0..\* mt\_role\_uzytkownikow.mt\_role\_uzy\_uzy\_fk  **1** mt\_uzytkownicy**.** PK\_mt\_uzytkownicy |

### mt\_sasiednie\_skrzyzowania

Tabela zawiera pary identyfikatorów skrzyżowań, które ze sobą sąsiadują, tzn. istnieje ulica (krawędź grafu) prowadząca bezpośrednio od jednego do drugiego skrzyżowania.

|  |  |  |
| --- | --- | --- |
| Nazwa kolumny | Typ danych | Opis |
| skrzyzowanie\_1 | integer not null | Identyfikator pierwszego skrzyżowania |
| skrzyzowanie\_2 | integer not null | Identyfikator drugiego skrzyżowania |

Ograniczenia:

| Nazwa ograniczenia | Typ | Kolumny | Opis |
| --- | --- | --- | --- |
| mt\_ss\_fk1 | «FK» | skrzyzowanie\_1 | Klucz obcy do tabeli mt\_dane\_skrzyżowania |
| mt\_ss\_fk2 | «FK» | skrzyzowanie\_2 | Klucz obcy do tabeli mt\_dane\_skrzyżowania |

Relacje:

| Kolumny | Asocjacja |
| --- | --- |
| (skrzyzowanie\_2 = id) | 0..\* mt\_sasiednie\_skrzyzowania.mt\_ss\_fk2  **1** mt\_dane\_skrzyzowania**.** PK\_mt\_dane\_skrzyzowania |
| (skrzyzowanie\_1 = id) | 0..\* mt\_sasiednie\_skrzyzowania.mt\_ss\_fk1  **1** mt\_dane\_skrzyzowania**.** PK\_mt\_dane\_skrzyzowania |

### mt\_taksowkarze

Tabela zawierająca dane taksówkarzy.

|  |  |  |
| --- | --- | --- |
| Nazwa kolumny | Typ danych | Opis |
| id | integer not null | Identyfikator taksówkarza |
| imie | varchar(30) not null | Imię taksówkarza |
| nazwisko | varchar(50) not null | Nazwisko taksówkarza |
| uzytkownik\_id | integer not null | Identyfikator użytkownika przypisanego do taksówkarza |
| czy\_aktywny | boolean not null default false | Czy taksówkarz zalogowany |
| polozenie\_x | numeric(10,6) | Współrzędna X położenia – długość geograficzna |
| polozenie\_y | numeric(10,6) | Współrzędna Y położenia – szerokość geograficzna |
| status | varchar(1) not null default ‘1’ | Status taksówkarza. ‘1’ – zatrudniony, ‘2’ – zwolniony. |

Ograniczenia:

| Nazwa ograniczenia | Typ | Kolumny | Opis |
| --- | --- | --- | --- |
| mt\_taksowkarze\_uzy\_FK | «FK» | uzytkownik\_id | Klucz obcy do tabeli mt\_uzytkownicy |
| PK\_mt\_taksowkarze | «PK» | id | Klucz główny tabeli |

Relacje:

| Kolumny | Asocjacja |
| --- | --- |
| (taksowkarz\_id = id) | 0..\* mt\_zamowienia.mt\_zamowienia\_taksowkarze\_fk  **1** mt\_taksowkarze**.** PK\_mt\_taksowkarze |
| (uzytkownik\_id = id) | 0..\* mt\_taksowkarze.mt\_taksowkarze\_uzy\_FK  **1** mt\_uzytkownicy**.** PK\_mt\_uzytkownicy |

### mt\_uzytkownicy

Tabela przechowująca dane użytkowników aplikacji.

|  |  |  |
| --- | --- | --- |
| Nazwa kolumny | Typ danych | Opis |
| id | integer not null | Identyfikator użytkownika |
| nazwa | varchar(50) not null | Nazwa użytkownika, login do aplikacji |
| haslo | varchar(50) not null | Hasło użytkownika zaszyfrowane algorytmem SHA1 |

Ograniczenia:

| Nazwa ograniczenia | Typ | Kolumny | Opis |
| --- | --- | --- | --- |
| PK\_mt\_uzytkownicy | «PK» | id | Klucz główny tabeli |

Relacje:

| Kolumny | Asocjacja |
| --- | --- |
| (uzytkownik\_id = id) | 0..\* mt\_role\_uzytkownikow.mt\_role\_uzy\_uzy\_fk  **1** mt\_uzytkownicy**.** PK\_mt\_uzytkownicy |
| (uzytkownik\_id = id) | 0..\* mt\_klienci.mt\_klienci\_uzytkownicy\_fk  **1** mt\_uzytkownicy**.** PK\_mt\_uzytkownicy |
| (uzytkownik\_id = id) | 0..\* mt\_taksowkarze.mt\_taksowkarze\_uzy\_FK  **1** mt\_uzytkownicy**.** PK\_mt\_uzytkownicy |

### mt\_zamowienia

Tabela przechowująca wszystkie zamówienia taksówki, bieżące, archiwalne oraz rezerwacje.

|  |  |  |
| --- | --- | --- |
| Nazwa kolumny | Typ danych | Opis |
| id | integer not null | Identyfikator zamówienia |
| czas\_zamowienia | timestamp not null | Czas dokonania zamówienia. Dla rezerwacji jest to czas, na który dokonano rezerwacji |
| start\_x | numeric(10,6) not null | Współrzędna X miejsca startowego – długość geograficzna |
| start\_y | numeric(10,6) not null | Współrzędna Y miejsca startowego – szerokość geograficzna |
| start\_adres | varchar(50) not null | Adres miejsca startowego |
| koniec\_x | numeric(10,6) | Współrzędna X miejsca końcowego – długość geograficzna |
| koniec\_y | numeric(10,6) | Współrzędna Y miejsca końcowego – szerokość geograficzna |
| koniec\_adres | varchar(50) | Adres miejsca końcowego |
| status | varchar(1) not null default ‘1’ | Status zamówienia.  ‘1’ – zamówienie złożone,  ‘2’ – przydzielono taksówkarza do zamówienia,  ‘3’ – taksówkarz odebrał klienta,  ‘4’ – zamówienie zakończone, ‘5’ – brak klienta, ‘6’ - zamówienie anulowane |
| taksowkarz\_id | integer | Identyfikator taksówkarza |
| klient\_id | integer | Identyfikator klienta, uzupełniony tylko dla zalogowanego klienta |
| uwagi | varchar(300) | Treść problemu zgłoszonego przez taksówkarza, uzupełniana dla anulowanego zamówienia |

Ograniczenia:

| Nazwa ograniczenia | Typ | Kolumny | Opis |
| --- | --- | --- | --- |
| mt\_zamowienia\_klienci\_FK | «FK» | klient\_id | Klucz obcy do tabeli mt\_klienci |
| mt\_zamowienia\_taksowkarze\_fk | «FK» | taksowkarz\_id | Klucz obcy do tabeli mt\_taksowkarze |
| PK\_mt\_zamowienia | «PK» | id | Klucz główny tabeli |

Relacje:

|  |  |
| --- | --- |
| Kolumny | Asocjacja |
| (klient\_id = id) | 0..\* mt\_zamowienia.mt\_zamowienia\_klienci\_FK  **1** mt\_klienci**.** PK\_mt\_klienci |
| (taksowkarz\_id = id) | 0..\* mt\_zamowienia.mt\_zamowienia\_taksowkarze\_fk  **1** mt\_taksowkarze**.** PK\_mt\_taksowkarze |

# Podsumowanie

Celem niniejszej pracy było wykonanie systemu realizującego funkcjonalność automatycznej dyspozytorni taksówek.

Przed rozpoczęciem implementacji, zapoznano się z aktualnie funkcjonującymi rozwiązaniami, które są wykorzystywane komercyjnie. Na tej podstawie przygotowano zakres przypadków użycia, który powinien zostać zrealizowany, aby powstały system mógł być użytkowany w różnych sytuacjach. Następnie, dokonano wyboru algorytmów realizujących główne funkcjonalności tj. algorytm wyznaczający najkrótszą trasę dla taksówki oraz algorytm kontroli dostępu do poszczególnych części systemu.

Zrealizowana w ramach niniejszej pracy aplikacja została wykonana zgodnie z zasadami poprawnego projektowania oraz programowania. Zadbano także o schludny i przejrzysty interfejs użytkownika. Ponadto przygotowano dokumentację techniczną obejmującą opis przypadków użycia oraz schemat bazy danych.

Ze względu na specyfikę aplikacji, kolejnym krokiem w jej rozwoju, który mógłby zostać wykonany w przyszłości, jest przeniesienie modułu klienta i taksówkarza na technologię umożliwiającą uruchomienie go na telefonie komórkowym. Dzięki temu, system mógłby być wykorzystywany na równi z istniejącymi rozwiązaniami.

# Bibliografia

1. http://system-taxi.pl/ (data dostępu: 2015-10-14)
2. https://tiskel.com/system-taxi (data dostępu: 2015-10-14)
3. „Grafy i rekurencje”, J.Marecki
4. https://maps.googleapis.com/maps/api/staticmap?center=50.061781,19.937242&size=640x640&zoom=15&scale=1 (data dostępu: 2015-10-14)
5. https://eclipse.org/downloads/ (data dostępu: 2015-10-14)
6. „Thinking in Java”, B. Eckel, Helion 2006
7. http://tomcat.apache.org/download-70.cgi (data dostępu 2015-10-14)
8. https://docs.oracle.com/javaee/5/tutorial/doc/bnagx.html (data dostępu 2015-10-14)
9. http://www.postgresql.org/download (data dostępu: 2015-10-14)
10. https://github.com/smolo/workspaceInz (data dostępu: 2015-10-14)
11. http://www.sparxsystems.com.au/products/ea/trial.html (data dostępu: 2015-10-14)