Corona taxi – Specyfikacja funkcjonalna, projekt grupowy

Arkadiusz Kasprzak
Jakub Komorowski
Marcin Kowalczyk

01.12.2020.

.

Spis treści

1 Opis ogólny, założenia projektowe

Głównym celem projektu jest stworzenie programu o nazwie "Corona Taxi", mającego pomóc Służbie Ochrony Zdrowia. Będzie to możliwe, dzięki optymalizacji ruchu Karetek Pogotowia - odbioru oraz transportu Pacjenta do najbliższego (w linii prostej) Ośrodka Medycznego. W przypadku braku wolnych miejsc w Ośrodku medycznym, pacjent zostanie przetransportowany do kolejnego, najbliższego Ośrodka Medycznego (jednakże w tym przypadku, odległość do kolejnych placówek jest mierzona po drogach). Informacje o:

- położeniu Pacjentów,
- położeniu Ośrodków Medycznych,
- położeniu Obiektów - neutralnych elementów, będących częścią kraju,
- drogach - połączeniach między Ośrodkami Medycznymi,
- liczbie wszystkich łóżek w danym Ośrodku Medycznym,
- liczbie wolnych łóżek w danym Ośrodku Medycznym,

będą dostarczane do programu przez Dyspozytora Pogotowia. Ograniczenia:

- Granice kraju będą ograniczane przez najbardziej oddalone od siebie punkty - jest to najmniejszy zbiór wypukły zawierający w sobie wszystkie Ośrodki Medyczne oraz Obiekty.
- Pacjenci znajdujący się poza granicami kraju nie będą obsługiwani.

Założenia:

- Karetka Pogotowia sama dojeżdża do miejsca pobytu Pacjenta (jej ruchem zaczynamy kierować w momencie podjęcia Pacjenta).
- Karetka Pogotowia może przewozić tylko jednego pacjenta na raz.
- W miejscach przecięcia się dróg znajdują się skrzyżowania.

Program będzie interaktywny, a korzystanie z niego będzie możliwe poprzez interfejs graficzny.

2 Opis funkcjonalności

Program będzie realizował następujące funkcje:

- Weryfikacja poprawności otrzymanych danych z obu plików tzn. sprawdzenie czy:
  - dane są odpowiednich typów.

  - dany plik nie jest pusty.
  - dane nie przekraczają założonych wartości.
- Dostarczenie pacjentów do najbliższych, niezajętych szpitali.
- Zmiana prędkości odtwarzania animacji.
- Wyświetlanie informacji o ruchach pacjentów.
- Dodanie pacjenta z poziomu GUI.

3 Format danych i struktura folderów

3.1 Struktura folderów

Wszystkie elementy składowe programu będą znajdowały się w odpowiednich folderach. W projekcie rozróżniamy cztery główne foldery:

I. data - folder, w którym będą przechowywane dane wejściowe używane przez program.

II. documentation - folder, w którym będzie znajdowała się cała dokumentacja projektu tzn. specyfikacja funkcjonalna i specyfikacja implementacyjna.

III. src - główny folder programu, który będzie zawierał cały kod odpowiedzialny za działanie programu.

IV. test - W tym folderze będą umieszczone wszystkie testy, które zostaną przeprowadzone, aby sprawdzić poprawne funkcjonowanie programu.

3.2 Format danych

Program będzie przyjmował dwa pliki. Będą one posiadały następujące informacje:

I.
  - Lista szpitali zawierająca:
    * unikalne id (liczba całkowita nieujemna)
    * unikalna nazwa (ciąg znaków, nie dłuższy niż 40 znaków)
    * współrzędne x (liczba całkowita)
    * współrzędne y (liczba całkowita)
    * liczba wszystkich łóżek (liczba całkowita nieujemna)
    * liczba wolnych łóżek (liczba całkowita nieujemna)
  - Lista obiektów zawierająca:
    * unikalne id (liczba całkowita nieujemna)
    * unikalna nazwa (ciąg znaków, nie dłuższy niż 40 znaków)

    - współrzędne x (liczba całkowita)
    - współrzędne y (liczba całkowita)
  - Lista dróg między szpitalami zawierająca:
    - unikalne id (liczba całkowita nieujemna)
    - id pierwszego szpitala (z wcześniej wspomnianej listy szpitali)
    - id drugiego szpitala (z wcześniej wspomnianej listy szpitali)
    - odległość między tymi szpitalami

II. – Lista pacjentów zawierająca:
    - unikalne id pacjenta (liczba całkowita nieujemna)
    - współrzędne x (liczba całkowita)
    - współrzędne y (liczba całkowita)

```
# Szpitale (id | nazwa | wsp. x | wsp. y | Liczba łóżek | Liczba wolnych łóżek)
1 | Szpital Wojewódzki nr 997 | 10 | 10 | 1000 | 100
2 | Krakowski Szpital Kliniczny | 100 | 120 | 999 | 99
3 | Pierwszy Szpital im. Prezesa RP | 120 | 130 | 99 | 0
4 | Drugi Szpital im. Naczelnika RP | 10 | 140 | 70 | 1
5 | Trzeci Szpital im. Króla RP | 140 | 10 | 996 | 0

# Obiekty (id | nazwa | wsp. x | wsp. y)
1 | Pomnik Wikipedii | -1 | 50
2 | Pomnik Fryderyka Chopina | 110 | 55
3 | Pomnik Anonimowego Przechodnia | 40 | 70

# Drogi (id | id_szpitala | id_szpitala | odległość)
1 | 1 | 2 | 700
2 | 1 | 4 | 550
3 | 1 | 5 | 800
4 | 2 | 3 | 300
5 | 2 | 4 | 550
6 | 3 | 5 | 600
7 | 4 | 5 | 750
```

Rysunek 1: Wzór pierwszego pliku.

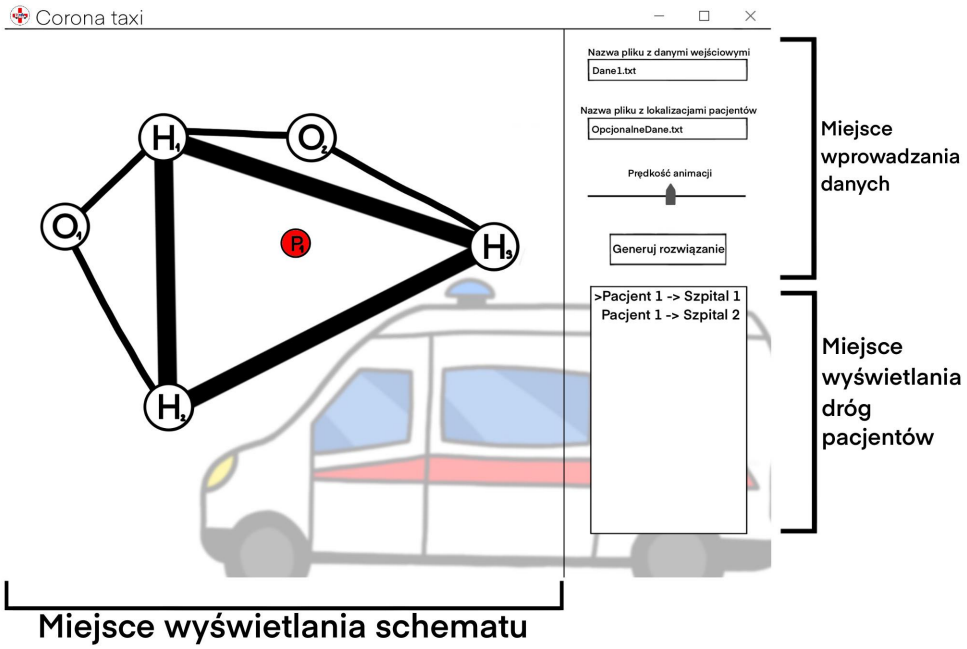
```
# Pacjenci (id | wsp. x | wsp.y)
1 | 20 | 20
2 | 99 | 105
3 | 23 | 40
```

Rysunek 2: Wzór drugiego pliku.

4 Opis graficznego interfejsu użytkownika

Graficzny interfejs użytkownika będzie znajdował się w nieskalowalnym oknie, wypełniającym cały ekran. Interfejs zostanie podzielony na dwie główne części (Patrz rys. 2): część w której wyświetli się mapa szpitali, obiektów i pacjentów, oraz część obsługową, umożliwiającą użytkownikowi: wprowadzenie nazw plików z danymi, ustawienie prędkości wyświetlania kolejnych przejazdów pacjentów, oraz sekcję wyświetlającą przejazdy kolejnych pacjentów między szpitalami.

Rys. 3. Ikona programu



Rys. 4. Wizualizacja GUI programu, z opisem poszczególnych części

5 Potencjalne komunikaty o błędach

W przypadku wprowadzenia nieprawidłowych danych wejściowych, program poinformuje o tym użytkownika poprzez następujące komunikaty:

- **Błędny format pliku!** – komunikat wyświetlany w przypadku, gdy dostarczony plik jest w nieodpowiednim formacie.
- **Błędna struktura pliku!** – komunikat wyświetlany w przypadku, gdy w pliku z danymi wejściowymi brakuje któregoś nagłówka, lub gdy liczba kresek pionowych „|" jest nieprawidłowa.
- **Niepoprawne dane!** – komunikat wyświetlany w przypadku, gdy któryś parametr będzie zawierał niepoprawne dane, np. w miejscu parametru id znajdzie się litera.
- **Podany plik wejściowy jest pusty!** - komunikat wyświetlany w przypadku, gdy dostarczony plik nie ma zawartości.
- **Dane w pliku nie mogą przekraczać następujących wartości:** - komunikat wyświetlany w przypadku, gdy dane w pliku będą zbyt duże. Po tym komunikacie zostaną wypisane maksymalne wartości poszczególnych typów danych przyjmowane przez program.

6 Testowanie

Poprawne działanie programu zostanie zweryfikowane za pomocą testów jednostkowych, napisanych we frameworku TestNG. Poza tym zostaną przeprowadzone manualne testy Graficznego Interfejsu Użytkownika.