

# Projekt Systemu Informatycznego

TransTime - System dynamicznej informacji pasażerskiej i  
analizy rozkładów jazdy w czasie rzeczywistym

**Autorzy:**

Adam Czakon

Jakub Czyż

Jakub Czajka

14 stycznia 2026

# Spis treści

<b>1</b>	<b>Koncepcja systemu</b>	<b>3</b>
<b>2</b>	<b>Diagram przypadków użycia</b>	<b>4</b>
<b>3</b>	<b>Dziedzinowy diagram klas</b>	<b>5</b>
3.1	Opis kluczowych klas dziedzinowych . . . . .	5
3.2	Relacje . . . . .	6
<b>4</b>	<b>Scenariusze przypadków użycia</b>	<b>7</b>
4.1	Scenariusz 1: Sprawdzenie rzeczywistego czasu przyjazdu (ETA) . . . . .	7
4.2	Scenariusz 2: Dodanie komunikatu o utrudnieniach . . . . .	7
4.3	Scenariusz 3: Wyszukiwanie rozkładu jazdy konkretnej linii . . . . .	8
4.4	Scenariusz 4: Estymacja czasu dojazdu (Proces Systemowy) . . . . .	8
4.5	Scenariusz 5: Śledzenie pojazdu na mapie interaktywnej . . . . .	8
4.6	Scenariusz 6: Aktualizacja bazy rozkładów (Import danych) . . . . .	9
4.7	Scenariusz 7: Obsługa kursu wypadającego z rozkładu (Anulowanie kursu)	9
<b>5</b>	<b>Specyfikacja DFD (Data Flow Diagram)</b>	<b>10</b>
5.1	Diagram wstępny (Poziom 0 - Kontekstowy) . . . . .	10
5.2	Dekompozycja procesów (Poziom 1 - systemowy) . . . . .	10
5.3	Magazyny danych (Data Stores) . . . . .	12
5.4	Opis przepływów danych (Słownik danych) . . . . .	12
5.5	Dekompozycja procesu poziomu 1-ego . . . . .	12
5.6	Procesy atomowe (Specyfikacja logiczna) . . . . .	13
<b>6</b>	<b>Systemowy diagram klas</b>	<b>14</b>
6.1	Specyfikacja techniczna klas . . . . .	14
6.2	Relacje techniczne . . . . .	15
6.3	Typy pomocnicze . . . . .	16
<b>7</b>	<b>Architektura systemu i projekt interfejsu</b>	<b>17</b>
7.1	Koncepcja architektury . . . . .	17
7.2	Interfejsy komunikacyjne . . . . .	17
7.3	Mocki widoków (User Interface) . . . . .	17
7.4	Schemat blokowy architektury . . . . .	17

# Spis rysunków

1	Diagram przypadków użycia systemu TransTime. . . . .	4
2	Dziedzinowy diagram klas systemu informacji pasażerskiej. . . . .	5
3	Diagram kontekstowy (DFD Poziom 0). . . . .	10
4	Dekompozycja procesów (DFD Poziom 1). . . . .	11
5	Dekompozycja procesu 2. . . . .	13
6	Systemowy diagram klas (widok implementacyjny). . . . .	14
7	Makiety interfejsu: Tablica odjazdów (lewo) oraz Mapa Live (prawo). . . .	17
8	Schemat architektury warstwowej systemu TransTime. . . . .	17

# 1 Koncepcja systemu

Celem projektu jest stworzenie systemu informatycznego umożliwiającego publikację aktualnych rozkładów jazdy komunikacji miejskiej w Internecie. W odróżnieniu od tradycyjnych, statycznych tabel, system ten oferuje dynamiczne śledzenie czasu odjazdu na podstawie rzeczywistych danych.

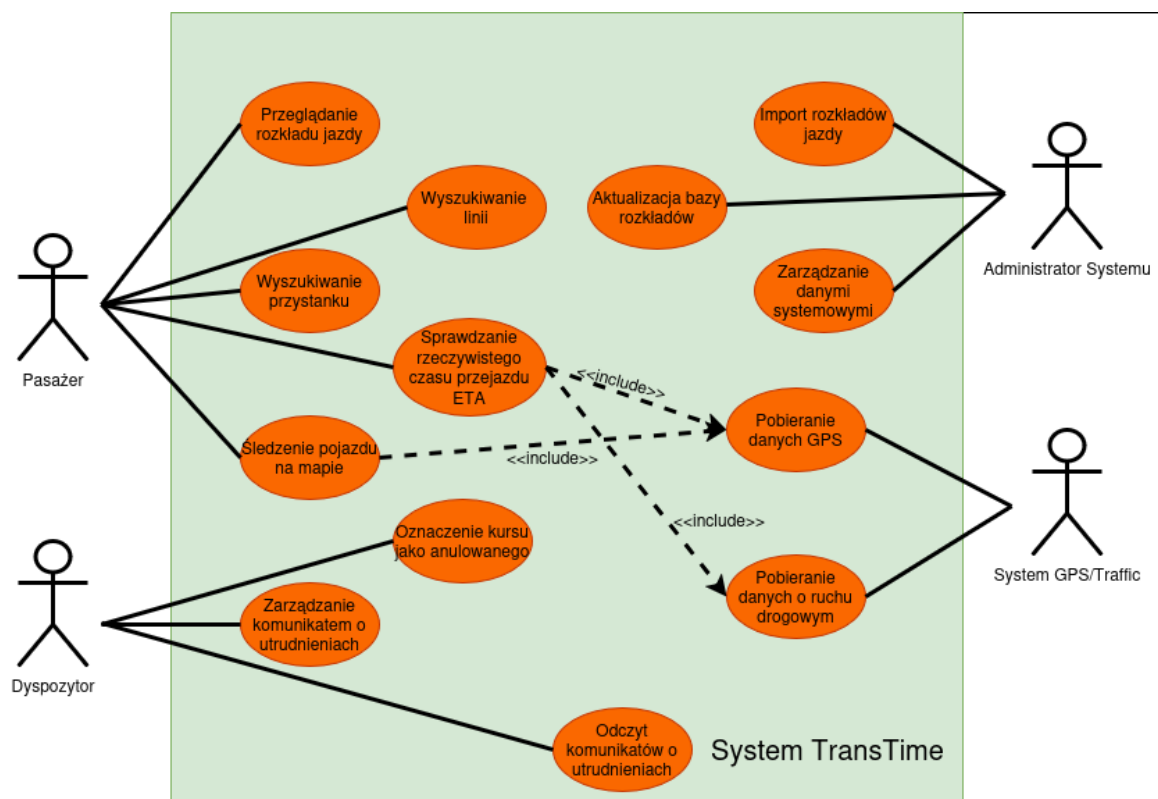
Główne założenia systemu to:

- **Publikacja rozkładów:** Udostępnienie rozkładów jazdy dla konkretnych linii oraz poszczególnych przystanków.
- **Analiza ruchu (Traffic):** Uwzględnienie aktualnego stanu przejezdności dróg w mieście.
- **Estymacja czasu (ETA):** Obliczanie faktycznego czasu dojazdu pojazdu do przystanku z uwzględnieniem opóźnień i zdarzeń losowych.
- **Informacja pasażerska:** Przekazanie użytkownikowi jasnej informacji, czy dany kurs nie wypadł z rozkładu i ile minut pozostało do jego przyjazdu.

System ma za zadanie zwiększyć komfort podróżnych poprzez dostarczenie wiarygodnych danych w czasie rzeczywistym, wzorując się na nowoczesnych systemach typu ITS (Intelligent Transportation Systems).

## 2 Diagram przypadków użycia

Poniższy diagram identyfikuje głównych aktorów systemu oraz kluczowe funkcjonalności (przypadki użycia) niezbędne do realizacji celów biznesowych systemu TransTime.



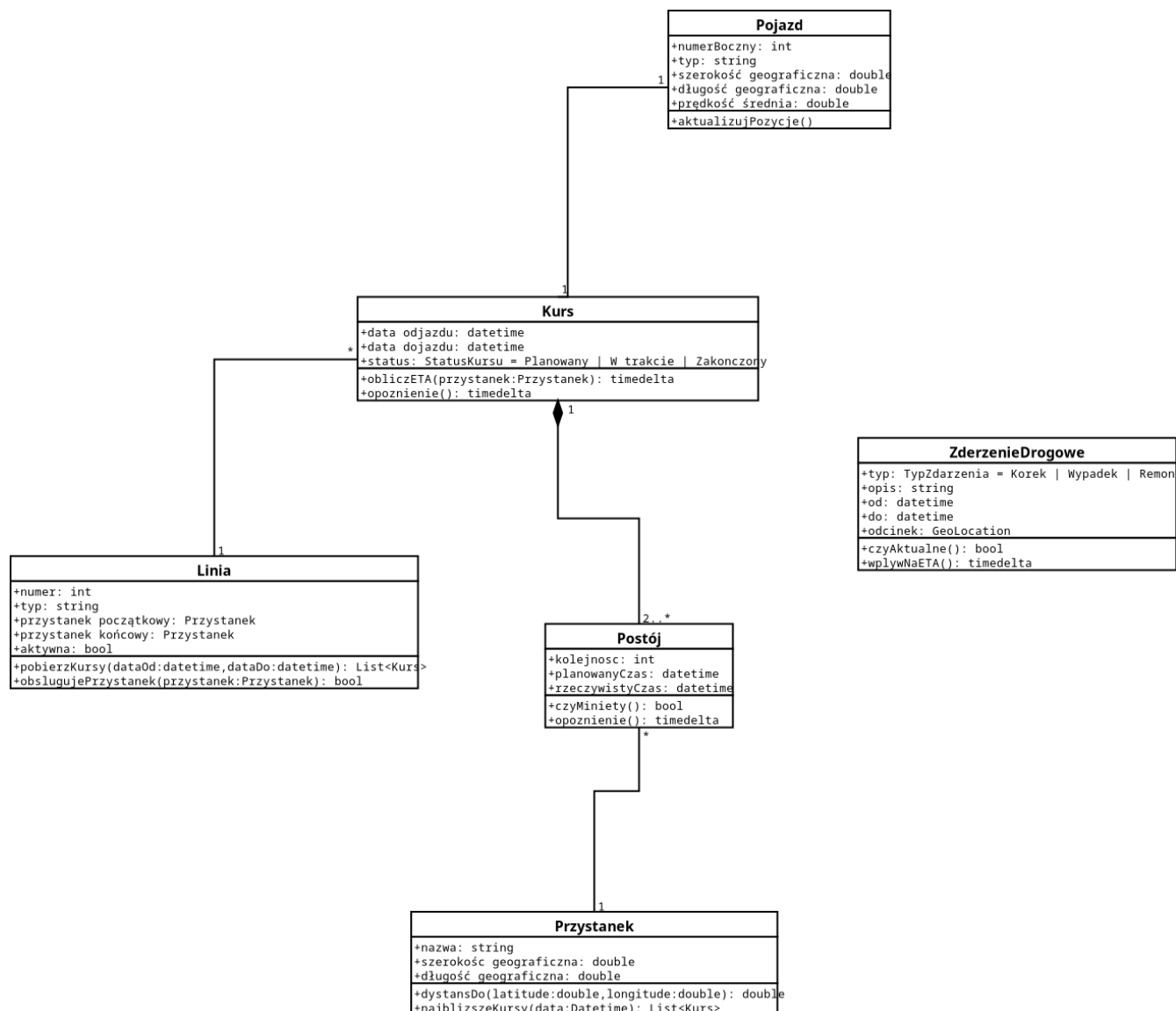
Rysunek 1: Diagram przypadków użycia systemu TransTime.

### Opis aktorów i funkcjonalności:

- **Pasażer:** Użytkownik końcowy, który wyszukuje połączenia, przegląda rozkłady linii oraz sprawdza estymowany czas przyjazdu (ETA).
- **Dyspozytor:** Pracownik MPK odpowiedzialny za wprowadzanie komunikatów o zdarzeniach losowych (np. awaria tramwaju, objazd).
- **System GPS/Traffic:** Zewnętrzne źródło danych dostarczające w trybie ciągłym informacje o współrzędnych pojazdów i natężeniu ruchu.
- **Administrator Systemu:** Odpowiedzialny za zarządzanie danymi podstawowymi i aktualizację bazy rozkładów.

### 3 Dziedzinowy diagram klas

Dziedzinowy diagram klas przedstawia kluczowe pojęcia biznesowe występujące w systemie oraz relacje zachodzące między nimi. Koncentruje się on na strukturze informacji o rozkładach jazdy i danych napływających w czasie rzeczywistym.



Rysunek 2: Dziedzinowy diagram klas systemu informacji pasażerskiej.

#### 3.1 Opis kluczowych klas dziedzinowych

- **Linia:** Reprezentuje numer linii (np. „Tramwaj 8”) oraz typ pojazdu. Grupuje kursy realizowane według rozkładu.
- **Kurs (Trip):** Konkretna realizacja linii o określonej godzinie, powiązana z pojazdem. Zawiera status i umożliwia obliczanie ETA.
- **Postój:** Reprezentuje postoje kursu na przystankach, zawiera informacje o kolejności i czasach planowanych oraz rzeczywistych.

- **Przystanek:** Punkt na mapie z nazwą i lokalizacją GPS, obsługiwany przez wiele linii i kursów.
- **Pojazd:** Fizyczna jednostka realizująca kurs, przesyłająca swoje położenie i prędkość.
- **Zdarzenie drogowe:** Informacja o utrudnieniach na drodze, mająca określony czas i obszar działania. Wpływa na obliczenia czasu przejazdu, ale nie jest trwale powiązana z kursem.

## 3.2 Relacje

Zastosowano następujące powiązania:

- Jedna **Linia** posiada wiele **Kursów**.
- Każdy **Kurs** składa się z wielu **Postojów** na **Przystankach**, ułożonych w określonej kolejności.
- **Pojazd** jest przypisany do dokładnie jednego **Kursu** w danym momencie.
- **Zdarzenia drogowe** mają przypisany obszar działania i wpływają dynamicznie na obliczanie czasu przejazdu kursów, lecz nie są z nimi trwale powiązane.

## 4 Scenariusze przypadków użycia

Poniższe tabele szczegółowo opisują przebieg interakcji użytkownika z systemem dla wybranych, kluczowych przypadków użycia.

### 4.1 Scenariusz 1: Sprawdzenie rzeczywistego czasu przyjazdu (ETA)

<b>Nazwa przypadku</b>	Sprawdzenie rzeczywistego czasu przyjazdu (Live)
<b>Aktor główny</b>	Pasażer
<b>Aktorzy wspierający</b>	System GPS, API Traffic
<b>Warunek wstępny</b>	Użytkownik znajduje się na ekranie wyboru przystanku.
<b>Przebieg główny</b>	<ol style="list-style-type: none"><li>1. Użytkownik wybiera konkretny przystanek z listy lub mapy.</li><li>2. System pobiera listę nadchodzących kursów dla tego przystanku.</li><li>3. System odczytuje aktualną pozycję pojazdów przypisanych do tych kursów.</li><li>4. System analizuje dane o natężeniu ruchu na trasie pojazdu.</li><li>5. System oblicza estymowany czas przyjazdu (ETA).</li><li>6. System wyświetla użytkownikowi odświeżaną listę odjazdów.</li></ol>
<b>Sytuacje wyjątkowe</b>	<ol style="list-style-type: none"><li>1a. Brak danych GPS z pojazdu – system wyświetla czas teoretyczny (z rozkładu) z adnotacją „brak danych live”.</li><li>2a. Kurs został anulowany przez dyspozytora – system wyświetla status „Wypadł z rozkładu”.</li></ol>

### 4.2 Scenariusz 2: Dodanie komunikatu o utrudnieniach

<b>Nazwa przypadku</b>	Zarządzanie komunikatem o utrudnieniach drogowych
<b>Aktor główny</b>	Dyspozytor
<b>Warunek wstępny</b>	Dyspozytor jest zalogowany do panelu administracyjnego.
<b>Przebieg główny</b>	<ol style="list-style-type: none"><li>1. Dyspozytor wybiera opcję „Dodaj komunikat”.</li><li>2. System wyświetla formularz wyboru linii lub obszaru miasta.</li><li>3. Dyspozytor wprowadza treść komunikatu (np. „Awaria sieci trakcyjnej”) i określa przewidywany czas trwania.</li><li>4. Dyspozytor zatwierdza komunikat.</li><li>5. System natychmiastowo publikuje informację na tablicach przystankowych online oraz w widoku mapy dla pasażerów.</li></ol>

### 4.3 Scenariusz 3: Wyszukiwanie rozkładu jazdy konkretnej linii

<b>Nazwa przypadku</b>	Przeglądanie rozkładu linii
<b>Aktor główny</b>	Pasażer
<b>Przebieg główny</b>	1. Użytkownik wprowadza numer linii w wyszukiwarce. 2. System wyświetla listę kierunków (pętli docelowych) dla tej linii. 3. Użytkownik wybiera kierunek oraz konkretny przystanek z trasy. 4. System generuje tabelaryczny rozkład jazdy (godziny i minuty) na dany dzień tygodnia.

### 4.4 Scenariusz 4: Estymacja czasu dojazdu (Proces Systemowy)

<b>Nazwa przypadku</b>	Estymacja czasu dojazdu na podstawie Traffic
<b>Aktor główny</b>	System (Proces automatyczny)
<b>Aktorzy wspierający</b>	System zewnętrzny (Google Maps API / Traffic API)
<b>Przebieg główny</b>	1. System identyfikuje aktualną pozycję pojazdu (GPS). 2. System pobiera dane o natężeniu ruchu na odcinkach drogi między pojazdem a kolejnymi przystankami. 3. System porównuje czas przejazdu teoretyczny z czasem uwzględniającym korki. 4. System aktualizuje przewidywaną godzinę przyjazdu w bazie danych czasu rzeczywistego.

### 4.5 Scenariusz 5: Śledzenie pojazdu na mapie interaktywnej

<b>Nazwa przypadku</b>	Wizualizacja pozycji pojazdu na mapie
<b>Aktor główny</b>	Pasażer
<b>Aktorzy wspierający</b>	Moduł map (np. OpenStreetMap)
<b>Przebieg główny</b>	1. Użytkownik wybiera opcję "Mapa Live". 2. System wyświetla mapę miasta z naniesionymi przystankami. 3. System pobiera pozycje wszystkich aktywnych pojazdów danej linii. 4. System nanosi ikony pojazdów na mapę, wskazując kierunek ich poruszania się. 5. Użytkownik klika w ikonę pojazdu, aby zobaczyć numer boczny i aktualne opóźnienie.



#### 4.6 Scenariusz 6: Aktualizacja bazy rozkładów (Import danych)

<b>Nazwa przypadku</b>	Import nowego rozkładu jazdy (GTFS)
<b>Aktor główny</b>	Administrator Systemu
<b>Przebieg główny</b>	<ol style="list-style-type: none"><li>1. Administrator przesyła plik z nowym harmonogramem do modułu zarządzania danymi.</li><li>2. System weryfikuje poprawność danych (spójność przystanków i linii).</li><li>3. System aktualizuje tabele rozkładów statycznych w bazie danych.</li><li>4. System generuje potwierdzenie pomyślnej aktualizacji.</li></ol>

#### 4.7 Scenariusz 7: Obsługa kursu wypadającego z rozkładu (Anulowanie kursu)

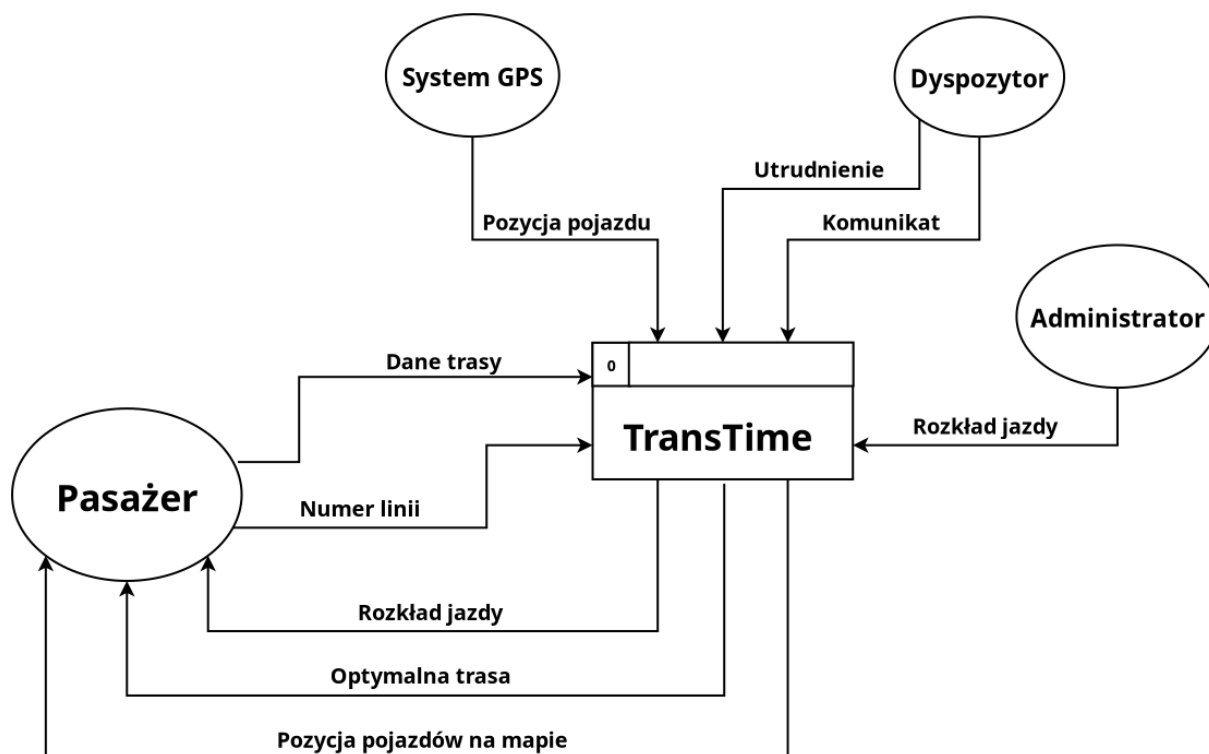
<b>Nazwa przypadku</b>	Oznaczenie kursu jako anulowanego
<b>Aktor główny</b>	Dyspozytor
<b>Aktorzy wspierający</b>	System (Proces automatyczny)
<b>Warunek wstępny</b>	Kurs jest widoczny w systemie jako aktywny, ale wystąpiły przeszkody w jego realizacji (awaria, brak kontaktu GPS).
<b>Przebieg główny</b>	<ol style="list-style-type: none"><li>1. System generuje alert o braku sygnału GPS z pojazdu przez ponad 5 minut lub o zgłoszonej awarii technicznej.</li><li>2. Dyspozytor odbiera powiadomienie w panelu sterowania.</li><li>3. Dyspozytor weryfikuje status pojazdu (np. poprzez kontakt radiowy).</li><li>4. Dyspozytor wybiera opcję „Anuluj kurs” w systemie dla danej linii i godziny.</li><li>5. System aktualizuje status kursu na „Wypadł z rozkładu”.</li><li>6. Informacja zostaje natychmiast rozesłana do modułów pasażerskich (Tablice Live, Mapa).</li></ol>
<b>Sytuacje wyjątkowe</b>	4a. Dyspozytor stwierdza, że to tylko błąd GPS – nie anuluje kursu, system nadal wyświetla czas teoretyczny z adnotacją „Brak danych live”.

## 5 Specyfikacja DFD (Data Flow Diagram)

Specyfikacja przepływu danych obrazuje sposób, w jaki informacje są pobierane, przetwarzane i składowane w systemie TransTime.

### 5.1 Diagram wstępny (Poziom 0 - Kontekstowy)

Diagram kontekstowy przedstawia system jako jedną funkcję procesową i jego interakcje z otoczeniem.



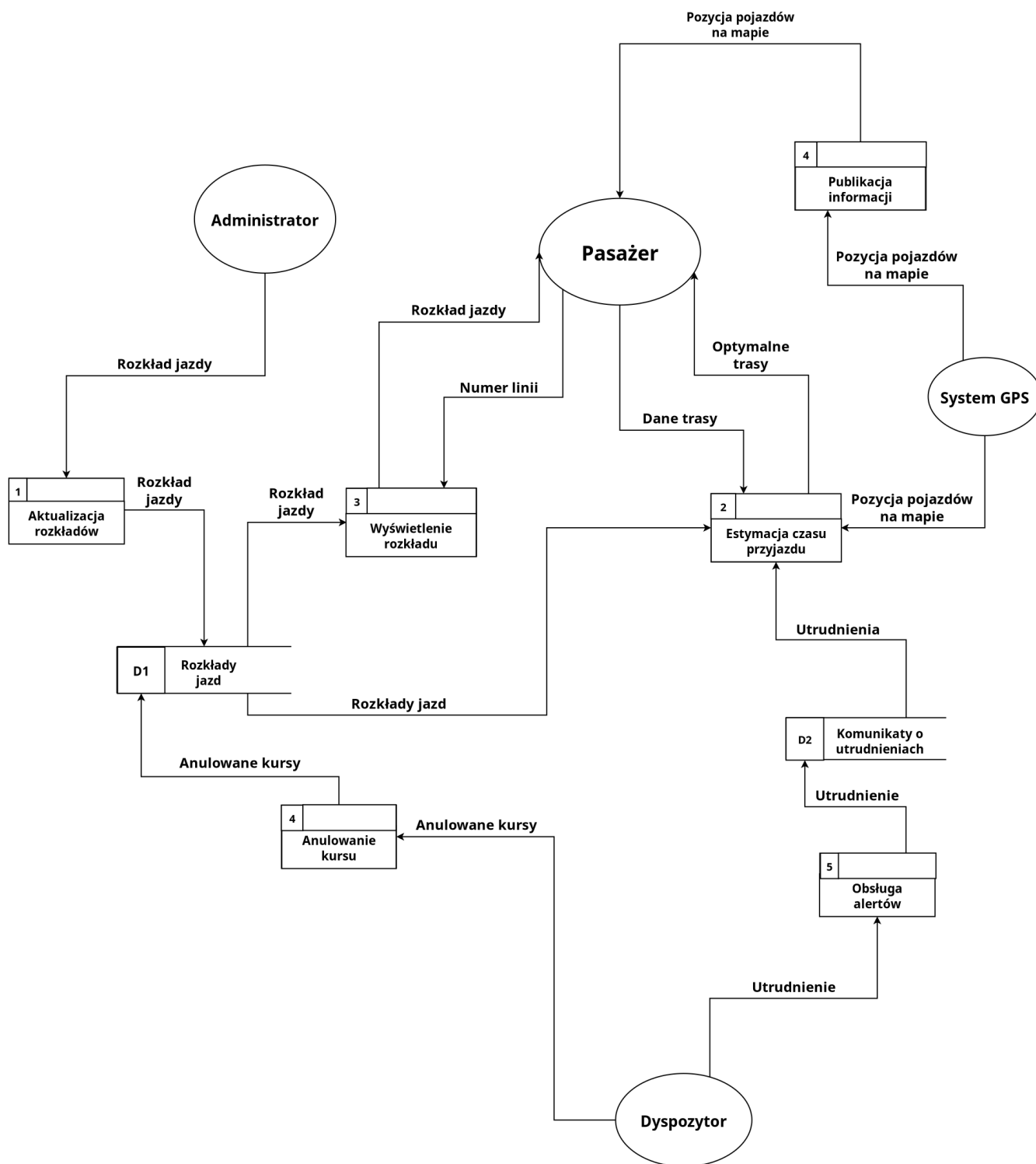
Rysunek 3: Diagram kontekstowy (DFD Poziom 0).

Główne przepływy danych na poziomie 0:

- **Pasażer:** Zapytanie o rozkład → System; Wyświetlenie ETA ← System.
- **System GPS:** Aktualna pozycja pojazdu → System.
- **Zarząd Dróg (Traffic API):** Dane o natężeniu ruchu → System.
- **Dyspozytor:** Parametry linii i komunikaty → System.

### 5.2 Dekompozycja procesów (Poziom 1 - systemowy)

Na tym poziomie system został rozbity na główne procesy funkcjonalne.



Rysunek 4: Dekompozycja procesów (DFD Poziom 1).

#### Wykaz procesów atomowych:

1. **P1. Aktualizacja rozkładów:** Przechowywanie statycznych godzin odjazdów.
2. **P2. Estymacja czasu przyjazdu (ETA):** Proces łączący dane statyczne, pozycję GPS oraz komunikaty o utrudnieniach w celu obliczenia faktycznego opóźnienia.

3. **P3. Wyświetleniu rozkładu:** Wyświetlenie informacji na temat kursów obsługiwanych przez daną linię.
4. **P4. Publikacja informacji:** Generowanie widoków dla pasażera (tablica przystankowa, mapa).
5. **P5. Obsługa alertów:** Rejestracja komunikatów o zdarzeniach drogowych.

### 5.3 Magazyny danych (Data Stores)

W systemie zidentyfikowano następujące bazy danych:

- **D1 Rozkłady jazdy:** Dane statyczne o liniach i przystankach.
- **D2 Komunikaty o utrudnieniach:** Baza aktywnych utrudnień i ogłoszeń.

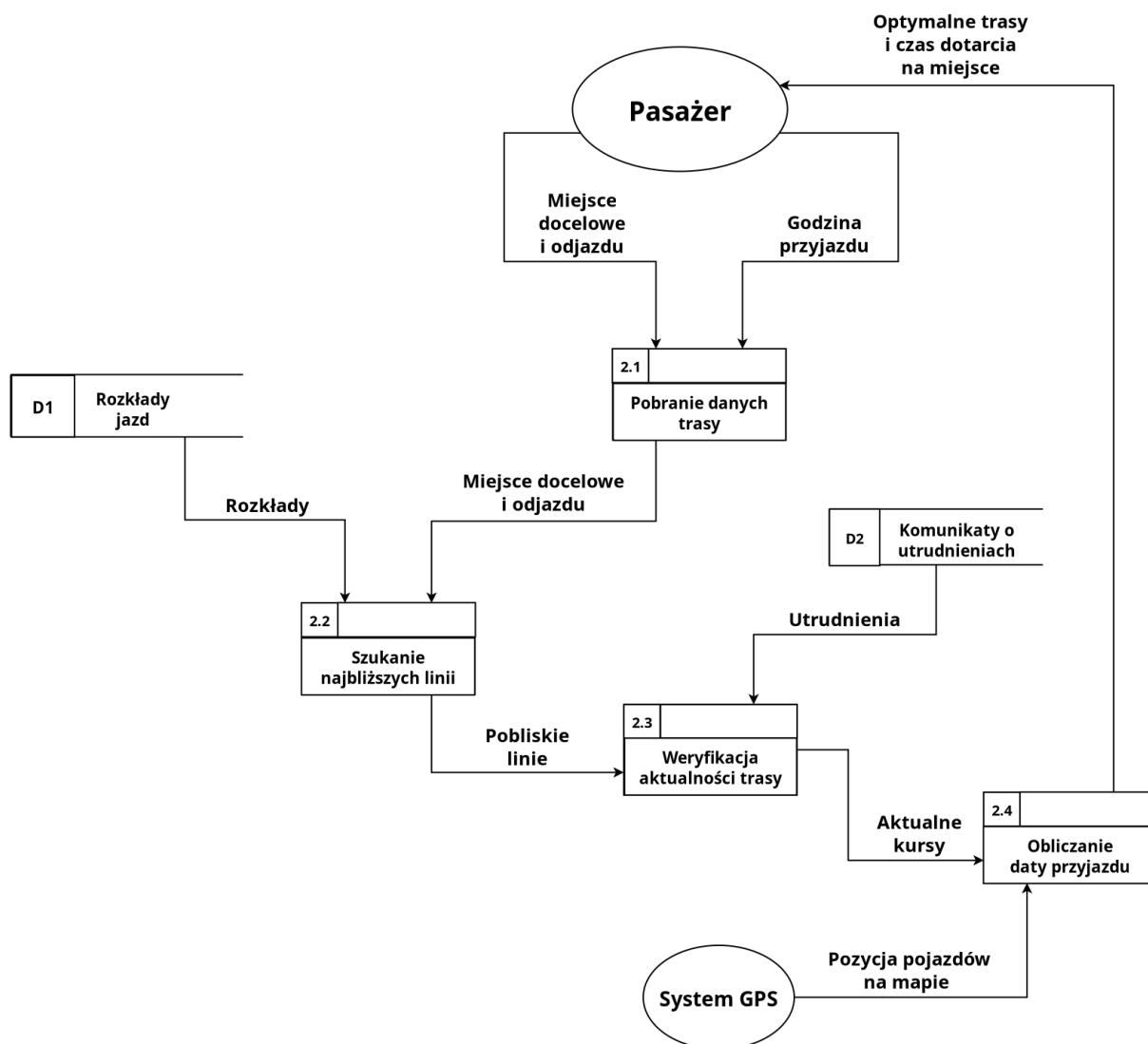
### 5.4 Opis przepływów danych (Słownik danych)

Poniższa tabela opisuje strukturę najważniejszych informacji krążących w systemie, widocznych na diagramach DFD.

Nazwa przepływu	Źródło / Cel	Zawartość (Atrybuty)
Dane GPS	Pojazd → P2	ID pojazdu, długość i szerokość geogr., prędkość, timestamp.
Dane Traffic	System GPS → P2	ID odcinka drogi, współczynnik zakorkowania (0-1), średnia prędkość.
Wynik ETA	P2 → Pasażer	ID kursu, ID przystanku, przewidywana minuta przyjazdu, status opóźnienia.
Zapytanie pasażera	Pasażer → P2	ID przystanku lub numer linii, koordynaty użytkownika.
Alert drogowy	Dyspozytor → P5	Typ zdarzenia, opis tekstowy, ID dotkniętych linii, czas wygaśnięcia.

### 5.5 Dekompozycja procesu poziomemu 1-ego

Na tym poziomie proces 2 został zdekomponowany na podprocesy.



Rysunek 5: Dekompozycja procesu 2.

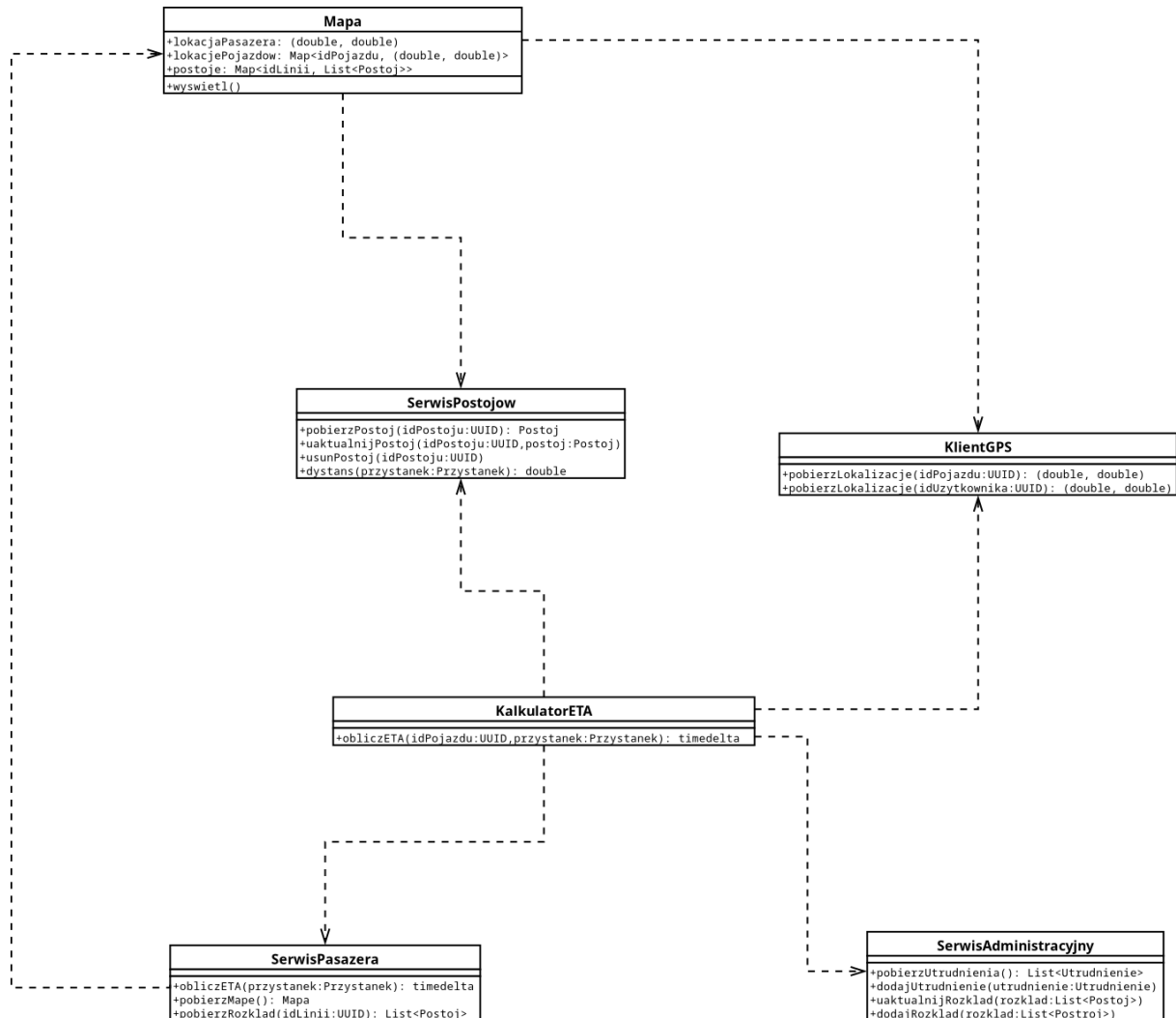
## 5.6 Procesy atomowe (Specyfikacja logiczna)

Najniższy poziom dekompozycji (procesy atomowe) realizuje następującą logikę:

- **P2.1 (Pobranie danych trasy):** Przekazuje miejsce docelowe oraz godzinę przyjazdu oczekiwaną przez pasażera do kolejnego procesu.
- **P2.2 (Szukanie najbliższych linii):** Pobiera rozkłady, które obejmują kursy z przystankami znajdującymi się w bliskiej odległości do miejsca docelowego oraz odjazdu.
- **P2.3 (Weryfikacja aktualności trasy):** Na bazie odnalezionych kursów i komunikatów o utrudnieniach przesiewa nieaktualne kursy.
- **P2.4 (Obliczanie daty przyjazdu):** Na podstawie GPS pojazdu i współrzędnych przystanku przewiduje pozostały czas do miejsca docelowego.

## 6 Systemowy diagram klas

Systemowy diagram klas przedstawia techniczną strukturę systemu, definiując typy danych, metody oraz szczegółowe powiązania między komponentami oprogramowania.



Rysunek 6: Systemowy diagram klas (widok implementacyjny).

### 6.1 Specyfikacja techniczna klas

- **TrackerPojazdow**: Klasa odpowiedzialna za śledzenie pojazdów i aktualizację ich pozycji na podstawie danych GPS.
  - Atrybuty: `- id: UUID`, `- szerokośćGeograficzna: double`, `- długośćGeograficzna: double`, `- średniaPrędkość: double`
  - Metody: `+ uaktualnijPozycję(): void`, `+ ETA(przystanek: Przystanek): TimeDelta`
- **KalkulatorETA**: Silnik obliczeniowy czasu przybycia pojazdu na dany przystanek.
  - Atrybuty: brak stanów przechowywanych na poziomie klasy (stateless)
  - Metody: `+ obliczETA(idPojazdu: UUID, przystanek: Przystanek): TimeDelta`

- Opis: Metoda wykorzystuje zewnętrzne serwisy, takie jak **SerwisPostojow** (dane o rozkładach i postojach) oraz **SerwisAdministracyjny** (utrudnienia), aby precyzyjnie obliczyć przewidywany czas przybycia.
- **SerwisPostojow**: Fasada odpowiedzialna za zarządzanie danymi o postojach pojazdów i ich rozkładach.
  - Metody: + `pobierzPostoj(idPostoju: UUID): Postoj`, + `uaktualnijPostoj(idPostoju: UUID, postoj: Postoj): void`, + `usunPostoj(idPostoju: UUID): void`, + `dystans(przystanek: Przystanek): double`
  - Opis: Zarządza danymi o postojach i pozwala na wyliczanie odległości do przystanku.
- **SerwisAdministracyjny**: Zarządza informacjami o utrudnieniach i rozkładach.
  - Metody: + `pobierzUtrudnienia(): List<Utrudnienie>`, + `dodajUtrudnienie(utrudnienie: Utrudnienie): void`, + `uaktualnijRozkład(rozkład: List<Postoj>): void`, + `dodajRozkład(rozkład: List<Postoj>): void`
  - Opis: Zapewnia dostęp do danych o utrudnieniach drogowych oraz aktualizuje rozkłady jazdy.
- **SerwisPasazera**: Udostępnia funkcjonalności dla pasażera, takie jak pobieranie map, rozkładów i obliczanie ETA.
  - Metody: + `obliczETA(przystanek: Przystanek): TimeDelta`, + `pobierzMape(): Mapa`, + `pobierzRozkład(idLinii: UUID): List<Postoj>`
- **Mapa**: Klasa reprezentująca mapę z lokalizacjami pasażerów, pojazdów oraz postojów.
  - Atrybuty: - `lokalizacjaPasazera: (double, double)`, - `lokalizacjePojazdow: Map<UUID, (double, double)>`, - `postoje: Map<UUID, List<Postoj>`
  - Metody: + `wyswietl(): void`
- **KlientGPS**: Klient komunikujący się z modułem GPS, zwracający pozycję pojazdu lub użytkownika.
  - Metody: + `pobierzLokalizację(idPojazdu: UUID): (double, double)`, + `pobierzLokalizację(idUzytkownika: UUID): (double, double)`

## 6.2 Relacje techniczne

W diagramie systemowym uwzględniono:

- **Asocjacje i zależności**:
  - **KalkulatorETA** zależy od **SerwisPostojow** i **SerwisAdministracyjny** w celu pobrania niezbędnych danych do wyliczeń.
  - **SerwisPasazera** korzysta z **KalkulatorETA** oraz **Mapa** do obsługi zapytań użytkownika.

- KlientGPS dostarcza dane lokalizacyjne dla TrackerPojazdow oraz Mapa.
- **Brak repozytoriów:** Na tym poziomie abstrakcji zdecydowano się nie modelować osobnych klas repozytoriów, gdyż serwisy pełnią funkcję warstwy dostępu do danych.

### 6.3 Typy pomocnicze

- **Typy pomocnicze:**
  - UUID – unikalny identyfikator obiektów takich jak pojazd, przystanek, postój itp.
  - TimeDelta – typ reprezentujący odstęp czasu.
  - Mapa – obiekt przechowujący aktualne lokalizacje i umożliwiający wizualizację.



## 7 Architektura systemu i projekt interfejsu

### 7.1 Koncepcja architektury

System został zaprojektowany w oparciu o **architekturę warstwową (n-tier architecture)**, co zapewnia separację logiki biznesowej od sposobu prezentacji danych.

- **Warstwa Prezentacji (Frontend):** Aplikacja webowa i mobilna komunikująca się z serwerem poprzez API. Odpowiada za renderowanie mapy i tablic odjazdów.
- **Warstwa Logiki (Backend):** Centralny serwer przetwarzający dane. Tu znajduje się silnik ETA oraz moduły integracji z GPS.
- **Warstwa Danych (Database):** Relacyjna baza danych przechowująca rozkłady statyczne oraz nierelacyjna baza typu Key-Value dla szybkich aktualizacji pozycji pojazdów.

### 7.2 Interfejsy komunikacyjne

Komunikacja między modułami odbywa się za pomocą standardu **REST API**. Kluczowe punkty styku (Endpoints):

- `GET /stops/{id}/departures` – pobiera listę odjazdów w czasie rzeczywistym.
- `POST /admin/alerts` – przesyła nowy komunikat o utrudnieniach.
- `GET /vehicles/positions` – pobiera współrzędne wszystkich pojazdów danej linii.

### 7.3 Mocki widoków (User Interface)

Poniższe makiety przedstawiają kluczowe ekrany aplikacji użytkownika końcowego.

Rysunek 7: Makiety interfejsu: Tablica odjazdów (lewo) oraz Mapa Live (prawo).

#### Opis widoków:

1. **Widok Tablicy Przystankowej:** Wyświetla numer linii, kierunek oraz czas do odjazdu. Czas rzeczywisty jest wyróżniony kolorem zielonym lub ikoną fal radiowych.
2. **Widok Mapy:** Interaktywna mapa z naniesioną pozycją użytkownika oraz ikonami autobusów/tramwajów poruszającymi się w czasie rzeczywistym.

### 7.4 Schemat blokowy architektury

Poniższy diagram (Rysunek 8) przedstawia fizyczny i logiczny podział systemu na komponenty oraz protokoły komunikacyjne użyte do ich integracji.

Rysunek 8: Schemat architektury warstwowej systemu TransTime.

#### Komponenty infrastruktury:

- **Load Balancer:** Rozdziela ruch między instancje serwera API (zapewnienie wysokiej dostępności).
- **Redis Cache:** Przechowuje najświeższe dane o pozycjach GPS, aby nie obciążać głównej bazy danych przy każdym zapytaniu pasażera.
- **Worker Services:** Niezależne procesy w tle, które co kilka sekund odpytują API Traffic i aktualizują estymacje czasowe.