Punktualność komunikacji miejskiej w Warszawie

Patryk Karp, Mateusz Roman, Hanna Moczek, Karolina Winczewska
11.07.2024 r.

1 Wprowadzenie

Celem niniejszego raportu jest analiza punktualności komunikacji miejskiej na przykładzie pojazdów poruszających się po warszawskich mostach. Badanie koncentruje się na ocenie rozkładu opóźnień tramwajów i autobusów względem pory dnia, tygodnia, linii oraz konkretnego mostu. Analiza ta ma na celu dostarczenie informacji na temat punktualności komunikacji publicznej w różnych warunkach czasowych i przestrzennych, co może być pomocne w zarządzaniu transportem oraz planowaniu podróży mieszkańców Warszawy.

1.1 Zakres analizy

1. Wybór przystanków
2. Pobranie danych z API ZTM
3. Analiza rozkładu opóźnień

2 Wybór przystanków

Zdecydowano się na porównanie opóźnień na Warszawskich mostach: Grota - Roweckiego, Gdańskim, Śląsko - Dąbrowskim, Świętokrzyskim, Poniatowskim, Łazienkowskim i Siekierkowskim. W tym celu wybrano po jednej linii autobusowej i tramwajowej (jeśli było to możliwe). Dla każdej linii wybrano po 7 przystanków, za przystanek "środkowy" obrano przystanek na moście.

- most Grota - Roweckiego : linia 114
- most Gdański: linie 500 i 6
- most śląsko - Dąbrowski: linie 190 i 26
- most Świętokrzyski: linia 162
- most Poniatowskiego : linie 521 i 9
- most Łazienkowski: linia 523
- most Siekierkowski: linia 148

3 Zbieranie danych - Patryk Karp i Mateusz Roman

Wykonane z pomocą pythonowego pakietu **warsaw_data_api**, korzystającego z otwartych danych udostępnianych przez miasto Warszawa na stronie internetowej api.um.warszawa.pl.

3.1 Przyjazdy

Funkcja zbierająca czasy przyjazdów znajduje się w pliku `collectArrivals.py`.

3.1.1 Przystanki

Przystanki zapisujemy w słowniku o następującej strukturze:
`{"LINIA": {"KIERUNEK": [(id_przystanku, nr_przystanku), ... ]}}`

Dane o przystankach (m.in. ich lokalizacja) są pobierane z API za pomocą funkcji z pliku `busStops.py`.

3.1.2 Parametry programu

- **dt** = 10 - krok czasowy, co ile są zbierane dane (w sekundach)
- **limit_odl** = 0.005 (ok. 0.5km) - promień okręgu (w stopniach), od kiedy łapany jest pojazd
- **limit_prob** = 6x15 (15 minut) - liczba prób na znalezienie nowego minimum/wjechanie do nowego okręgu
- **limit_prob_gr** = 6x2 (2 minuty) - to samo dla ostatnich przystanków na trasie

3.1.3 Główna pętla programu

1. Tworzymy słownik **'vehicles'**, w którym przechowujemy zlokalizowane pojazdy, podzielone na linie i brygady.
   `{'517': {'3': veh, '7':veh}, ...}`
2. Pobieramy lokalizację autobusów i tramwajów z api i zapisujemy do stworzonego słownika **'vehicles'**.
3. Obliczamy odległość od wszystkich pojazdów z wybranych przez nas linii (z podziałem na te linie) do odpowiadających im przystanków na tych liniach.

4. Aktualizujemy odległości w słowniku **'tracked_veh'**, który ma następującą postać:
`{"LINIA": {"BRYGADA": {"KIERUNEK1": [(min_odleglosc_od_przystanku, czas_osiagniecia_minimum, pozostale_proby)], "KIERUNEK2": [...]}}}`

   (a) Odległość jest aktualizowana tylko wtedy, jeśli pojazd jest w okręgu (**'limit_odl'**) i jeśli jest to nowe minimum odległości od pojazdu do danego przystanku

   (b) Przy tej okazji ustawiamy `pozostale_proby` na **'limit_prob'** lub **'limit_prob_gr'** wyjaśnione w poprzedniej sekcji.

   (c) Zapisujemy także czas tej zmiany (`czas_osiagniecia_minimum`). Będzie to czas przyjazdu na dany przystanek.

5. Jeśli dla ostatnich przystanków na trasie, na których był dany pojazd, minął **'limit_prob'** lub **'limit_prob_gr'**, to znaczy, że zapisujemy autobus do pliku (przejechał przez wszystkie przystanki lub przejechał do pewnego przystanku i dalej przestał wysyłać sygnał GPS).

   (a) Na tym etapie musimy też wyznaczyć kierunek w jakim jechał pojazd, ponieważ dotychczas dane były zbierane dla obydwu. Robimy to porównując różnice czasów przyjazdu pomiędzy ostatnim a pierwszym przystankiem dla danego kierunku.

   (b) Znając kierunek, zapisujemy przyjazdy dla danej brygady, korzystając z danych w słowniku **tracked_veh**.

3.2 Rozkłady jazdy

Jak dobrze wiemy, rozkład jazdy w Warszawskiej komunikacji, nie jest identyczny każdego dnia. Należy, więc go pobierać najlepiej na początku każdego dnia. Zrealizowano to przy pomocy skryptu 'fetchSchedule.py'. Wewnątrz jego zostały zaimplementowane dwie funkcje:

- 'fetchScheduleByIds(przystanki)', który pozyskuje równolegle z API 'wawztm' przy pomocy ThreadPoolExecutor, rozkłady jazdy po id przystanków.

  INPUT: Przyjmuje przystanki w formie listy, gdzie argumentem jest np.: ("2098", "01", "141").

  OUTPUT: Funkcja void, która zapisuje pozyskane planowe przyjazdy pojazdów do pliku w formacie:
  przystanek;nr przystanku;linia;dzień;godzina docelowa

  LOG: Kontrolnie wypisuje czas uzyskiwania wszystkich rozkładów jazdy.

- 'fetchScheduleByNames(przystankiNAMES)', który pozyskuje równolegle z API 'wawztm' przy pomocy ThreadPoolExecutor, rozkłady jazdy po nazwie przystanków.

INPUT: Przyjmuje przystanki w formie listy, gdzie argumentem jest np.: (”Międzynarodowa”, ”01”, ”141”).

OUTPUT: Funckja void, która zapisuje pozyskane planowe przyjazdy pojazdów do pliku: ”busSchedules.csv” w formacie:
przystanek;nr przystanku;linia;dzień;godzina docelowa.

LOG: Kontrolnie wypisuje czas uzyskiwania wszystkich rozkładów jazdy.

3.3 Końcowe przetwarzanie zebranych danych

3.3.1 Naprawa błędów

Po zebraniu, dane zostają poddane wstępnej obróbce. Dokonano tego przy pomocy krótkiego skryptu 'repairData.py'. W celu zneutralizowania występujących błędów niezgodności danych z rozkładem jazdy wykonano następujące zmiany:

- Usunięto rekordy, dla których wystąpił prefiks nazwy przystanku 'KS Polonia'.
- Zmieniono również numer dla przystanku 'pl.Na rozdrożu' w kierunku 'PKP Olszynka Grochowska' z 55 na 05, gdyż podczas procesu powstawania skryptu, zmienił się numer przystanku i należało to również uwzględnić.

3.3.2 Przetworzenie danych

Finalnym aspektem, umożliwiającym bardziej przyjemną analizę danych jest dodanie skryptu 'processing.py', który zajmuje się konwersją zebranych danych oraz danych rozkładów jazdy zapisanych w plikach o prefixach ”rj” i ”dane”. Skrypt ten przyjmuje na wejście:

- sfp - schedule file path,
- dfp - data file path.

A zwraca na wyjście dwa słowniki:

- D - Data dict,
- S - Schedule dict.

W postaci:
D = {`linia: {'Nazwa przystanku': {numer: {'data': [godzina1, godzina2, ...]}}}}`},

S = {`linia: {'Nazwa przystanku': {numer: {'data': [godzina1, godzina2, ...]}}}}`.

Przyjęta forma przechowywania danych, znacznie ułatwia późniejszą ich analizę.

4 Analiza wyników - Hanna Moczek i Karolina Winczewska

Stworzone do analizy funkcje znajdują się w pliku *Analiza_danych.py*. Za opóźnienie przyjęto najkrótszą różnicę czasu między godziną przyjazdu a godziną z rozkładu, co pozwala analizować opóźnienia z perspektywy osoby czekającej na autobus na przystanku. Takie podejście sprawia, że osoba wsiadająca poza punktem z utrudnieniami na trasie nie jest świadoma dotychczasowego opóźnienia pojazdu, ale lepiej ilustruje opóźnienie na konkretnym odcinku trasy.

4.1 Obliczenie opóźnienia

Funkcja *oblicz_opóźnienie*, od każdej danej z rozkładu odejmuje godziny przyjazdów, a następnie wybiera najmniejszą wartość bezwzględną dla tej różnicy. Na podstawie tych danych funkcja zwraca średnie opóźnienie dla konkretnych godzin z rozkładu.

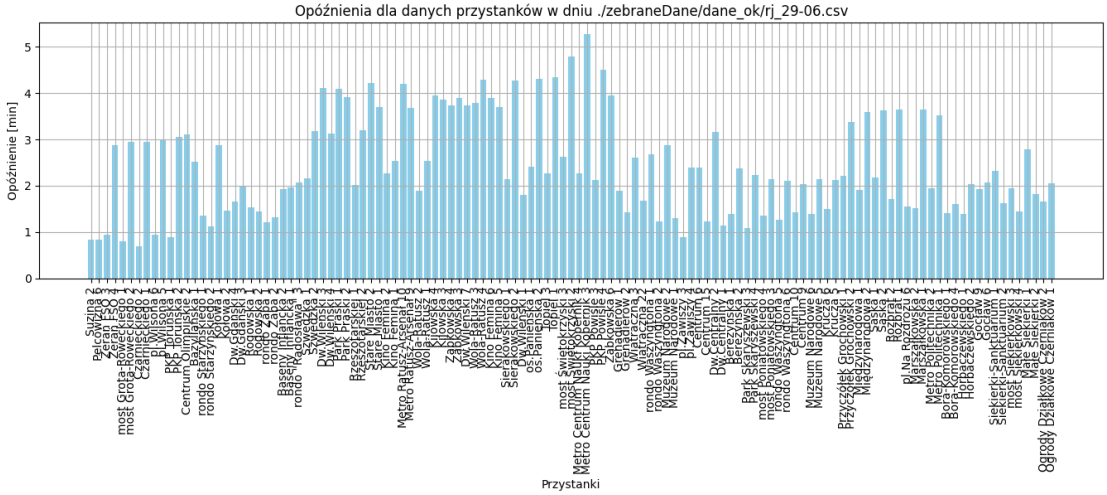
4.2 Przetwarzanie danych

W funkcji *przetwarzanie_danych* średnie opóźnienia są przypisywane do nowych słowników, które opisują zależności opóźnień od przystanku, linii, mostu, pory dnia i całego dnia.

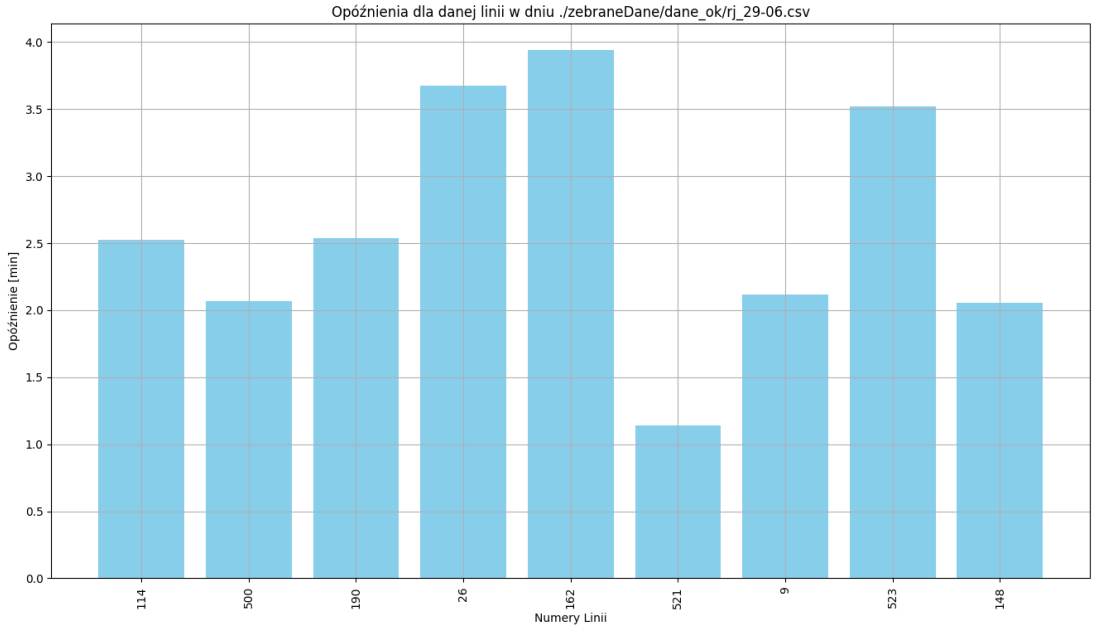
Zależność opóźnienia od pory dnia obliczano osobno (funkcja *opóźnienia_pory*). Godziny przyjazdów przyporządkowano do pór dnia, a następnie do odpowiedniego słownika.

W ostatnim kroku stworzono wykresy przedstawiające analizowane dane. Na osi x umieszczono klucze słowników, a na osi y wartości opóźnień odpowiadające tym kluczom.

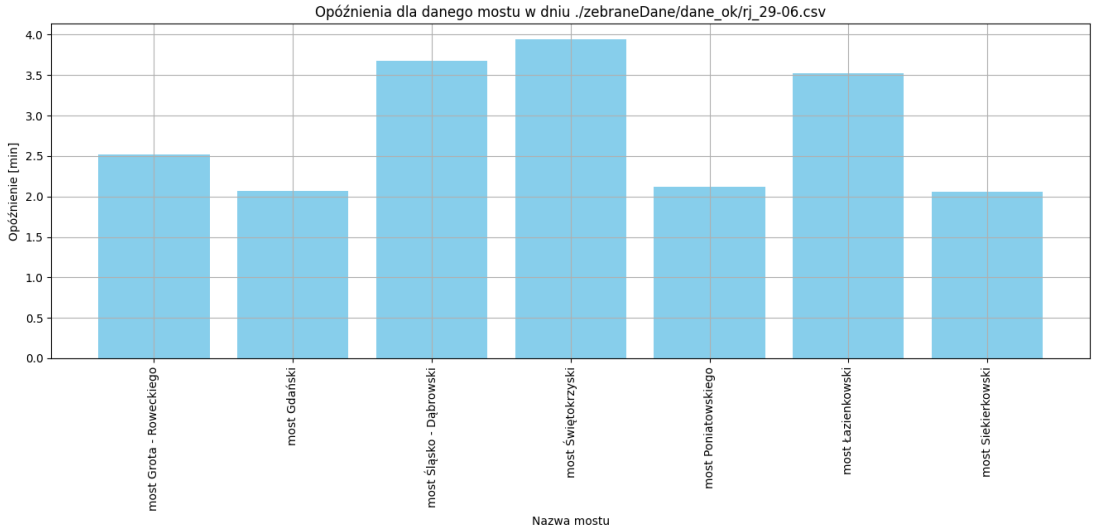
4.3 29 czerwca



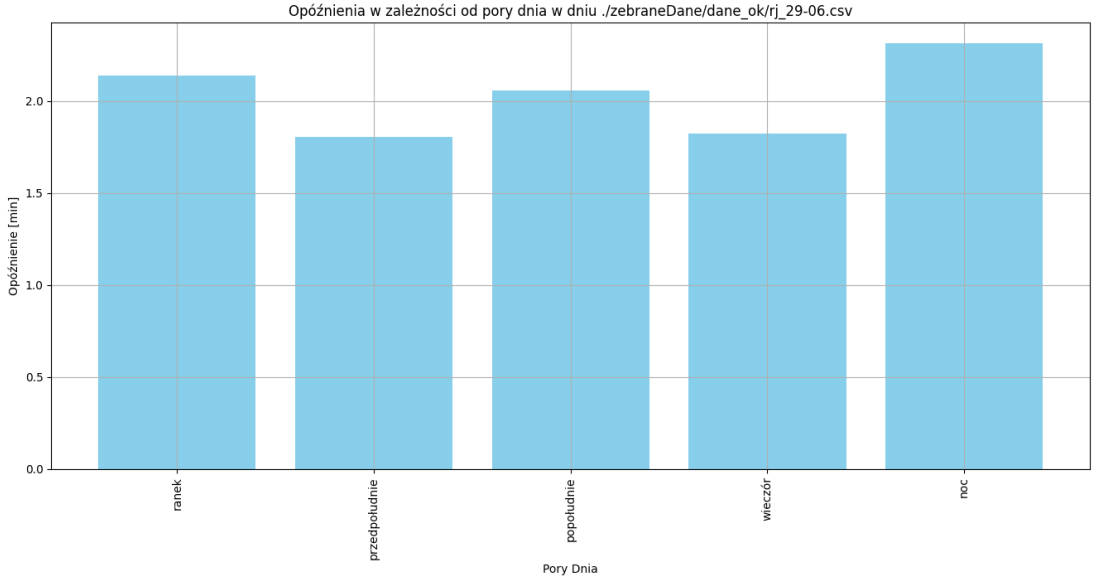
Rysunek 1. Wykres przedstawiający opóźnienia dla danych przystanków w dniu 29 czerwca 2024r.



Rysunek 1. Wykres przedstawiający opóźnienia na danej linii w dniu 29 czerwca 2024r.



Rysunek 1. Wykres przedstawiający opóźnienia dla danego mostu w dniu 29 czerwca 2024r.

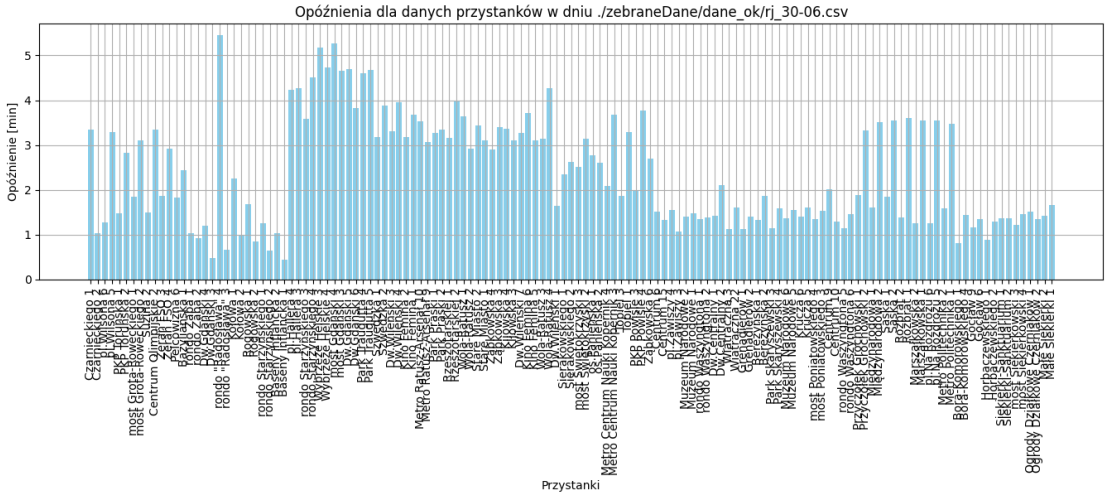


Rysunek 1. Wykres przedstawiający opóźnienia dla danej pory dnia w dniu 29 czerwca 2024r.

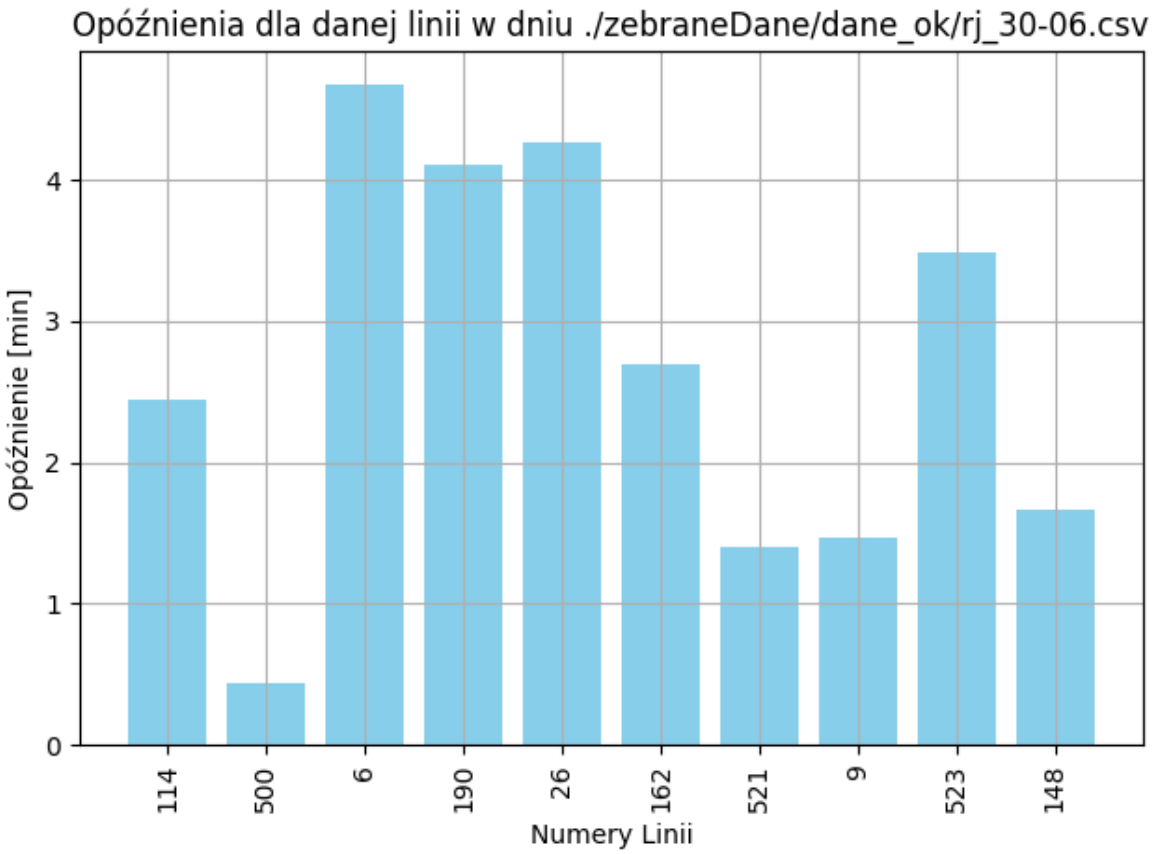
W sobotę, 29 czerwca, największe opóźnienia zarejestrowano na przystanku Me-

tro Centrum Nauki Kopernik 03. Najbardziej opóźniona była linia 162, która przekracza Wisłę mostem Świętokrzyskim. Najszybciej rzekę w Warszawie można było przekroczyć mostem Gdańskim lub Siekierkowskim. Najpunktualniejsza była linia 521. Średnia opóźnień dla most Poniatowskiego została zawyżona przez kursujący na niej tramwaj linii 9. Największe opóźnienia odnotowano w nocy (po godzinie 20), co może być związane z wyjazdem mieszkańców stolicy na wakacje i utrudnieniami w ruchu na mostach.

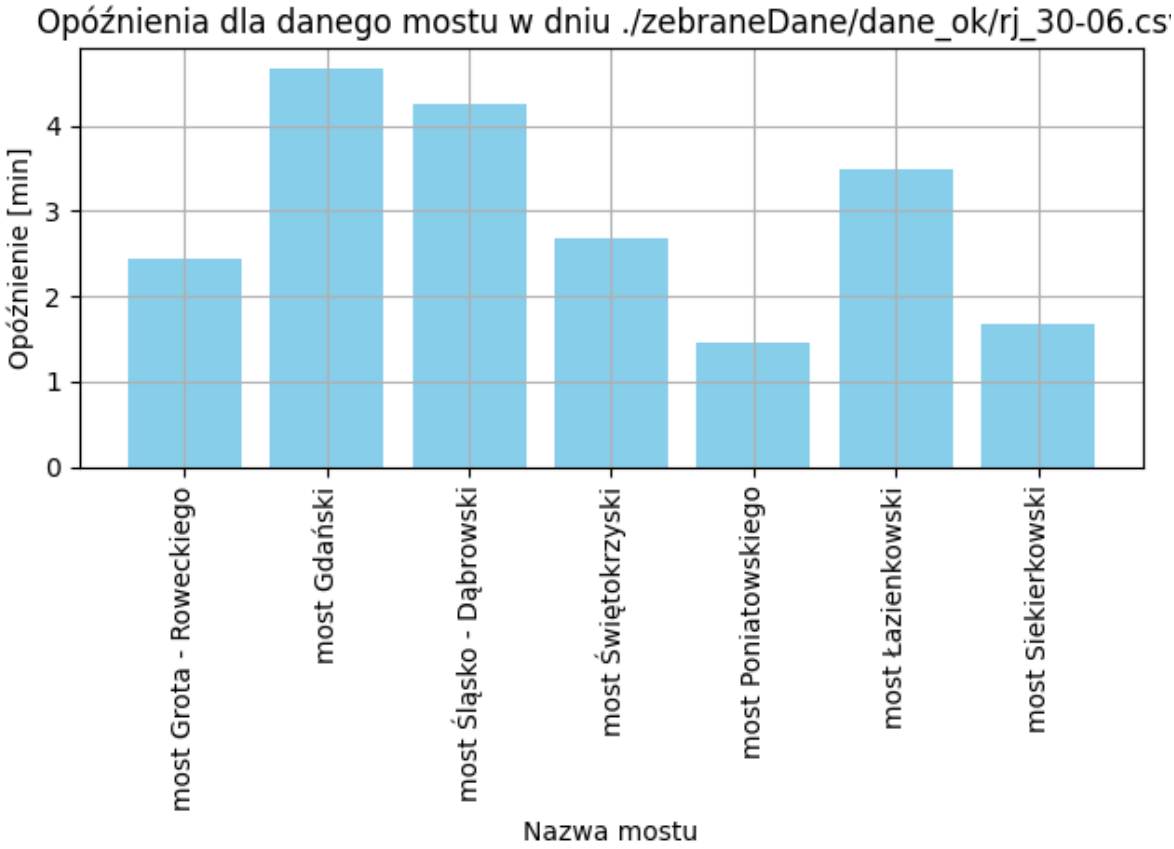
4.4 30 czerwca



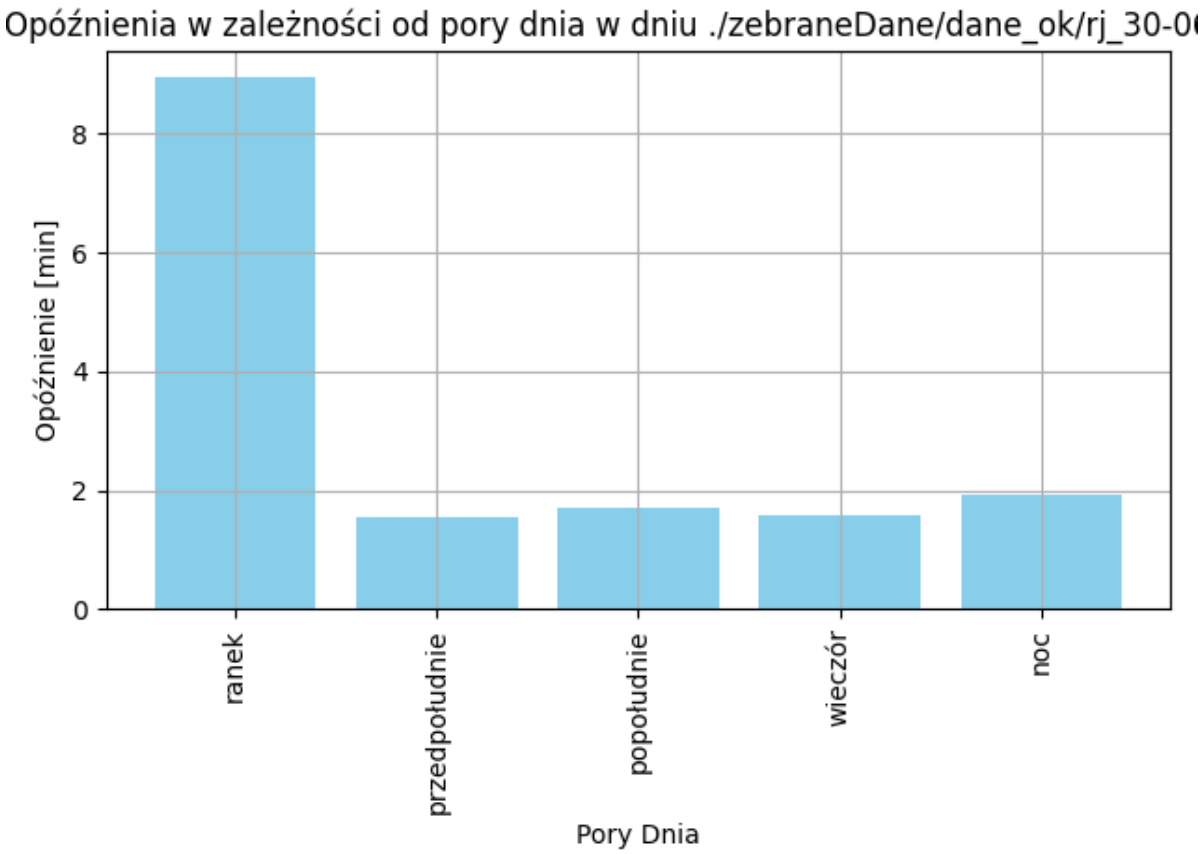
Rysunek 1. Wykres przedstawiający opóźnienia dla danego przystanku w dniu 30 czerwca 2024r.



Rysunek 1. Wykres przedstawiający opóźnienia na danej linii w dniu 30 czerwca 2024r.



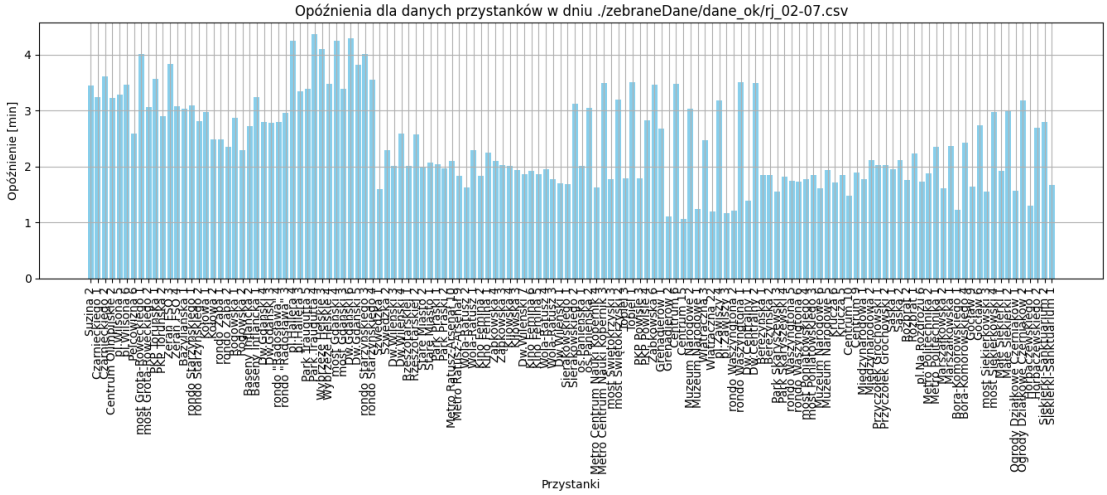
Rysunek 1. Wykres przedstawiający opóźnienia dla danego mostu w dniu 30 czerwca 2024r.



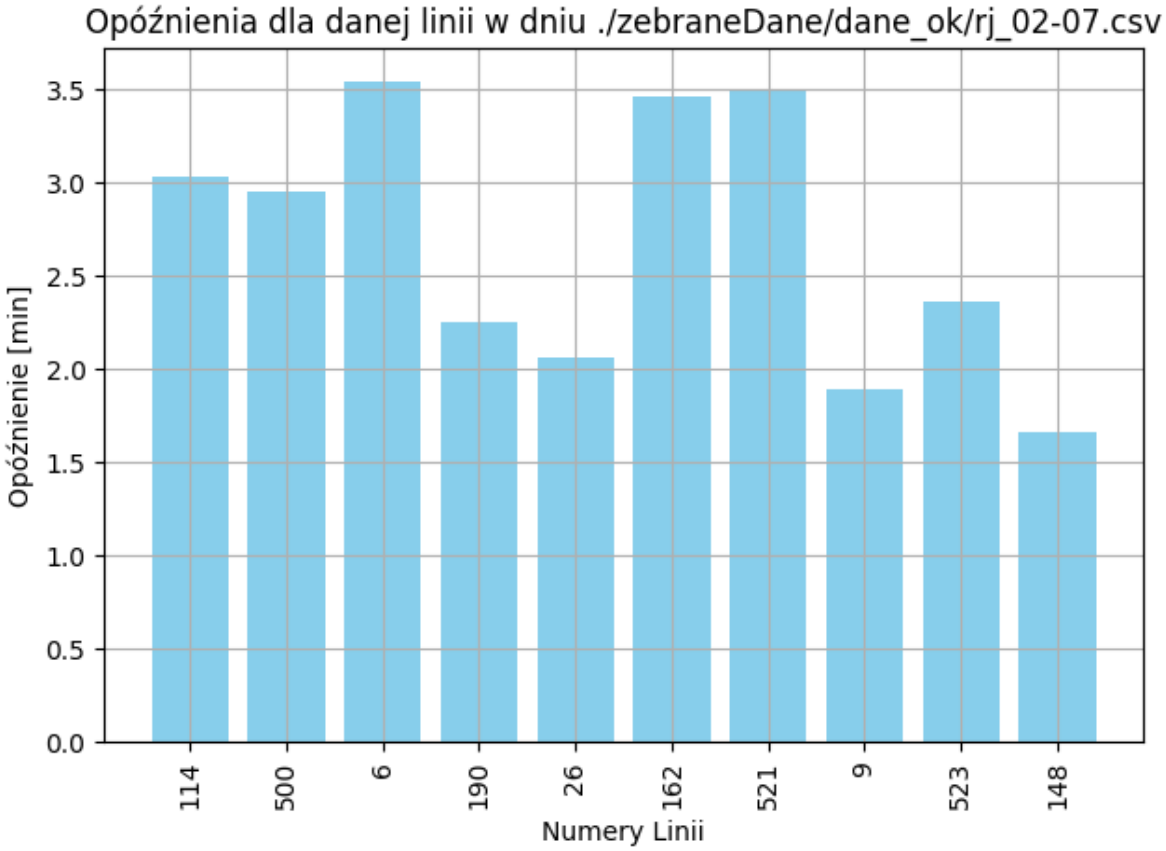
Rysunek 1. Wykres przedstawiający opóźnienia dla danej pory dnia w dniu 30 lipca 2024r.

W niedzielę, 30 czerwca, największe opóźnienia odnotowano na przystanku Rondo Radosława 04. Najbardziej opóźnioną linią była 6, kursująca po moście Gdańskim. Najszybciej rzekę można było przekroczyć mostem Poniatowskiego. Najbardziej punktualną linią była 500. Średnia opóźnień dla most Gdańskiego została zawyżona przez kursujący na niej tramwaj linii 6. Największe opóźnienia występowały rano (do godziny 9), co może być związane z wyjazdami mieszkańców stolicy na wakacje.

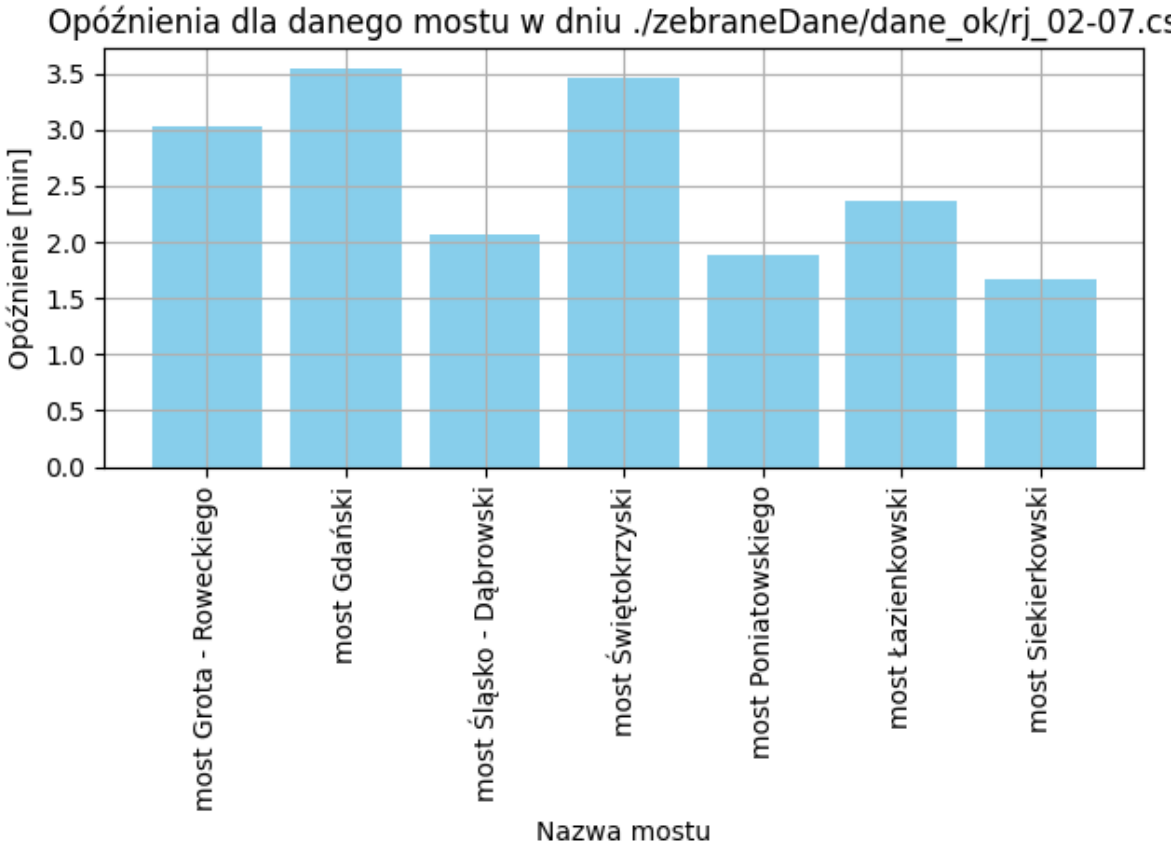
4.5 2 lipca



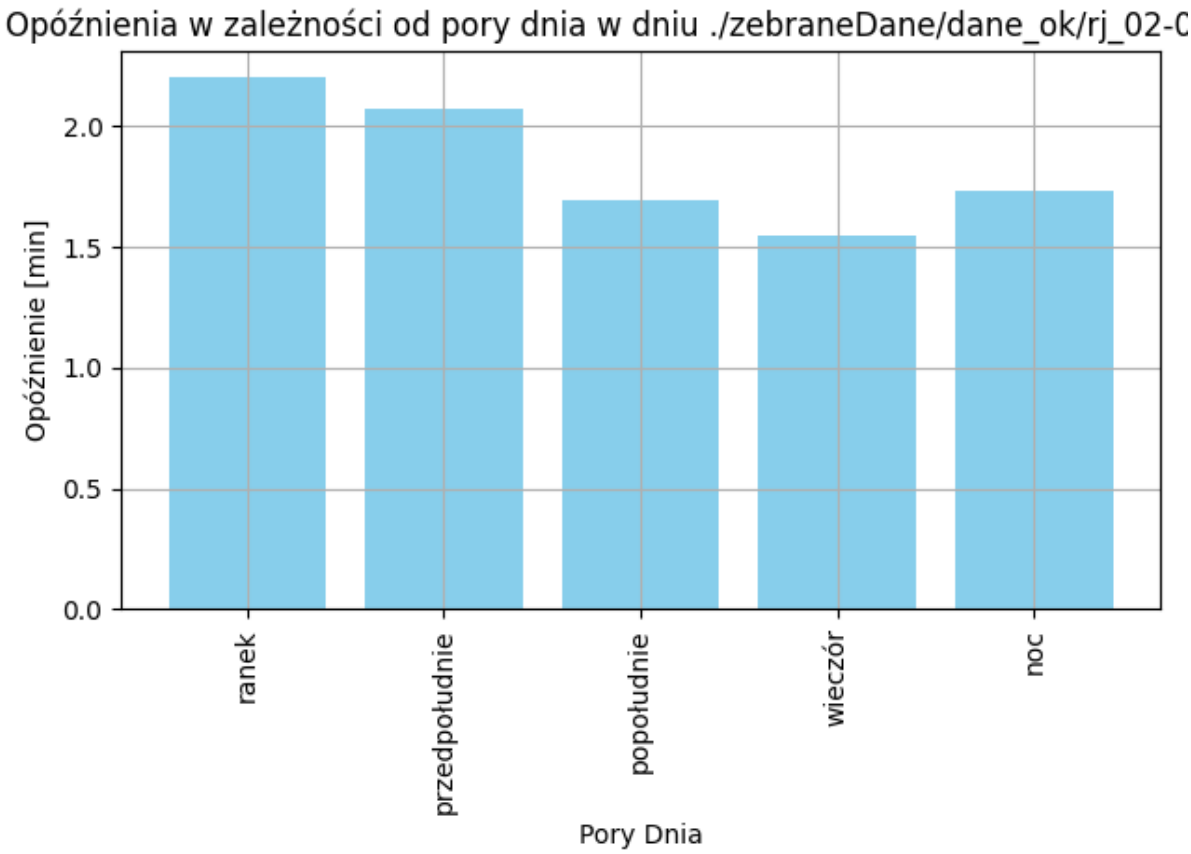
Rysunek 1. Wykres przedstawiający opóźnienia dla danego przystanku w dniu 2 lipca 2024r.



Rysunek 1. Wykres przedstawiający opóźnienia na danej linii w dniu 2 lipca 2024r.



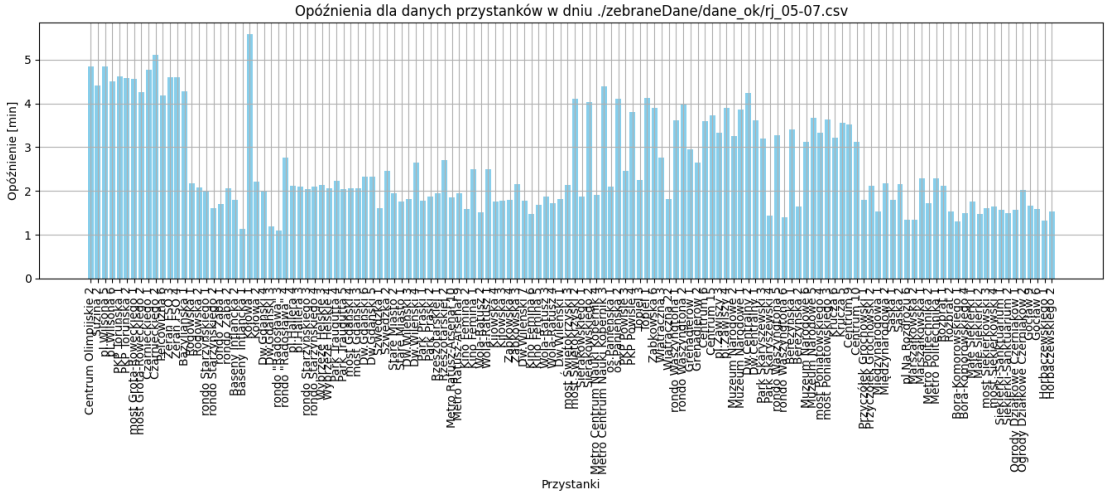
Rysunek 1. Wykres przedstawiający opóźnienia dla danego mostu w dniu 2 lipca 2024r.



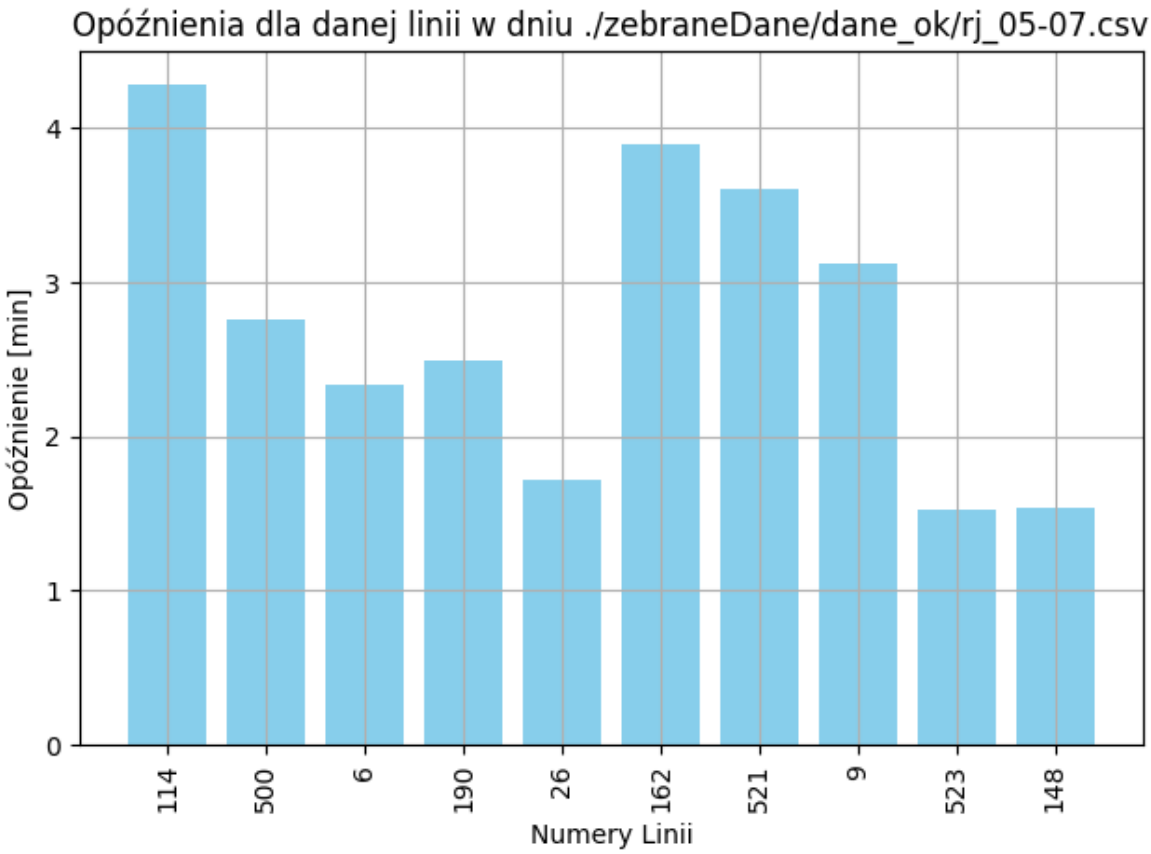
Rysunek 1. Wykres przedstawiający opóźnienia dla danej pory dnia w dniu 2 lipca 2024r.

We wtorek, 2 lipca, największe opóźnienia odnotowano na przystanku Park Traugutta 04. Najbardziej opóźnioną linią była 6, która zmienia brzeg Wisły mostem Gdańskim. Najszybciej rzekę w Warszawie można było przekroczyć mostem Siekierkowskim, dzięki czemu najpunktualniejsza była linia 148. Największe opóźnienia występowały rano (do godziny 9), co może być związane z wyjazdami do pracy.

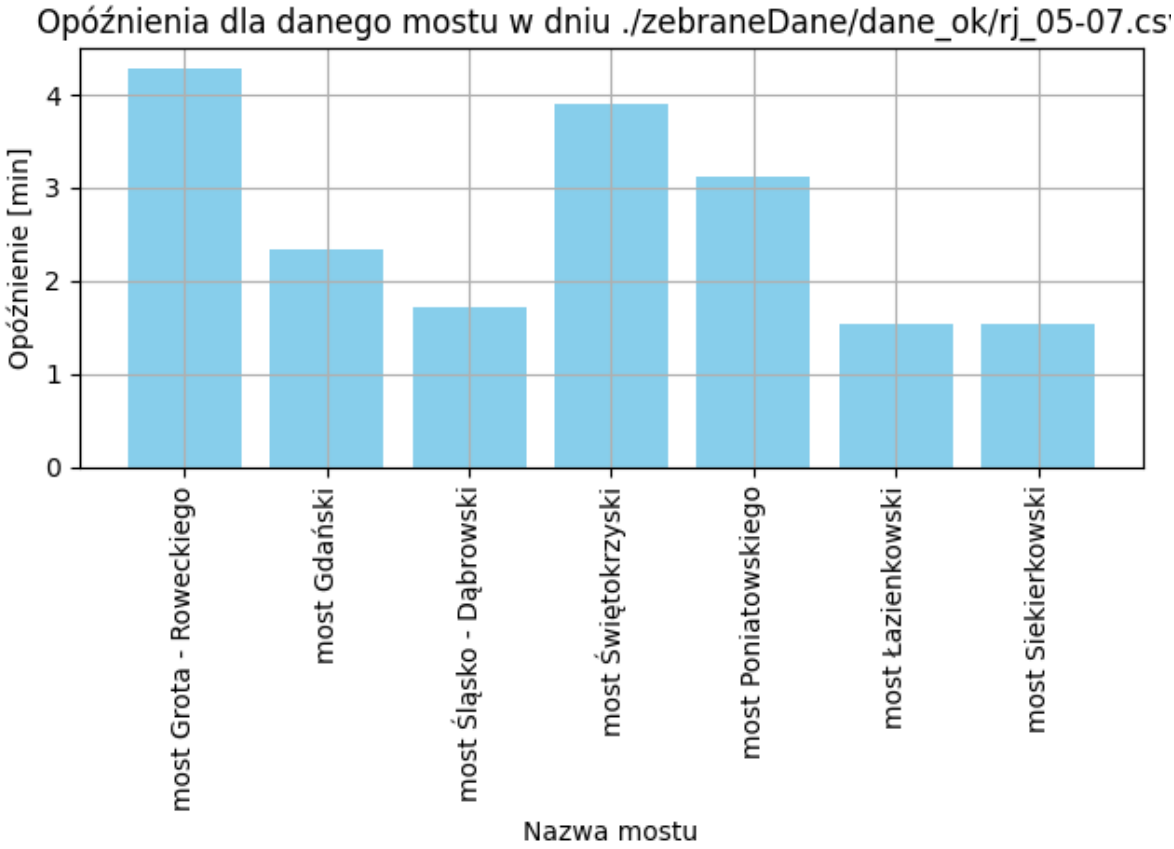
4.6 5 lipca



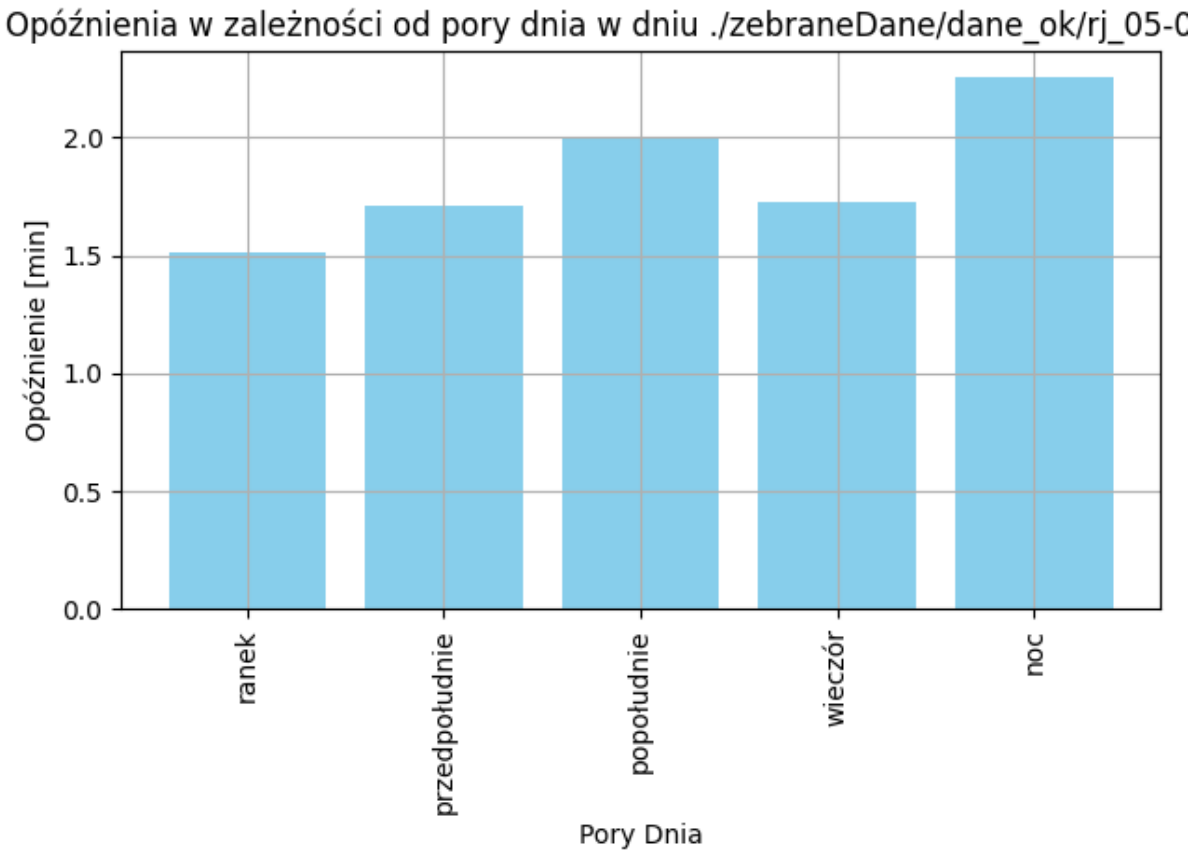
Rysunek 1. Wykres przedstawiający opóźnienia dla danego przystanku w dniu 5 lipca 2024r.



Rysunek 1. Wykres przedstawiający opóźnienia na danej linii w dniu 5 lipca 2024r.



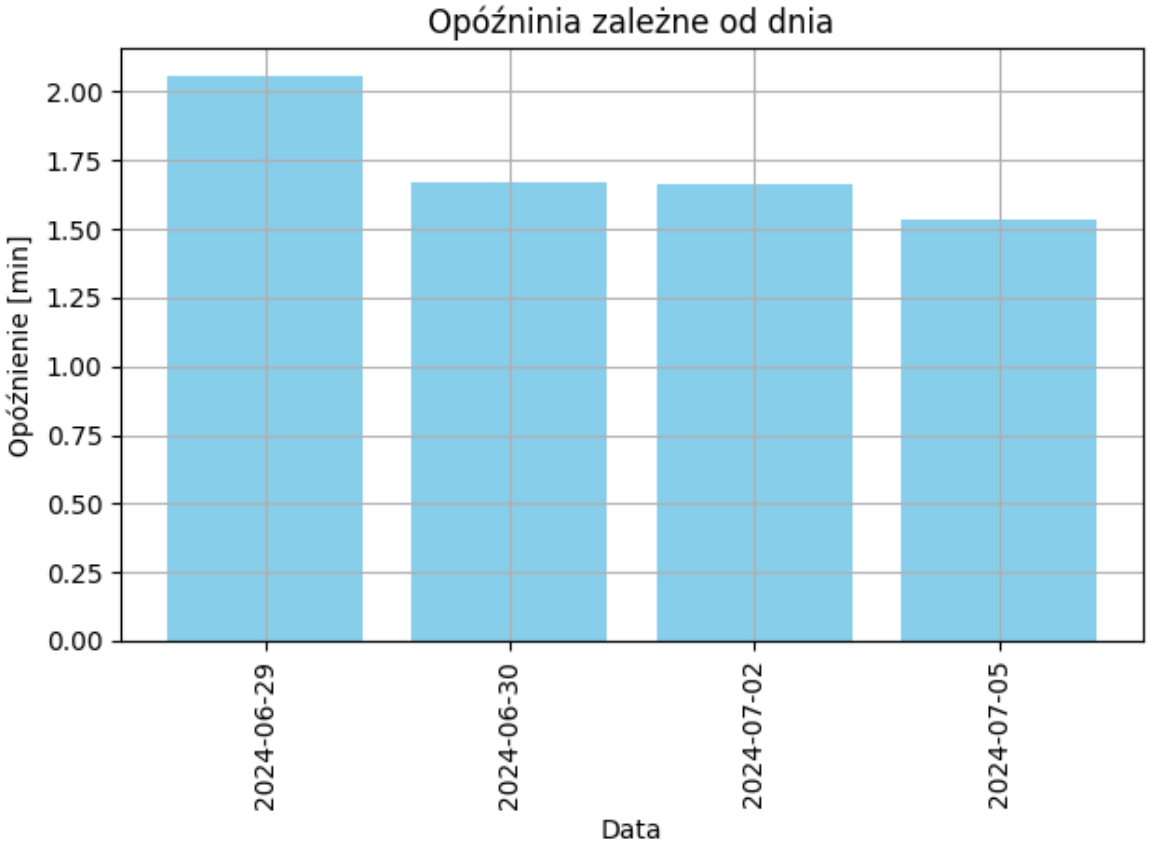
Rysunek 1. Wykres przedstawiający opóźnienia dla danego mostu w dniu 5 lipca 2024r.



Rysunek 1. Wykres przedstawiający opóźnienia dla danej pory dnia w dniu 5 lipca 2024r.

W piątek, 5 lipca, największe opóźnienia odnotowano na przystanku Kołowa 01. Najbardziej opóźnioną linią była 114, która przekracza Wisłę mostem Grota - Roweckiego. Najszybciej rzekę w Warszawie można było przekroczyć mostem Łazienkowskim lub Siekierkowskim. Najbardziej punktualne były linie 523 i 148, które kursowały odpowiednio przez most Łazienkowski i Siekierkowski. Największe opóźnienia występowały w nocy (po godzinie 20), co może być związane z wyjazdami mieszkańców stolicy na weekend oraz utrudnieniami w ruchu na mostach.

4.7 Średnia z dni



Rysunek 1. Wykres przedstawiający opóźnienia dla wszystkich analizowanych dni

Największe opóźnienia w Warszawie miały miejsce w sobotę, 29 czerwca. Mogły być one związane z rozpoczęciem wyjazdów na wakacje, które zaczęły się tydzień wcześniej. Najlepszym dniem do podróży po stolicy okazał się piątek, gdy znaczna część Warszawiaków opuszczała miasto na weekendowy odpoczynek.

5 Wnioski

Warszawa w okresie wakacyjnym jest miastem pełnym remontów. Z tego powodu przystanki zmieniają swoje numery lub czasowo znikają z sieci komunikacyjnej ZTM, co jest związane z przerwą od nauki w szkołach. Z tego powodu korzystanie z API było utrudnione. Jednak po uporządkowaniu danych, aby zgadzały się z rozkładami jazdy, można było zaobserwować interesujące zależności dotyczące opóźnień na mostach stolicy. Analiza danych wykazała, że duże znaczenie na wielkość opóźnień mają pora dnia i dzień tygodnia. Okazuje się jednak, że warszawski transport publiczny jest stosunkowo punktualny z perspektywy pasażera czekającego na

przystanku, przynajmniej w okresie letnim.