|  |  |  |
| --- | --- | --- |
| wordml://75.png |  | wordml://76.png |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | |  | | --- | | Imię i nazwisko studenta: Artur Kąkol | | | |  | | --- | | Nr albumu: 143251 | | | |  | | --- | | Studia drugiego stopnia | | | |  | | --- | | Forma studiów: stacjonarne | | | |  | | --- | | Kierunek studiów: Informatyka | | | |  | | --- | | Specjalność/profil: Algorytmy i technologie internetowe | | |  |
|  |  |
| |  | | --- | | **PRACA DYPLOMOWA MAGISTERSKA** | | |
| |  | | --- | | Tytuł pracy w języku polskim: System internetowy wspierający marszrutyzację pojazdów | | Tytuł pracy w języku angielskim: Web-based system supporting Vehicle Routing Problems | |  | | |
| |  |  |  |  | | --- | --- | --- | --- | |  | |  | | --- | | Potwierdzenie przyjęcia pracy | | | |  | |  | | --- | | Opiekun pracy |   *podpis* | |  | | --- | | Kierownik Katedry/Zakładu |   *podpis* | |  | dr inż. Krzysztof Bruniecki |  | | |
| |  | | --- | | Data oddania pracy do dziekanatu: | | |

|  |  |  |  |
| --- | --- | --- | --- |
| wordml://75.png | |  | wordml://76.png |
|  | **OŚWIADCZENIE** | | |
|  | |  | | --- | | Imię i nazwisko: Artur Kąkol  Data i miejsce urodzenia: 27.05.1992, Kościerzyna  Nr albumu: 143251  Wydział: Wydział Elektroniki, Telekomunikacji i Informatyki  Kierunek: informatyka  Poziom studiów: II stopnia  Forma studiów: stacjonarne | | | |
|  | |  | | --- | | Ja, niżej podpisany(a), wyrażam zgodę/nie wyrażam zgody\* na korzystanie z mojej pracy dyplomowej zatytułowanej: Licznik uśmiechów  do celów naukowych lub dydaktycznych.1 | | | |
| |  |  | | --- | --- | | Gdańsk, dnia .................................. | .....................................................  *podpis studenta* | | | | |
|  | |  | | --- | | Świadomy(a) odpowiedzialności karnej z tytułu naruszenia przepisów ustawy z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (Dz. U. z 2006 r., nr 90, poz. 631) i konsekwencji dyscyplinarnych określonych w ustawie Prawo o szkolnictwie wyższym (Dz. U. z 2012 r., poz. 572 z późn. zm.),2 a także odpowiedzialności cywilno-prawnej oświadczam, że przedkładana praca dyplomowa została opracowana przeze mnie samodzielnie.  Niniejsza(y) praca dyplomowa nie była wcześniej podstawą żadnej innej urzędowej procedury związanej z nadaniem tytułu zawodowego.  Wszystkie informacje umieszczone w ww. pracy dyplomowej, uzyskane ze źródeł pisanych i elektronicznych, zostały udokumentowane w wykazie literatury odpowiednimi odnośnikami zgodnie z art. 34 ustawy o prawie autorskim i prawach pokrewnych.  Potwierdzam zgodność niniejszej wersji pracy dyplomowej z załączoną wersją elektroniczną. | | | |
| |  |  | | --- | --- | | Gdańsk, dnia .................................. | .....................................................  *podpis studenta* | | | | |
|  | Upoważniam Politechnikę Gdańską do umieszczenia ww. pracy dyplomowej w wersji elektronicznej w otwartym, cyfrowym repozytorium instytucjonalnym Politechniki Gdańskiej oraz poddawania jej procesom weryfikacji i ochrony przed przywłaszczaniem jej autorstwa. | | |
| |  |  | | --- | --- | | Gdańsk, dnia ................................. | .....................................................  *podpis studenta* | | | | |
|  | |  | | --- | | \*) niepotrzebne skreślić | | | |
| |  | | --- | |  | | | | |
| |  |  |  |  | | --- | --- | --- | --- | | |  | | --- | | 1 | | |  | | --- | | Zarządzenie Rektora Politechniki Gdańskiej nr 34/2009 z 9 listopada 2009 r., załącznik nr 8 do instrukcji archiwalnej PG. | | | |  | | --- | | 2 | | |  | | --- | | Ustawa z dnia 27 lipca 2005 r. Prawo o szkolnictwie wyższym: | | |  | |  | | --- | | Art. 214 ustęp 4. W razie podejrzenia popełnienia przez studenta czynu podlegającego na przypisaniu sobie autorstwa istotnego fragmentu lub innych elementów cudzego utworu rektor niezwłocznie poleca przeprowadzenie postępowania wyjaśniającego. | | |  | |  | | --- | | Art. 214 ustęp 6. Jeżeli w wyniku postępowania wyjaśniającego zebrany materiał potwierdza popełnienie czynu, o którym mowa w ust. 4, rektor wstrzymuje postępowanie o nadanie tytułu zawodowego do czasu wydania orzeczenia przez komisję dyscyplinarną oraz składa zawiadomienie o popełnieniu przestępstwa. | | | | | |

Streszczenie

Celem pracy jest utworzenie systemu pozwalającego na wspieranie marszrutyzacji pojazdów.

W ramach projektu powstała aplikacja pozwalająca na wyznaczanie tras dla pojazdów. Aplikację można podzielić na dwie części. Pierwszą z nich jest aplikacja kliencka oparta na frameworku Angular, pozwalająca na zarządzanie magazynami oraz odbiorcami. Pozwala także na wyświetlanie tras na mapie. Drugą częścią są usługi RESTowe pozwalające na obliczanie rozwiązania na podstawie zadanych parametrów.

**Słowa kluczowe:** marszrutyzacja pojazdów

**Dziedzina nauki i techniki, zgodnie z wymogami OECD:** Nauki inżynieryjne i techniczne. Elektrotechnika, elektronika, inżynieria informatyczna.

Abstract

**Keywords:**

**OECD field of science and technology (FOS) classification:**.

Spis treści

[Streszczenie 3](#_Toc492842372)

[Abstract 4](#_Toc492842373)

[Spis treści 5](#_Toc492842374)

[1. Wstęp 6](#_Toc492842375)

[1.1. Cele pracy 7](#_Toc492842376)

[1.2. Tematyka pracy 7](#_Toc492842377)

[1.3. Planowanie tras pojazdów 7](#_Toc492842378)

[1.3.1. Dokładne metody obliczeniowe 8](#_Toc492842379)

[1.3.1.1. Branch-and-cut algorithms 8](#_Toc492842380)

[1.3.2. Heurystyczne metody obliczeniowe 8](#_Toc492842381)

[1.3.2.1. Savings algorithms 8](#_Toc492842382)

[1.3.2.1.1. Parallel version 8](#_Toc492842383)

[1.3.2.1.2. Sequential version 8](#_Toc492842384)

[1.3.2.2. The sweep algorithm 8](#_Toc492842385)

[1.3.2.3. Petal algorithms 8](#_Toc492842386)

[1.3.2.4. Cluster-first, route-second algorithms 8](#_Toc492842387)

[1.3.2.5. Improvement heuristic 8](#_Toc492842388)

[1.3.3. Metaherystyczne metody obliczeniowe 8](#_Toc492842389)

[1.3.3.1. Ant algorithms 8](#_Toc492842390)

[1.3.3.2. Tabu Search 8](#_Toc492842391)

[2. Przykłady gotowych rozwiązań 9](#_Toc492842392)

[2.1. Jsprit, Open-VRP, OptaPlaner 9](#_Toc492842393)

[3. Wizja systemu 10](#_Toc492842394)

[4. Opis systemu 11](#_Toc492842395)

[4.1. Architektura 11](#_Toc492842396)

[4.2. Część kliencka 11](#_Toc492842397)

[4.2.1. Dodawanie magazynu lub odbiorcy 12](#_Toc492842398)

[4.2.1. Zmiana ustawień wybranego punktu 12](#_Toc492842399)

[4.2.2. Obliczanie wyniku 13](#_Toc492842400)

[4.2.3. Logowanie informacji 13](#_Toc492842401)

[4.2.4. Wykorzystane technologie 13](#_Toc492842402)

[4.3. Część serwerowa 13](#_Toc492842403)

[4.3.1. Schemat klas 13](#_Toc492842404)

[4.3.2. Przykładowe zapytanie do serwera 14](#_Toc492842405)

[4.3.3. Proces przetwarzania żądania 14](#_Toc492842406)

[4.3.4. Obliczanie odległości pomiędzy punktami 15](#_Toc492842407)

[4.4. Komunikacja 15](#_Toc492842408)

[4.5. Dostęp z urządzeń mobilnych 15](#_Toc492842409)

[5. Walidacja rozwiązania 16](#_Toc492842410)

[6. Podsumowanie 17](#_Toc492842411)

[Wykaz literatury 18](#_Toc492842412)

[Wykaz tabel 19](#_Toc492842413)

[Wykaz rysunków 20](#_Toc492842414)

# Wstęp

We współczesnym świecie transport odgrywa bardzo ważną rolę we współczesnej gospodarce. Zarówno w usługach(transport ludzi) jak i w przemyśle(dostarczanie towarów). Dynamiczny rozwój rynku wymusza na przedsiębiorstwach utrzymania konkurencyjności. Można ją uzyskać na kilka sposobów, między innymi poprzez niższą cenę. Aktualnie przyjmuje się, że koszty paliwa w przedsiębiorstwie zajmującym się transportem drogowym mieszczą się w przedziale 15%-20%[[1]](#endnote-1). Obniżając długości tras można w dużym stopniu przyczynić się do końcowej ceny towaru.

Kolejnymi ważnym aspektem jest też, jakość realizowanych usług. Dlatego warto zadbać o to, żeby czas realizacji był jak najkrótszy oraz sprawność transportu była jak najwyższa. Pomóc w tym może oraz powinien dedykowany system internetowy, dzięki któremu w łatwy oraz szybki sposób można by było wyznaczyć trasy dla poszczególnych pojazdów.

## Cele pracy

Celem pracy jest utworzenie systemu internetowego wspomagającego marszrutyzację pojazdów. Pozwalał on będzie na łatwe zarządzanie punktami docelowymi oraz położeniem magazynu. Każdy z punktów docelowych posiada wymagania, którym należy sprostać. System ten też wspierał będzie urządzenia mobilne.

Dodatkowo ocenione oraz porównane zostaną poszczególne sposoby obliczania tras dla poszczególnych pojazdów we flocie. Porównane zostaną algorytmy oraz zostanie pokazany wpływ, jaki wpływ na ostateczne rozwiązanie ma sposób obliczania odległości pomiędzy poszczególnymi odbiorcami lub magazynem.

## Tematyka pracy

Niniejszy dokument skupia się na zakresie marszrutyzacji pojazdów oraz planowaniu tras dla dostaw towarów. Zagadnienia w tym zakresie mogą być bardzo skomplikowane oraz złożone. Ze względu na rodzaj różnych zastosowań oraz uwarunkowań do rozważenia można wyróżnić wiele wariantów problemu. Najważniejszymi jednak są:

* Zagadnienie komiwojażera (ang. *Travelling Salesman Problem*)
* Klasyczny problem marszrutyzacji (ang. *Vehicle Routing Problem*)

Zagadnienie komiwojażera to najprostszy wariant planowania dostaw. Uwzględnia on tylko jeden środek transportu, bez dodatkowych ograniczeń. Polega na wyznaczenie najkrótszej trasy, która odwiedza wszystkich obiorców oraz wrócić z powrotem do magazynu.

Klasyczny problem marszrutyzacji odpowiada na pytanie: „Jaki jest optymalny zestaw tras dla floty pojazdów w celu dostarczenia dóbr dla określonych odbiorców?”. Po raz pierwszy został zaprezentowany przez G.B. Dantziga oraz R. H. Ramsera w 1959 roku[[2]](#endnote-2). Jako, że zadany problem, jest rozwinięciem problemu komiwojażera to należy do problemów NP-trudnych[[3]](#endnote-3). Istnieją liczne odmiany klasycznego zagadnienia marszrutyzacji.

Jednym z nich jest Capacitated Vehicle Routing Problem (CVRP), w którym flota ma ograniczoną pojemność, która nie może zostać przekroczona. Dodatkowo odbiorca może być odwiedzony tylko i wyłącznie przez jeden pojazd. Magazyn jest jeden dla wszystkich odbiorców. To głównie tą odmiana zostanie wykorzystana w danej pracy.

Kolejnym jest Vehicle Routing Problem with Time Windows (VRPTW). W tym rodzaju każdy z odbiorów przypisane ma ramy czasowe definiujące, kiedy dany odbiorca musi zostać obsłużony. Dopuszcza się, że pojazd może przybyć przed czasem do odbiorcy, jednak w takim wypadku musi czekać na rozładunek/załadunek.

Innymi przykładami mogą być: Vehicle Routing Problem with Multiple Trips (VRPMT), Vehicle Routing Problem with Pickup and Delivery (VRPPD) lub Open Vehicle Routing Problem (OVRP).

## Planowanie tras pojazdów

Planowanie tras pojazdów może być realizowane przez dwa różne rodzaje metod: dokładne oraz przybliżone. Wybór sposobu opiera się głównie na rozmiarze problemu, tj. na ilości odbiorców, ilości ograniczeń oraz typie floty. W przypadku problemów o mniejszym skomplikowaniu zasadnym jest używanie metod dokładnych, natomiast dla bardziej złożonych problemów ze względu na swoją złożoność obliczeniową są zdyskwalifikowane. Metody przybliżone w znacznym stopniu przyspieszają proces obliczeniowy, jednak ich rozwiązanie niekoniecznie jest optymalne (jednak może takie być).

### Dokładne metody obliczeniowe[[4]](#endnote-4)

### Branch-and-cut algorithms

### Heurystyczne metody obliczeniowe[[5]](#endnote-5)

### Savings algorithms

Algorytm Clarke i Wright jest najprawdopodobniej najbardziej znanym algorytmem heurystycznym dla problemów klasy VRP. Może być zastosowany do problemów, w których ilość pojazdów we flocie jest zmienna.

### Parallel version

### Sequential version

### The sweep algorithm

### Petal algorithms

### Cluster-first, route-second algorithms

### Improvement heuristic

### Metaherystyczne metody obliczeniowe

### Ant algorithms

### Tabu Search

# Przykłady gotowych rozwiązań

# Wizja systemu

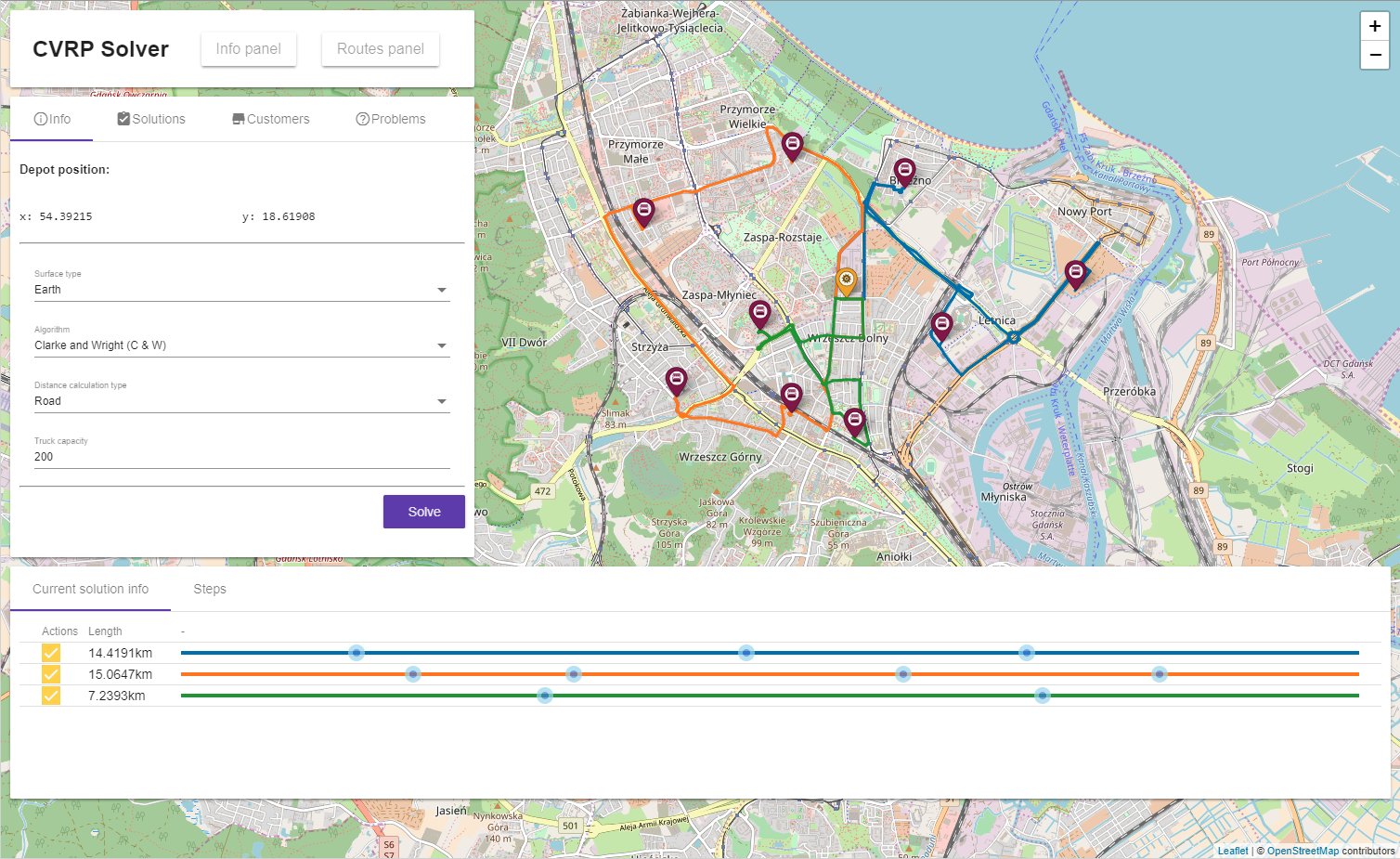
# Opis systemu

## Architektura

Stworzony system został oparty na architekturze klient-serwer. Pozwoliło to na łatwe rozdzielenie obsługi mapy, niezależnego wyświetlania wyników obliczeń oraz części odpowiedzialnej za obliczenia. Pozwoliło to także na ewentualne implementacje innych aplikacji korzystających z części serwerowej – na przykład dedykowanej aplikacji mobilnej. Poniższy diagram przestawia ogólny zarys. Poszczególne części będą opisane w kolejnych sekcjach.

## Część kliencka

Aplikacja kliencka pozwala na edycję magazynów, odbiorców oraz innych ustawień.

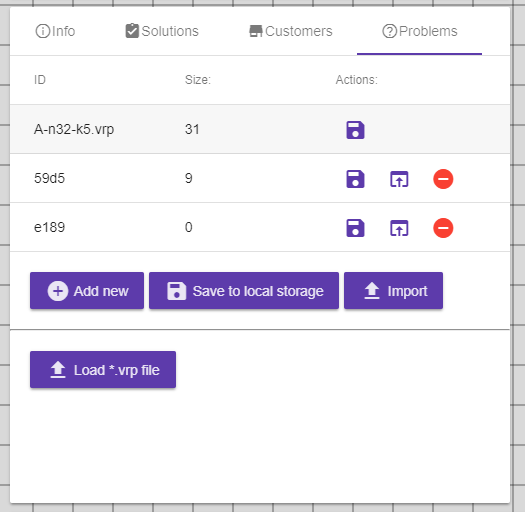


Rysunek Widok aplikacji klienckiej

Jak można zauważyć, większą część aplikacji zajmuje mapa, która aktualnie wyświetla wyliczone trasy. Po lewej stronie znajduje się menu odpowiedzialne na ustawienie parametrów wybranego elementu lub ogólnych ustawień, takich jak wybór algorytmu czy sposób obliczania odległości pomiędzy punktami. Natomiast dolny panel wyświetla aktualnie wyliczone rozwiązanie oraz pośrednie stany działania algorytmu, jeżeli takie istnieją.

### Dodawanie problemu

Dodawanie oraz przełączanie się pomiędzy problemami dostępne jest na zakładce „Problems” dostępnej na panelu dostępnym po lewej stronie ekranu.



Rysunek 2 Menu pozwalające na dodawanie magazynu lub odbiorcy

Aplikacja w tym samym momencie może pracować na kilku problemach. Można się dowolnie pomiędzy nimi przełączać. Na powyższym rysunku można zobaczyć 3 problemy wyświetlone w tabeli. Pierwsza kolumna w tej tabeli oznacza unikalne ID problemu. Następna zatytułowana „*Size*” oznacza liczbę odbiorców w danym problemie. Ostatnią kolumną są akcje, które są dostępne na podanym problemie.

Pierwszą z nich jest export. Problem exportowany jest w formacie *json*. W wygenerowanym pliku znajdują się wszystkie ustawienia oraz wcześniej obliczone rozwiązania.

Kolejną akcją jest załadowanie problemu. Opcja ta nie jest dostępna dla pierwszego z nich, ponieważ jest ona aktualnie wczytany.

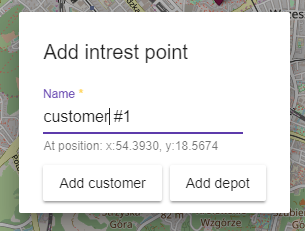
Ostatnią akcją jest usunięcie problemu.

Pod tabelą znajdują się cztery przyciski. Przycisk o nazwie „*Add new*” odpowiedziany jest za dodawanie nowego problemu z domyślnymi ustawieniami. Kolejnym jest „*Save to local storage*”, który zapisuje wszystkie problemy do pamięci lokalnej przeglądarki, dzięki czemu po ponownym otwarciu strony możemy edytować te problemu, nad którymi wcześniej pracowaliśmy. Kolejny przycisk - „*Import*”. Pozwala na wczytanie wcześniej wyeksportowanych problemów w formacie \*.json. Ostatnim przyciskiem dostępnym na tej zakładce jest „*Load \*.vrp file*”, który pozwala na wczytanie problemu z pliku \*.vrp.

Plik ten możemy podzielić na dwie sekcje, część specyfikacji oraz część z danymi. Pierwsza nich składa się z informacji na temat formatu danych natomiast druga zawiera konkretne dane na temat odbiorców oraz magazynu.

### Dodawanie magazynu lub odbiorcy

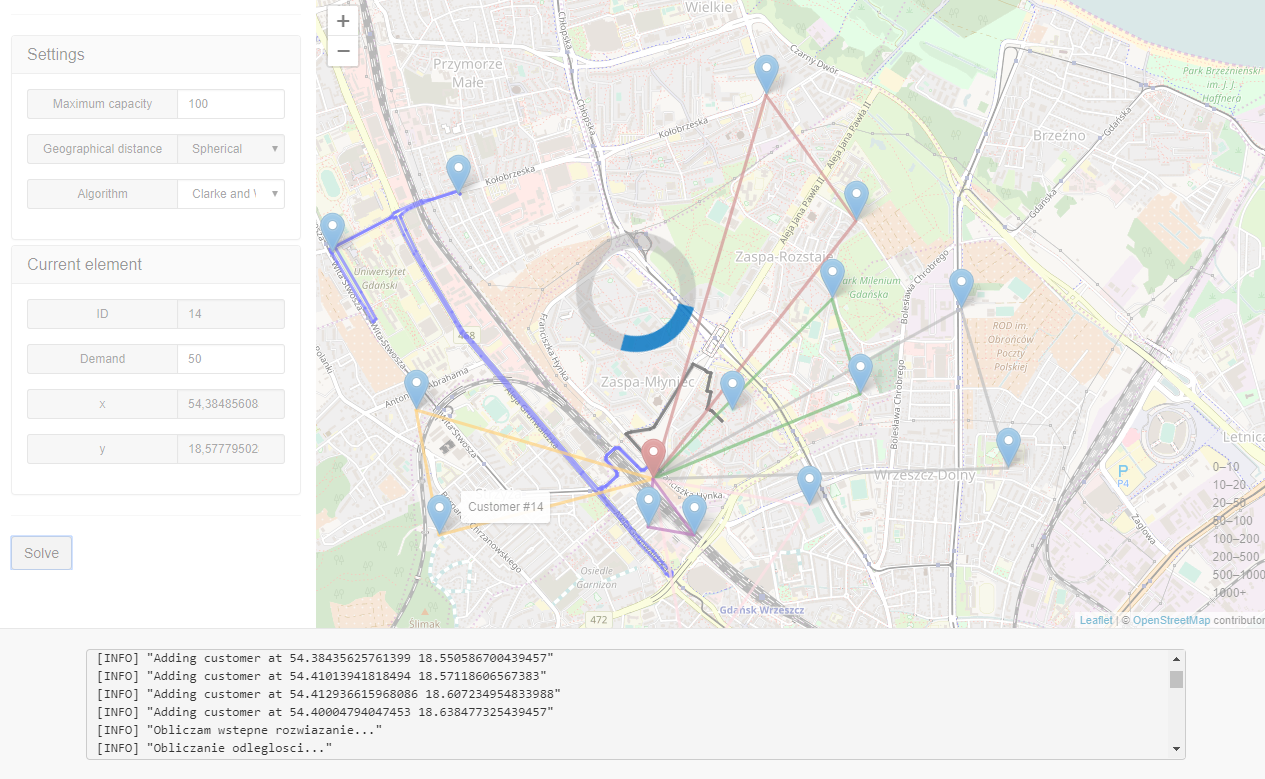
Aby dodać magazyn lub odbiorcę należy nacisnąć lewym przyciskiem myszy na mapę a następnie z menu wybrać, co chcemy dodać.



Rysunek 3 Menu pozwalające na dodawanie magazynu lub odbiorcy

### Obliczanie wyniku

Po poprawnym uzupełnieniu danych oraz wybraniu odpowiednich proces obliczania można rozpocząć poprzez naciśnięcie przycisku „*Solve*”. Jako, że proces ten może trwać długo, pokazywany jest ekran ładowania przysłaniający mapę oraz ustawienia. W tle można jednak zobaczyć etapy pośrednie dla wyliczania końcowego rozwiązania, inne w zależności od użytego algorytmu lub ustawień.



Rysunek Ładowanie rozwiązania

### Logowanie informacji

Informacje na temat rozwiązania oraz wszystkie szczegółowe informacje na temat etapów procesu zapisywane są w logach, które dostępne są w dolnej części aplikacji.

### Wykorzystane technologie

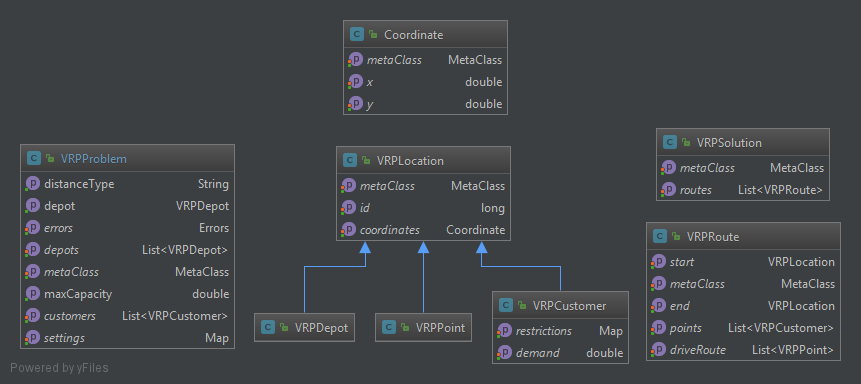
//TODO://

## Część serwerowa

Część serwerowa odpowiedzialna jest za obliczanie rozwiązania dla podanego problemu. Aplikacja została napisana w języku Groovy przy pomocy frameworka Grails w wersji 3.2.4.

### Schemat klas

//TODO:// Dokonczyc



### Przykładowe zapytanie do serwera

{

"depots":[

{

"coordinates":{

"x":54.374258435906924,

"y":18.617620468139652

},

"id":0

}

],

"customers":[

{

"coordinates":{

"x":54.37955760256642,

"y":18.58242988586426

},

"demand":50,

"id":0

}

],

"settings":{

"algorithm":"savings",

"distance":"air",

"geo\_distance":"spherical",

"capacity":200

}

}

### Proces przetwarzania żądania

Po otrzymaniu przez serwer na podstawie podanych danych tworzy obiekt klasy *VRPProblem*. Przekazywany jest on do serwisu odpowiedzialnego za obługę problemu. Ten wyszukuje wszystkie zarejestrowane

TODO://

### Obliczanie odległości pomiędzy punktami

Istnieje kilka możliwości obliczania odległości pomiędzy poszczególnymi punktami na mapie. W aplikacji zostały wybrane dwie z nich, odległość drogowa (obliczanie najkrótszej trasy samochodowej) oraz lotnicza (bezpośrednia).

W celu obliczenia odległości drogowej potrzebne są dane na temat dróg. Tutaj podobnie jak w wypadku części klienckiej wykorzystano mapy OpenStreetMap[[6]](#endnote-6). Dane na temat dróg można pobrać ze strony [http://download.geofabrik.de].

## Komunikacja

Jako, że proces obliczania rozwiązania może być czasochłonny zdecydowano się na komunikację ciągłą z serwerem, dzięki czemu możemy mieć ciągły pogląd, na jakim etapie obliczeń się znajdujemy bez potrzeby ciągłego odpytywania serwera. W tym celu wykorzystano technologię WebSocket[[7]](#endnote-7).

//TODO: Wstawić diagram http vs ws

## Dostęp z urządzeń mobilnych

Aplikacja może być też uruchamiana na urządzeniach mobilnych. Strona dostosowuje swoją zawartość do rozdzielczości ekranu. Na mniejszych rozdzielczościach ukrywane są logi aplikacji, natomiast ustawienia zwijane są to górnego menu i mogą być rozwinięte w każdym momencie. Dzięki temu cały ekran zajmuje mapa, którą możemy dowolnie manipulować.

# Walidacja rozwiązania

# Podsumowanie

Wykaz literatury

Wykaz tabel

Wykaz rysunków

1. http://www.wneiz.pl/nauka\_wneiz/frfu/77-2015/FRFU-77-349.pdf [↑](#endnote-ref-1)
2. https://andresjaquep.files.wordpress.com/2008/10/2627477-clasico-dantzig.pdf [↑](#endnote-ref-2)
3. http://www.sciencedirect.com/science/article/pii/S0166218X04002860 [↑](#endnote-ref-3)
4. http://orbit.dtu.dk/files/102382394/Recent\_exact\_algorithms\_for\_solving\_the\_vehicle\_routing\_problem\_under\_capacity\_and\_time\_window\_constraints.pdf [↑](#endnote-ref-4)
5. https://www.researchgate.net/profile/Frederic\_Semet/publication/227724716\_Classical\_heuristics\_for\_the\_vehicle\_routing\_problem/links/0046352a1584d32a82000000/Classical-heuristics-for-the-vehicle-routing-problem.pdf [↑](#endnote-ref-5)
6. openStreetMap [↑](#endnote-ref-6)
7. http://jmesnil.net/stomp-websocket/doc/ [↑](#endnote-ref-7)