Politechnika Wrocławska

Etap 2: System do zarządzania działalnością salonu samochodowego

---

Sprawozdanie

Prowadzący:
dr inż. Bogumiła Hnatkowska

Wykonali:
Jakub Staniszewski 266876
Kamil Wojcieszak 264487
Kasjan Kardaś 263505

Wrocław, 18 Grudzień 2025r.

Spis treści

1) Cel

Dokument przedstawia decyzje i ich uzasadnienie oraz ograniczenia i ważne elementy projektu systemu rozwiązania, które wpływają na jego implementację.

2) Cele i ograniczenia architektoniczne

Wyróżniono zostały następujące cele, które system powinien spełniać:

2.1) Funkcjonalne:

- Sklep internetowy salonu
  - Użytkownik ma możliwość przeglądania katalogu części samochodowych w sklepie internetowym salonu
  - Użytkownik ma możliwość dodawania i usuwania produktów z koszyka w sklepie internetowym salonu
  - Użytkownik ma możliwość wystawiania opinii produktom w sklepie internetowym salonu
  - Użytkownik ma możliwość zakupu produktów (części) w sklepie internetowym salonu
- Salon samochodowy
  - Użytkownik ma możliwość rezerwacji wizyty w salonie samochodowym (wraz z wyborem rodzaju wizyty [jazda próbna, konsultacja, zakup], wybór terminu wizyty spośród dostępnych)
  - Użytkownik ma możliwość przeglądania oferty salonu
- Serwis samochodowy salonu
  - Użytkownik ma możliwość rezerwacji wizyty w serwisie salonu (wraz z wyborem rodzaju wizyty [przegląd, naprawa, wymiana opon, wymiana oleju], opisaniem przyczyny wizyty, podaniem szczegółów technicznych samochodu)
  - Użytkownik ma możliwość przeglądania cennika usług w serwisie samochodowym

2.2) Niefunkcjonalne:

- Przechowywane hasła muszą być bezpieczne
- System powinien zapewniać bezpieczne przesyłanie informacji
- Aplikacja powinna poprawnie działać na urządzeniach mobilnych
- System powinien obsługiwać dwie wersje językowe

3) Decyzje i ich uzasadnienie

Cel	Sposób osiągnięcia (Taktyki)
Przechowywane hasła muszą być bezpieczne	A. Haszowanie haseł
System powinien zapewniać bezpieczną komunikację pomiędzy klientem a serwerem	A. Szyfrowanie komunikacji z użyciem protokołu HTTPS
Aplikacja powinna poprawnie działać na urządzeniach mobilnych	A. Projektowanie responsywne interfejsu użytkownika
System powinien obsługiwać dwie wersje językowe	A. Pliki słowników językowych

Tabela 1: Decyzje architektoniczne i ich uzasadnienie

4) Mechanizmy architektoniczne

Poniżej przedstawiono mechanizmy architektoniczne wspierające wyżej wymienione taktyki.

4.1) Haszowanie haseł

Kontekst: System przechowuje dane uwierzytelniające użytkowników, które należą do danych wrażliwych i muszą być odpowiednio zabezpieczone przed nieautoryzowanym dostępem, w szczególności w przypadku wycieku bazy danych.

Decyzja: Zastosowano mechanizm haszowania haseł użytkowników przy użyciu bezpiecznych algorytmów kryptograficznych, takich jak bcrypt lub Argon2. Hasła nie są przechowywane w postaci jawnej.

Uzasadnienie: Algorytmy bcrypt i Argon2 są odporne na ataki brute-force dzięki wykorzystaniu mechanizmów solenia oraz regulowanej złożoności obliczeniowej. Ich zastosowanie znacząco zwiększa poziom bezpieczeństwa systemu w porównaniu do prostych funkcji skrótu (np. SHA-256).

Konsekwencje: Zwiększenie bezpieczeństwa danych użytkowników kosztem niewielkiego wzrostu czasu potrzebnego na proces uwierzytelniania, co jest akceptowalne z punktu widzenia wymagań niefunkcjonalnych systemu.

4.2) Szyfrowanie komunikacji z użyciem protokołu HTTPS

Kontekst: System internetowy umożliwia przesyłanie danych użytkowników pomiędzy klientem a serwerem, w tym danych logowania oraz innych informacji wrażliwych.

Decyzja: Zdecydowano się na realizację całej komunikacji sieciowej z wykorzystaniem protokołu HTTPS, opartego na protokole TLS.

Uzasadnienie: HTTPS zapewnia poufność, integralność oraz autentyczność przesyłanych danych.

Konsekwencje: Konieczność konfiguracji certyfikatów TLS oraz niewielki narzut wydajnościowy, który jest pomijalny w stosunku do uzyskanych korzyści bezpieczeństwa.

4.3) Projektowanie responsywne interfejsu użytkownika

Kontekst: System ma być dostępny dla użytkowników korzystających z różnych urządzeń, w tym komputerów stacjonarnych, tabletów oraz smartfonów o zróżnicowanych rozdzielczościach ekranów.

Decyzja: Interfejs użytkownika został zaprojektowany zgodnie z zasadami Responsive Web Design (RWD), z wykorzystaniem technologii CSS Flexbox, CSS Grid oraz media queries.

Uzasadnienie: Zastosowanie RWD umożliwia utrzymanie jednej wersji interfejsu, która automatycznie dostosowuje się do rozmiaru ekranu. Ogranicza to koszty utrzymania aplikacji oraz poprawia użyteczność systemu na urządzeniach mobilnych.

Konsekwencje: Większa złożoność arkuszy stylów, jednak poprawa dostępności i komfortu użytkowania systemu na różnych platformach.

4.4) Pliki słowników językowych

Kontekst: System przewiduje obsługę dwóch wersji językowych interfejsu użytkownika.

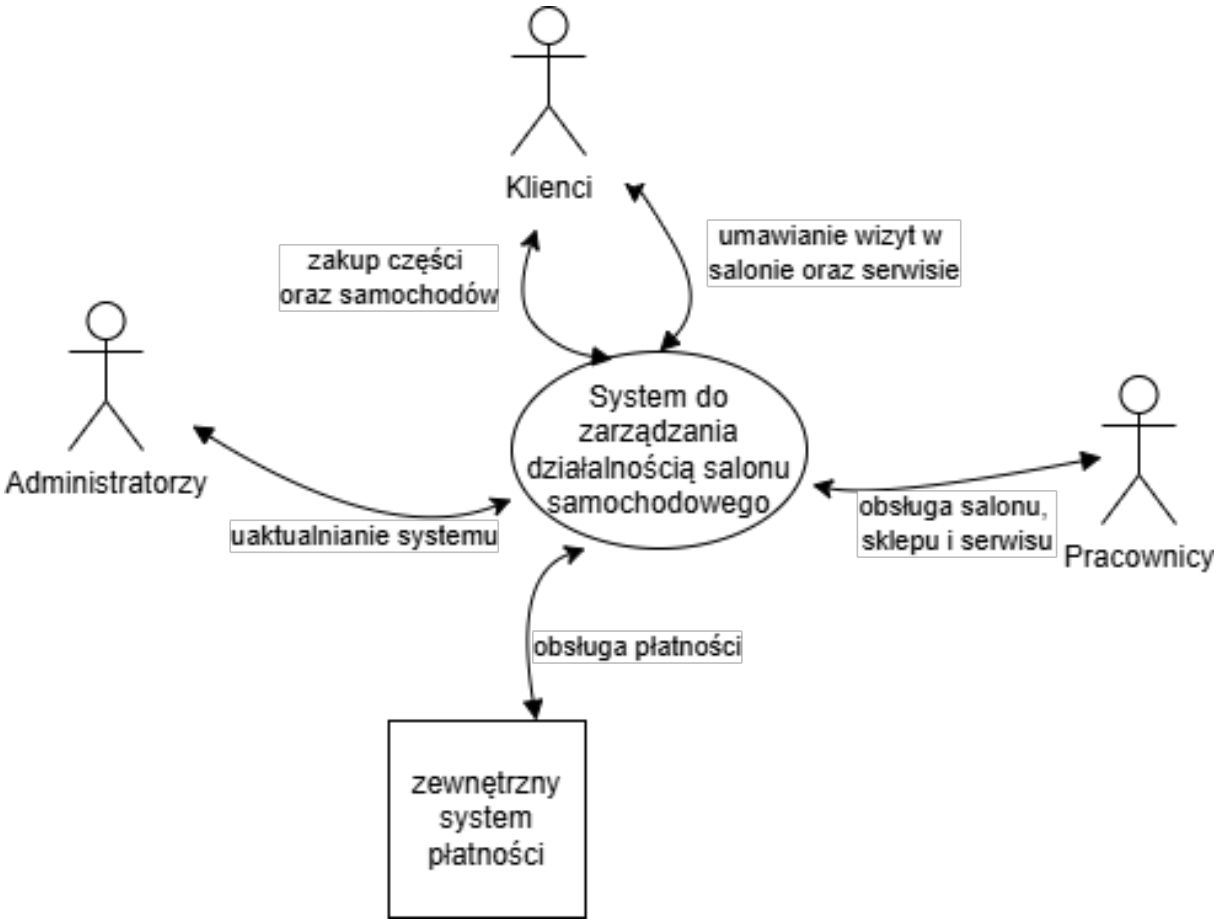
Decyzja: Zastosowano zewnętrzne pliki słowników językowych, osobne dla każdej wersji językowej, zawierające wszystkie etykiety i komunikaty interfejsu użytkownika.

Uzasadnienie: Oddzielenie warstwy językowej od logiki aplikacji umożliwia łatwe dodawanie nowych wersji językowych bez ingerencji w kod źródłowy systemu. Rozwiązanie to zwiększa elastyczność oraz ułatwia utrzymanie aplikacji.

Konsekwencje: Konieczność zarządzania dodatkowymi plikami konfiguracyjnymi, jednak znaczne uproszczenie procesu internacjonalizacji systemu.

5) Widoki architektoniczne

5.1) Widok kontekstowy



Rysunek 1: Diagram kontekstowy systemu

5.2) Scenariusze interakcji

Do realizacji wybrano zakup części i samochodów oraz umawianie wizyt w salonie i serwisie realizowane przez system do zarządzania działalnością salonu samochodowego. Interakcje składają się z żądań HTTP wysyłanych przez Klienta do systemu.

5.3) Interfejsy integracyjne - poziom logiczny

5.3.1) Interfejs 1: System do zarządzania działalnością salonu samochodowego - Zewnętrzny serwis płatności (poza zakresem wymagań do implementacji)

Opis	Zewnętrzna usługa obsługi płatności oraz stanu konta użytkownika. Na tę usługę kierowana jest obsługa płatności w systemie	
Status	Planowany	
	Aplikacja źródłowa	Aplikacja docelowa
Nazwa aplikacji	Sklep salonu	Zewnętrzna bramka płatności np. PayU, Przelewy24, Tpay
Technika integracji	HTTPS	HTTPS
Mechanizm autentykacji	nagłówek HTTP	nagłówek HTTP
Kontrakt danych	Identyfikator użytkownika, identyfikator zamówienia, kwota i waluta płatności, identyfikator płatności, identyfikator transakcji, informacje o błędach, stan konta użytkownika	
Czy interfejs manipuluje na danych wrażliwych (RODO)?	Nie, identyfikator nie jest objęty RODO.	
Strona inicjująca	Sklep salonu	
Model komunikacji	Synchronicznie na żądanie użytkownika	
Wydajność	Zależne od liczby użytkowników systemu, szacowane około 1000 / godz.	
Wolumetria	Szacowana ilość wywołań w jednostce czasu znacząco dłuższej niż w określeniu wydajności. Potrzebne do szacowania np. przestrzeni dyskowej niezbędnej do obsługi interfejsu.	
Wymagana dostępność	99,9%	

Tabela 2: Interfejs integracyjny - System z zewnętrzną bramką płatności

5.3.2) Interfejs 2: Klient - System do zarządzania działalnością salonu samochodowego

Opis	Wewnętrzna usługa obsługi zakupu części i samochodów przez użytkownika	
Status	Planowany	
	Aplikacja źródłowa	Aplikacja docelowa
Nazwa aplikacji	Sklep salonu	Sklep salonu
Technika integracji	HTTPS	HTTPS
Mechanizm autentykacji	nagłówek HTTP	nagłówek HTTP
Kontrakt danych	Identyfikator użytkownika, identyfikator zamówienia, kwota płatności, identyfikator płatności, identyfikator transakcji, informacje o błędach	
Czy interfejs manipuluje na danych wrażliwych (RODO)?	Nie, identyfikator nie jest objęty RODO.	
Strona inicjująca	Klient	
Model komunikacji	Synchronicznie na żądanie użytkownika	
Wydajność	Zależne od liczby użytkowników systemu, szacowane około 1000 / godz.	
Wolumetria	Ok 100 / godzinę	
Wymagana dostępność	99,9%	

Tabela 3: Interfejs integracyjny - Klient ze Sklepem salonu (zakup części i samochodów)

5.3.3) Interfejs 3: Klient - System do zarządzania działalnością salonu samochodowego

Opis	Wewnętrzna usługa obsługi umawiania wizyt w salonie przez użytkownika	
Status	Planowany	
	Aplikacja źródłowa	Aplikacja docelowa
Nazwa aplikacji	Strona salonu	Strona salonu
Technika integracji	HTTPS	HTTPS
Mechanizm autentykacji	nagłówek HTTP	nagłówek HTTP
Kontrakt danych	Identyfikator użytkownika, identyfikator wizyty, identyfikator samochodu, identyfikator pracownika, rodzaj wizyty, data wizyty, status wizyty	
Czy interfejs manipuluje na danych wrażliwych (RODO)?	Nie, identyfikator nie jest objęty RODO.	
Strona inicjująca	Klient	
Model komunikacji	Synchronicznie na żądanie użytkownika	
Wydajność	Zależne od liczby użytkowników systemu, szacowane około 1000 / godz.	
Wolumetria	Ok 15 / godzine	
Wymagana dostępność	70%	

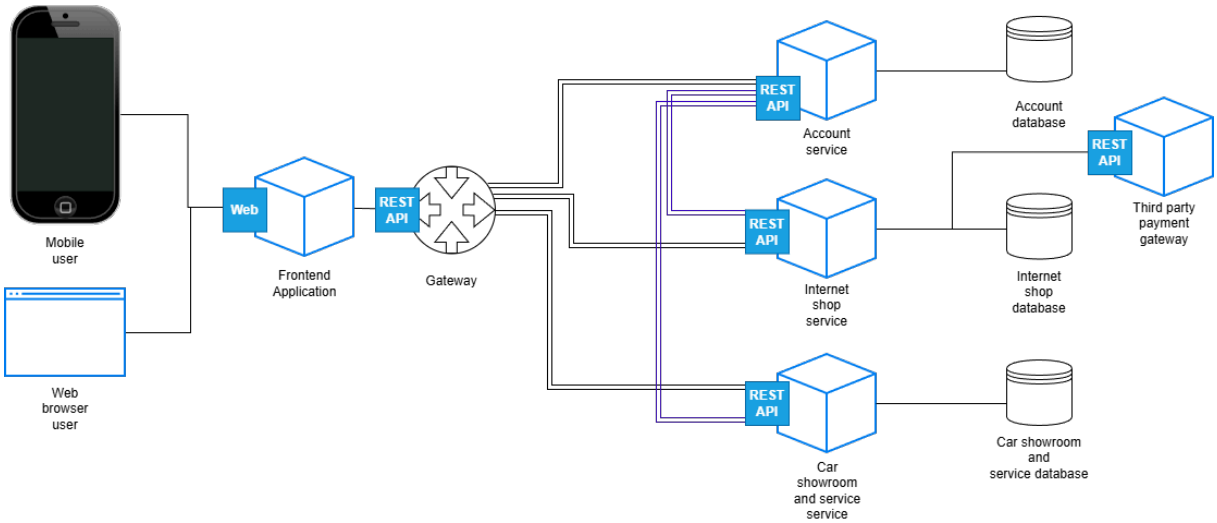
Tabela 4: Interfejs integracyjny - Klient ze Stroną salonu (wizyty w salonie)

5.3.4) Interfejs 4: Klient - System do zarządzania działalnością salonu samochodowego

Opis	Wewnętrzna usługa obsługi umawiania wizyt w serwisie przez użytkownika	
Status	Planowany	
	Aplikacja źródłowa	Aplikacja docelowa
Nazwa aplikacji	Strona serwisu	Strona serwisu
Technika integracji	HTTPS	HTTPS
Mechanizm autentykacji	nagłówek HTTP	nagłówek HTTP
Kontrakt danych	Identyfikator użytkownika, identyfikator wizyty, identyfikator pracownika, rodzaj wizyty, data wizyty, status wizyty, cena usługi, nazwa usługi, opis usługi, marka samochodu, model samochodu, rocznik samochodu, opis problemu	
Czy interfejs manipuluje na danych wrażliwych (RODO)?	Nie, identyfikator nie jest objęty RODO.	
Strona inicjująca	Sklep salonu	
Model komunikacji	Synchronicznie na żądanie użytkownika	
Wydajność	Zależne od liczby użytkowników systemu, szacowane około 1000 / godz.	
Wolumetria	Ok 3 / godzinę	
Wymagana dostępność	70%	

Tabela 5: Interfejs integracyjny - Klient ze Stroną serwisu (wizyty w serwisie)

6) Widok funkcjonalny



Rysunek 2: Diagram komponentów

Niebieskie połączenia na powyższym diagramie symbolizują komunikację między Account Service, a pozostałymi serwisami.

Rola serwisów:

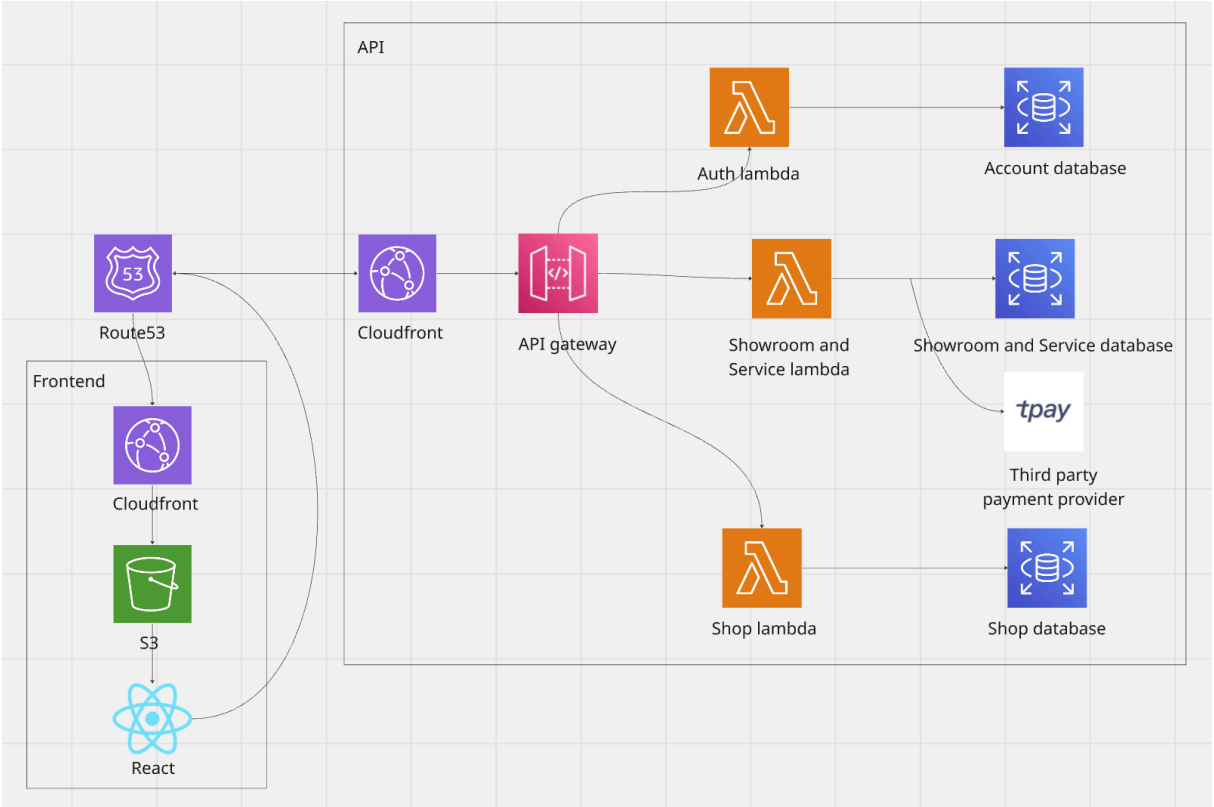
- Internet shop service (serwis sklepu internetowego)
  - możliwość przeglądania katalogu części samochodowych w sklepie internetowym salonu
  - możliwość dodawania i usuwania produktów z koszyka w sklepie internetowym salonu

- możliwość wystawiania opinii produktom w sklepie internetowym salonu
- możliwość zakupu produktów (części) w sklepie internetowym salonu

- Car showroom and service service (serwis salonu i serwisu samochodowego)
  - możliwość rezerwacji wizyty w salonie samochodowym (wraz z wyborem rodzaju wizyty [jazda próbna, konsultacja, zakup], wybór terminu wizyty spośród dostępnych)
  - możliwość przeglądania oferty salonu
  - możliwość rezerwacji wizyty w serwisie salonu (wraz z wyborem rodzaju wizyty [przegląd, naprawa, wymiana opon, wymiana oleju], opisaniem przyczyny wizyty, podaniem szczegółów technicznych samochodu)
  - możliwość przeglądania cennika usług w serwisie samochodowym
- Account service (serwis kont użytkowników)
  - możliwość rejestracji i logowania się do systemu

7) Widok rozmieszczenia

7.1) Diagram rozmieszczenia



Rysunek 3: Diagram rozmieszczenia systemu

7.2) Opis architektury systemu

System zbudowany jest w oparciu o architekturę mikroserwisową, wykorzystującą usługi AWS. Całość została podzielona na dwa główne obszary: Frontend oraz API.

7.2.1) Warstwa Frontend

Warstwa Frontend obejmuje:

- **Route 53** - usługa DNS AWS, która kieruje ruch użytkowników do odpowiednich zasobów
- **Frontend CloudFront** - sieć CDN (Content Delivery Network) zapewniająca szybki dostęp do zasobów statycznych aplikacji frontendowej

- **Bucket** - kontener S3 przechowujący statyczne pliki aplikacji frontendowej (HTML, CSS, JavaScript)
- **UI** - interfejs użytkownika zbudowany w technologii React

Warstwa ta jest odpowiedzialna za prezentację danych użytkownikowi końcowemu oraz obsługę interakcji.

7.2.2) Warstwa API

Warstwa API składa się z następujących komponentów:

- **API CloudFront** - sieć CDN cachująca odpowiedzi API i zmniejszająca opóźnienia
- **API Gateway** - centralny punkt wejściowy do wszystkich mikroserwisów, zarządzający routingiem żądań, autoryzacją oraz ograniczeniami ruchu

7.2.2.1) Mikroserwisy (Lambda Functions)

System wykorzystuje funkcje Lambda, które realizują różne funkcjonalności biznesowe:

- **Auth Lambda** - obsługa uwierzytelniania i autoryzacji użytkowników, połączona z bazą danych Account Database
- **Showroom and Service Lambda** - zarządzanie ekspozycją pojazdów, prezentacją oferty oraz usługami serwisowymi i naprawami, współdzieląca bazę danych Showroom and Service Database
- **Shop Lambda** - obsługa procesów związanych ze sprzedażą i transakcjami, z dostępem do Shop Database oraz integracja z zewnętrznym dostawcą płatności (tpay)

7.2.3) Warstwa danych

System wykorzystuje rozdzielone bazy danych dla poszczególnych obszarów funkcjonalnych:

- **Account Database** - przechowuje dane użytkowników, informacje o kontach i uprawnieniach. Dedykowana dla funkcji Auth Lambda
- **Showroom and Service Database** - wspólna baza danych dla funkcjonalności związanych z wystawą pojazdów oraz serwisem. Zawiera informacje o pojazdach w showroomie, historię serwisową, zlecenia napraw i szczegóły usług
- **Shop Database** - przechowuje dane transakcyjne, zamówienia, koszyki zakupowe oraz historię zakupów klientów

7.2.3.1) Integracje zewnętrzne

- **Third Party Payment Provider (tpay)** - zewnętrzny system płatności zintegrowany z Shop Lambda, obsługujący transakcje finansowe, weryfikację płatności oraz zwroty

7.2.4) Komunikacja między komponentami

System charakteryzuje się następującym przepływem danych:

1. Użytkownik łączy się przez Route 53, który kieruje ruch do Frontend CloudFront
2. Frontend CloudFront pobiera zasoby statyczne z Bucket S3 i dostarcza je do przeglądarki użytkownika
3. UI (React) komunikuje się z API poprzez API CloudFront i API Gateway
4. API Gateway kieruje żądania do odpowiednich funkcji Lambda w zależności od typu operacji
5. Funkcje Lambda przetwarzają żądania, komunikując się z dedykowanymi bazami danych:
   - Auth Lambda z Account Database
   - Showroom and Service Lambda z Showroom and Service Database
   - Shop Lambda z Shop Database oraz zewnętrznym systemem płatności tpay
6. Wyniki przetwarzania są zwracane przez API Gateway do warstwy frontendowej

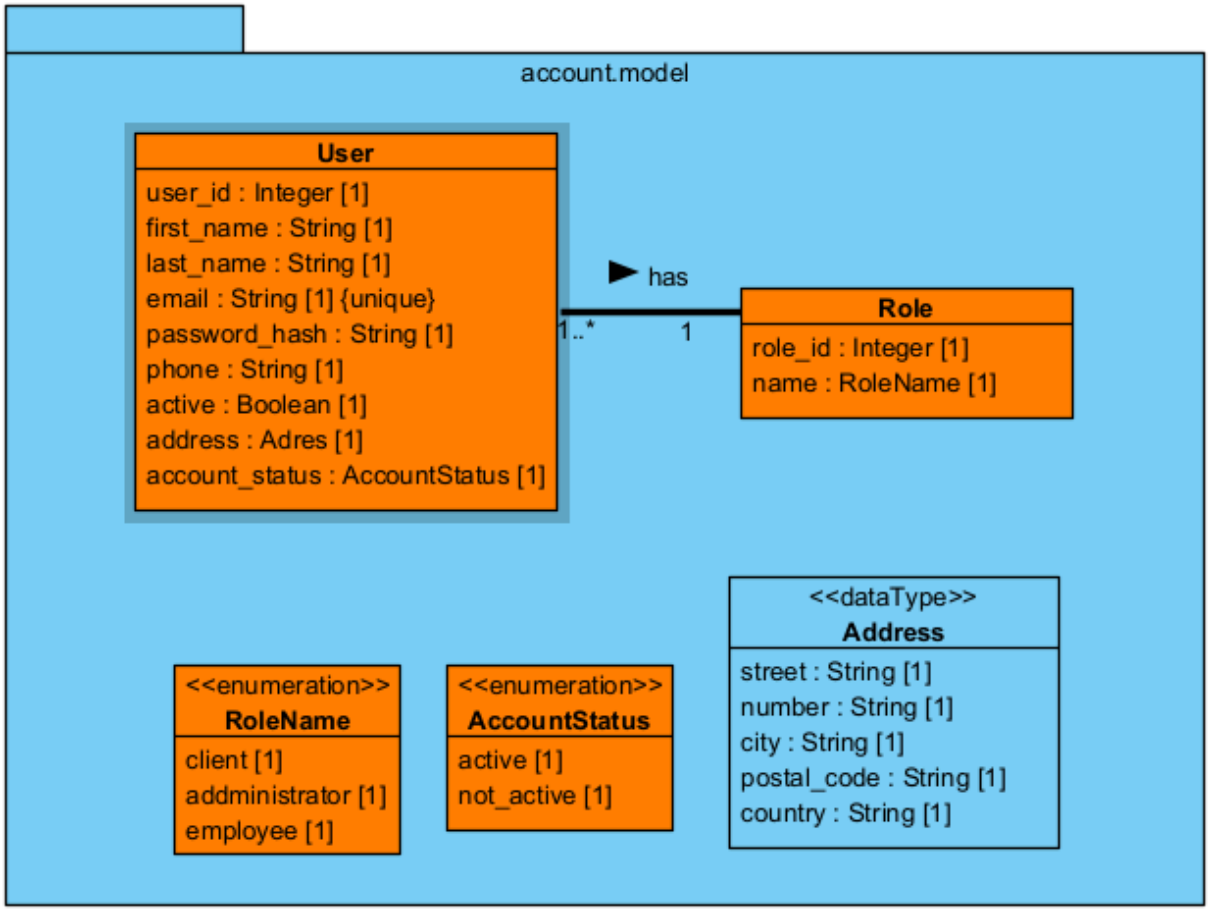
7.2.5) Zalety przyjętego rozwiązania, rozdzielone bazy danych ograniczają ryzyko nieautoryzowanego dostępu

- **Optymalizacja kosztów** - model płatności za faktyczne użycie zasobów (pay-as-you-go)
- **Separacja odpowiedzialności** - każdy mikroserwis realizuje dedykowaną funkcjonalność biznesową z własną bazą danych
- **Izolacja danych** - niezależne bazy danych dla różnych obszarów funkcjonalnych zwiększają bezpieczeństwo i ułatwiają zarządzanie
- **Elastyczność integracji** - możliwość łatwej wymiany dostawcy płatności dzięki luźnemu powiązaniu z systemem zewnętrznymci od obciążenia
- **Wydajność** - użycie CloudFront w obu warstwach minimalizuje opóźnienia
- **Bezpieczeństwo** - API Gateway zapewnia centralne zarządzanie autoryzacją i dostępem
- **Optymalizacja kosztów** - model płatności za faktyczne użycie zasobów (pay-as-you-go)
- **Separacja odpowiedzialności** - każdy mikroserwis realizuje dedykowaną funkcjonalność biznesową

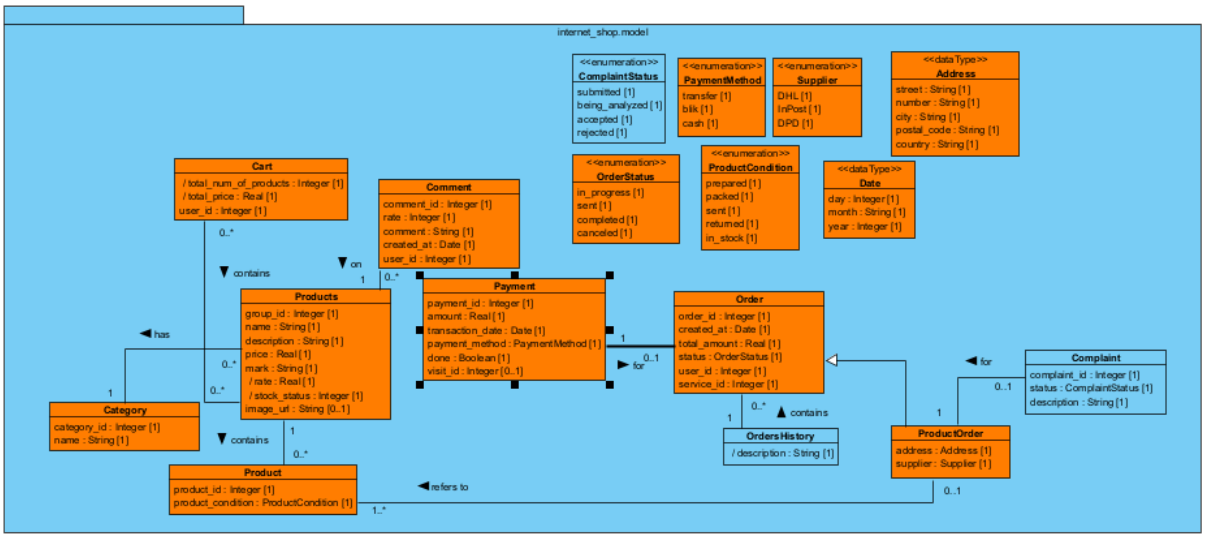
8) Widok informacyjny

8.1) Model informacyjny

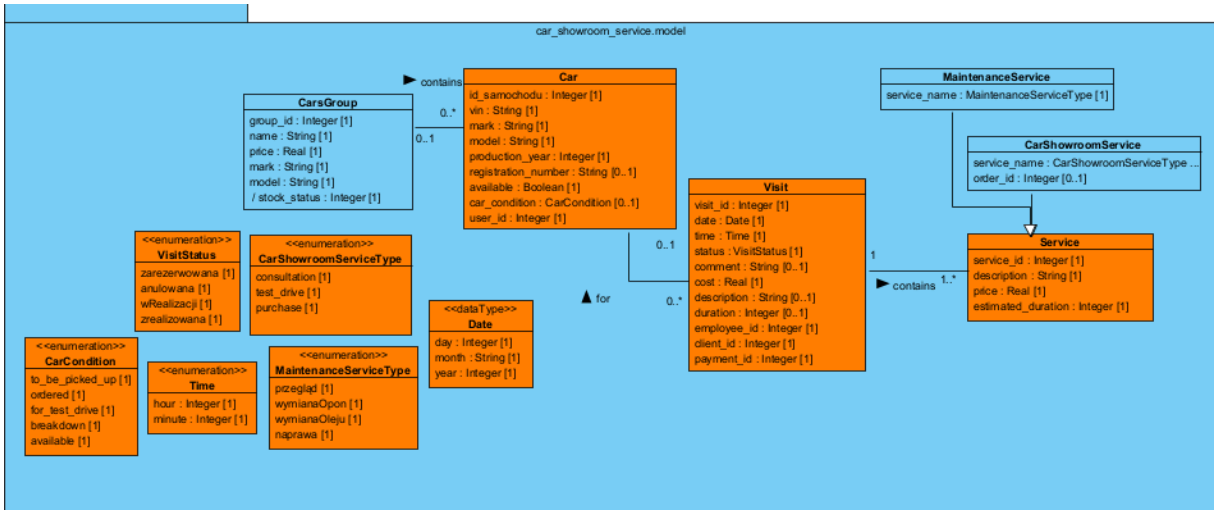
Modele informacyjne został wykonany dla całego systemu.



Rysunek 4: Model informacyjny - Konta użytkowników



Rysunek 5: Model informacyjny - Sklep internetowy



Rysunek 6: Model informacyjny - Salon samochodowy i serwis

8.2) Projekt bazy danych

Projekt bazy danych został wkonany nie dla całego systemu, lecz dla wymagań, które zostały wybrane do implementacji.

Ogólne informacje nt. bazy danych (osobna tabela dla każdej bazy)	
SID/Service Name	SKLEP_DB
Nazwa serwera	DeEs.myqnapcloud.com
Port	1521
Type	Oracle Database 19c
Kodowanie znaków	AL32UTF8
Opis	Relacyjna baza danych sklepu internetowego. Baza przechowuje dane produktów, koszyków, zamówienia.
Technologie	Sequences, Constraints

Tabela 6: Ogólne informacje o bazie danych sklepu internetowego

Backup

Backup
Dane przechowywane w systemie sklepu obejmują katalog produktów i kategorii, koszyki zakupowe, zamówienia oraz opinie o produktach. Dane te mają kluczowe znaczenie dla poprawnego funkcjonowania systemu, w szczególności w zakresie realizacji zakupów i rozliczeń. Pełne kopie zapasowe bazy danych wykonywane są raz w tygodniu, natomiast kopie przyrostowe wykonywane są codziennie w godzinach nocnych w celu ograniczenia wpływu na dostępność systemu. Przechowywane są trzy ostatnie pełne kopie zapasowe, co umożliwia odtworzenie danych po awarii sprzętowej lub w przypadku błędów logicznych (np. niezamierzonego usunięcia danych).

Tabela 7: Strategia backupu bazy danych sklepu

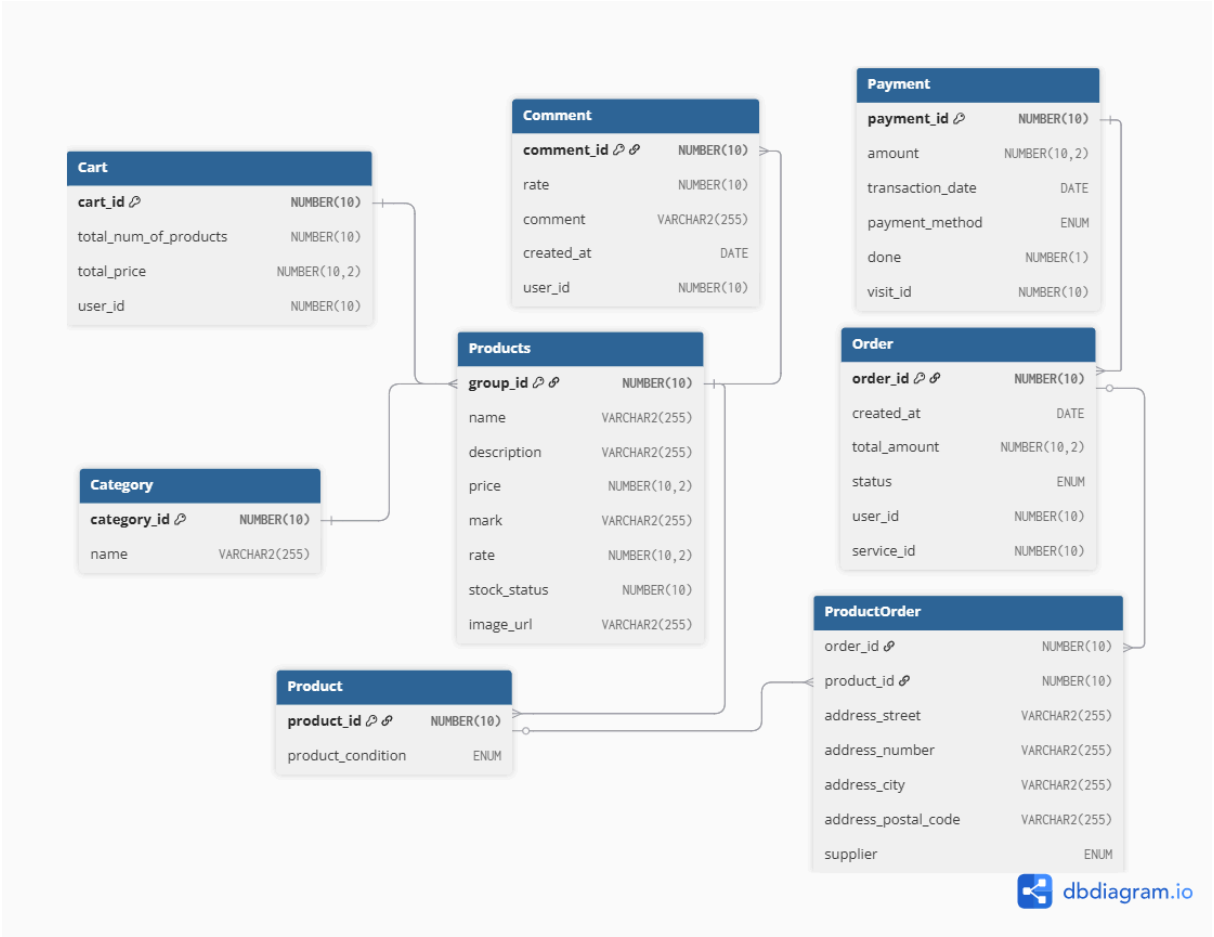
Schemat

Informacje o schemacie	
Nazwa	SKLEP_SCHEMA
Początkowa pojemność	≈10 MB + metadane
Przyrost pojemności (rok)	≈30 MB + metadane
Niezbędne prawa	CREATE TABLE, CREATE SEQUENCE, INSERT, UPDATE, DELETE, SELECT

Tabela 8: Informacje o schemacie bazy sklepu

Pojemność początkowa bazy została oszacowana poprzez zsumowanie przewidywanych rozmiarów rekordów dla każdej tabeli (products, category, cart, order, comment) na podstawie liczby kolumn oraz typów danych (number, varchar, date), a następnie powiększona o przestrzeń na metadane bazy (definicje tabel, klucze główne i obce, constraints, sekwencje).

Przyrost danych roczny oszacowano na podstawie przewidywanej liczby nowych zamówień oraz opinii, a także średniego rozmiaru pojedynczego rekordu w tabelach order, productOrder oraz comment, które generują największy wolumen danych w czasie.



Rysunek 7: Diagram bazy danych - Sklep internetowy

Ogólne informacje nt. bazy danych (osobna tabela dla każdej bazy)	
SID/Service Name	SALON_SERWIS_DB
Nazwa serwera	DeEs.myqnapcloud.com
Port	1521
Type	Oracle Database 19c
Kodowanie znaków	AL32UTF8
Opis	Relacyjna baza danych obsługująca salon samochodowy oraz serwis samochodowy.
Technologie	Sequences, Constraints

Tabela 9: Ogólne informacje o bazie danych salonu i serwisu

Backup

Backup
Dane przechowywane w bazie obejmują ofertę pojazdów, katalog usług serwisowych, rezerwacje wizyt w salonie i serwisie wraz z opisami usterek i szczegółami pojazdów. Dane te są kluczowe dla obsługi klientów, planowania pracy salonu oraz realizacji usług serwisowych. Pełne kopie zapasowe bazy danych wykonywane są raz w tygodniu, natomiast kopie przyrostowe wykonywane są codziennie w godzinach nocnych w celu minimalizacji wpływu na dostępność systemu. Przechowywane są trzy ostatnie pełne kopie zapasowe, co umożliwia odtworzenie danych do wybranego punktu w czasie w przypadku awarii lub błędu logicznego.

Tabela 10: Strategia backupu bazy danych salonu i serwisu

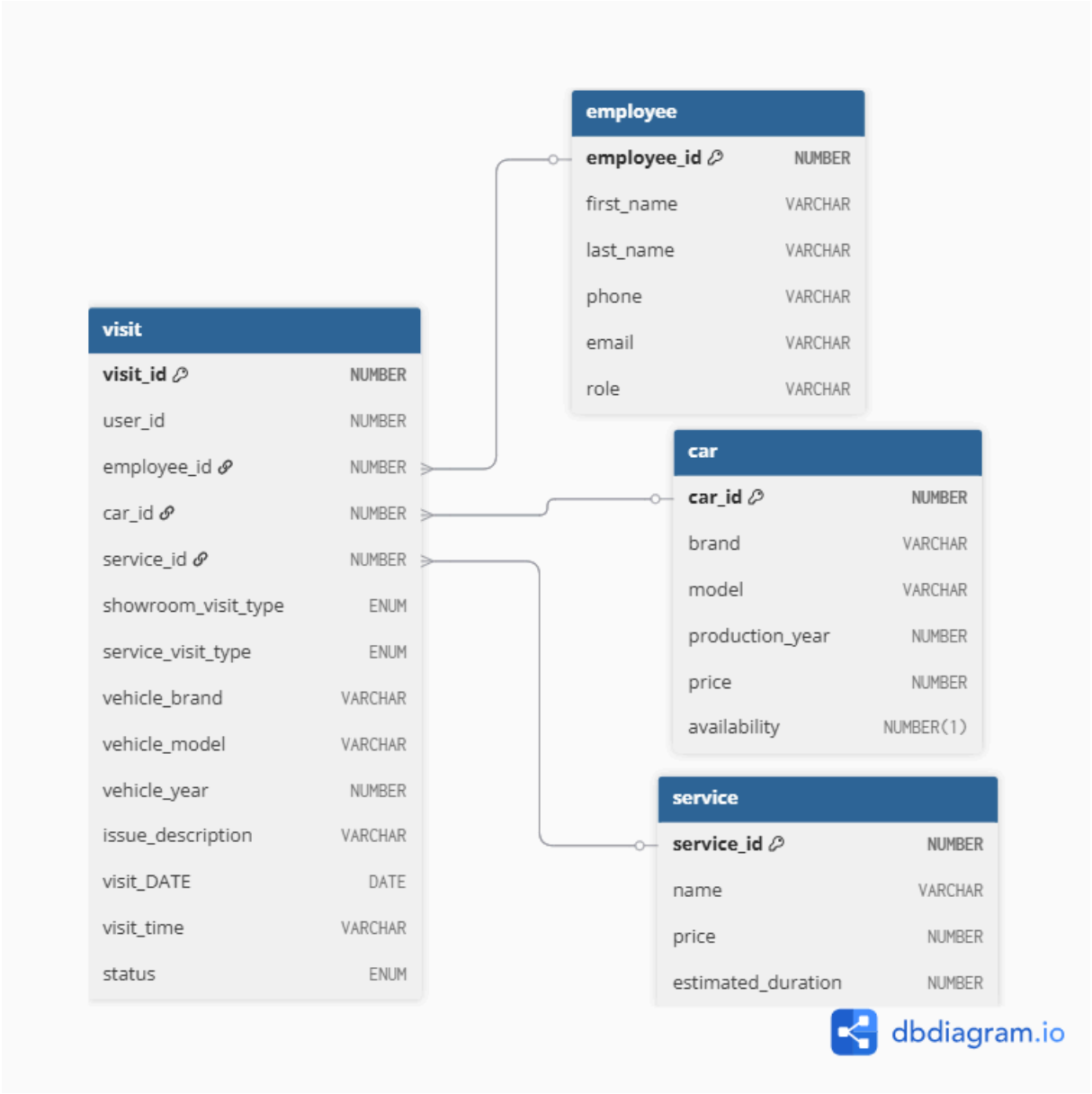
Schemat

Informacje o schemacie	
Nazwa	SALON_SERWIS_SCHEMA
Początkowa pojemność	≈10 MB + metadane
Przyrost pojemności (rok)	≈30 MB + metadane
Niezbędne prawa	CREATE TABLE, CREATE SEQUENCE, INSERT, UPDATE, DELETE, SELECT

Tabela 11: Informacje o schemacie bazy salonu i serwisu

Początkowa pojemność bazy została oszacowana na podstawie przewidywanej liczby rekordów w tabelach car, service, visit oraz employee, a także średniego rozmiaru rekordów zawierających opisy pojazdów, usług oraz wizyt. Dodatkowo uwzględniono przestrzeń na metadane bazy danych, takie jak definicje tabel, klucze główne i obce, ograniczenia integralności oraz sekwencje.

Roczny przyrost danych wynika głównie ze wzrostu liczby rezerwacji wizyt w salonie i serwisie, które są zapisywane w tabelach visit. Tabele car oraz service rosną wolniej, co powoduje umiarkowany i przewidywalny wzrost całkowitego rozmiaru bazy danych.



Rysunek 8: Diagram bazy danych - Salon i serwis samochodowy

Ogólne informacje nt. bazy danych (osobna tabela dla każdej bazy)	
SID/Service Name	ACCOUNT_DB
Nazwa serwera	DeEs.myqnapcloud.com
Port	1521
Type	Oracle Database 19c
Kodowanie znaków	AL32UTF8
Opis	Relacyjna baza danych przechowująca konta użytkowników, role oraz statusy kont.
Technologie	Sequences, Constraints

Tabela 12: Ogólne informacje o bazie danych kont użytkowników

Backup

Backup
Dane przechowywane w bazie kont obejmują informacje identyfikacyjne użytkowników, dane kontaktowe, role oraz statusy kont. Dane te są krytyczne dla bezpieczeństwa systemu oraz kontroli dostępu. Pełne kopie zapasowe bazy danych wykonywane są raz w tygodniu, natomiast kopie przyrostowe wykonywane są codziennie w godzinach nocnych w celu minimalizacji wpływu na dostępność systemu. Przechowywane są trzy ostatnie pełne kopie zapasowe, co umożliwia odtworzenie danych do wybranego punktu w czasie w przypadku awarii lub błędu logicznego.

Tabela 13: Strategia backupu bazy danych kont

Schemat

Informacje o schemacie	
Nazwa	ACCOUNT_SCHEMA
Początkowa pojemność	≈1 MB + metadane
Przyrost pojemności (rok)	≈10 MB + metadane
Niezbędne prawa	CREATE TABLE, CREATE SEQUENCE, INSERT, UPDATE, DELETE, SELECT

Tabela 14: Informacje o schemacie bazy kont użytkowników

Początkowa pojemność bazy została oszacowana na podstawie liczby rekordów w tabelach users oraz roles. Dane użytkowników zawierają głównie krótkie pola tekstowe (imię, nazwisko, e-mail, telefon), dlatego pojedyncze rekordy są niewielkie. Dodatkowo uwzględniono przestrzeń na metadane bazy danych, takie jak definicje tabel, klucze główne, klucze obce oraz sekwencje.

Roczny przyrost danych wynika głównie ze wzrostu liczby użytkowników systemu. Tabele słownikowe, takie jak role, pozostają praktycznie stałe, co powoduje przewidywalny i niewielki wzrost rozmiaru bazy.

Rysunek 9: Diagram bazy danych - Konta użytkowników

9) Widok wytwarzania

9.1) Wprowadzenie

Widok wytwarzania przedstawia technologie, narzędzia oraz proces wdrażania aplikacji webowej do zarządzania salonem samochodowym. System składa się z frontendu w React oraz backendu opartego na funkcjach AWS Lambda w TypeScript, z infrastrukturą zarządzaną przez AWS CDK.

9.2) Technologie

9.2.1) Frontend

- **Framework:** React

- **Język:** TypeScript
- **Hosting:** AWS S3 + CloudFront
- **Routing DNS:** AWS Route53

9.2.2) Backend

- **Język:** TypeScript
- **Runtime:** AWS Lambda (Node.js)
- **API:** API Gateway (REST)
- **Bazy danych:** Amazon RDS (relacyjne bazy danych)

9.2.3) Infrastruktura jako Kod

- **Narzędzie:** AWS CDK
- **Język:** TypeScript
- **Wersjonowanie:** Git

9.2.4) Usługi AWS

- **Route53** - zarządzanie DNS
- **CloudFront** - dystrybucja CDN
- **S3** - hosting statycznej strony
- **API Gateway** - warstwa API
- **Lambda** - funkcje serverless
- **RDS** - bazy danych relacyjne
- **Secrets Manager** - zarządzanie sekretami

9.3) Architektura wdrożenia

System jest wdrażany jako aplikacja serverless w chmurze AWS. Użytkownicy łączą się przez Route53 do CloudFront, który dystrybuuje:

- Statyczne pliki frontendu (React) z S3
- Requesty API przez API Gateway do funkcji Lambda

9.4) Funkcje Lambda

System składa się z trzech głównych funkcji Lambda napisanych w TypeScript:

9.4.1) Auth Lambda

Odpowiedzialna za autoryzację i uwierzytelnianie użytkowników.

- **Połączenia:** Account Database
- **Operacje:** logowanie, rejestracja, zarządzanie sesjami

9.4.2) Showroom and Service Lambda

Obsługuje funkcjonalności związane z salonem i serwisem samochodowym.

- **Połączenia:** Showroom and Service Database
- **Operacje:** zarządzanie pojazdami, rezerwacje wizyt serwisowych, harmonogram

9.4.3) Shop Lambda

Obsługuje sklep internetowy z częściami i akcesoriami.

- **Połączenia:**
  - Shop Database
  - Third party payment provider (TPay)
- **Operacje:** katalog produktów, koszyk, zamówienia, integracja z systemem płatności

9.5) Bazy danych

System wykorzystuje trzy oddzielne bazy danych RDS:

- **Account Database** - dane użytkowników i autoryzacji
- **Showroom and Service Database** - dane o pojazdach, wizytach, serwisie
- **Shop Database** - katalog produktów, zamówienia

Separacja baz danych zapewnia lepszą skalowalność i izolację danych pomiędzy domenami biznesowymi.

9.6) Proces wdrażania

Wdrożenie systemu odbywa się automatycznie za pomocą AWS CDK:

1. **Przygotowanie kodu:**
   - Kompilacja frontendu React do statycznych plików
   - Kompilacja kodu TypeScript backendu
2. **Synteza infrastruktury:**

   ```
   cdk synth
   ```

   Generuje szablony CloudFormation
3. **Wdrożenie:**

   ```
   cdk deploy --all
   ```

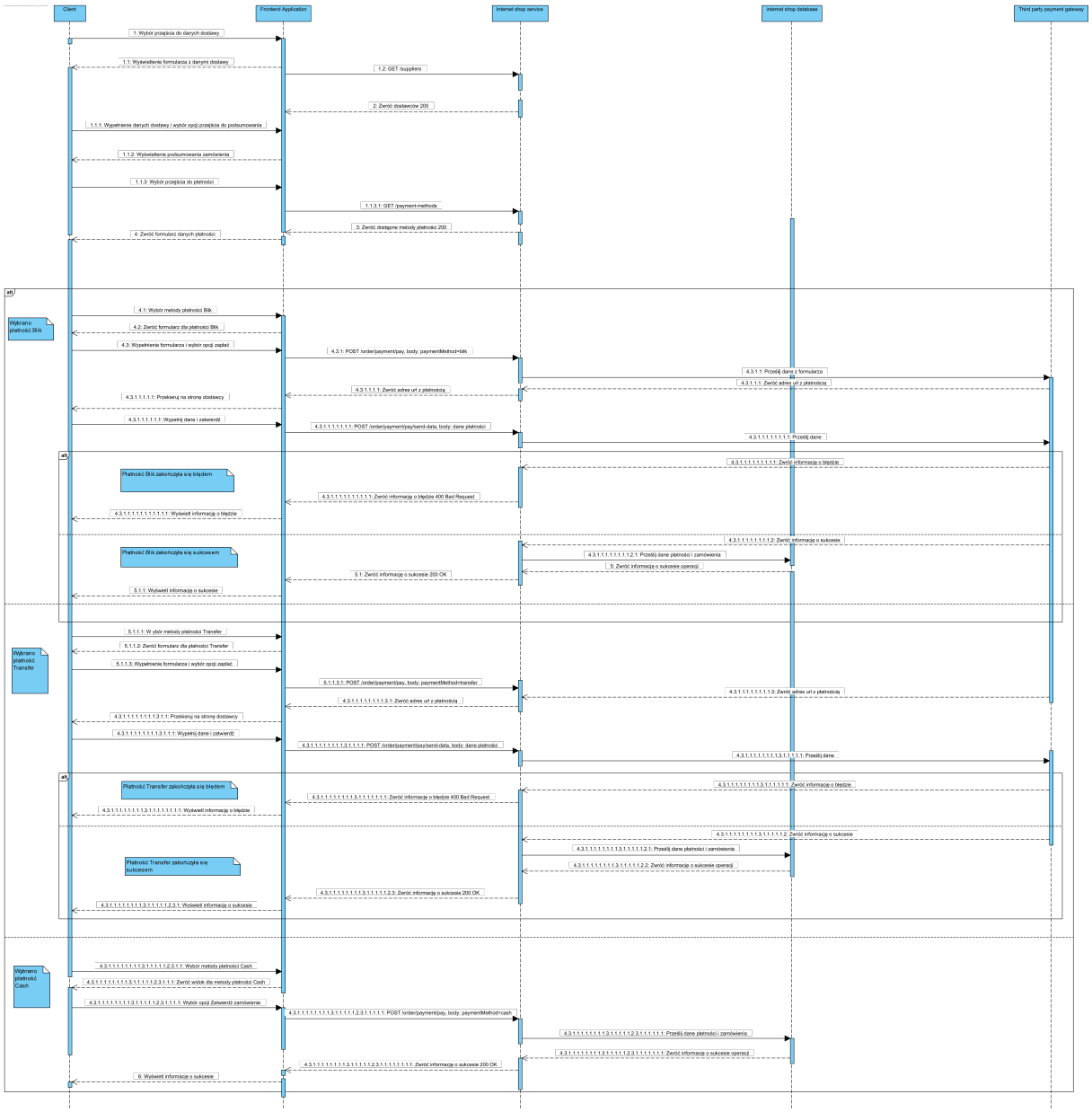
   Tworzy/aktualizuje wszystkie zasoby AWS
4. **Invalidacja cache:** Po wdrożeniu frontendu automatyczna invalidacja CloudFront

9.7) Struktura projektu

```
salon-samochodowy/
├── frontend/              # Aplikacja React
│   ├── src/
│   └── public/
├── backend/               # Funkcje Lambda (TypeScript)
│   ├── auth/
│   ├── showroom-service/
│   └── shop/
├── infra/                 # Infrastruktura AWS CDK
│   └── lib/
│       ├── frontend-stack.ts
│       ├── api-stack.ts
│       ├── database-stack.ts
│       └── lambda-stack.ts
└── shared/                # Współdzielone typy i utilities
```

10) Realizacja przypadków użycia

Dla przypadku użycia: zakup produktów z koszyka


Client
Frontend Application
Internet shop service
internet shop database
Third party payment gateway
1: Wybór przejścia do danych dostawy
1.1: Wyświetlenie formularza z danymi dostawy
1.2: GET /suppliers
2: Zwróć dostawców 200
1.1.1: Wypełnienie danych dostawy i wybór opcji przejścia do podsumowania
1.1.2: Wyświetlenie podsumowania zamówienia
1.1.3: Wybór przejścia do płatności
1.1.3.1: GET /payment-methods
3: Zwróć dostępne metody płatności 200
4: Zwróć formularz danych płatności
alt
Wybrano płatność Blik
4.1: Wybór metody płatności Blik
4.2: Zwróć formularz dla płatności Blik
4.3: Wypełnienie formularza i wybór opcji zapłać
4.3.1: POST /order/payment/pay, body: paymentMethod=blik
4.3.1.1: Prześlij dane z formularza
4.3.1.1.1: Zwróć adres url z płatnością
4.3.1.1.1.1: Zwróć adres url z płatnością
4.3.1.1.1.1.1: Przekieruj na stronę dostawcy
4.3.1.1.1.1.1.1: Wypełnij dane i zatwierdź
4.3.1.1.1.1.1.1.1: POST /order/payment/pay/send-data, body: dane płatności
4.3.1.1.1.1.1.1.1.1: Prześlij dane
alt
4.3.1.1.1.1.1.1.1.1.1: Zwróć informacje o błędzie
Płatność Blik zakończyła się błędem
4.3.1.1.1.1.1.1.1.1.1.1: Zwróć informacje o błędzie 400 Bad Request
4.3.1.1.1.1.1.1.1.1.1.1.1: Wyświetl informacje o błędzie
4.3.1.1.1.1.1.1.1.1.1.1.2: Zwróć informacje o sukcesie
Płatność Blik zakończyła się sukcesem
4.3.1.1.1.1.1.1.1.2.1: Prześlij dane płatności i zamówienia
5: Zwróć informacje o sukcesie operacji
5.1: Zwróć informacje o sukcesie 200 OK
5.1.1: Wyświetl informacje o sukcesie
Wybrano płatność Transfer
5.1.1.1: W ybór metody płatności Transfer
5.1.1.2: Zwróć formularz dla płatności Transfer
5.1.1.3: Wypełnienie formularza i wybór opcji zapłać
5.1.1.3.1: POST /order/payment/pay, body: paymentMethod=transfer
4.3.1.1.1.1.1.1.1.3: Zwróć adres url z płatnością
4.3.1.1.1.1.1.1.1.3.1: Zwróć adres url z płatnością
4.3.1.1.1.1.1.1.1.3.1.1: Przekieruj na stronę dostawcy
4.3.1.1.1.1.1.1.1.3.1.1.1: Wypełnij dane i zatwierdź
4.3.1.1.1.1.1.1.1.3.1.1.1.1: POST /order/payment/pay/send-data, body: dane płatności
4.3.1.1.1.1.1.1.1.3.1.1.1.1.1: Prześlij dane
alt
4.3.1.1.1.1.1.1.1.3.1.1.1.1.1.1: Zwróć informacje o błędzie
Płatność Transfer zakończyła się błędem
4.3.1.1.1.1.1.1.1.3.1.1.1.1.1.1.1: Zwróć informacje o błędzie 400 Bad Request
4.3.1.1.1.1.1.1.1.3.1.1.1.1.1.1.1.1: Wyświetl informacje o błędzie
4.3.1.1.1.1.1.1.1.3.1.1.1.1.1.2: Zwróć informację o sukcesie
Płatność Transfer zakończyła się sukcesem
4.3.1.1.1.1.1.1.1.3.1.1.1.1.2.1: Prześlij dane płatności i zamówienia
4.3.1.1.1.1.1.1.1.3.1.1.1.1.2.2: Zwróć informacje o sukcesie operacji
4.3.1.1.1.1.1.1.1.3.1.1.1.1.2.3: Zwróć informacje o sukcesie 200 OK
4.3.1.1.1.1.1.1.1.3.1.1.1.1.2.3.1.1: Wyświetl informacje o sukcesie
Wybrano płatność Cash
4.3.1.1.1.1.1.1.1.3.1.1.1.1.2.3.1.1.1: Wybór metody płatności Cash
4.3.1.1.1.1.1.1.1.3.1.1.1.1.2.3.1.1.1.1: Zwróć widok dla metody płatności Cash
4.3.1.1.1.1.1.1.1.3.1.1.1.1.2.3.1.1.1.1: Wybór opcji Zatwierdź zamówienie
4.3.1.1.1.1.1.1.1.3.1.1.1.1.2.3.1.1.1.1.1: POST /order/payment/pay, body: paymentMethod=cash
4.3.1.1.1.1.1.1.1.3.1.1.1.1.2.3.1.1.1.1.1.1: Prześlij dane płatności i zamówienia
4.3.1.1.1.1.1.1.1.3.1.1.1.1.2.3.1.1.1.1.1.1.1: Zwróć informacje o sukcesie operacji
4.3.1.1.1.1.1.1.1.3.1.1.1.1.2.3.1.1.1.1.1.1.1.1: Zwróć informacje o sukcesie 200 OK
6: Wyświetl informacje o sukcesie