System wirtualnej spedycji do trybu wieloosobowego gry ETS 2 / ATS

Opis realizacji

Rafał Zakrzewski, Artur Zienkiewicz, Paweł Wyrzykowski

Streszczenie

Aplikacja webowa imitująca system spedycji do wieloosobowego trybu gry Euro Truck Simulator, która umożliwia oddawanie raportów z tras, systemu stawek i nagród oraz inne. W głównej mierze aplikacja urozmaica rozrywkę od strony gracza o realistyczne działania w systemie spedycji.

Spis treści

1. Wstęp

1.1. Ogólna charakterystyka projektu

Projekt został głównie stworzony dla graczy którzy wiążą swoje hobby z jazdą ciężarówką przez świat. Dzięki naszej aplikacji, gracz może odczuć realizm. Aplikacja jest obsługiwana przez wszystkie współczesne przeglądarki, na każdym urządzeniu. Aplikacja oferuje szeroki wachlarz opcji i funkcjonalności, w połączeniu z serwerem backendowym, który służy za pośrednika bezpieczniej wymiany danych interfejsu użytkownika z bazą danych.

1.2. Dlaczego taki temat?

Duża część osób interesująca się spedycją, często wykorzystuje wolny czas na rozgrywkę w Euro Truck Simulator 2, a szczególnie jego tryb Multi-Player. Niestety po dłuższej rozgrywce tryb MultiPlayer staje się nudny. System wirtualnej spedycji urozmaica graczom rozrywkę wprowadzając do rozrywki odrobinę realizmu

1.3. Przegląd istniejących rozwiązań

Virtual Speditor – jeden z oficjalnych rozwiązań. Głównym celem Virtual Speditor jest tworzenie własnych tras, które gracz może później wykorzystać w świecie gry. Program również pozwala na modyfikację aktualnych zleceń wybrane przez gracza (miejsce początkowe, miejsce docelowe oraz ładunek).

2. Zakres

2.1. Opis biznesowy projektu

System wirtualnej spedycji opiera się w głównej mierze na wykorzystaniu współczesnych technologii wykorzystywanych przy tworzeniu aplikacji internetowych. Ze względu na obecny trend wykorzystywany w aplikacjach internetowych został zastosowany skromny, intuicyjny interfejs graficzny dla oka użytkownika. Jednym z priorytetów, który został postawiony na początku aplikacji, był to błyskawiczny czas realizacji zapytania i otrzymania odpowiedzi. Głównym celem aplikacji było wprowadzenie immersji do świata fikcyjnego, w którym osoby grające w ETS2/ATS mogą poczuć odrobinę realizmu.

2.2. Analiza funkcjonalna

Dostęp do aplikacji będzie dostępny za pomocą przeglądarki. Funkcje dostępne na aplikacji, będą odpowiadały szczególnym funkcjom w grze.

WF.01	Użytkownik jest rejestrowany przez właściciela lub zastępcę spedycji.
WF.02	Użytkownik może awansować o nowe stanowiska.
WF.03	Użytkownik może wzbogacać swoje doświadczenie, poprzez zdobywanie nowych umiejętności.
WF.04	Użytkownik prowadzi system rozliczania się, oraz obszerny opis trasy, tak jak w prawdziwym życiu.

Tabela 1 Wymagania funkcjonalne

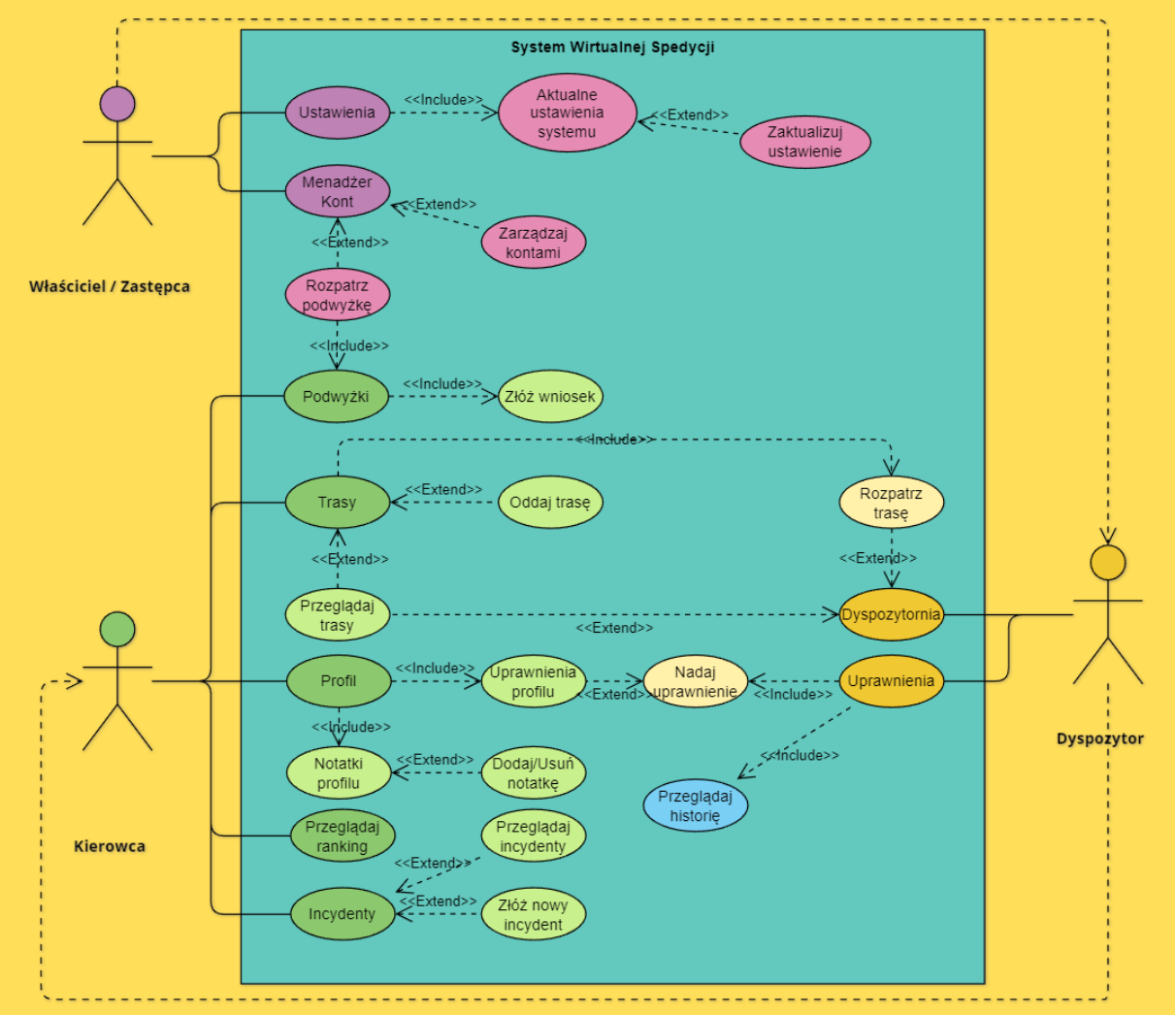
2.3. Analiza niefunkcjonalna

WNF.01	Aplikacja posiada interfejs graficzny, który jest prosty w obsłudze
WNF.02	Odpowiedź z serwera bazodanowego będzie wynosiła nie więcej niż 1 sekundę.

WNF.03	Ranking generowany jest automatycznie poprzez zapytania do bazy danych.

Tabela 2 Wymagania niefunkcjonalne

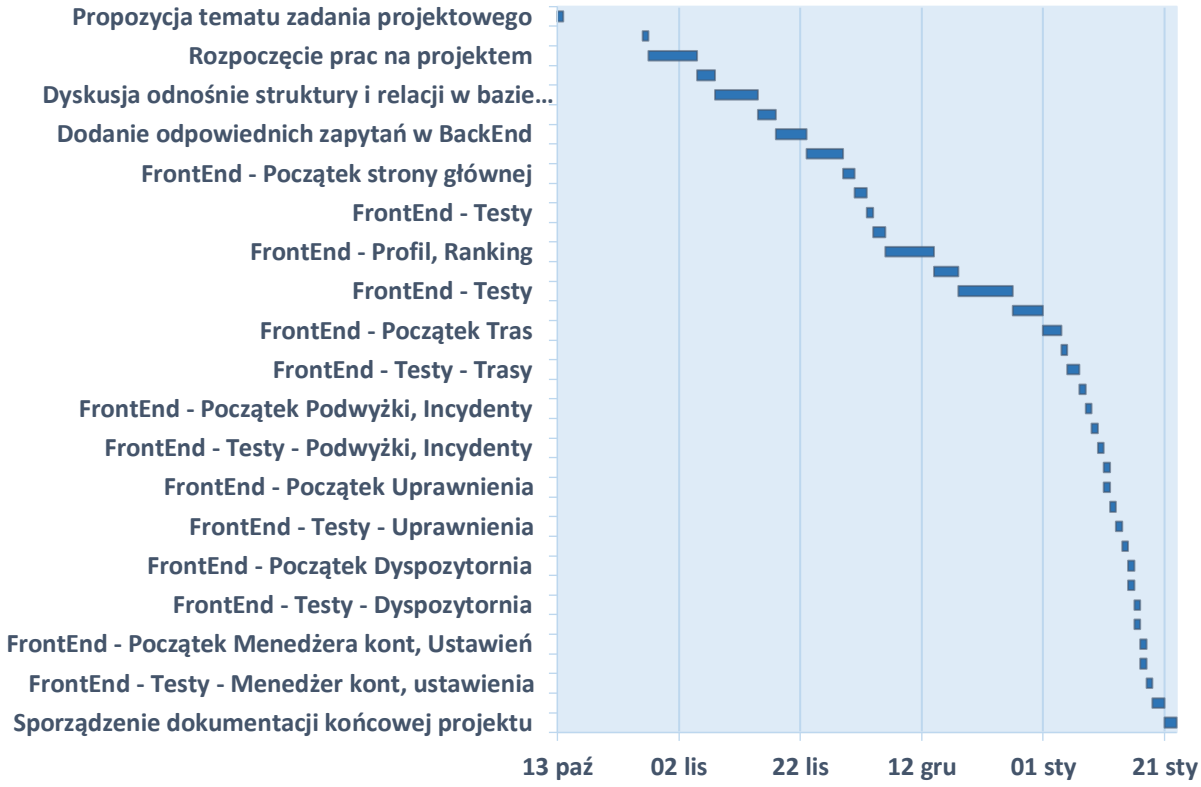
2.4. Przypadki użycia



Rysunek 1 Diagram przypadków użycia

2.5. Harmonogram zadań

Wykres 1 Udział członków zespołu w realizacji zadania



Wykres 2 Wykres Ganta

3. Metodyka pracy

4. Diagramy

4.1. Diagram komponentów

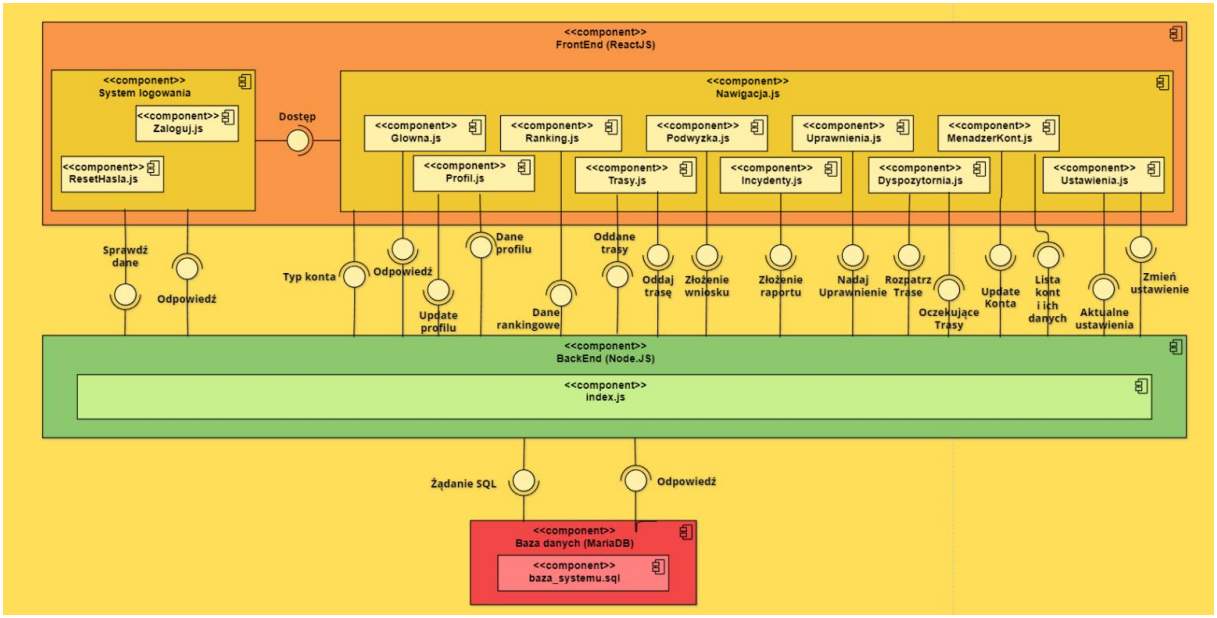


Diagram 1 Diagram komponentów

4.2. Diagram rozlokowania

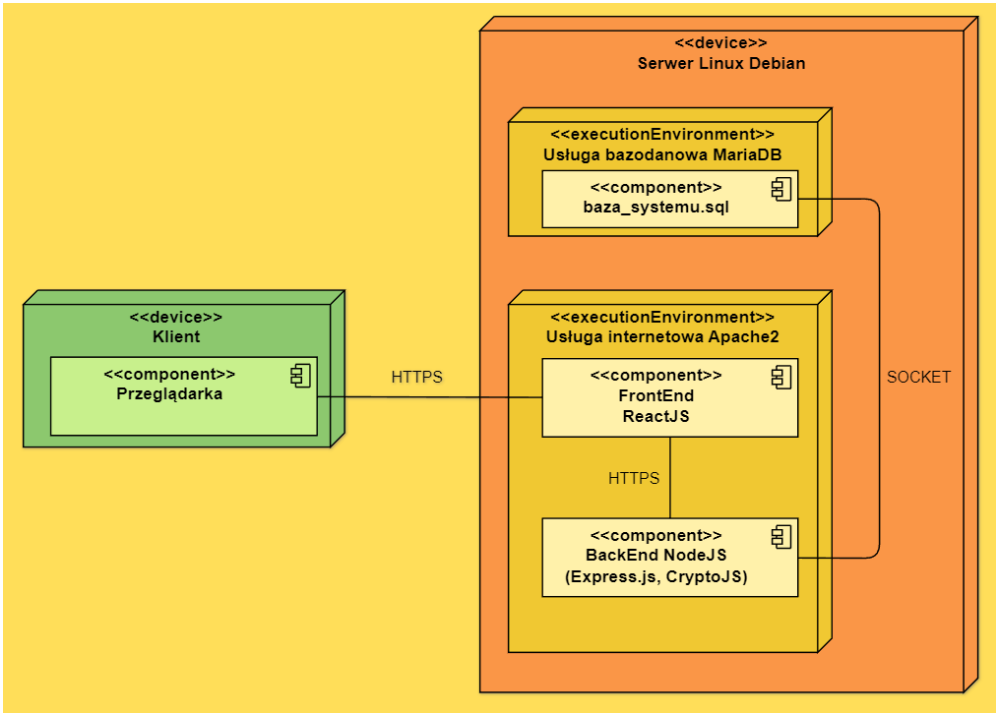


Diagram 2 Diagram rozlokowania

4.3. Diagram stanów

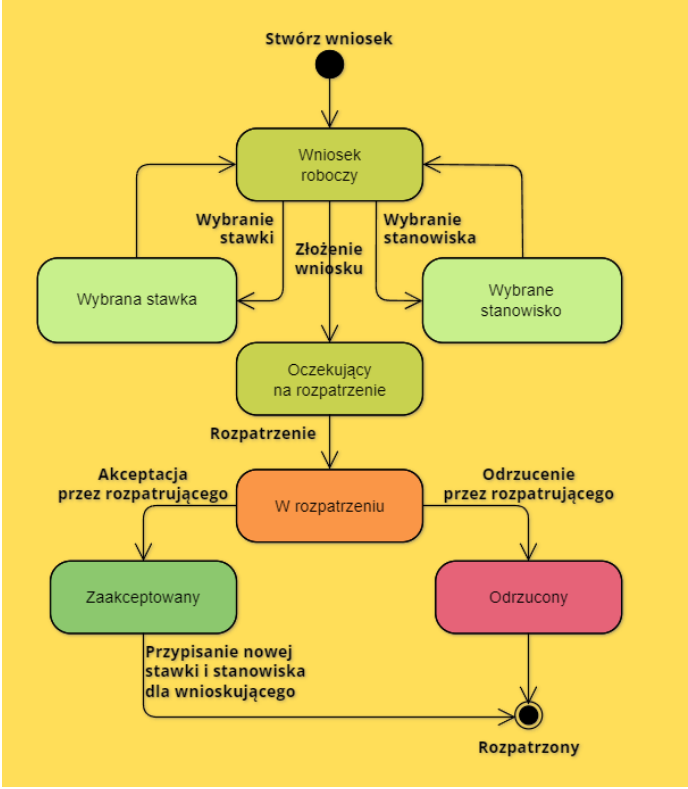


Diagram 3 Diagram stanów

4.4. Protokołowy diagram stanów

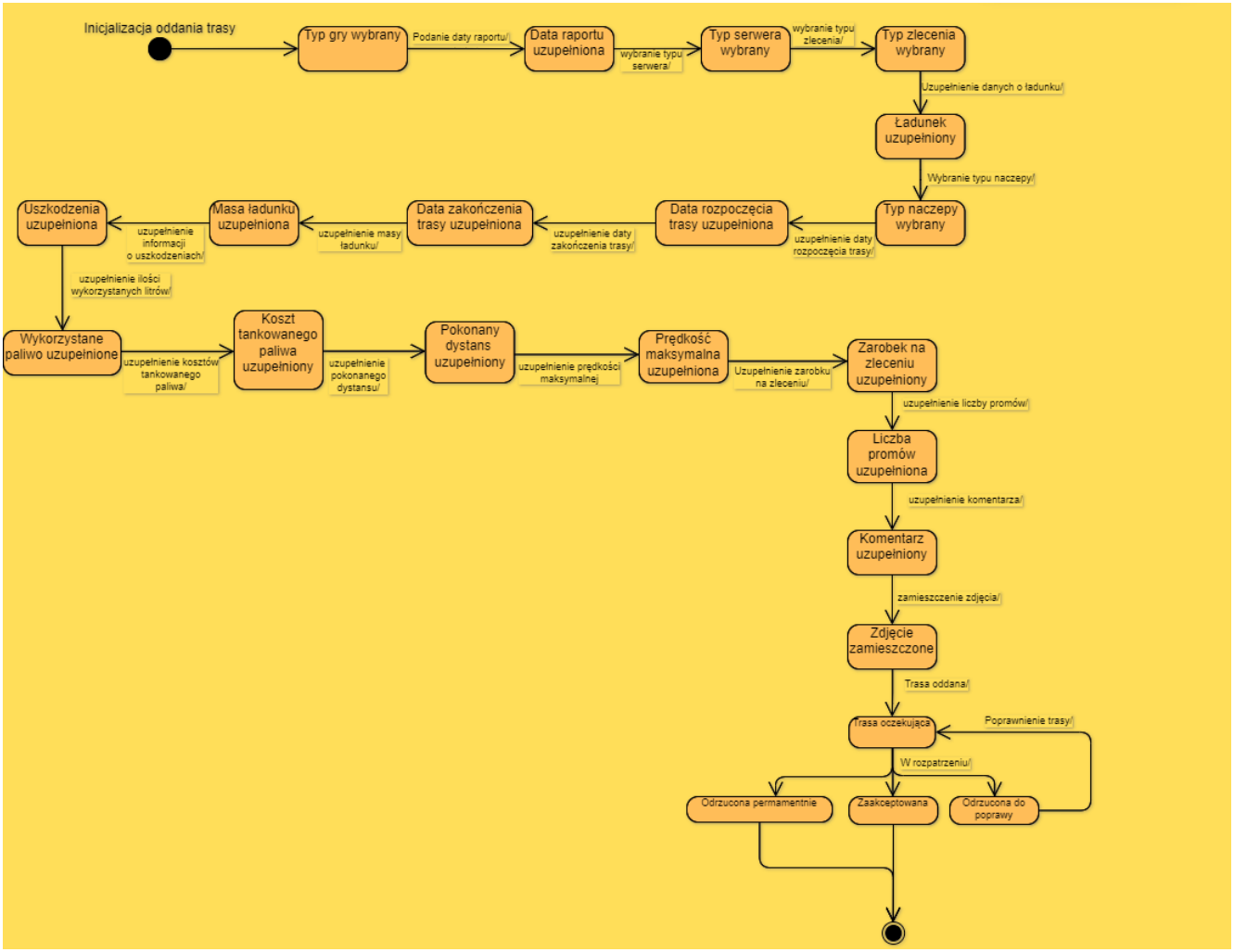


Diagram 4 Protokołowy diagram stanów

5. Projekt architektury

5.1. Wybór technologii

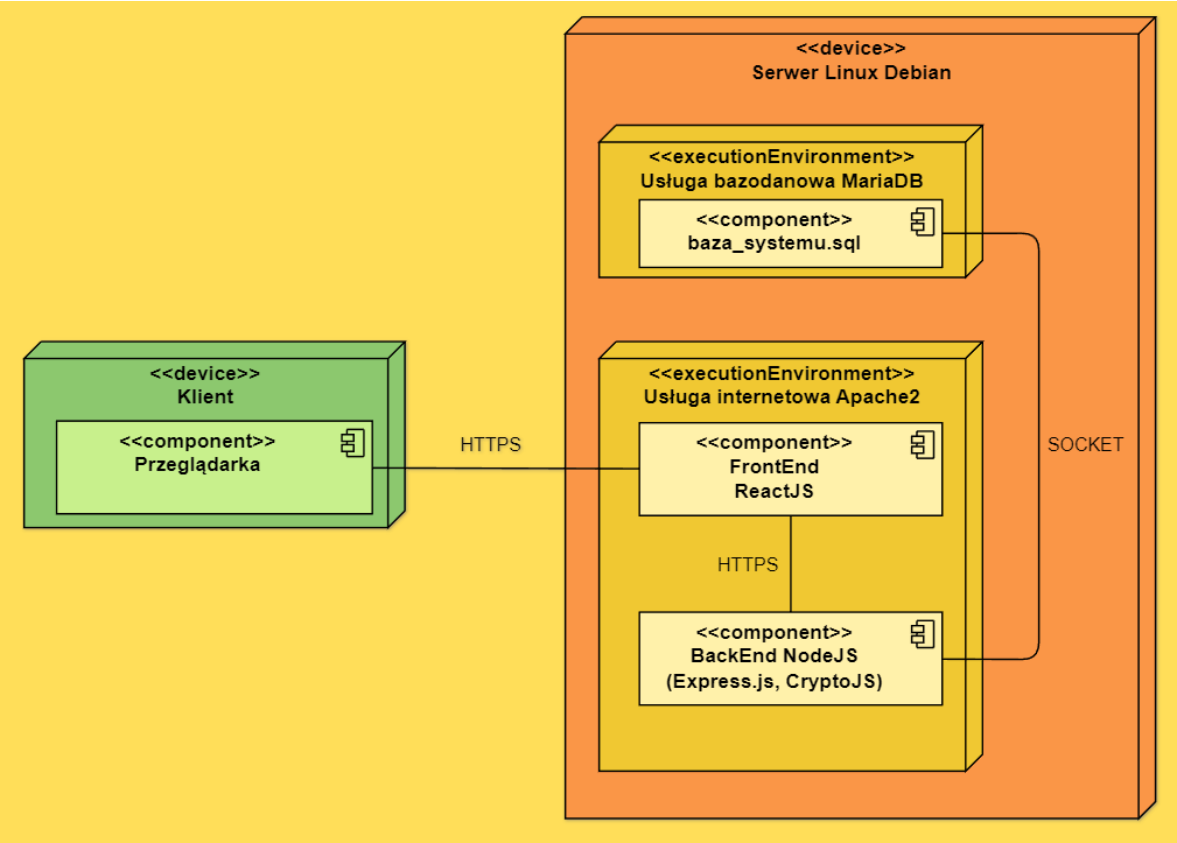
Aplikacja korzysta z:

- Bazy danych – oddzielna usługa serwerowa bazodanowa, w przypadku naszej aplikacji MariaDB, który przetrzymuje niezbędne dane do funkcjonowania strony
- NodeJS – wieloplatformowe środowisko uruchomieniowe o otwartym kodzie do tworzenia aplikacji typu server-side napisanych w języku JavaScript.
- ReactJS – biblioteka języka programowania, która wykorzystywana jest do tworzenia interfejsów graficznych aplikacji internetowych.
- Express.JS – jest to back-endowy framework, do budowania interfejsów API.

5.2. Projekt architektury aplikacji

4.2 Diagram rozlokowania

Urządzenia klienckie z poziomu własnej przeglądarki łączą się protokołem HTTPS do FrontEnd'u

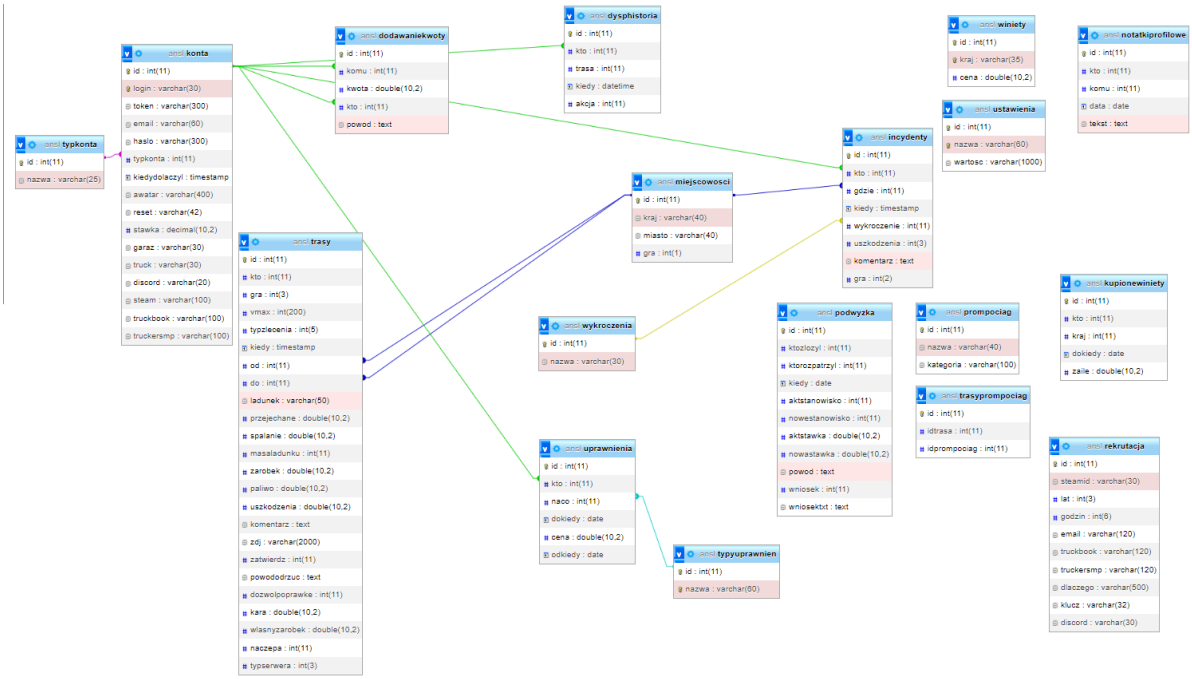


Rysunek 2 Diagram rozlokowania

Urządzenia klienckie z poziomu własnej przeglądarki łączą się protokołem HTTPS do FrontEnd'u aplikacji serwera. FrontEnd został zaprojektowany wykorzystując framework ReactJS. Wszelkie wykonywane operacje w aplikacji wysyłane są protokołem HTTPS do BackEnd'u który służy za bezpieczny pośrednik wstawiania/aktualizowania/usuwania/odczytywania danych w bazie systemu na usłudze bazodanowej MariaDB. BackEnd stworzony został w środowisku Node.JS z wykorzystaniem bibliotek takich jak: Express.js oraz CryptoJS. BackEnd łączy się z usługą bazodanową za pomocą socket'u systemu operacyjnego Linux.

5.3. Projekt bazy danych

Baza danych ma za zadanie przechowywać informacje. Do projektu aplikacji została wykorzystana relacyjna baza danych, w której tabele są powiązane ze sobą przy pomocy relacji.



Rysunek 3 Widok projektu bazy danych

Poniżej przykładowa struktura tabeli dla tabeli ‘dysphistoria’.

```
CREATE TABLE `dysphistoria` (
  `id` int(11) NOT NULL,
  `kto` int(11) NOT NULL,
  `trasa` int(11) NOT NULL,
  `kiedy` datetime NOT NULL DEFAULT current_timestamp(),
  `akcja` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

Listing 1 Tworzenie tabeli 'dysphistoria'

```
ALTER TABLE `dodawaniekwoty`
  ADD CONSTRAINT `dodawaniekwoty_ibfk_1` FOREIGN KEY (`komu`)
REFERENCES `konta` (`id`),
  ADD CONSTRAINT `dodawaniekwoty_ibfk_2` FOREIGN KEY (`kto`)
REFERENCES `konta` (`id`);

--
-- Ograniczenia dla tabeli `dysphistoria`
--
ALTER TABLE `dysphistoria`
  ADD CONSTRAINT `dysphistoria_ibfk_1` FOREIGN KEY (`kto`)
REFERENCES `konta` (`id`);

--
-- Ograniczenia dla tabeli `incydenty`
--
ALTER TABLE `incydenty`
  ADD CONSTRAINT `incydenty_ibfk_1` FOREIGN KEY (`gdzie`)
REFERENCES `miejscowosci` (`id`),
  ADD CONSTRAINT `incydenty_ibfk_2` FOREIGN KEY (`kto`)
REFERENCES `konta` (`id`),
  ADD CONSTRAINT `incydenty_ibfk_3` FOREIGN KEY (`wykroczenie`)
REFERENCES `wykroczenia` (`id`);

--
-- Ograniczenia dla tabeli `konta`
--
ALTER TABLE `konta`
  ADD CONSTRAINT `konta_ibfk_1` FOREIGN KEY (`typkonta`)
REFERENCES `typkonta` (`id`);

--
-- Ograniczenia dla tabeli `trasy`
--
ALTER TABLE `trasy`
  ADD CONSTRAINT `trasy_ibfk_3` FOREIGN KEY (`od`) REFERENCES
`miejscowosci` (`id`),
  ADD CONSTRAINT `trasy_ibfk_4` FOREIGN KEY (`do`) REFERENCES
`miejscowosci` (`id`);

--
-- Ograniczenia dla tabeli `uprawnienia`
--
ALTER TABLE `uprawnienia`
  ADD CONSTRAINT `uprawnienia_ibfk_1` FOREIGN KEY (`kto`)
REFERENCES `konta` (`id`),
  ADD CONSTRAINT `uprawnienia_ibfk_2` FOREIGN KEY (`naco`)
REFERENCES `typyuprawnien` (`id`);
COMMIT;
```

Listing 2 Kod tworzący relacje w bazie danych

6. Implementacja

6.1. Opis podstawowych struktur danych i struktur sterowania

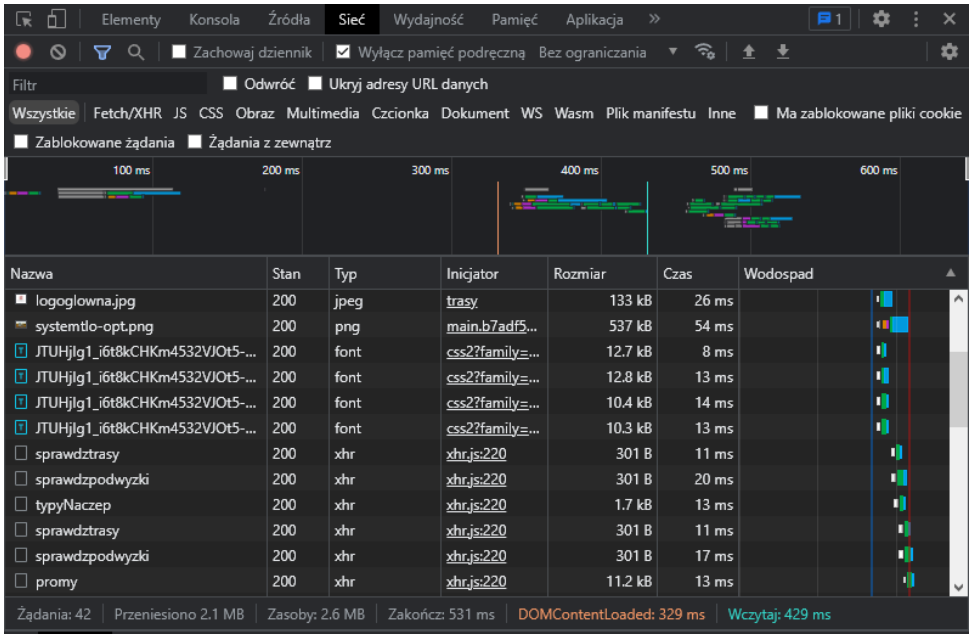
FrontEnd wysyła i odbiera dane z BackEndu za pomocą protokołu HTTPS i typów żądań: POST oraz GET. Pomiędzy FrontEnd'em oraz BackEnd'em zachodzi powiązanie struktury folderów. Folder frontendu /awatary oraz /trasy wskazuje na foldery o takich samych nazwach w strukturze plików backendu.

Backend służy głównie jako bezpieczny pośrednik wymiany danych. Zanim użytkownik otrzyma dane do wyświetlenia na stronie internetowej, backend wpierw wysyła zapytanie do bazy danych MariaDB. Struktura bazy danych przedstawiona jest w rozdziale 5.3.

6.2. Testy

Przeprowadzono testy dla wszystkich podstron systemu, aby zapobiec nieprzewidzianych zachowań, które mogą skutkować naruszeniem prywatności innych użytkowników systemu. Wszelkie wprowadzane dane przez użytkownika weryfikowane są zarówno po stronie klienta jak i po stronie serwera. Zapytania do SQL nie narażone są na SQL Injection, ponieważ zapobiega sama w sobie takim sytuacjom biblioteka używana do operacji na bazie danych.

System pod względem wydajnościowym działa błyskawicznie.



Rysunek 4 Zrzut ekranu z zachowania operacji sieciowych

Powyższy wycinek ekranu z zachowania operacji sieciowych wykazuje, że przykładowa podstrona Trasy, która zawiera w sobie wiele danych i ewentualnych zdjęć, dodatkowo ładowane wszelkie obrazy flag państw i tła aplikacji wykonuje się w 531 milisekund uwzględniając fakt, że aplikacja działa na budżetowym hostingu i czas mierzony był dla wyłączonej pamięci podręcznej.

7. Użytkowanie

7.1. Instrukcja wdrożeniowa

FrontEnd'owa część systemu została napisana i skompilowana przy użyciu frameworku ReactJS. Pliki „build" trzeba umieścić na jakimkolwiek hostingu, która oferuje usługi witrynowe. Zaleca się skorzystania z usługi „Apache2" i skonfigurowania wirtualnego hosta tej usługi w następujący sposób:

```
<VirtualHost *:443>
          ServerName      domena.com
          ServerAlias     www.domena.com
          DocumentRoot    /sciezkaFrontendu/build/
          <IfModule mod_headers.c>
                 Header set Access-Control-Allow-Origin "*"
         </IfModule>
         SSLEngine               on
         Protocols       h2 h2c http/1.1
         Include /etc/letsencrypt/options-ssl-apache.conf
         <Directory /sciezkaFrontendu/build>
                 Options Indexes FollowSymLinks MultiViews
                 AllowOverride All
                 Require all granted
                 RewriteEngine on
                 RewriteCond %{REQUEST_FILENAME} -f [OR]
                 RewriteCond %{REQUEST_FILENAME} -d
                 RewriteRule ^ - [L]
                 RewriteRule ^ index.html [L]
         </Directory>
         SSLCertificateFile      /sciezkaCertDomeny.pem
        SSLCertificateKeyFile /sciezkaPrivCertDomeny.pem
</VirtualHost>
```

Listing 3 Konfiguracja hosta

BackEnd aplikacji wymaga uruchomienia pliku index.js w środowisku Node.js. Niezbędne biblioteki wykorzystane w kodzie można zainstalować komendą: npm install , w terminalu na ścieżce folderu backendu. Aby backend działał jako proces w tle, na hostingu zaleca się skorzystanie z usługi ekranowej „screen". Komenda: screen -dmS „backend" node index.js spowoduje uruchomienie ekranu backend, w którym aplikacja działa do momentu własnoręcznego wyłączenia ekranu lub restartu hostingu. Backend przechowuje również zdjęcia do których Frontend fizycznie nie ma dostępu. Ważne jest zatem aby utworzyć foldery powiązane w strukturze plików frontendu, wskazujące na folder zdjęć backendu. W systemie Linux można to wykonać za pomocą komendy:
ln -s /sciezkaBackend/awatary /sciezkaFrontend/build/img/awatary
ln -s /sciezkaBackend/trasy /sciezkaFrontend/build/img/

Wymagane jest również stworzenie poddomeny dla tej usługi i ustawieniu jej w usłudze Apache2 następująco:

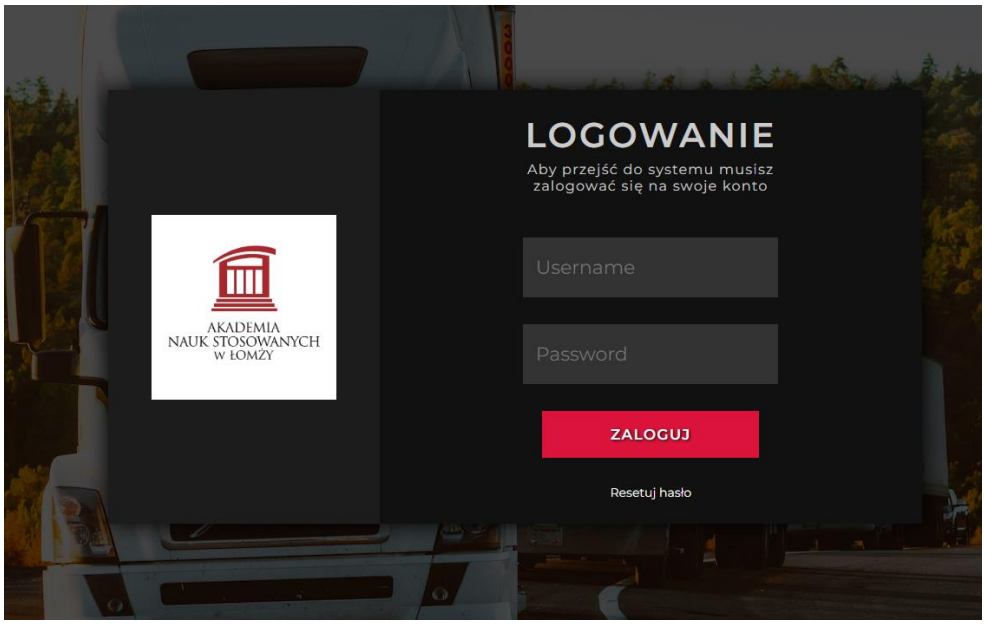
```
<VirtualHost *:443>
        ServerName      backend.domena.com
        ServerAlias     www.backend.domena.com
        SSLEngine on
        Protocols       h2 h2c http/1.1
        SSLCertificateFile /sciezkaCertDomeny.pem
        SSLCertificateKeyFile /sciezkaPrivCertDomeny.pem
        Include /etc/letsencrypt/options-ssl-apache.conf
        ProxyPreserveHost on
        ProxyPass / http://localhost:3003/
        ProxyPassReverse / http://localhost:3003/
</VirtualHost>
```

Listing 4 Tworzenie poddomeny

Niezbędne jest również wykorzystanie usługi bazodanowej np. MariaDB. Trzeba stworzyć konto oraz bazę danych i zaimportować do niej plik SQL. Ostatnim finalnym etapem jest podmiana wartości pliku .env w backendzie na aktualne dane stworzonej bazy danych. Po wykonaniu tych czynności aplikacja jest gotowa do użytku.

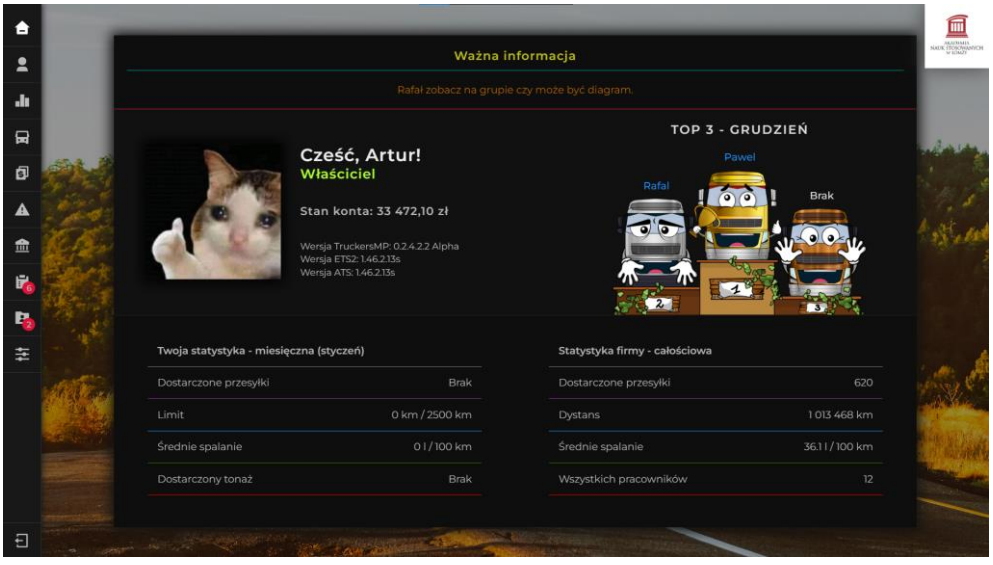
7.2. Instrukcja użytkowania

Użytkownik loguje się za pomocą loginu i hasła w oknie logowania.



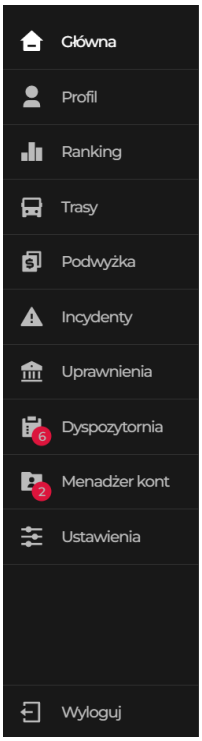
Rysunek 5 Ekran logowania Systemu Wirtualnej Spedycji

Po zalogowaniu, ukazuje się użytkownikowi strona główna, która składa się w głównej mierze, z częściowych informacjach o użytkowniku, takich jak: nick, stan konta, miesięczna statystyka, rankingu oraz statystyki firmy.



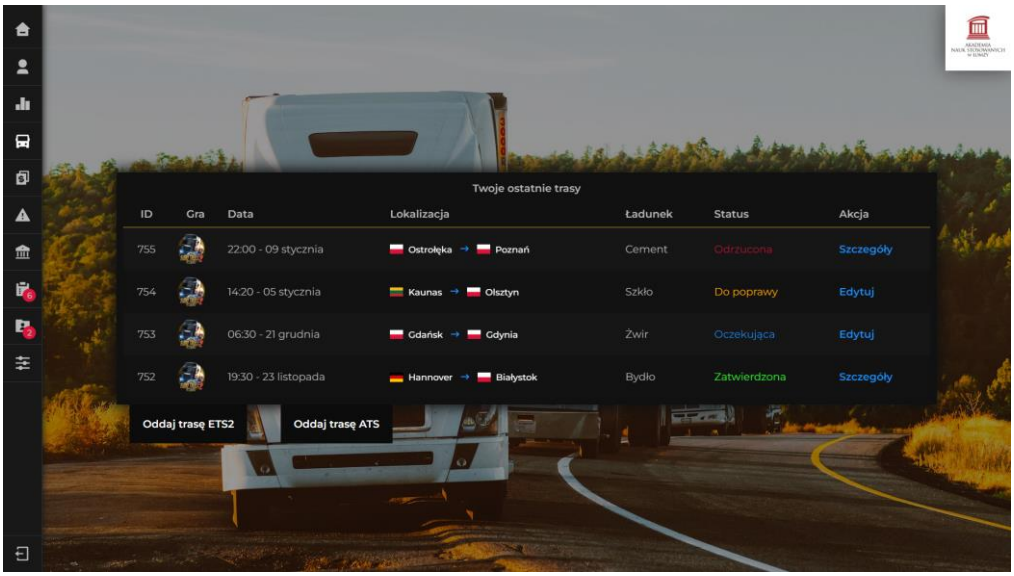
Rysunek 6 Strona główna systemu wirtualnej spedycji

Z lewej strony ekranu, znajduje się menu, które w fazie neutralnej jest zminimalizowane. Po przejechaniu w obszar menu, zostaje rozwinięte i ukazane są nazwy poszczególnych podstron aplikacji.

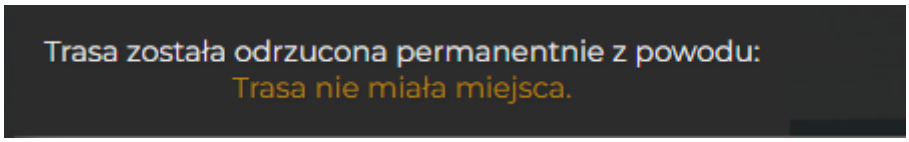


Rysunek 7 Menu systemu wirtualnej spedycji

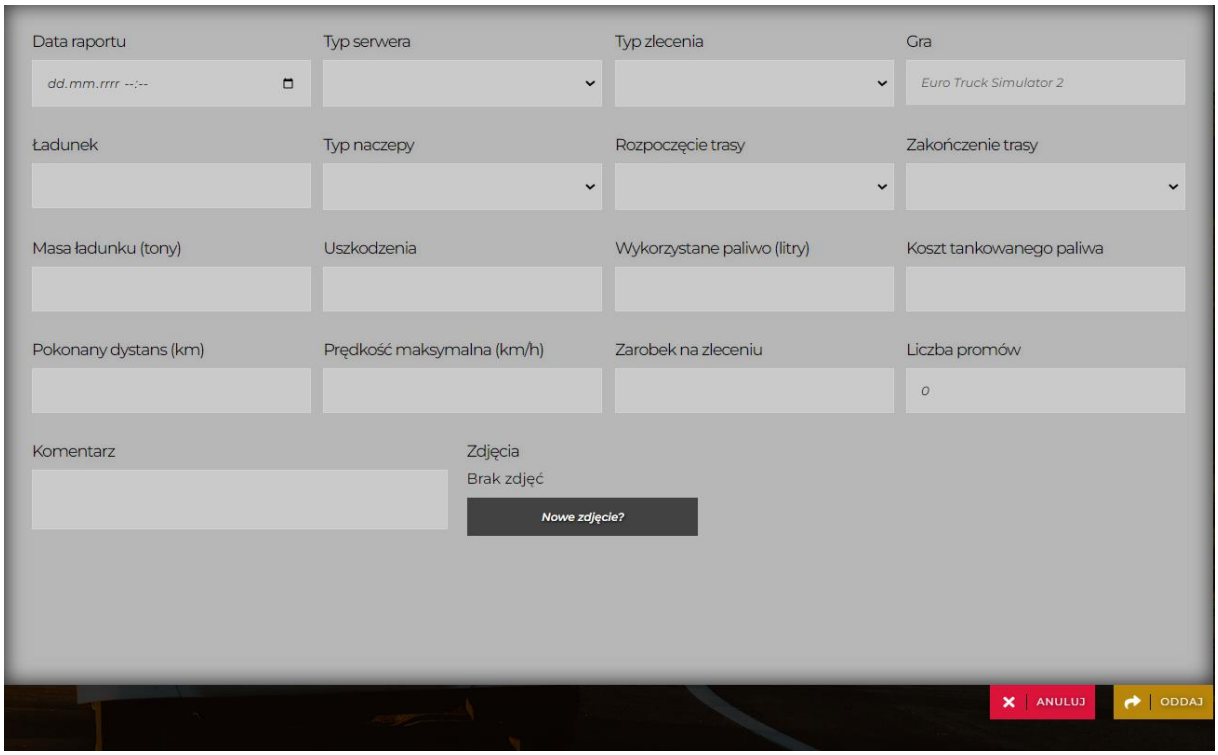
Gracz ma możliwość oddania raportu o trasie, po wybraniu z menu rozwijanego zakładki dla użytkownika wyświetlają się trasy, które zostały przez niego złożone, z podstawowymi informacjami takich jak ID trasy, rodzaj gry (czy ETS2 lub ATS), lokalizacją początkową, lokalizacją docelową, typem ładunku, statusie trasy(zaakceptowana, oczekująca, do poprawy, odrzucona). Jeżeli trasa została odrzucona luz zaakceptowana w rubryce „Akcja” mamy możliwość podejrzenia szczegółów o trasie. W przypadku trasy odrzuconej, mamy informację o powodzie odrzucenia danej trasy.



Rysunek 8 Zrzut ekranu z Tras



Rysunek 9 Informacja w przypadku odrzucenia trasy



Rysunek 10 Okno dialogowe oddawania tras

8. Podsumowanie

8.1. Opis celów zrealizowanych i niezrealizowanych

8.1.1. Cele zrealizowane

CZ.01	Oddawanie tras z podziałem na wersje gry
CZ.02	Dodanie funkcji administracyjnych dla właściciela.
CZ.03	Dodanie menedżera kont
CZ.04	Stworzenie rankingu

Tabela 3 Cele zrealizowane

8.1.2. Cele niezrealizowane

CNZ.01	Brak serwera poczty, przez co administrator musi wysłać wygenerowany przez system token do resetu hasłą
CNZ.02	Brak formularza rejestracyjnego. Administrator musi dodać ręcznie użytkownika

Tabela 4 Cele niezrealizowane

8.2. Opis problemów jakie wynikły podczas pracy i jak z nimi poradzono

Na komputerach o bardziej złożonym zestawie reguł firewalla w instytucjach publicznych, frontend nie mógł skomunikować się z backendem który działał na hostingu na niestandardowym porcie. Zastosowano w takim przypadku usługę Apache2, gdzie skonfigurowano proxy dla backendu aby traktowany był i obsługiwany po protokole HTTPS.

Kolejnym problemem są oddzielne ścieżki dla frontendu i backendu, które wynikają z nałożonych struktur drzewa plików przez inne usługi takie jak Apache2. W takiej sytuacji gdy w systemie wgrywane są zdjęcia, backend może je zapisać tylko w swoim folderze, do którego frontend nie ma fizycznego dostępu. Jako, że systemem operacyjnym hostingu aplikacji jest Linux - zastosowano rozwiązanie tworzenia folderów powiązanych (odpowiednik skrótów w Windowsie), komendą: ln -s /sciezkaDoZdjecBackend/ /sciezkaFrontend/zdjecia

8.3. Wskazanie możliwych kierunków rozbudowy systemu

Szczególnym możliwym kierunkiem możliwym do zrealizowania w celu rozbudowy systemu jest stworzenie panelu tachografu, gdzie użytkownicy wypełniali by formularze związane z postojem, czasem pracy, przebytymi kilometrami.

Kolejnym kierunkiem możliwym do realizacji jest Karta Paliwowa, która działałaby na zasadzie dodawania rabatów, gratisów podczas tankowania na stacji. Gdy podczas rozrywki gracz zatankowałby określoną ilość paliwa, dostawałby punkty, które mógłby później wymienić na rabat.

8.4. Wnioski

Projekt o nazwie „System wirtualnej spedycji" zakończył się sukcesem. Aplikacja webowa działa i została przetestowana na hostingu. Zaimplementowane zostały funkcjonalności zgodnie z założeniami jakie miała spełniać aplikacja

Jednakże, warto zwrócić uwagę, że proces projektowania aplikacji webowej jest bardzo złożony, oraz w przypadku tworzenia aplikacji webowej, która ma za zadanie urozmaicić rozgrywkę graczom, nie da się zamknąć projektu w określonych ramach, gdyż zawsze tworzenie aplikacji zaczepia o różne aspekty, których oczekują gracze. Nie inaczej było z systemem wirtualnej spedycji. Nasza grupa miała za zadanie, przed stworzeniem samego projektu, zapoznać się z elementami, które dla gracza wprowadzą coś nowego i sam gracz będzie chętnie z tego korzystał.

Bardzo ciekawą częścią naszej pracy było stworzenie właśnie czegoś, z czym nie można spotkać się podczas codziennej rozgrywki w tą grę i sam fakt, że mimo funkcjonalności, które zostały wdrożone w ten projekt, sam projekt ma wielką ilość kierunków rozwoju samej aplikacji.

9. Literatura

[1] P.Kamiński , *React. Wstęp do programowania* wyd. Helion

[2] T. Sochacki, *JavaScript. Techniki zaawansowane* wyd. Helion

[3] Meta Platforms. *React – A JavaScript library for building user interfaces*. https://reactjs.org

[4] OpenJS Foundation, *Node.js v17.9.1 documentation*, https://nodejs.org/docs/latestv17.x/api/synopsis.html

[5] ExpressJS, *Fast, unopinionated, minimalist web framework for Node.js* https://expressjs.com/en/4x/api.html

[6] S.Pasquali, *Node.js. Projektowanie, wdrażanie i utrzymywanie aplikacji*, Helion

[7] D.Herron, *Platforma Node.js. Przewodnik webdevelopera. Wydanie III*, Helion

[8] V.M Grippa, S.Kuzmichev, *MySQL. Jak zaprojektować i wdrożyć wydajną bazę danych. Wydanie II*, Helion

[9] R.J.T Dyer, *Learning MySQL and MariaDB. Heading in the Right Direction with MySQL and MariaDB*, Packt Publishing

10. Spisy

10.1. Spis rysunków

10.2. Spis tabel

10.3. Spis listingów

10.4. Spis diagramów

10.5. Spis wykresów