



**Politechnika
Śląska**

PROJEKT INŻYNIERSKI

System do monitorowania położenia geoprzestrzennego obiektów

Jakub SIBIK

Nr albumu: 300420

Kierunek: Informatyka

Specjalność: Informatyczne Systemy Mobilne i Przemysłowe

PROWADZĄCY PRACĘ

dr inż. Łukasz Wyciślik

KATEDRA Informatyki Stosowanej

Wydział Automatyki, Elektroniki i Informatyki

Gliwice 2025

Tytuł pracy

System do monitorowania położenia geoprzestrzennego obiektów

Streszczenie

(Streszczenie pracy – odpowiednie pole w systemie APD powinno zawierać kopię tego streszczenia.)

Słowa kluczowe

lokalizator, lokalizacja, pojazd, kierowca

Thesis title

Thesis title in English

Abstract

(Thesis abstract – to be copied into an appropriate field during an electronic submission – in English.)

Key words

(2-5 keywords, separated by commas)

Spis treści

1	Wstęp	1
2	Analiza tematu	3
3	Wymagania i narzędzia	5
4	Specyfikacja zewnętrzna	9
5	Specyfikacja wewnętrzna	11
6	Weryfikacja i walidacja	13
7	Podsumowanie i wnioski	15
	Bibliografia	17
	Spis skrótów i symboli	21
	Źródła	23
	Lista dodatkowych plików, uzupełniających tekst pracy	25
	Spis rysunków	27
	Spis tabel	29

Rozdział 1

Wstęp

Początki technologii do określania pozycji sięgają lat 60. XX wieku. Powstał wtedy system NAVSAT (ang. Navigation Satellite System) - będący pierwszym satelitarnym systemem nawigacyjnym. Został on opracowany przez Stany Zjednoczone oraz wykorzystywany był przez tamtejszą marynarkę wojenną. W latach 70. XX wieku postanowiono wprowadzić międzynarodowy standard, dzięki czemu powstał system GPS (ang. Global Positioning System), który jest używany po dziś dzień.

Długi czas obecności tego systemu na rynku zaowocował jego rozwojem, jak również dostępnością dla przeciętnych użytkowników. W efekcie tego, GPS jest wsparciem dla ludzi w wielu dziedzinach. W obecnych czasach ponad połowa światowej populacji posiada smartfony, które to mają wbudowane systemy GPS. Pozwala to przede wszystkim na sprawną nawigację do celu czy dokładne ustalenie pozycji danej osoby. Samochody również są w posiadaniu znacznej części populacji, a ponadto stanowią dosyć znaczną część budżetu domowego. Z tego względu ludzie zaopatrują się w lokalizatory samochodowe. Podczas kupna takiego urządzenia, klient otrzymuje zazwyczaj dostęp do strony internetowej, na której jest w stanie sprawdzać położenie swojego samochodu, w którym został umieszczony lokalizator. Takie rozwiązania są dosyć proste, niewystarczające dla wielu użytkowników, wygląd interfejsu również pozostawia wiele do życzenia. Wraz z rozwojem komputerów oraz smartfonów, zwiększają się możliwości do stworzenia aplikacji do zarządzania lokalizatorami w pojazdach, która oferowałaby większą ilość funkcjonalności, oraz która byłaby atrakcyjniejsza niż podstawowe odpowiedniki.

Celem niniejszej pracy inżynierskiej jest stworzenie aplikacji przeglądarkowej pozwalającej na zarządzanie lokalizatorami, pojazdami i kierowcami oraz możliwość wyświetlania ich tras na mapie. Praca obejmuje proces i sposób tworzenia oprogramowania, specyfikacje: zewnętrzną i wewnętrzną, testowanie, jak również efekty i wnioski.

Rozdział 2

Analiza tematu

Lokalizatorów samochodowych na rynku jest wiele. W zależności od ceny, możemy otrzymać dodatkowe funkcjonalności, mniej lub bardziej znaczące, są to między innymi czujnik wstrząsu, monitorowanie prędkości czy podsłuch. Producenci zazwyczaj posiadają własne strony internetowe, do których klient otrzymuje bezpłatny dostęp po zakupie produktu. Pozwalają one zwykle powiązać lokalizator z kontem użytkownika i śledzić na bieżąco jego lokalizację.

W celu stworzenia aplikacji do monitorowania lokalizatorów dla jak największej liczby ludzi, korzystających z tego typu urządzeń, warto przeanalizować kilka kwestii. Niezbędną funkcją lokalizatora jest możliwość jego konfiguracji, aby wysyłał dane na konkretny adres IP oraz port. Umożliwi to dostęp aplikacji do lokalizacji urządzenia. Z tego powodu odrzucono produkty o najniższej cenie na rynku, gdyż nie oferują one wymaganej do działania programu funkcji. Kolejnym znaczącym aspektem wyboru lokalizatora jest jego cena w kontekście klienta. Należy wykluczyć najdroższe opcje, aby nie ograniczyć ilości potencjalnych użytkowników aplikacji. Uwzględniając powyższe wymagania, wybrano model Tk108. Dodatkowym atutem jest jego akumulator, którego pojemność wynosi 10000 mAh, dzięki czemu nie wymaga częstego ładowania.

Podczas przeprowadzania analizy tematu, należało również wybrać stos technologiczny, w którym aplikacja będzie tworzona. Ze względu na liczne biblioteki usprawniające proces rozwijania oprogramowania, część funkcjonalną programu postanowiono napisać w języku Java. W celu konfiguracji aplikacji przeglądarkowej została wybrana platforma programistyczna, jaką jest Spring Boot - jedna z bardziej popularnych opcji przy tego typu programach pisanych w języku Java. Drugą częścią, która zostanie stworzona, jest część interfejsu. W tym przypadku wybrano bibliotekę React, będącą powszechnym narzędziem, oferującym wiele przydatnych funkcji. Biblioteka oparta będzie na języku TypeScript - jest to rozszerzenie języka JavaScript, które wymaga określania typów zmiennych, dzięki czemu można uniknąć dużej liczby błędów w przyszłości, które pojawiają się w JavaScript

przy braku typowania. Należy się również zastanowić nad przechowywaniem danych. Służą do tego bazy danych. W niniejszej aplikacji w celu tworzenia i obsługi takiej bazy zostanie użyty język PostgreSQL. Jest to uwarunkowane danymi, które pojawiają się w systemie, a mianowicie dane geograficzne - długość i szerokość geograficzna. PostgreSQL posiada rozszerzenie - PostGIS - umożliwiające zapis długości i szerokości geograficznej w jednej kolumnie, jako punkt na Ziemi.

Rozdział 3

Wymagania i narzędzia

Tworzona aplikacja ma ułatwić użytkownikowi zarządzanie lokalizatorami, kierowcami i pojazdami jednocześnie oraz pozwalać na wyświetlanie tras poszczególnych kierowców i pojazdów. Wymagania aplikacji należy podzielić na funkcjonalne i нефункционаłne. Przyjrzyjmy się w pierwszej kolejności wymaganiom funkcjonalnym. Poniżej wypisano przewidziane przypadki użycia programu.

Dla niezalogowanego użytkownika (przedstawia je również diagram widoczny na rys 3.1):

- Rejestracja - gdy użytkownik wchodzi na stronę aplikacji, może się zarejestrować, czyli stworzyć bezpłatne konto, które będzie przypisane do podanego przez niego w formularzu rejestracji adresu email
- Logowanie - w przypadku uprzedniej rejestracji w systemie, użytkownik może zalogować się na istniejące konto

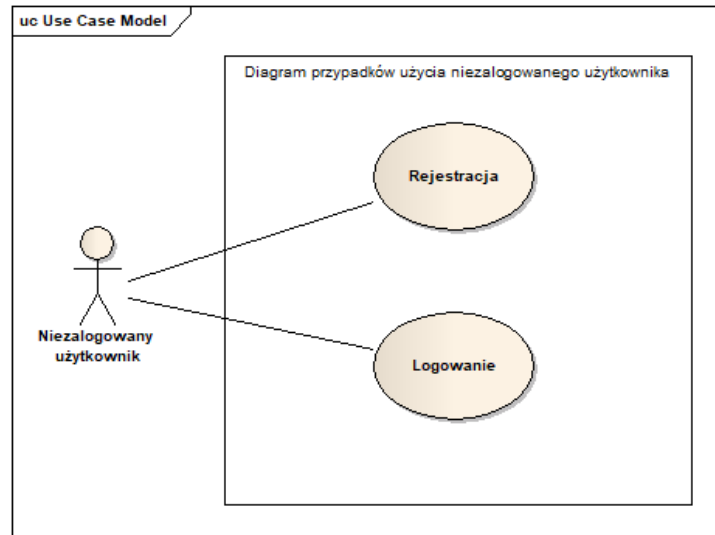
Dla zalogowanego użytkownika (przedstawia je również diagram widoczny na rys 3.2):

- Dodanie obiektu
 - Dodanie kierowcy - użytkownik ma możliwość dodania kierowcy do swojego konta, podając ich imię oraz nazwisko
 - Dodanie pojazdu - pozwala dodać pojazd do konta poprzez podanie jego marki, modelu, numeru rejestracyjnego oraz numeru VIN
 - Dodanie lokalizatora - w ten sposób użytkownik może dodać posiadane lokalizatory TK108 do swojego konta, podając jego nazwę, numer seryjny oraz typ
- Dodanie powiązania
 - Dodanie kierowcy pojazdu - ten przypadek umożliwia powiązanie kierowcy z pojazdem w podanym przez użytkownika okresie czasu

- Dodanie lokalizatora w pojeździe - działa na takiej zasadzie, jak powyższy podpunkt, umożliwia powiązanie lokalizatora z pojazdem w danym okresie czasu
- Wyświetlenie trasy
 - Wyświetlenie trasy kierowcy - użytkownik może wyświetlić na mapie trasę przebytą przez danego kierowcę w wybranym przez siebie okresie czasu
 - Wyświetlenie trasy pojazdu - analogicznie do powyższego przypadku, możliwość wyświetlenia trasy pokonanej przez pojazd w danym czasie
- Usunięcie obiektu
 - Usunięcie kierowcy - powoduje usunięcie istniejącego kierowcy z systemu
 - Usunięcie pojazdu - daje możliwość usunięcia wybranego pojazdu z aplikacji
 - Usunięcie lokalizatora - podobnie do powyższych, skutkuje usunięciem danego lokalizatora
- Usunięcie powiązania
 - Usunięcie kierowcy pojazdu - usuwa czasowe powiązanie kierowcy oraz pojazdu
 - Usunięcie lokalizatora w pojeździe - analogicznie, użytkownik może się w ten sposób pozbyć powiązania lokalizatora z pojazdem z danego okresu czasu
- Usunięcie trasy z mapy
 - Usunięcie trasy kierowcy - powoduje zniknięcie z mapy trasy kierowcy z danego okresu czasu
 - Usunięcie trasy pojazdu - pozwala usunąć z mapy wcześniej wyświetloną trasę pojazdu

Przejdźmy teraz do wymagań niefunkcjonalnych:

- Niskie wymagania systemowe - aplikacja powinna być dostępna dla każdego urządzenia posiadającego przeglądarkę internetową oraz dostęp do internetu. W tym celu wymagane jest, aby program był aplikacją przeglądarką.
- Skalowalność interfejsu użytkownika - wymagane jest, aby program był dostosowany do różnej wielkości oraz rozdzielczości ekranów. Użytkowanie aplikacji na smartfonie powinno być równie sprawne i nie sprawiające problemów, jak podczas korzystania z komputera stacjonarnego lub laptopa.

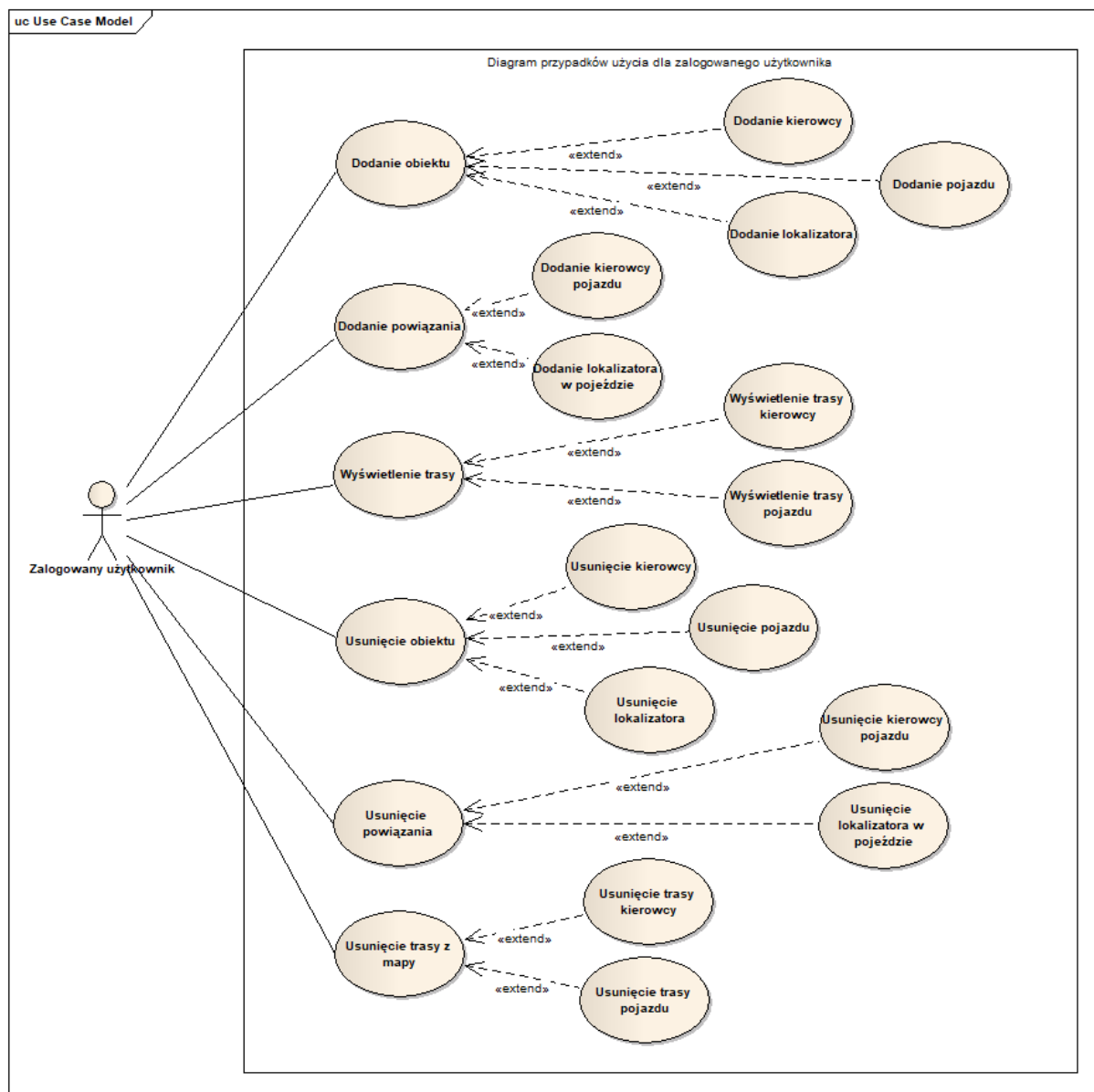


Rysunek 3.1: Diagram przypadków użycia niezalogowanego użytkownika.

- Prosta obsługa - ze względu na duże i zróżnicowane grono potencjalnych użytkowników, założono, iż aplikacja będzie możliwie jak najmniej skomplikowana w obsłudze. Dostęp do każdej funkcji programu powinien być możliwy przy wykonaniu minimalnej ilości kliknięć myszy (lub ekranu w przypadku urządzeń dotykowych), nieprzekraczającej trzech.

Aby przystąpić do tworzenia aplikacji, należało przygotować odpowiednie narzędzia

- wymagania funkcjonalne i нефункционалне
- przypadki użycia (diagramy UML) – dla prac, w których mają zastosowanie
- opis narzędzi, metod eksperymentalnych, metod modelowania itp.
- metodyka pracy nad projektowaniem i implementacją – dla prac, w których ma to zastosowanie



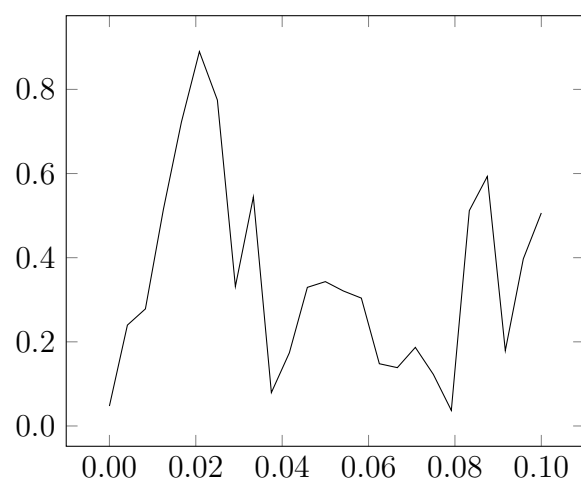
Rysunek 3.2: Diagram przypadków użycia zalogowanego użytkownika.

Rozdział 4

Specyfikacja zewnętrzna

Jeśli „Specyfikacja zewnętrzna”:

- wymagania sprzętowe i programowe
- sposób instalacji
- sposób aktywacji
- kategorie użytkowników
- sposób obsługi
- administracja systemem
- kwestie bezpieczeństwa
- przykład działania
- scenariusze korzystania z systemu (ilustrowane zrzutami z ekranu lub generowanymi dokumentami)



Rysunek 4.1: Podpis rysunku po rysunkiem.

Rozdział 5

Specyfikacja wewnętrzna

Jeśli „Specyfikacja wewnętrzna”:

- przedstawienie idei
- architektura systemu
- opis struktur danych (i organizacji baz danych)
- komponenty, moduły, biblioteki, przegląd ważniejszych klas (jeśli występują)
- przegląd ważniejszych algorytmów (jeśli występują)
- szczegóły implementacji wybranych fragmentów, zastosowane wzorce projektowe
- diagramy UML

Krótką wstawka kodu w linii tekstu jest możliwa, np. **int a**; (biblioteka `listings`). Dłuższe fragmenty lepiej jest umieszczać jako rysunek, np. kod na rys 5.1, a naprawdę długie fragmenty – w załączniku.

```
1 class test : public basic
2 {
3     public:
4         test (int a);
5         friend std::ostream operator<<(std::ostream & s,
6                                         const test & t);
7     protected:
8         int _a;
9
10 };
```

Rysunek 5.1: Pseudokod w `listings`.

Rozdział 6

Weryfikacja i walidacja

- sposób testowania w ramach pracy (np. odniesienie do modelu V)
- organizacja eksperymentów
- przypadki testowe zakres testowania (pełny/niepełny)
- wykryte i usunięte błędy
- opcjonalnie wyniki badań eksperymentalnych

Tabela 6.1: Nagłówek tabeli jest nad tabelą.

ζ	metoda						
	alg. 1	alg. 2	alg. 3			alg. 4, $\gamma = 2$	
			$\alpha = 1.5$	$\alpha = 2$	$\alpha = 3$	$\beta = 0.1$	$\beta = -0.1$
0	8.3250	1.45305	7.5791	14.8517	20.0028	1.16396	1.1365
5	0.6111	2.27126	6.9952	13.8560	18.6064	1.18659	1.1630
10	11.6126	2.69218	6.2520	12.5202	16.8278	1.23180	1.2045
15	0.5665	2.95046	5.7753	11.4588	15.4837	1.25131	1.2614
20	15.8728	3.07225	5.3071	10.3935	13.8738	1.25307	1.2217
25	0.9791	3.19034	5.4575	9.9533	13.0721	1.27104	1.2640
30	2.0228	3.27474	5.7461	9.7164	12.2637	1.33404	1.3209
35	13.4210	3.36086	6.6735	10.0442	12.0270	1.35385	1.3059
40	13.2226	3.36420	7.7248	10.4495	12.0379	1.34919	1.2768
45	12.8445	3.47436	8.5539	10.8552	12.2773	1.42303	1.4362
50	12.9245	3.58228	9.2702	11.2183	12.3990	1.40922	1.3724

Rozdział 7

Podsumowanie i wnioski

- uzyskane wyniki w świetle postawionych celów i zdefiniowanych wyżej wymagań
- kierunki ewentualnych danych prac (rozbudowa funkcjonalna ...)
- problemy napotkane w trakcie pracy

Bibliografia

- [1] Imię Nazwisko i Imię Nazwisko. *Tytuł strony internetowej*. 2021. URL: <http://gdzies/w/internecie/internet.html> (term. wiz. 30.09.2021).
- [2] Imię Nazwisko, Imię Nazwisko i Imię Nazwisko. „Tytuł artykułu konferencyjnego”. W: *Nazwa konferencji*. 2006, s. 5346–5349.
- [3] Imię Nazwisko, Imię Nazwisko i Imię Nazwisko. „Tytuł artykułu w czasopiśmie”. W: *Tytuł czasopisma* 157.8 (2016), s. 1092–1113.
- [4] Imię Nazwisko, Imię Nazwisko i Imię Nazwisko. *Tytuł książki*. Warszawa: Wydawnictwo, 2017. ISBN: 83-204-3229-9-434.

Dodatki

Spis skrótów i symboli

DNA kwas deoksyrybonukleinowy (ang. *deoxyribonucleic acid*)

MVC model – widok – kontroler (ang. *model-view-controller*)

N liczebność zbioru danych

μ stopnień przyleżności do zbioru

\mathbb{E} zbiór krawędzi grafu

\mathcal{L} transformata Laplace’a

Źródła

Jeżeli w pracy konieczne jest umieszczenie długich fragmentów kodu źródłowego, należy je przenieść w to miejsce.

```
1 if (_nClusters < 1)
2     throw std::string ("unknown_number_of_clusters");
3 if (_nIterations < 1 and _epsilon < 0)
4     throw std::string ("You should set a maximal number of
        iteration or minimal difference — epsilon.");
5 if (_nIterations > 0 and _epsilon > 0)
6     throw std::string ("Both number of iterations and minimal
        epsilon set — you should set either number of iterations
        or minimal epsilon.");
```

Lista dodatkowych plików, uzupełniających tekst pracy

W systemie do pracy dołączono dodatkowe pliki zawierające:

- źródła programu,
- dane testowe,
- film pokazujący działanie opracowanego oprogramowania lub zaprojektowanego i wykonanego urządzenia,
- itp.

Spis rysunków

3.1	Diagram przypadków użycia niezalogowanego użytkownika.	7
3.2	Diagram przypadków użycia zalogowanego użytkownika.	8
4.1	Podpis rysunku po rysunkiem.	10
5.1	Pseudokod w <code>listings</code>	11

Spis tabel

6.1	Nagłówek tabeli jest nad tabelą.	14
-----	--	----