POLITECHNIKA WROCŁAWSKA

WYDZIAŁ ELEKTRONIKI

PROJEKT z BAZ DANYCH

System zarządzania warsztatem samochodowym

Termin zajęć: Środa, 9:15–11:00

AUTOR/AUTORZY:

Marcin Bober

Indeks: 249426

E-mail: 249426@student.pwr.edu.pl

Rafał Rzewucki

Indeks: 248926

E-mail: 248926@student.pwr.edu.pl

PROWADZĄCY ZAJĘCIA:

dr inż. Roman Ptak, W4/K9

Wrocław 2020

Spis treści

1. Wstęp

1.1 Cel projektu

Warsztaty samochodowe często nie mogą sobie poradzić z zapisywaniem i zarządzaniem kolejką napraw pojazdów swoich klientów. Klienci warsztatów samochodowych chcieliby wiedzieć, kiedy ich pojazd będzie mógł zostać naprawiony. Celem projektu jest stworzenie systemu, który umożliwi zarządzanie kolejką napraw przez mechaników warsztatu, co będą mogli na bieżąco obserwować klienci warsztatu.

1.2 Zakres projektu

Serwis jest zaprojektowany dla dużego warsztatu samochodowego, aby zautomatyzować część czynności zarówno po stronie klientów warsztatu jak i po stronie mechaników. Został zaprojektowany jako baza do bardziej rozbudowanego serwisu. Będzie on przygotowany do tego, aby w przyszłości zostać rozbudowany o sklep z częściami, jak i o dodatkowy system powiadomień dla użytkowników o działaniach w serwisie.

Będzie on gromadził w bazie danych informacje na temat klientów warsztatu, naprawianych pojazdów oraz usterek, które zostały w nich naprawione. co więcej będzie przechowywany rejestr mechaników oraz zapis wszystkich wydarzeń z systemu. Aby umożliwić korzystanie z systemu, zostanie stworzona aplikacja webowa, która w przejrzysty sposób ułatwi dostęp do danych.

2. Analiza wymagań

2.1 Opis działania i schemat logiczny systemu

Gdy klient wykryje usterkę w swoim pojeździe będzie mógł dodać nową naprawę bezpośrednio do kolejki napraw w warsztacie. Natychmiast po dodaniu nowej naprawy mechanik będzie mógł odpowiednio zmienić status usterki w systemie w zależności od postępu w jej usuwaniu. Klient przez cały czas ma wgląd do tego co aktualnie jest robione w jego sprawie i gdy mechanik zakończy naprawę jego pojazdu, klient będzie wiedział, kiedy może go odebrać.

2.2 Wymagania funkcjonalne

Możliwe będzie logowanie jako klient, mechanik lub jako administrator.

Obserwator ma możliwość do:

- przeglądania oferty warsztatu,
- dostęp do informacji kontaktowych.

Klient ma możliwość:

- dodawania swoich pojazdów do listy pojazdów,
- wprowadzania nowych aut do kolejki napraw,
- przeglądania stanu kolejki napraw,
- przeglądania statusu naprawy swojego pojazdu.

Mechanik ma uprawnienia do:

- organizacja wizyt w warsztacie,
- zarządzanie naprawą.

Administrator ma uprawnienia do:

- przypisywaniem ról do użytkowników,
- zarządzaniem rejestrem użytkowników oraz pojazdów.

2.3 Wymagania niefunkcjonalne

2.3.1 Wykorzystywane technologie i narzędzia

Aplikacja dostępowa zostanie zbudowana w języku wysokiego poziomu jakim jest PHP. Jest to prosty język wysokiego poziomu, który w prosty sposób pozwala zwizualizować dane dla użytkowników systemu oraz umożliwia interakcję z systemem. Dzięki niemu i wbudowanym w niego metodą możliwe jest połączenie z bazą danych oraz wykonywanie do niej zapytań.

2.3.2 Wymagania dotyczące bazy danych

Baza danych musi obsługiwać przechowywanie danych w formie tabel. Aplikacja będzie wymagać minimum 3 tabel o różnych rozmiarach, przy czym chcemy zachować możliwość rozbudowania aplikacji w przyszłości.

Szacunkowa liczba napraw w ciągu roku wynosi około 1000, przy czym musimy założyć, że każda naprawa będzie przeprowadzana na innym pojeździe. w najbardziej rozbudowanym przypadku każdy pojazd będzie miał innego właściciela, toteż zakładamy średnią ilość nowych użytkowników w ciągu roku na poziomie równym 1000. w sumie daje to około 3000 nowych rekordów w bazie danych w ciągu jednego roku. Dla każdego silnika baz danych jest to bardzo mała ilość, dlatego nie determinuje to wyboru silnika bazy.

Bardziej miarodajnym czynnikiem będzie ilość odczytów danych z bazy, ponieważ mechanicy w warsztacie prawdopodobnie będą cały czas zalogowani do systemu i co pewien okres czasu dane wyświetlane w serwisie będą musiały zostać zaktualizowane, tak aby zawsze można mieć dostęp do najnowszych danych.

Częstość wykonywania operacji dla najważniejszych encji:

	Wstawianie	**Modyfikacja**	**Usuwanie**	**Wyszukiwanie**
Użytkownicy	często	rzadko	b. rzadko	często
Adresy	często	rzadko	b. rzadko	często
Kraje	średnio	b. rzadko	b. rzadko	często
Auta	często	rzadko	b. rzadko	często
Kolejka napraw	często	często	b. rzadko	często
Naprawy	często	rzadko	b. rzadko	często
Producenci Aut	b. rzadko	b. rzadko	b. rzadko	często
Modele aut	b. rzadko	b. rzadko	b. rzadko	często
Zapis działań w systemie	b. często	b. rzadko	b. rzadko	często

Przy dodawaniu nowych danych do bazy będzie używany mechanizm transakcji, który umożliwi wycofanie wprowadzonych zmian, jeśli wystąpi jakikolwiek błąd przy dodawaniu. Jako serwer bazy danych został wybrany MariaDB wraz z mechanizmem składowania InnoDB z kilku istotnych powodów:

- jest on w pełni darmowy,
- jego uruchomienie nie wymaga praktycznie żadnej konfiguracji,
- nie ma wygórowanych wymagań sprzętowych,
- przechowuje dane również po zaniku zasilania,
- przez swoją prostotę budowy i działania operacje tak zapisu jak i odczytu z bazy będą wykonywane w bardzo krótkim czasie,
- posiada wbudowaną obsługę transakcji.

2.3.3 Wymagania dotyczące bezpieczeństwa systemu

Dla klientów zostanie zaprojektowany system logowania z użyciem adresów email i haseł. Hasła będą hashowane co zagwarantuje brak możliwości ich późniejszego odczytania nawet przez administratora serwisu.

Połączenia z serwerem będą szyfrowane co stanowi podstawę bezpieczeństwa przesyłania danych między urządzeniem klienta, a serwerem aplikacji.

2.4 Przyjęte założenia projektowe

- dostarczenie przystępnej aplikacji dla warsztatów i ich klientów,
- stworzenie stabilnej i odpornej na błędy aplikacji dostępowej do bazy danych,
- utworzenie aplikacji bazodanowej, która będzie gwarantować spójność oraz bezpieczeństwo danych.

3. Projekt systemu

3.1 Projekt bazy danych

3.1.1 Analiza rzeczywistości i uproszczony model konceptualny

W rzeczywistości występują klienci warsztatu samochodowego, ich pojazdy oraz usterki tych pojazdów. Wymagany jest model, który pozwoli jak najefektywniej przechowywać dane o każdym przedmiocie.

Co więcej ważne jest również, aby mechanik, który rozpoczyna naprawę auta wiedział co wcześniej było zmieniane przy tym aucie, można powiedzieć, że pomocna dla niego będzie historia napraw danego auta.

Administrator serwisu musi mieć możliwość przejrzenia wszystkich zdarzeń jakie zaszły w serwisie, aby być w stanie wykryć ewentualne błędy. Taka historia zdarzeń może być również pomocna, jeśli wystąpią jakiekolwiek inne problemy, w których takie dane mogą być pomocne.

Odzwierciedlenie tej części rzeczywistości w postaci modelu bazy danych zostało przedstawione na modelu logicznym jak i fizycznym bazy danych.

3.1.2 Model logiczny i normalizacja

Baza danych musi przechowywać podstawowe dane o użytkownikach serwisu, ich uprawnieniach do działań w serwisie, adresach zamieszkania oraz dane kontaktowe. Auta dodawane przez klientów powinny być przypisane do danego użytkownika oraz posiadać podstawowe parametry pojazdu.

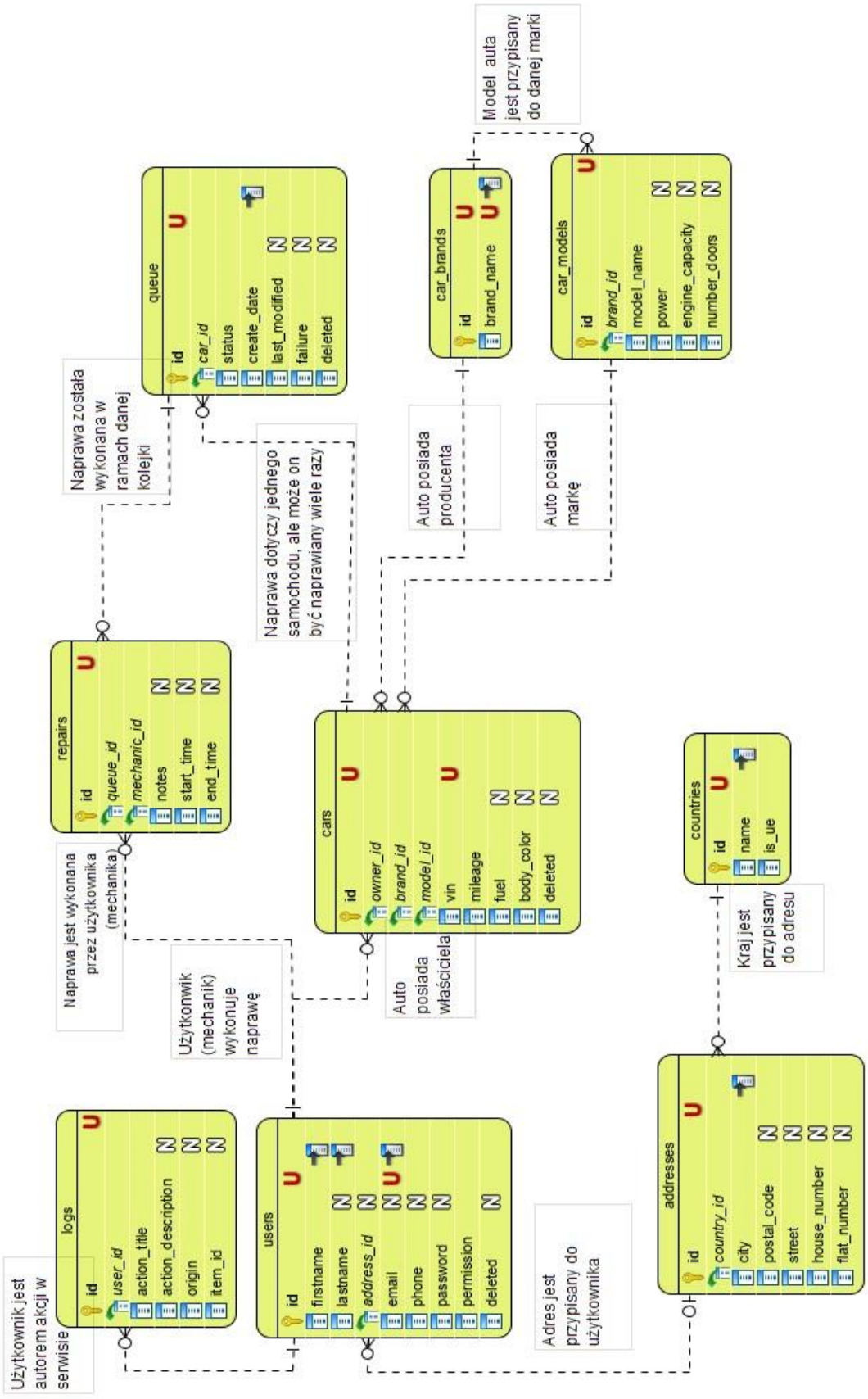
Kolejka napraw musi zawierać dane o tym jaki samochód był, jest lub będzie naprawiany, status naprawy, informację od klienta co się dzieje z autem, datę dodania do kolejki oraz dodatkowe informacje jak notatki od mechanika po zakończeniu naprawy.

Dane adresowe użytkowników mogą zostać zapisane w oddzielnej tabeli, aby można było w prosty sposób usystematyzować dane adresowe. Kraj w adresie może zostać znormalizowany w osobnej tabeli, która dodatkowo będzie zawierać informacje o tym czy dany kraj znajduje się w Unii Europejskiej. Taka informacja może być pomocna dla pracownika wystawiającego fakturę dla klienta jak i dla przyszłych funkcjonalności serwisu, które nie zostały uwzględnione w tym projekcie.

Również dane o producencie jak i modelu auta mogą zostać wyodrębnione i z racji na ich ograniczoną ilość mogą zostać usystematyzowane, aby były jednolite w całym serwisie.

3.1.3 Model fizyczny i ograniczenia integralności danych

Model konceptualny i logiczny bazy danych:

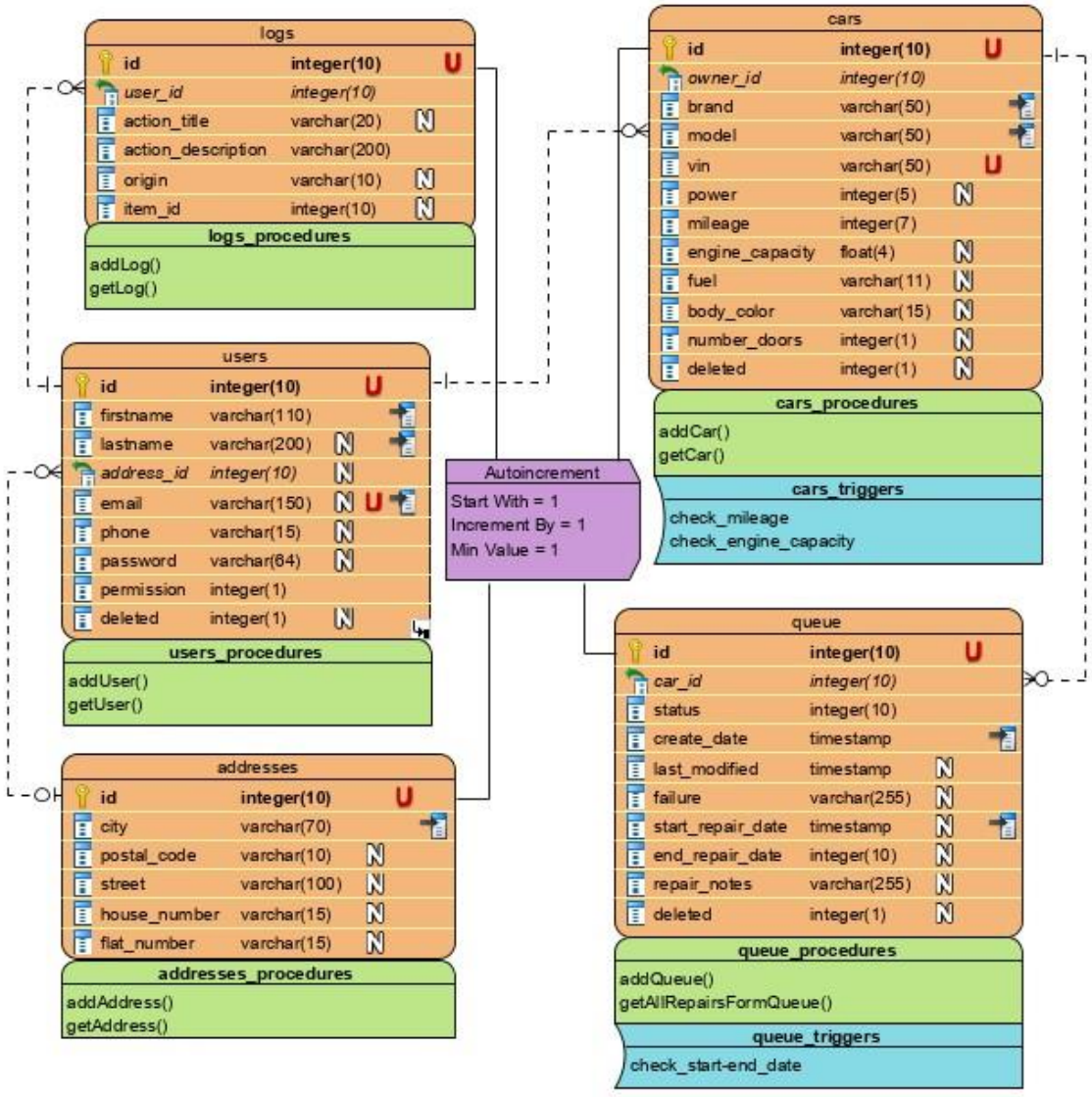


Rysunek 1 Model konceptualny bazy danych

Relacje występujące w bazie danych:

- Relacja jeden do wielu między tabelą *users* a tabelą *cars* obrazuje fakt, że dany użytkownik serwisu jest posiadaczem danego pojazdu, jeden użytkownik może posiadać wiele pojazdów,
- Relacja jeden do wielu między *cars* a *queue* obrazuje sytuację, gdzie jeden samochód może być naprawiany wiele razy,
- Relacja jeden do wielu między *queue* a *repairs* pokazuje, która naprawa została wykonana w ramach danej zaplanowanej naprawy. Może być wiele napraw w ramach jednej kolejki,
- Relacja jeden do wielu między *users* a *repairs* zaznacza który mechanik wykonał daną naprawę,
- Relacja jeden do wielu między *car_brands* oraz *car_models* a *cars* pozwala, aby w bazie danych było wiele aut tego samego producenta, odpowiednio modeli,
- Relacja jeden do wielu między *addresses* a *users* daje możliwość, aby w systemie wiele użytkowników współdzieliło jeden adres.
- Relacja jeden do wielu między *countries* a *addresses* obrazuje sytuację, gdzie istnieje wiele adresów zamieszkania w jednym kraju.

Model fizyczny bazy danych:

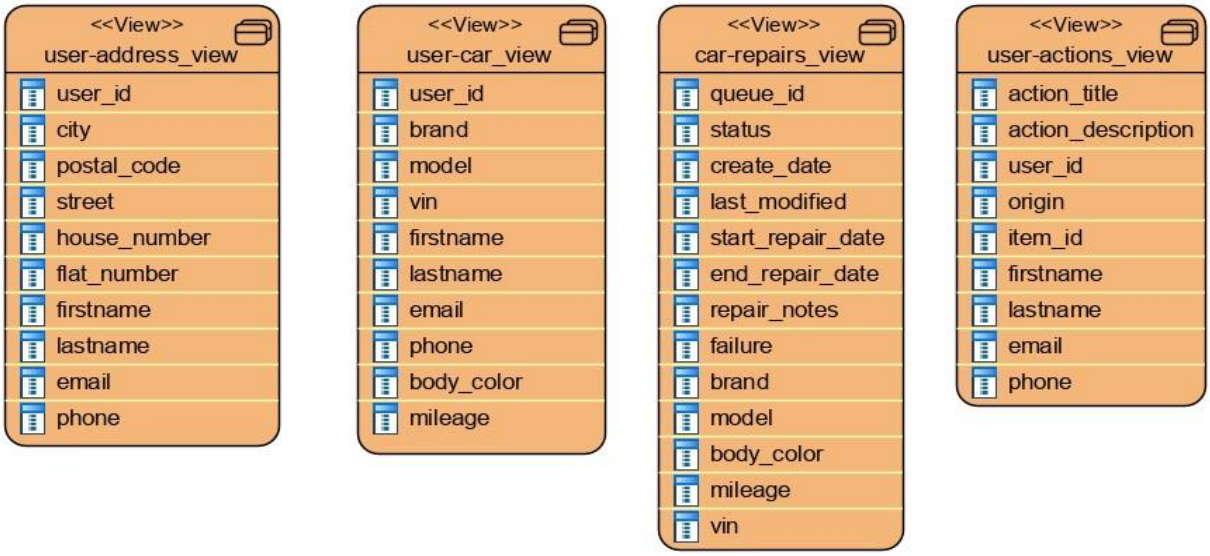


Rysunek 2 Model fizyczny bazy danych

Poczynione uproszczenia i uogólnienia:

- Tabela *cars* mogłaby mieć jako klucz główny kolumnę *vin,* ponieważ jest on niesztucznym kluczem dla każdego rekordu jednak jest on bardzo długi, co mogłoby wpłynąć na wydajność, dlatego zdecydowano się na dodanie sztucznego klucza.
- Tabela zawierająca adresy użytkowników została scalona z tabelą użytkowników. Powodem takiego uproszczenia był fakt, że użytkownik w serwisie zawsze będzie miał podany jeden adres. Osobna tabela adresów byłaby uzasadniona w sytuacji gdy wiele użytkowników współdzieliło by jeden adres, jednak taka sytuacja występowała by szczególnie często tylko w przypadku dużych firm gdzie każdy pracownik firmy byłby osobnym użytkownikiem w serwisie, jednak w rzeczywistości takie firmy mają specjalne osoby oddelegowane do zajmowania się takimi sprawami, co powoduje, że w serwisie nawet dla dużych firm zarejestrowanych pod jednym adresem nie będzie to powodowało nadmiarowości danych w serwisie.
- Kraj z adresu został całkowicie usunięty. Powodem takiej decyzji było założenie, że system będzie obsługiwał klientów głównie z Polski. Nawet jeśli zdarzy się wyjątek użytkownika z poza granic Polski, to jest małe prawdopodobieństwo, że adres z innego państwa będzie miał swój odpowiednik w Polsce
- Producenci aut jak i modele aut zostały zaimplementowane bezpośrednio w tabeli cars. Jest to spowodowane faktem, że rozbicie to na osobne tabele wymagało by moderacji za każdym razem, gdy na rynku pojawi się nowy model pojazdu. Stwarzało by to dodatkowe obowiązki dla administratora systemu oraz możliwe okresowe problemy dla klientów. Takie rozwiązanie pozwoli klientom wpisać markę i model pojazdu bezpośrednio do bazy danych. Co więcej chcemy, aby użytkownicy mieli swobodę w dodawaniu swoich aut, które mogły ulec modyfikacją po opuszczeniu fabryki.
- Tabela z danymi poszczególnych napraw została również uproszczona do kilku istotnych pól w tabeli kolejki napraw. Dawała ona jedynie dwie dodatkowe możliwości, które w tym serwisie są zbędne. Można było z niej odczytać, który mechanik wykonywał naprawę, jednak taką informację można również uzyskać z zapisu zdarzeń w systemie. Można było mieć dwie naprawy w jednej kolejce napraw, jednak taka sytuacja występowała by niezwykle rzadko, ponadto można ją uprościć do tego, że notatka po naprawie w kolejce napraw będzie odpowiednio dłuższa.

Projekt widoków w fizycznej bazie danych:



- user-address_view – służy do wygodnego pobierania danych użytkownika wraz z danymi adresowymi,
- user-car_view – systematyzuje wybieranie danych aut danego użytkownika,
- car-repairs_view – pobieranie kolejki napraw wraz z danymi samochodu,
- user-actions_view – umożliwia proste pobieranie danych o działaniach użytkownika w systemie.

3.1.4 Inne elementy schematu – mechanizmy przetwarzania danych

W tabeli *users* zostanie utworzony index na kolumnach *firstname*, *lastname* oraz *email* zostanie utworzony index, który usprawni wyszukiwanie użytkowników.

Indeks zostanie również ustanowiony w tabeli *queue* odpowiadającej kolejce napraw na kolumnie *create_date*. Pomoże on sortować dane w zależności od daty dodania do kolejki.

Opis wyzwalaczy zaprojektowanych w fizycznym modelu bazy danych:

- check_mileage – uruchamiany przed modyfikacją rekordu. Sprawdza, czy nowo wprowadzony przebieg jest większy od poprzedniego
- check_engine_capacity - uruchamiany przed dodaniem i modyfikacją rekordu. Sprawdza, czy pojemność silnika jest większa od zera,
- check_start-end_date - uruchamiany przed modyfikacją rekordu. Sprawdza, czy data zakończenia naprawy jest późniejsza niż data rozpoczęcia.

3.1.5 Projekt mechanizmów bezpieczeństwa na poziomie bazy danych

Zabezpieczenie przed nieuprawnionym dostępem będzie się odbywać poprzez zabezpieczenie bazy danych hasłem, które będzie przechowywane w formie zaszyfrowanej. Dostęp do hasła jak i do bazy danych będzie miał administrator, właściciel bazy danych oraz osoby wskazane przez właściciela.

Zabezpieczenie przed utratą danych będzie zapewnione poprzez wykonywanie cyklicznych kopii zapasowych całej bazy danych. Po uruchomieniu usługi w warsztacie można również rozszerzyć bezpieczeństwo poprzez przechowywanie dodatkowej bazy danych w chmurze jako kopię tylko do odczytu, która również może umożliwić odzyskać dane w przypadku awarii serwera umieszczonego w warsztacie.

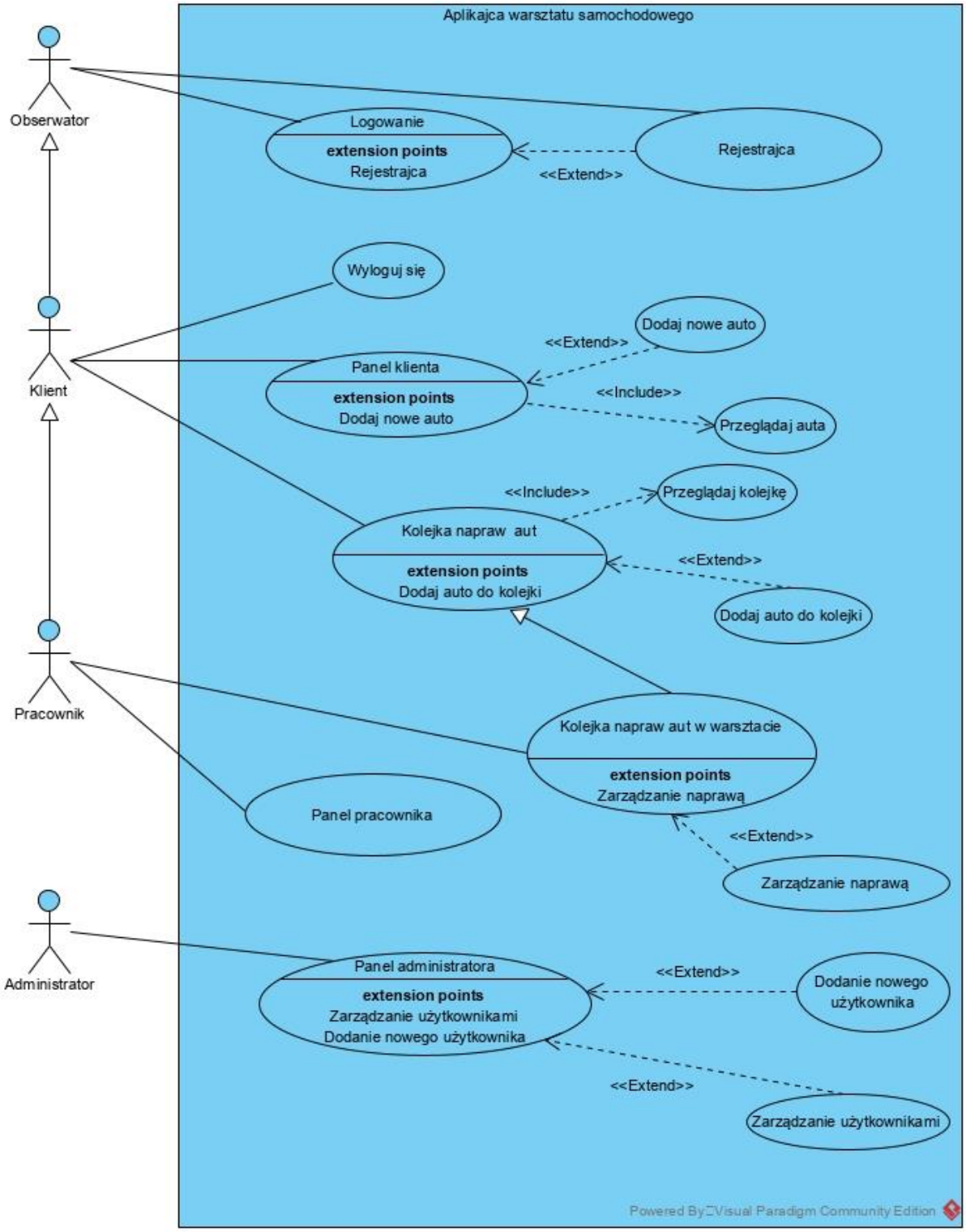
Aby serwis mógł współpracować z bazą danych wymagane jest utworzenie dwóch użytkowników. Użytkownik zwykły oraz użytkownik przeznaczony dla administratora, z rozszerzonymi uprawnieniami. Projekt uprawnień znajduje się poniżej.

	Wstawianie		**Modyfikacja**		**Usuwanie**		**Wyszukiwanie**	
Użytkownik	Zwykły	Admin	Zwykły	Admin	Zwykły	Admin	Zwykły	Admin
users	Tak	Tak	Tak	Tak	Nie	Tak	Tak	Tak
addresses	Tak	Tak	Tak	Tak	Nie	Tak	Tak	Tak
cars	Tak	Tak	Tak	Tak	Nie	Tak	Tak	Tak
queue	Tak	Tak	Tak	Tak	Nie	Tak	Tak	Tak
logs	Tak	Tak	Nie	Tak	Nie	Tak	Nie	Tak

3.2 Projekt aplikacji użytkownika

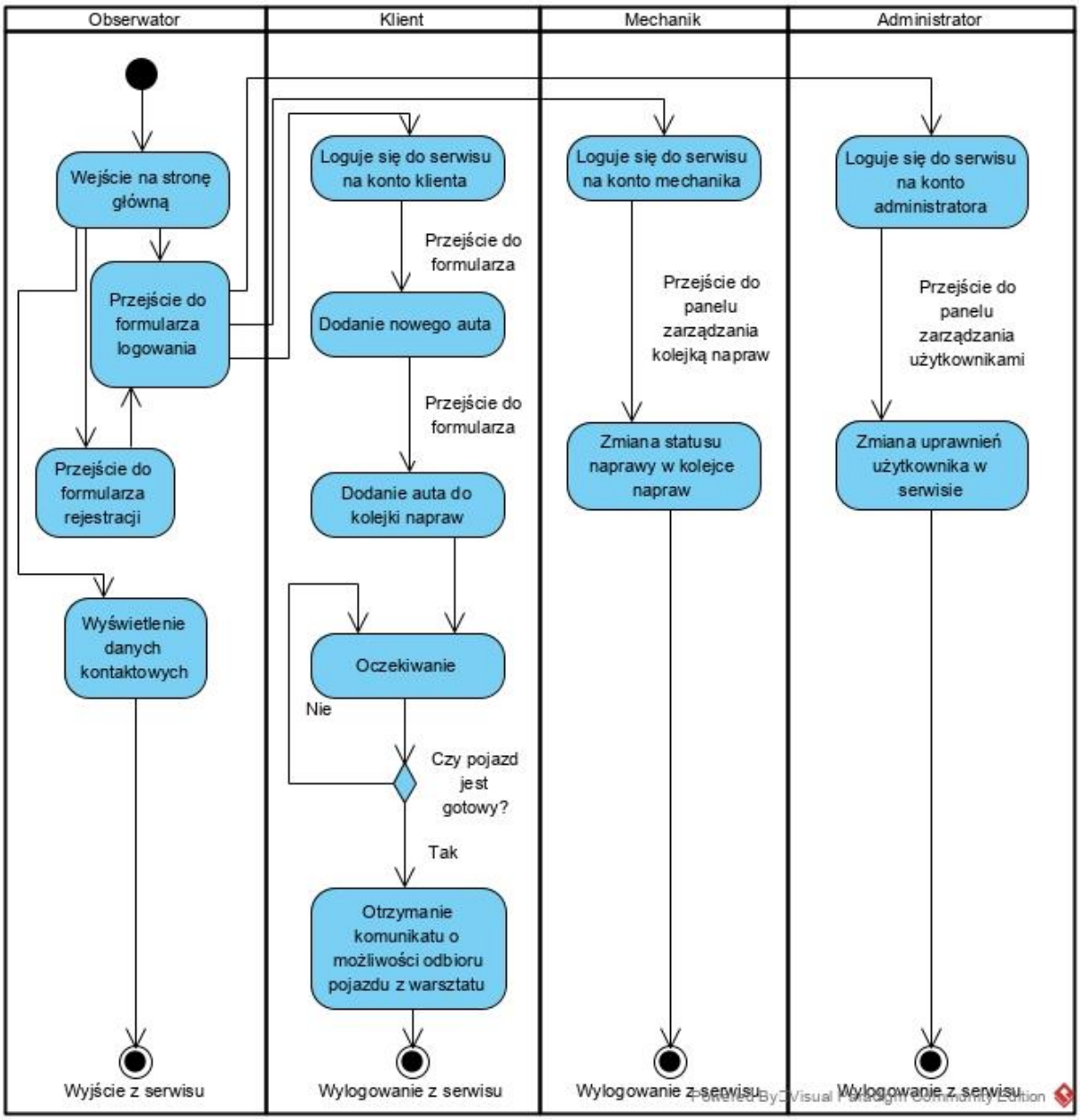
3.2.1 Architektura aplikacji i diagramy projektowe

Diagram przypadków użycia:



Rysunek 3 Diagram przypadków użycia

Diagram czynności:



Rysunek 4 Diagram czynności

3.2.2 Interfejs graficzny i struktura menu

Jako że jesteśmy w posiadaniu prototypu aplikacji dostępowej, zdecydowaliśmy się nie umieszczać makiet.

Użytkownik ma możliwość przejścia w następujące miejsca w serwisie:

- Strona główna – wyświetla stronę główną serwisu,
- Kolejka napraw – wyświetla aktualną kolejkę napraw w warsztacie
- Panel Klienta – przechodzi do panelu klienta:
    - Dodaj auto – klient ma możliwość dodać nowe auto do serwisu,
    - Dodaj auto do kolejki – w tym miejscu klient może dodać swój pojazd do kolejki napraw.

- Kontakt – przenosi na stronę, na której znajdują się dane kontaktowe,
- Zaloguj – przenosi do formularza umożliwiającego zalogowanie,
- Wyloguj – wylogowanie użytkownika z serwisu.

3.2.3 Projekt wybranych funkcji systemu

Najważniejsze funkcjonalności serwisu w formie nagłówków metod, które je będą realizować:

- **pobierzUzytkownikow**() – zwraca dane wszystkich użytkowników zarejestrowanych w serwisie,
- **pobierzUzytkownika**(email) – zwraca dane użytkownika o podanym adresie e-mail
- **zarejestrujUzytkownika**(email, haslo, imie, nazwisko) – tworzy nowego użytkownika w serwisie,
- **zalogujUzytkownika**(id_uzytkownika) – loguje użytkownika do serwisu
- **dodajAuto**(id_wlasciciela, marka, model, vin, przebieg) – dodaje auto o podanych danych i przypisuje je do użytkownika,
- **pobierzKolejke**() – pobierz aktualną kolejkę napraw,
- **dodajDoKolejki**(id_auta, usterka) – dodaje auto do kolejki napraw,
- **pobierzAuto**(id_auta) – pobiera informacje o aucie,

Funkcjonalności istotne z punktu widzenia mechaników warsztatu:

- **zmienStatusNaprawy**(id_naprawy) – zmienia status podanej naprawy odpowiednio na przyjęto do realizacji, w trakcie naprawy i do odbioru,
- **zakonczNaprawe**(id_naprawy) – kończy realizację naprawy, mechanik ma możliwość wprowadzić informacje o tym co zostało zrobione oraz kwotę jaką klient musi uiścić za naprawę.

Funkcjonalności istotne z punktu widzenia administratora serwisu:

- **zmienUprawnieniaUzytkownika**(id_uzytkownika) – zmienia uprawnienia użytkownika odpowiednio klient, mechanik, administrator.
- **pobierzZdarzeniaSerwisu**() – umożliwia pobranie wszystkich zdarzeń z serwisu, aby można było je wyświetlić administratorowi.

3.2.4 Metoda podłączania do bazy danych – integracja z bazą danych

Połączenie z bazą danych będzie odbywać się poprzez rozszerzenie języka PHP o nazwie PDO. Zapewnia ono powłokę, którą można użyć, aby komunikować się z bazą danych i wykonywać wszelkie zapytania. Za pomocą tego rozszerzenia możliwa jest również pełna obsługa transakcji bazy danych.

3.2.5 Projekt zabezpieczeń na poziomie aplikacji

Każdy użytkownik będzie miał przypisany indywidualny zestaw adresu e-mail i hasła, dzięki którym możliwe będzie zalogowanie do serwisu. Dodatkowo aplikacja będzie monitorowała zmianę adresu IP zalogowanego użytkownika, aby wykryć przechwycenie sesji przez innego użytkownika

Program przed dodaniem nowego użytkownika do bazy, jego email zostanie sprawdzony funkcje „filter_input” z filtrem „FILTER_VALIDATE_EMAIL”, co więcej hasło zostanie zaszyfrowane poprzez użycie wbudowanej w język PHP funkcji password_hash(). Domyślnie używa ona algorytmu bcrypt, który gwarantuje optymalną złożoność szyfrowania.