UNIWERSYSTET WSB MERITO W POZNANIU

Paweł Jaruszewski  
Numer albumu: 156712

**Aplikacja internetowa wspomagająca pracę dostawców w przedsiębiorstwie zajmującym się sprzedażą artykułów biurowych**

**Praca Inżynierska**

**Promotor  
dr inż. Katarzyna Skroban**

Praca została zweryfikowana przez Jednolity System  
Antyplagiatowy, dopuszczam pracę do egzaminu dyplomowego

**……………………………………………….....................................**

podpis promotora

**Kierunek: Informatyka  
Specjalność: Programowanie**

WARSZAWA ROK 2024

**SPIS TREŚCI**

[WSTĘP 4](#_Toc169955300)

[ROZDZIAŁ 1 SYSTEM INFORMATYCZNY JAKO PODSTAWOWE NARZĘDZIE DO FUNKCJONOWANIA PRZEDSIĘBIORSTWA 9](#_Toc169955301)

[1.1. System informatyczny 9](#_Toc169955302)

[1.2. Bazy danych 10](#_Toc169955303)

[1.3. ORM 13](#_Toc169955304)

[1.3.1. Prisma jako narzędzie zwiększające produktywność 13](#_Toc169955305)

[1.4. Rest Api 15](#_Toc169955306)

[1.5. Języki programowania 16](#_Toc169955307)

[1.6. Interfejs użytkownika frameworki 19](#_Toc169955308)

[1.7. Systemy kontroli wersji 22](#_Toc169955309)

[1.8. Modelowanie systemów 23](#_Toc169955310)

[1.9. Opis przedsiębiorstwa 26](#_Toc169955311)

[ROZDZIAŁ 2 ANALIZA OBSZARU APLIKACJI INTERNETOWYCH WSPOMAGAJĄCYCH PRACĘ KIEROWCÓW 29](#_Toc169955312)

[2.1. Kontekst biznesowy oraz technologiczny 29](#_Toc169955313)

[2.2. Analiza procesów przedsiębiorstwa 31](#_Toc169955316)

[2.3. Analiza potrzebnych funkcjonalności 32](#_Toc169955318)

[2.3. Aspekty prawne i regulacyjne 34](#_Toc169955321)

[2.4. Sformułowanie problemu 37](#_Toc169955322)

[ROZDZIAŁ 3 OPIS SYSTEMU ORAZ BADANIA 38](#_Toc169955323)

[3.1. Start projektu 38](#_Toc169955325)

[3.2. System na diagramach 46](#_Toc169955326)

[3.3. Badania do aplikacji 49](#_Toc169955327)

[3.4. Rozważanie hipotezy i problemu badawczego 50](#_Toc169955328)

[ROZDZIAŁ 4 WYNIKI BADAŃ DOTYCZĄCE ISTOTNYCH FUNKCJONALNOŚCI 51](#_Toc169955329)

[4.1. Wyniki badań dotyczące aplikacji obecnie używanych 51](#_Toc169955331)

[4.2. Pożądane funkcjonalności wśród interesariuszy 57](#_Toc169955332)

[4.3. Implementacja 58](#_Toc169955333)

[4.4. Rozwój aplikacji 59](#_Toc169955334)

[4.5. Analiza problemu badawczego oraz hipotezy 61](#_Toc169955335)

[ZAKOŃCZENIE 62](#_Toc169955336)

[BIBLIOGRAFIA 65](#_Toc169955337)

[AKTY NORMATYWNE 67](#_Toc169955338)

[SPIS RYSUNKÓW 68](#_Toc169955339)

[SPIS TABEL 69](#_Toc169955340)

[STRESZCZENIE 70](#_Toc169955341)

# WSTĘP

W środowisku biznesowym stale dynamicznie się rozwijającym ważne jest doskonalenie procesów oraz wprowadzanie nowych technologii. Szczególnie w  przedsiębiorstwach zajmujących się sprzedażą i dystrybucją lokalną, poprzez dowóz bezpośredni danych artykułów do klienta, logistyka oraz komunikacja odgrywają ważną rolę w codziennym prosperowaniu firmy. Praca skupiać będzie się na przedsiębiorstwie zajmującym się sprzedażą oraz dowozem artykułów biurowych do klienta, które odbywa się na następny dzień roboczy. Tematem tej pracy inżynierskiej będzie: „Aplikacja internetowa wspomagająca pracę dostawców w firmie zajmującej się sprzedażą artykułów biurowych”.

Głównym celem pracy dyplomowej jest pokazanie na przykładzie aplikacji internetowej w przedsiębiorstwie zajmującym się sprzedażą artykułów biurowych jak, można usprawnić komunikację między osobami odpowiedzialnymi za zamawianie towarów z hurtowni zewnętrznych oraz kierowcami odpowiedzialnymi za dowóz towaru do klienta, odbiór materiałów z hurtowni, zrobienie zakupów dla klienta, odbieranie wszelkich zwrotów oraz reklamacji. Już same jakże różne ale powiązane ze sobą zadania, mogą powodować dużo problemów komunikacyjno- logistycznych, które też są zauważane w postaci odroczenia dostawy czy  nieekonomicznej jazdy kilkakrotnie po odbiór z powodu niedoprecyzowania co do ilości odbieranych towarów. Dodatkowo, jeśli cel zostanie osiągnięty, może wpłynąć na  efektywność pracy kierowców oraz ekonomiczność w eksploatacji pojazdów.

Przystępując do badań, sformułowano następującą hipotezę główną „Wprowadzenie aplikacji internetowej wspomagającej pracę dostawców w firmie zajmującej się sprzedażą artykułów biurowych przyczyni się do zwiększenia efektywności działań logistycznych oraz poprawy obsługi klienta”. Praca przedstawi odpowiedź na to pytanie, która w ciągu używania aplikacji może okazać się pozytywna oraz negatywna z powodu nierozwiązania problemów logistyczno-komunikacyjnych.

Rozważając pierwszy rozdział, można sformułować hipotezę poboczną „Technologie użyte do stworzenia dobrej i wydajnej aplikacji internetowej”. Pierwszy rozdział przedstawi technologię oraz opisze, czemu akurat w aplikacji należy użyć takich, a  nie innych.

Patrząc na drugi rozdział, można rozważyć hipotezę „Aplikacje na rynku, których używa przedsiębiorstwo wystarczą kierowcą oraz osobą zamawiającym do płynnej oraz  zrozumiałej komunikacji”. Wiadomo, że aplikację na rynku są, ale czy są one dostosowane do przedsiębiorstwa zajmującego się sprzedażą artykułów biurowych i czy są  one wystarczające.

W trzecim rozdziale postawiona sformułowana hipoteza to „Stworzenie aplikacji internetowej w szkielecie Next.js nie jest trudnym wyzwaniem, jeśli zostaną zastosowane metodyki projektowania systemów informatycznych, co pozwoli na efektywne usprawnienie komunikacji w przedsiębiorstwie.”. Rozdział wykorzysta metody badawcze i sprawdzi, jak trudnym wyzwaniem jest stworzenie aplikacji internetowej do wspomagania komunikacji w przedsiębiorstwie.

W ostatnim rozdziale można sformułować hipotezę „Interesariusze nie są zadowoleni z obecnych rozwiązań”. Wyniki badań pokażą czy ze wszystkich funkcjonalności kierowcy są niezadowoleni czy istnieje jakaś, która jest przydatna

Przedsiębiorstwo korzysta z różnych aplikacji internetowych czy też mobilnych wspomagającą komunikację. Problemem z aplikacjami dostępnymi na rynku jest taki, że jeśli są bardziej skomplikowane wymagają wprowadzenia jednorazowej opłaty lub  wykupienia miesięcznej subskrypcji czego przedsiębiorstwo wolałoby uniknąć, a  darmowe aplikacje nie spełniają oczekiwań po względem jakości wykonania czy też  płynności lub przejrzystości. Kolejnym nieudogodnieniem jest oczywiście brak dopasowania do modelu przedsiębiorstwa co skutkuje właśnie szukaniem kolejnych spełniających pojedyncze funkcję. Także, głównym celem tej pracy jest zbadanie potrzeb i  oczekiwań od „idealnej” aplikacji i próba stworzenia, przekształcenia czy też dodania funkcjonalności i zamknięcie to w jednej aplikacji internetowej oraz dzięki możliwości ustawiania responsywności strony będzie działać oraz wyglądać przejrzysto na monitorach komputera oraz na telefonach komórkowych, czyli płynna komunikacja będzie dostępna również dla kierowców podczas wykonywania swoich zadań.

Główny problem badawczy przyjął postać następującego pytania „Jakie funkcjonalności aplikacji mogłyby efektywnie wspomagać pracę kierowców w firmie zajmującej się sprzedażą artykułów biurkowych, mając na uwadze optymalizację procesów logistycznych?”. Żeby odpowiedzieć na to pytanie potrzebne będzie wykorzystanie metod badawczych empirycznych, które pomogą zebrać informacje oraz wymagania od  interesariuszy aplikacji.

Pierwszym rozdziale problem badawczy przyjmuje pytanie „Czy istnieją odpowiednie technologie do stworzenia aplikacji internetowej?”. Odpowiedź na to pytanie, przy obecnym stacku technologicznym najprawdopodobniej będzie pozytywna jednak celem jest przedstawienie tych technologii.

Drugi rozdział rozważy problem badawczy „Funkcjonalności, które są w aplikacjach istniejących na rynku, są wystarczające?”. Wynik rozważań pokaże czy trzeba dopisywać funkcjonalności przydatne czy funkcjonalności już istniejące wystarczą.

Dla trzeciego rozdziału problem badawczy formułuje się następująco „Czy stworzenie aplikacji internetowej w szkielecie next.js jest trudnym wyzwaniem?”. Tworzenie systemu informatycznego wymaga wybrania technologii oraz rozrysowania systemu na diagramach, a więc rozdział rozjaśni czy stworzenie takiego systemu jest trudnym wyzwaniem.

Ostatni rozdział przyjmie problem badawczy, który brzmi następująco „Funkcjonalności, które powinny pojawić się w aplikacji jako pierwsze?”. Najważniejsze funkcjonalności powinny pojawiać się jako pierwsze, a w ciągu rozwoju aplikacji dodawać te mniej ważne.

Metodami badawczymi wykorzystanymi w pracy będą metody empiryczne takie jak: metodę obserwacyjną biorąc pod uwagę liczbę populacji czyli obserwacja indywidualna czy  grupowa. Polegać będzie na obserwowaniu indywidualnym i grupowym potrzeb oraz  oczekiwań względem aplikacji wspomagająca pracę kierowców, dzięki czemu potencjalni użytkownicy aplikacji mogą podzielić się swoimi doświadczeniami oraz pomysłami na  funkcjonalności w aplikacji, dzięki czemu aplikacja będzie dostosowana ściśle do  przedsiębiorstwa w którym będzie wykorzystywana.

Praca składa się z czterech rozdziałów poprzedzonych wstępem oraz zakończeniem. W pierwszym rozdziale przedstawiono teorię, gdzie wyjaśnione będą podstawowe definicję oraz pojęcia, wyjaśnione będą również technologię na których opierać będzie się aplikacja internetowa takie jak framework, bazy danych oraz biblioteki. Projekt w całości napisany będzie w popularnym języku javascript używanym po stronie frontedowej aplikacji czy też stron internetowych, ale również może być używany po stronie backendowej do  pisania API czy też serwisów. Do wspomagania języka typowanego dynamicznie, użyta będzie nakładka typescript, która pozwala na typowanie statyczne dzięki temu wszelkie błędy typowania będzie można wykryć już podczas pisania kodu, a nie jego kompilacji. Aplikacja będzie używać frameworka Nextjs który jest frameworkiem full-stackowym pozwalającym budować strukturę frontendową wykorzystująca bibliotekę react js która jest oparta na javascripcie służącą do tworzenia interfaców. Umożliwia stworzenia API dedykowanego pod aplikację oraz korzystać będzie z nierelacyjnej bazy danych mongodb która dzięki swojej prostocie często jest wykorzystywana w mniejszych aplikacjach, do tego baza danych jest dostępna w darmowej wersji z przestrzenia do wykorzystania 512 mb oraz  wykorzystuje opcję hostingu europejskiego poprzez Amazon Aws, Google Cloud lub  Microsoft Azure oraz pozwala na łatwe wykorzystanie biblioteki do uwierzytelniania, która opierać się będzie na rolach (kierowca/zamawiający) W projekcie wykorzystane będą dedykowane pod framework biblioteki komponentów user interface takie jak next ui oparte również na bibliotece react js oraz bilbioteki icon takie jak react icons. Narzędziem ORM będzie prisma, która umożliwia bezpośrednie mapowanie między bazą danych, a kodem javascript/typescipt. Mimo, iż służy do mapowania baz obiektowo relacyjnych można użyć również do bazy mongodb, która nie jest bazą relacyjną. W pierwszym rozdziale będzie przedstawiony również precyzyjny opis badanego obszaru czyli przedsiębiorstwo oraz jego wewnętrzne prosperowanie pod względem komunikacyjnym pomiędzy osobami zamawiającymi oraz kierowcami wykonującymi swoje zadania poza placówką firmy. Opisane również będą problemy oraz jej diagnoza w ujęciu praktycznym czyli zostaną opisane powody powstawania problemów, zmiany sytuacji w ciągu czasu, stan obecny przedsiębiorstwa, interesariuszy oraz jakie efekty może przynieść rozwiązanie problemu lub  też przedstawienie opisu stanu pożądanego.

Drugi rozdział zawiera wybrany obszar badań ze szczegółowym określeniem jego miejsca w ramach danej dyscypliny oraz ewentualnych uregulowań prawnych. Omówione będzie specyfikacja działania firmy w tym podział na marki produktów dostępnych w  danych hurtowniach zewnętrznych gdzie godzina odbioru oraz forma różni się od siebie. Przedstawione oraz przeanalizowane będą wymagania, a także potrzeby kierowców z  którymi spotykają się w swojej codziennej pracy wraz z badaniem obszarów, w którym aplikacja będzie spełniać swoją funkcję. Określona będzie rola jaką będzie pełnić aplikacja w  codziennym prosperowaniu przedsiębiorstwa, w tym jej potencjalna efektywność, koszty utrzymania. Aplikacja będzie wymagała wprowadzenia danych osobowych oraz numeru rejestracyjnego pojazdu przypisanego do danego pracownika oraz przechowywaniu to  w  bazie danych wiec podane będą uregulowania prawne dotyczących ochrony danych osobowych jak i bezpieczeństwa przetwarzania danych. W ostatnich podpunktach będą przedstawione dalsze perspektywy rozwoju, ewolucję technologii oraz wyjaśnienie szczegółowego problemu badawczego jakim jest „Czy aplikację internetową wspomagającą pracę kierowców można uogólnić do większości przedsiębiorstw prosperujących w obszarze sprzedaży oraz dowozu artykułów biurowych”.

Trzeci rozdział przedstawi tworzenie systemu od samego początku. Pokarze również system na diagramach jak diagram klas, diagram obiektów oraz diagram przypadków użycia. Pokarze w jaki sposób przeprowadzane będą badania na przedsiębiorstwie do systemu. To wszystko obali lub podtrzyma hipotezę postawioną w tym rozdziale.

W czwartym rozdziale zawarte będą wyniki badań własnych oraz propozycję dotyczące nowych rozwiązań oraz wyniki analizy zebranych danych. Prezentacja wyników badań, w formie tabeli bądź wykresów tak, żeby zebrane dane były jak najbardziej czytelne oraz łatwe do zrozumienia. Zawierać również będzie wyniki własnych obserwacji. Przedstawione będą również propozycję dotyczące nowych rozwiązań lub udoskonaleń, które wynikać będą z zebranych danych. Rozdział zawiera również identyfikację ewentualnych ograniczeń badań.

Wynikiem pracy będzie przedstawienie będzie przedstawienie w jaki sposób dzięki prostemu systemowi informatycznemu tzn. aplikacji internetowej można usprawnić komunikację w zespole oraz jak wpływać będzie to na sposób organizacji przedsiębiorstwa oraz w jaki sposób ustawni jej procesy.

Źródłami użytymi do opracowania problemu określonego w temacie będzie literatura naukowa zdobyta poprzez bibliotekę uniwersytecką oraz znalezioną w sieci poprzez platformę Google Scholar. Wykorzystana zostanie również dokumentacja techniczna bibliotek oraz frameworków w celu dokładnego przybliżenia działania poszczególnych fragmentów.

# ROZDZIAŁ 1 SYSTEM INFORMATYCZNY JAKO PODSTAWOWE NARZĘDZIE DO FUNKCJONOWANIA PRZEDSIĘBIORSTWA

## System informatyczny

System informatyczny to kompleksowy zespół systemów komputerowych, sieci oraz oprogramowania, który służy do przetwarzania informacji. W jego skład wchodzi zazwyczaj jeden lub więcej komputerów, które mogą być połączone w sieć lub działać samodzielnie, wraz z odpowiednim oprogramowaniem oraz różnymi urządzeniami pomocniczymi, takimi jak urządzenia peryferyjne (drukarki, skanery) oraz nośniki danych (np. dyski twarde, dyskietki). System informatyczny może mieć charakter uniwersalny lub być zaprojektowany do realizacji określonych zadań, na przykład rezerwacji biletów lotniczych czy prowadzenia finansowo-księgowych operacji przedsiębiorstwa. Współczesne systemy informatyczne odgrywają kluczową rolę w zarządzaniu i operacjach organizacji, umożliwiając efektywne gromadzenie, przetwarzanie i analizowanie danych. Dzięki integracji z różnorodnymi narzędziami i technologiami, systemy te wspierają wiele aspektów działalności biznesowej, od zarządzania łańcuchem dostaw i logistyką, przez produkcję, aż po marketing i sprzedaż. Rozwój i wdrażanie nowoczesnych systemów informatycznych pozwala przedsiębiorstwom na zwiększenie efektywności operacyjnej, poprawę jakości usług oraz uzyskanie przewagi konkurencyjnej na rynku.[[1]](#footnote-1)

Systemy informatyczne można sklasyfikować według zasięgu dziedzinowego oraz funkcji zarządzania, które wspierają. Pod względem zasięgu dziedzinowego wyróżnia się trzy główne typy systemów. System cząstkowy, działający w obrębie jednej konkretnej dziedziny, obejmuje wybrane funkcje lub sub funkcje użytkowe. System dziedzinowy operuje w ramach jednej dziedziny przedmiotowej zarządzania, obsługując niemal wszystkie lub wszystkie jej funkcje użytkowe. Natomiast system wielodziedzinowy integruje funkcje użytkowe co najmniej dwóch różnych dziedzin przedmiotowych, a współczesne systemy informatyczne zazwyczaj należą do tej kategorii, różniąc się liczbą obsługiwanych dziedzin oraz złożonością powiązań. Ze względu na zakres dziedzinowy można wyróżnić cztery rodzaje systemów: system prosty, który nie obsługuje działalności podstawowej organizacji; system bazowy, obsługujący podstawowe funkcje związane z domenami strategicznymi przedsiębiorstwa oraz wybrane inne dziedziny; system rozwinięty, który obsługuje standardowe agendy organizacji gospodarczej oraz dodatkowe dziedziny; oraz system kompleksowo zintegrowany, który wykorzystuje zaawansowane technologie w dziedzinach takich jak komputerowo wspomagane projektowanie wyrobów (CAD), procesów (CAP), zapewnienia jakości (CAQ), oraz wytwarzania (CAM), prowadząc do powstania systemów CIM (komputerowo zintegrowane wytwarzanie). Pod względem wspomagania funkcji zarządzania systemy informatyczne dzielą się na cztery typy. Systemy ewidencyjno-sprawozdawcze (SES), stosowane najwcześniej (lata 1955-1975), to zintegrowane systemy przetwarzania danych (ZSPD) specjalizujące się w obsłudze ewidencji i podstawowej obróbce danych, generujące tabulogramy o stałej strukturze. System informowania kierownictwa (SIK), oparty na SES, zawiera interfejs użytkownika umożliwiający generowanie zbiorów danych o dowolnie definiowanym zakresie i strukturze, stosowany od około 1975 roku jako zintegrowany system informacyjny (ZISK). System wspomagania decyzji (SWD) rozszerza funkcjonalność systemów typu MIS, umożliwiając symulację różnych scenariuszy decyzyjnych i dialogową pracę z użytkownikiem, posiadając wydzieloną bazę procedur decyzyjnych. Systemy z bazą wiedzy (SBW) realizują idee sztucznej inteligencji, wspierając użytkowników w procesach decyzyjnych i rozwiązywaniu problemów zarówno ustrukturyzowanych, jak i nieustrukturyzowanych, i mieć posiadać specyficzne cechy strukturalne, technologiczne i funkcjonalne, aby być zaliczone do tego typu. Te klasyfikacje systemów informatycznych ukazują ewolucję i różnorodność rozwiązań stosowanych w zarządzaniu organizacjami, od prostych systemów ewidencyjnych po zaawansowane systemy wspomagające decyzje oparte na sztucznej inteligencji.[[2]](#footnote-2)

## Bazy danych

Bazy danych używane są praktycznie we wszystkich aplikacjach czy też stronach internetowych z których korzystamy na co dzień. Dzięki nim możemy zapisywać, analizować, przetwarzać rozmaite zbiory danych. Spójrzmy na to w ten sposób, że  po  wejściu na daną stronę może okazać się, że będziemy musieli utworzyć konto, żeby w pełni móc skorzystać z usług. Wejdziemy więc na podstronę z formularzem rejestracyjnym, gdzie po wypełnieniu formularza i zaakceptowaniu polityki prywatności danej strony klikniemy by nas zarejestrowało. Gdy wszystko się zgadza nasze dane wpisane w formularz „powędrują” do takiej bazy danych gdzie zostaną zapisane. Bazy te dzielą się  na dwa podstawowe typy relacyjne (z ang. Relational Database Management System, RDBMS) oraz nierelacyjne (z ang, Not Only SQL, NoSQL). Wybór typu bazy danych zależy od indywidualnych potrzeb projektu. W aplikacji użyta zostanie baza danych nierelacyjna mongodb, dlatego też praca będzie oparta również o ten typ. Będzie korzystać z bezpłatnego hostingu poprzez Amazon AWS z serwerem europejskim umieszczonym we Frankfurcie, w  Niemczech. Dostępne w darmowej wersji będzie 512mb co będzie wystarczające do tego projektu. W fazie rozwoju będzie możliwość rozszerzyć dostępną przestrzeń za opłatą.

Systemy zarządzania bazami danych opartych na relacjach (z ang. Relational Database Management System, RDBMS), takie jak Oracle lub PostgreSQL, cieszą się w  obecnych czasach dużym uznaniem wśród programistów, którzy doceniają istotę wydajności baz danych oraz jej normalizacji. Jednakże, koncepcja NoSQL staje się coraz bardziej popularna, zwłaszcza jeśli chodzi o rozwiązania sterowania danymi (z ang. modern data-driven solution). Chociaż niektórzy programiści są sceptyczni jeśli chodzi o  nierelacyjne bazy danych, lecz wraz z rozwojem oraz rozpowszechnieniem się rozwiązań typu BigData, NoSQL stało się niezbędną formą dla wielu systemów. Warto pamiętać również, że jedno nie wyklucza drugiego to znaczy, że bazy relacyjne i nierelacyjne mogą współistnieć w jednym systemie. Ponadto, ogólnie uważa się, że wyszukiwanie danych jest prostsze w relacyjnych systemach RDBMS, dlatego też często niezbędne jest stosowanie obu rodzajów rozwiązań równocześnie.[[3]](#footnote-3)

Termin NoSQL, wywodzący się z języka angielskiego „Not Only SQL”, który odnosi się do systemów zarządzania danymi, nieopierającymi się na modelu relacyjnym. Odrzucenie tego modelu skutkuje możliwością przechowywania danych bez konieczności stosowania ściśle określonych schematów. Dodatkiem stosowania systemów NoSQL w  projektach jest brak potrzeby stosowania złączeń, co w znacznym stopniu ułatwia skalowanie w poziomie i skutkuje bardziej wydatnymi zapytaniami. Termin NoSQL częstokrotnie mylony jest z daną technologią, lecz w praktyce opisywany jest jako zbiór rozwiązań, które służą do przechowywania danych, różniących się od modelu relacyjnego. Termin „Not Only SQL” podkreśla możliwość używania elementów języka SQL, zwłaszcza składniowo podobnych języków zapytań.[[4]](#footnote-4)

Chcąc porównać relacyjne bazy danych oraz bazy NoSQL, który nie posiada relacji oraz często nie ma zdefiniowanego języka zapytań SQL jak i nie mają z góry określonego schematu może posłużyć się przykładem. Możemy założyć, że będziemy implementować system do przechowywania danych o pracownikach pewnej organizacji. Każdy z  pracowników może posiadać po kilka numerów telefonów.

Tabela 1 Telefony\_komórkowe

|  |  |  |  |
| --- | --- | --- | --- |
| Id\_Telefon | Pracownik\_Id | Numer tel. | Typ\_Id |
| 1 | 1 | 516245333 | 1 |
| 2 | 2 | 508202332 | 2 |
| 2 | 3 | 668512563 | 1 |

Tabela 2 Pracownicy

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Pracowni\_Id | Imię | Nazwisko | Wiek | Wynagrodzenie | Wydział\_Id |
| 1 | Paweł | Kowalski | 28 | 6000 | 1 |
| 2 | Gaweł | Nowak | 32 | 5000 | 2 |

Tabela pierwsza (Telefony\_komórkowe), będzie zawierać informacje o numerach telefonów, którymi posługują się pracownicy. Druga Tabela (Pracownicy), będzie zawierać podstawowe dane o pracownikach takie jak: imię, nazwisko, wiek, wynagrodzenie oraz klucz po którym będzie można zlokalizować dział w którym dany pracownik pracuje. Bez wątpienia, można zauważyć co najmniej kilka relacji. Pierwsza występuje pomiędzy tabelami Telefony oraz Pracownicy, gdzie wykorzystano klucz obcy Pracownik\_Id do  dokonania realizacji, który wskazuje na konkretny wiersz w tabeli drugiej (Pracownicy). Analogicznie wygląda to z Wydział\_Id oraz Typ\_Id. Takie podejście jest typowe dla  relacyjnych baz danych to znaczy, że wszystkie dane muszą być odpowiednio (atomowo) podzielone pomiędzy kolumnami. Ważne żeby nie trzymać w tej samej tabeli np. imienia i nazwiska (złączenie danych), z powodu niezgodności z tzw. pierwszą postacią normalną. Teraz należy rozważyć jak dany przykład mógłby zostać zrealizowany w  podejściu NoSQL?

Tabela 3 Wszystkie\_dane

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Pracownik\_Id:  1 | Imię:  Paweł | Nazwisko:  Kowalski | Wiek:  28 | Zarobki:  6000 | Tel\_prywatny:  516245333 | Tel\_służbowy:  605223456 |
| Pracownik\_Id:  2 | Imię:  Gaweł | Nazwisko:  Nowak | Wiek:  32 | Zarobki:  5000 | Tel\_domowy:  425234555 | Tel\_służbowy: |

Tabela trzecia różni się od przykładów z użyciem bazy SQL w dwóch miejscach. Pierwszą różnicą jest różnorodność wierszach, każdy może zawierać różne „kolumny”. Zauważyć można również, że nie ma tutaj z góry przyjętego formatu. Widać wyraźnie, że  jeden wiersz może zawierać informacje o telefonie prywatnym, podczas gdy drugi dotyczy telefonu służbowego. W podejściu NoSQL nie istnieją typowe relacje, takie jak  możemy znaleźć w tradycyjnych systemach baz danych, gdzie jest narzucana struktura przez silnik zarządzania bazą danych, dlatego też można stworzyć pola z „kluczami” podstawowymi i obcymi ale to programista jest odpowiedzialny za utrzymywanie oraz egzekwowanie ich. Konsekwencją tego jest to że bazy NoSQL są bazami z nierelacyjnymi modelami danych. W kontekście przykładu z telefonami, NoSQL może być preferowany nad RDBMS w sytuacjach, gdzie dane różnią się pomiędzy poszczególnymi rekordami. W  Bazach nierelacyjnych łatwiej jest osiągnąć skalowanie horyzontalne, co pozwoli na  bardziej elastyczne zażądanie zmiennymi informacjami. W przeciwieństwie do modelu relacyjnego, w którym dodanie nowej kolumny wyłącznie dla jednego rekordu jest niezgodne z normami. W modelu nierelacyjnym struktura danych może przyjmować formę par kluczy i wartości. W przypadku adresów, dokumentów o różnych strukturach lub  niestandardowych encji, utrzymanie tych informacji jest zwykle prostsze w niż w  bazach RDBMS.[[5]](#footnote-5)

## ORM

Interfejs dostępu do baz danych jest narzędziem programistycznym, ułatwiającym dostęp oraz manipulację danymi w bazach danych. Jednym z popularniejszych podejść jest mapowanie obiektowo-relacyjne (ORM), które pozwala na integrowanie struktury obiektowej systemu z bazą danych o charakterze relacyjnym co może być mylące z powodu tego, że bazą danych użytą w pracy jest baza NoSQL (Mongodb), która również pozwala na  korzystanie z ORM. Dzięki rozwiązaniom ORM, programiści programujących w takich językach jak C#, Java czy javascript są w stanie manipulować danymi bez konieczności korzystania z języka SQL. Zamiast operować na rekordach, aplikacje mogą używać gotowych obiektów, które są wynikiem zapytań do bazy danych.[[6]](#footnote-6)

W projekcie zostanie użyte Prisma ORM, która jest otwartym narzędziem do  mapowania obiektowo-relacyjnego. Składa się z następujących części: Prisma Client to  bezpieczny w typach, automatycznie generowany kreator zapytań dla Node.js oraz TypeScript, Prisma Migrate jest to system migracyjny, Prisma Studio która jest graficznym interfejsem użytkownika służącym do przeglądania oraz edycji danych w bazie danych i  można uruchamiać tylko lokalnie. Prisma Client może być użyty w dowolnej aplikacji backendowej Node.js opartej na architekturze REST API, GraphQL, gRPC lub w innym rozwiązania wymagającym dostępu do bazy danych.

Każdy projekt wykorzystujący narzędzie z Prisma ORM zaczyna się od pliku schematu Prisma. Schemat ten pozwala programistą na zdefiniowanie modeli aplikacji w  bardzo intuicyjnym języku modelowania danych zawierających również połączenie z  bazą danych oraz definiuje generator. W schemacie trzeba skonfigurować trzy rzeczy: źródło danych określające połączenie z bazą danych za pomocą zmiennej środowiskowej, generator, który wskazuje potrzebę wygenerowania klienta Prismy oraz model danych dla przykładu model o nazwie Użytkownik będzie zawierać pola o nazwie ID które za typ przyjmie string (tekstowy typ danych) oraz można ustawić, że będzie wygenerowany automatycznie następnie imię które również będzie stringiem i będzie to pole obowiązkowe oraz numer telefonu, który ustawimy typ danych jako number (w języku typescript jest to  typ określający wszystkie cyfry całkowite oraz zmiennoprzecinkowe). Warto zaznaczyć, że w innych językach programowania np. Java, liczby całkowite mają osobne typy INT, BYTE, SHORT, LONG oraz liczby zmiennoprzecinkowe DOUBLE czy FLOAT.

* + 1. Prisma jako narzędzie zwiększające produktywność

Patrząc na relacyjne bazy danych i pracę z nimi, które stanowi wąskie gardło w  rozwoju aplikacji debugowanie zapytań SQL lub złożonych obiektów ORM często pochłania godziny czasu programistycznego. Prisma ORM ułatwia programistom zrozumienie zapytań do bazy danych, dostarczając bezpiecznie typowo API do przesyłana zapytań do bazy danych, które zwracają obiekty JavaScript, więc wnioskując głównym powodem używania Prisma ORM jest zwiększenie produktywności programistów. Poważnym problemem narzędzi bazodanowych, istniejących w ekosystemie Node.js i  TypeScript jest konieczność dokonania poważnego wyboru pomiędzy produktywnością, a  kontrolą. W czystym języku zapytań SQL istnieje pełna kontrola, lecz niska produktywność z powodu czystych ciągów SQL do bazy danych, które jest uciążliwe oraz wiąże się z dużym nakładem pracy. Kolejnym problemem jest brak bezpieczeństwa typów dla wyników zapytania, które można ręcznie wprowadzić, ale znów wymaga to ogromnego wkładu pracy oraz dużych przeorganizowań za każdym razem, gdy zmienia się schemat bazy danych lub zapytania, aby zachować synchronizację typów. Istnieją również kreatory zapytań SQL w których jest wysoka kontrola i średnia produktywność. Największym ich minusem jest ciągłe myślenie ze strony programisty w kategoriach SQL co wiąże się z  kosztem poznawczym oraz praktycznym przekładaniem danych relacyjnych na obiekty. OMR-y służą głównie do przejścia przez różnice między obiektami, a podstawowymi elementami bazy danych czyli relacją.[[7]](#footnote-7)

TypeORM to narzędzie ORM, które działa na różnych platformach, takich jak NodeJS, Browser, Cordova, PhoneGap, Ionic, React Native, NativeScript, Expo i Electron, i może być używane zarówno z TypeScript, jak i JavaScript (ES2021). Jego głównym celem jest wspieranie najnowszych funkcji JavaScript oraz oferowanie dodatkowych funkcji, które ułatwiają tworzenie aplikacji korzystających z baz danych - od małych aplikacji z kilkoma tabelami po rozbudowane aplikacje korporacyjne z wieloma bazami danych. TypeORM obsługuje zarówno wzorce Active Record, jak i Data Mapper, co odróżnia go od innych ORM-ów JavaScript. Dzięki temu można tworzyć wysokiej jakości, luźno powiązane, skalowalne i łatwe w utrzymaniu aplikacje w bardzo efektywny sposób. TypeORM czerpie inspirację z innych popularnych ORM-ów, takich jak Hibernate, Doctrine i Entity Framework.[[8]](#footnote-8)

Bookshelf to lekki JavaScriptowy ORM dla Node.js, zbudowany na bazie Knex SQL query buildera, obsługujący PostgreSQL, MySQL i SQLite3. Oferuje interfejsy oparte na Promisach i callbackach, wsparcie dla transakcji oraz elastyczne ładowanie relacji, w tym eager/nested-eager i asocjacje polimorficzne. Jego prosty, literacki kod jest łatwy do zrozumienia i rozszerzania, nie narzuca konkretnego schematu walidacji, a także umożliwia przejście do surowego interfejsu Knex w przypadku niestandardowych zapytań. Bookshelf czerpie inspiracje z wzorca Data Mapper, dostarczając bibliotekę do typowych zadań związanych z zapytaniami do baz danych oraz zarządzaniem relacjami między obiektami w JavaScript.

## Rest Api

REST (z ang. Representational State Transfer) jest to architektura używana do  projektowania usług używanych na wielu różnych platformach oraz środowiskach. Ma na celu wsparcie interoperacyjności oraz sieci WWW (z ang. World Wide Web). Nieskończoność dostępu i uniwersalność konsumpcji na różnych platformach to dwie kluczowe cechy tej architektury. Jest powszechnie stosowana sposobem publikowania usług w internecie. API (z ang. Application Programming Interface) czyli interfejsy programowania aplikacji oraz REST są standardową częścią projektowania mikro usług.

REST API jest znane jako API internetowe, które składa się z punktów końcowych. Każdy punkt jest konkretnie zaimplementowaną funkcjonalnością procesu biznesowego. Interfejsy te są dostępne za pomocą protokołu HTTP (z ang. Hypertext Transfer Protocol), zawierającego standardowe zapytania takie jak: GET, POST,DELETE oraz PUT. RESTful API jest wywoływane za pomocą adresu URI (z ang. Uniform Resource Identifier).

Wyzwaniem jakim stało przed REST API było zdefiniowanie standardowego formatu komunikacji czyli zapytanie (z ang. Request) i odpowiedzi (z ang. Response). Na  początku stosowano nieformalny tekst opisu interfejsów. Następnie pojawiły się dokumenty JSON (z ang. JavaScript Object Notation), który był standardem jako zwykły tekstm łatwy do identyfikacji oraz przetwarzania dla maszyn w różnych sieciach i  platformach. Jednym z wyzwań pozostaje stworzenie standaryzowanego sposobu opisu usług REST. Specyfikacja OpenAPI jest jednym z rozwiązań tego dylematu. Luźne powiązania i elastyczny dostęp do usług REST mogą otwierać szerokie możliwości przesłania niepoprawnych danych wejściowych w żądaniu, co może prowadzić do błędów w usłudze, które mogą zostać niezauważone podczas standardowych procesów testowych i  statycznej analizy kodu. Różne dodatkowe biblioteki dostarczanych przez różnych dostawców stosowane w procesie tworzenia interfejsów REST, wprowadza dodatkową złożoność w testowaniu. W takim razie wykrywanie i rozwiązywanie ewentualnych błędów w usłudze staje się kluczowe dla zapewnienia jej stabilności, zwłaszcza gdy jest ona kluczowa dla danego przedsiębiorstwa obsługującego klienta końcowego. Niezależnie jakie dane zostaną przesłane jako żądanie, odpowiedź powinna być dostarczona w sposób elegancki, minimalizując potencjalne zakłócenia i błędy czasu wykonania. Różne wysiłki badawcze zostały podjęte w celu złagodzenia wyzwań związanych z testowaniem interfejsów REST API poprzez wprowadzenia różnych platform i automatycznego generowania podejść do testów jednostkowych. Jednak nadal są braki w przeglądu stanu wiedzy.[[9]](#footnote-9)

Na rynku są też inne podejścia do tworzenia interfejsów programistycznych, są nimi na przykład GraphQL czy gRPC. Zestawiając je ze sobą można dopatrzeć się różnic w  pomiarach czasu odpowiedzi wydajności czy też ilości przesłanych danych. Jeśli rozpatrywanym przypadkiem była by ilość pobieranych danych to możemy zauważyć, że  interfejs GraphQL najlepiej sobie z tym poradził, jednakże jeśli dane byłyby średnie oraz małej wartości pobranych rekordów, to wydajność REST API i GraphQL byłyby bardzo zbliżone z niewielką przewagą na korzyść REST API. Przewaga REST API również dotyczy małej ilości danych w przypadku prostych struktur zapytań. Technologia gRPC osiągnęła słabsze wyniki wydajnościowe dwu, a czasem nawet czterokrotnie. GRPC wyróżnia się od  REST API i GraphQL pobieraniem o połowę mniej danych podczas zapytań do serwera. Gdyby rozważać przykład, którym jest dodanie użytkownika do bazy danych, zauważalna jest przewaga REST API nad innymi technologiami, który osiąga w kryterium wydajnościowym i czasu odpowiedzi lepsze wyniki, dzięki czemu osiąga dwukrotnie lepsze wyniki. Możemy zauważyć, że dokonanie ostatecznego wyboru interfejsu jest nieco bardziej skomplikowane niż mogłoby się wydawać na pierwszy rzut oka. Przy podejmowaniu decyzji co do sposobu implementacji aplikacji, trzeba wziąć pod uwagę wiele czynników, takich jak wielkość danych przesyłanych pomiędzy serwerem, a użytkownikiem, ilość użytkowników oraz wydajność urządzeń. Rozważyć również należy czy zależy wygodę w korzystaniu z  interfejsu. Wielu programistów najczęściej wybiera REST API, ze względu na swoją popularność oraz wygodę lecz nie zawsze jest to najlepszy wybór. REST API może być najlepszym wyborem w przypadku dużej ilości danych, ale również dobrze sprawdziłby się GraphQL, jednakże przy małej ilości danych warto również rozważyć korzystanie z gRPC.[[10]](#footnote-10)

## Języki programowania

Systemy informatyczne nie mogłyby mieć miejsca bez języków programowania. Wraz z rozwojem informatyki powstaje wiele ich rodzajów, służących do różnych rzeczy. Tak na przykład język JavaScript służy do tworzenia stron internetowych ze strony wizualnej (front-end) oraz ze strony obsługi serwera itp. (back-end). Jest to narzędzie które powstało już w latach dziewięćdziesiątych ubiegłego wieku i przeszło długą drogę rozwoju. Jednym z najważniejszych wydarzeń jest integracja go z Wasm (WebAssembly), który jest formatem instrukcji binarnych umożliwiających wysokowydajne wykonanie kodu w  przeglądarkach internetowych. Pozwala programistom na uruchomienie kodu napisanego w językach takich jak C, C++ czy Rust obok JavaScriptu, otwierając nowe możliwości dla aplikacji internetowych co zapewnia ścieżkę do poprawy wydajności aplikacji internetowych. Programiści mogą wykorzystać Wasm do przeniesienia zadań wymagających dużych zasobów takich jak renderowanie czy obliczanie w celu osiągnięcia praktycznie natywnej szybkości w przeglądarce. JavaScript odgrywa kluczową rolę w  budowaniu progresywnych aplikacji webowych (PWA), umożliwiając dynamiczne i  interaktywne funkcję, które sprawiają, że aplikacje są atrakcyjne dla użytkownika. W  miarę wzrostu popularności tych aplikacji, programiści muszą zmierzyć się z  dostosowaniem do nowego paradygmatu budowania aplikacji internetowych. Przyszłość JavaScriptu jest ściśle wiąże się z nieustannie zmieniającym się krajobrazem ram i bibliotek. Wiodące biblioteki takie jak React, Angular czy Vue.js mają znaczący wpływ w jaki developerzy projektują interfejsy użytkownika oraz aplikacje jednostronicowe. Dodatkowo na rynku pojawiają się nowe biblioteki, które muszą sprostać konkretnym potrzebą takim jak zarządzanie stanem, routowanie, renderowanie po stronie serwera. Dlatego dla  programistów jest tak ważne śledzenie najnowszych trendów w celu pozostaniem konkurencyjnym w dynamicznie rozwijającej się branży tworzenia aplikacji internetowych. Dodatkowo język JavaScript jest językiem bardzo elastycznym wykorzystywanym w  rozwoju międzyplatformowym. Biblioteka jak na przykład React Native oparta na tym języku pozwala na budowanie aplikacji mobilnym na urządzenia z androidem jak i oparte na ios, a Electron umożliwia programistom budowanie aplikacji desktopowych. To  wszystko sprawia, że język JavaScript jest atrakcyjny dla programistów, którzy za  pomocą jednego kodu mogą dotrzeć do wielu platform.[[11]](#footnote-11)

Istnieje również język oparty na JavaScript i jest nim TypeScript. Różnica pomiędzy tymi językami jest to, że TypeScript jest rozszerzony o system typów. By został odczytany przez przeglądarkę musi zostać skompilowany do JavaScriptu. Wynalazcą tego języka była firma Microsoft z wkładem Googla. Jest on najbardziej udanym spośród języków stopniowo typowanych oraz najpopularniejszym językiem kompilowanym do JavaScriptu. Warto wspomnieć, że każdy program napisany w JavaScript jest również programem TypeScript. Dodanie do TypeScriptu typów pomaga zaoszczędzić wiele godzin debugowania. Gdy firmy tworzące duże aplikacje mające miliony linijek kodu, muszą coś w nim zmienić typy pozwalają dokładniej poznać co spowodują zmiany dla pewnych modułów na poziomie

kompilacji.[[12]](#footnote-12)

Najpopularniejszym językiem wykorzystywanym do tworzenia aplikacji internetowych jest PHP. Wiele aplikacji już istniejących oraz systemów wykorzystuje właśnie ten język co jest najlepszą motywacją do ciągłego rozwoju. Ten język również posiada wiele szkieletów jak na przykład Laravel, który jest wykorzystywany do niewielkich projektów, niewymagających wielkiego wkładu finansowego. Kolejnym popularnym szkieletem jest Symfony, przeznaczony głównie do projektów złożonych i wymaga większych wkładów finansowych.[[13]](#footnote-13)

Python to również bardzo popularny język, szczególnie w dzisiejszych czasach ze  względu na sztuczną inteligencję oraz uczenie maszynowe. Jest to język wysokiego poziomu, który powinien być dla ludzi prosty do pisania oraz czytania oraz dla komputerów do przetwarzania i odczytywania. Python używa interpretera napisanego w jezyku C do  przekształcenia kodu na język maszynowy. [[14]](#footnote-14) Sam język ma wiele zalet takich jak działanie na wielu platformach i systemach operacyjnych czy istniejące wiele bibliotek posiadające Pythonowe API, a nawet możliwość korzystania z bibliotek języka C (.so, .dll). Dodatkowo Python jest wygodniejszy w nauce niż wiele innych języków, ponieważ kod realizujący tą samą funkcjonalność przy podobnym poziomie obsługi błędów i czytelności, zazwyczaj jest krótszy niż kod w C, co ułatwia jego prezentację i omówienie.[[15]](#footnote-15)

Java jest również jednym z najpopularniejszych i najlepszych języków używanym na całym świecie, który pozwala tworzyć programy na poziomie przedsiębiorstwa. Język ten oparty jest na języku Oak, wynalezionym przez Sun Microsystems na początku lat dziewięćdziesiątych. Celem tego języka było stworzenie nowej generacji inteligentnych urządzeń, które działają niezależnie od systemu operacyjnego. By to osiągnąć programiści z Sun Microsystems planowali użyć języka C++, ale nie doszło to do skutku. Oak nie odniósł jednak sukcesu i już w 1995 roku zmienili nazwę na Java oraz wprowadzili zmiany w celu rozwijania WWW. Java jest językiem zorientowanym obiektowo (OOP – Object-Oriented Programming), bardzo podobnym do C++. Został jednak uproszczony przez usunięcie części, które są najbardziej podatne na błędy. Kod napisany w języku Java jest  konwertowany na kod bajtowy po kompilacji, a kod bajtowy jest odczytywany przez interpreter.[[16]](#footnote-16)

C++ jest wielozadaniowym językiem programowania. Stworzony w Bella Labs w  1979 roku przez Biyarne'a Stroustrupa w celu rozszerzenia możliwości języka programowania C oraz wprowadzić programowanie obiektowe. Początkowo został nazwany „C with Classes” co znaczy C z klasami, które są używane do programowania obiektowego, lecz w 1983 roku został zmieniony na nazwę obecną C++. Język ten ma zdolność kompilowania programów napisanych w języku C, lecz bez wzajemności, gdyż język C nie posiada takiej funkcji. Najczęściej język ten jest używany do w produkcji komponentów związanych z systemami operacyjnymi. Jest też używany do produkcji gier, tworzenia programów kilent-serwer.[[17]](#footnote-17)

## Interfejs użytkownika frameworki

Angular, React czy też Vue.js są najpopularniejszymi szkieletami dla JavaScriptu. Są wykorzystywane do do tworzenia interfejsów graficznych za pomocą SPA (z ang. Single Page Application). Podając przykład Reacta komponenty wielokrotnego użytku wykorzystane w interfejsie opisywane za pomocą JSX, który jest formatem zapisu kodu HTML wewnątrz języka JavaScript, jeżeli chodzi o TypeScript jest to TSX. W interfejsie Vue używane są komponenty, które wykorzystują szablony HTML do swojej implementacji. Angular, który jest następcą Google Angular JS, jest rozwijany przez ten sam zespół. Budowa Angulara narzuca programiście podział komponentów na trzy elementy, którymi są logika, interfejs użytkownika oraz style, podczas gdy Vue trzyma w jednym pliku wszystkie komponenty. W większości nowoczesnych stron internetowych wykorzystuje się  JS (JavaScript) do zwiększenia responsywności i interaktywności. W tradycyjnych witrynach, nowy dokument HTML pobierany jest za każdym razem, kiedy użytkownik przechodzi na nową stronę, co jest powolne. Natomiast w nowoczesnych szkieletach elementy są pobierane tylko wtedy gdy elementy zostaną zmienione co znacznie przyspiesza działanie strony.[[18]](#footnote-18)

Angular jest frameworkiem, który występuje głównie w dwóch wersjach jakimi są  Angular JS oraz Angular 2+. Angular JS jest to starsza wersja oparta na języku JavaScript, niekontynuowaną już przez twórców. Natomiast Angular 2+ jest nowszym rozwiązaniem, opratym na TypeScript i HTML. Angular 2+ jest wykorzystywany do  tworzenia aplikacji klienckich w HTML, CSS, JavaScript lub TypeScript. Zapewnia aplikacjom czytelną i elastyczną strukturę co ułatwia jej zrozumienie i utrzymanie co  sprawia, że ten sam kod można wykorzystywać w różnych aplikacjach. Aplikacje te składają się z modułów NgModule, które są kontenerami dal kodu związanego z daną domeną aplikacji. Zapewniają kontekst kompilacji swoim elementom i zawierają składniki dostawców usług i inne pliki. Aplikacje napisane przy użyciu Angulara posiadają moduł główny nazywany AppModule, który inicjuje uruchomienie aplikacji oraz może zawierać wiele modułów podrzędnych. Angular częściowo opiera się na wzorcu MVC (z ang. Model View Controller), w którym komponent pełni rolę kontrolera za to szablon – widoku. Szkielet ten zapewnia kontrolę nad skalowaniem aplikacji, umożliwiając budowę modeli danych w RxJSm Imutable.js lub innym modelu push w celu zapewnienia szybkości i  wydajności aplikacji internetowych.[[19]](#footnote-19)

React który jest frameworkiem pierwotnie rozwijanym przez Facebook jest  odpowiednikiem w języku JavaScript dla biblioteki XHP w PHP, umożliwiającej tworzenie niestandardowych komponentów. Zadebiutował w maju 2013 roku, skupiając się głównie na renderowaniu interfejsu użytkownika. Jego największymi zaletami jest niewielki nakład nauki i używania i lżejsza konstrukcja zestawiając go z innymi technologiami. Rozwój jest istotny w ramach zmiany podejścia do tworzenia aplikacji internetowych. Jego  największym osiągnięciem jest optymalizacja funkcji DOM, który jest sposobem reprezentacji XML i HTML, co ma znaczenie ponieważ manipulacja funkcjami DOM jest  kosztowna pod względem zasobów obliczeniowych. Został zaprojektowany tak, żeby  manipulacje DOM były jak najbardziej zminimalizowane poprzez wykorzystanie wirtualnego DOM co przyśpiesza działanie aplikacji.[[20]](#footnote-20)

Vue jest to nowoczesny framework JavaScript, często porównywanym do Angulara czy Reacta. Stworzony została przez firmę Google i jest bardziej skoncentrowany na  warstwie widoku oraz jest bardziej uproszczony od Angulara. W przypadku Vue również wykorzystywany jest wirtualny DOM tak jak w przypadku Reacta, ale bez konieczności korzystania z oddzielnej modułowej biblioteki. Opisywany jest jako framework postępowy, który jest elastyczny i skalowalny, nadając się do małych projektów jak i  do  pełnowymiarowych aplikacji jednostronicowych. Jego zaletą jest to, że nie jest ściśle związany z wzorcem MVVM. Aktywni programiści cały czas rozwijają Vue i dostosowują go do potrzeb społeczności. Framework posiada automatyczne śledzenie zależności komponentów, optymalizując tym samym czas i pracę programisty. Z powodu swojej szybkości i efektywności umożliwia skupienie się na istotnych funkcjonalnościach aplikacji.[[21]](#footnote-21)

Oceniając wydajność technologii używając testów oraz analizę wsparcia technicznego oferujących przez developerów danego frameworka można zauważyć, że  najszybciej rozwijającym oraz uruchamiającym się szkieletem jest Angular. Zawdzięcza to efektywnemu przetwarzaniu procesów związanych z uruchamianiem aplikacji, radzi sobie również z rożnymi przeglądarkami internetowymi. Jeśli weźmiemy pod uwagę framework wykorzystywany do tworzenia małych i średnich aplikacji to będzie nim Vue. Proces budowy i uruchamiania aplikacji w Vue jest podobny do tego w React, wiec ciężko stwierdzić, który szkielet działa szybciej. Angular otrzymujący wsparcie od strony Google jest często aktualizowany, co gwarantuje wydawanie nowych wersji co 6 miesięcy. Developerzy przywiązują dużą wagę do udoskonaleń tego szkieletu. W przypadku Reacta jego rozwój następuje poprzez interakcję ze społecznością programistów. Jest  aktualizowany poprzez zmiany wysyłane przez twórców oraz zewnętrznych twórców, które muszą przejść przez proces recenzji na platformie GitHub. Każdy programista może wprowadzić zmiany, które muszą być zaakceptowane przez twórców, którzy reagując krytycznie na błędy znalezione w starszych wersjach. Vue z drugiej strony jest najmłodszym z wymienionych frameworków i chociaż jest stworzony przez byłego developera Angulara, nie cieszy się dużą społecznością profesjonalistów, którzy wspierają tą technologie. Licencja open source pomaga ciągłemu wzrostu społeczności korzystającej z Vue.[[22]](#footnote-22)

Jeśli poddalibyśmy testom wszystkie wymienione szkielety, podzielonym na siedem kategorii opartych na działaniach CRUD (Create, Read, Update, Delete), można zauważyć, że czas tworzenia się wierszy podczas ładowania strony najlepiej prezentuje się w  frameworku React, ze względu na wydajność mikro serwisów użytych w tej technologii. Jeśli rozpatrywać odświeżanie strony to wszystkie szkielety wypadają podobnie, co sugeruje niemal identyczną wydajność procesów przetwarzania podglądu załadowanych informacji. Angular wypada najszybciej, kiedy przychodzi do wyboru wiersza poprzez jego zaznaczenie. Szybszy jest o średnio 30%. Biorąc pod lupę edycję losowych wierszy, React okazuje się najmniej wydajnym szkieletem. Spowodowane jest to czasem wykonania zewnętrznych funkcji edycji danych. Patrząc na usuwanie wierszy od pięćdziesięciu do  tysiąca rekordów jest się w stanie zauważyć, że Angular radzi sobie bardzo szybko z  usuwaniem małych ilości danych, sugerując, że częste oczyszczanie danych z małych pakietów jest najbardziej optymalne w tej technologii. Patrząc na szybsze usuwanie większych pakietów danych, React i Vue radzą sobie najlepiej przy czym czas miał rozkład logarytmiczny. Testując utworzenie dziesięciu tysięcy rekordów, Vue wykazuje się  największą wydajnością co prawdopodobnie wynikało z efektywności procesów używanych w tej technologii. Z drugiej strony React jest najwolniejszym frameworkiem w  tworzeniu danych, głównie z powodu korzystania z zewnętrznych funkcji. Framework dzięki dobrej komunikacji między komponentami zbliżył się do Vue. Rozpatrując ilość kodu który jest potrzebny do utworzenia testowych aplikacji, szkielet Vue wymaga go najmniej, choć różnice między konkurencyjnymi technologiami są niewielkie.[[23]](#footnote-23)

Vue jako szkielet interfejsów użytkownika wydaje się być najszybszy, osiągając najlepszą wydajność dzięki wykorzystaniu mikro procesów, sprawnie komunikujących się oraz efektywnie operujących na dużych zbiorach danych. Na drugim miejscu plasuje się na  drugim miejscu pod względem szybkości zwłaszcza jeśli chodzi o małe zbiory danych. React, który uplasował się na ostatnim miejscu z wymienionych szkieletów, wypada najwolniej, zwłaszcza na operacjach związanych z edycją danych, chociaż jego wewnętrzne funkcje działają szybko. Warte zauważenia jest to, że zewnętrzne funkcjonalności mogą spowolnić całą aplikację. Podsumowując, wszystkie frameworki mogą być wydajne, należy jednak być na bieżąco z nowymi technologiami i ich dokumentacją.[[24]](#footnote-24)

## Systemy kontroli wersji

VCS (z ang. Version Control Systems), czyli systemy kontroli wersji jest oprogramowaniem zezwalającym na przeglądanie oraz gromadzenie kolejnych wersji kodu źródłowego. Patrząc wstecz systemem kontroli wersji cieszącym się największą popularnością przez wiele lat był CVS, a później zgodny z nim SVN (z ang. Subversion). Lecz w dzisiejszych czasach największą popularnością cieszy się stworzony przez Linusa Torvaldsa system o nazwie Git. Różni się on od poprzedników tym, że Git jest rozproszonym systemem kontroli wersji co oznacza, że projekty magazynowane są na głównym serwerze, ale również na indywidualnych stanowiskach pracy. Narzędzie Git umożliwia właśnie ich  scalanie i synchronizację[[25]](#footnote-25).

Programiści używający Gita na co dzień muszą posługiwać się specyficzną terminologią, która to określa poszczególne czynności, powodując, że na początku stosowania systemu, może utrudniać zrozumienie, co tak naprawdę się dzieje. Miejscami gdzie profesjonaliści przetrzymują kolejne wersje kodu źródłowego, noszą imię repozytoriów. Istnieją ich dwa rodzaje lokalne oraz zdalne. Częstym skrótem używanym do  skrócenia długiego słowa repozytorium używa się słowa „repo”. Pliki zapisywane są  w  trzech różnych lokalizacjach: w folderze projektu nazywanym również jako katalog roboczy (z ang. working directory), w lokalnym „repo” znajdującym się w podkatalogu .git w folderze projektu oraz w zdalnym repozytorium przechowywanym na serwerze, na  przykład na platformach takich jak: GitLab, GitHub, Azure DevOps, Bitbucket czy inne. Wszelkie modyfikacje plików w projekcie są monitorowane, co znaczy, że każda zmiana pliku jest śledzona. Cyklicznie należy zaakceptować te zmiany poprzez dokonanie jednego zatwierdzenia co w terminologii Gita nazywane jest commitem. Czasownik angielskiego pochodzenia commit oznacza przesładnie zmienionej części projektu do lokalnego repozytorium. W języku polskim proces ten nazywa się zatwierdzeniem zmian. Następnie możemy wypchnąć zmiany do zdalnego repozytorium co w terminologii znaczy commit push. Przed wypchnięciem zmian powinno się zobaczyć czy repozytorium nie ma  na  serwerze nowszej wersji projektu, działanie te nazwane jest fetch, na polski pobranie. Zestawienie push i fetch używane jest do synchronizacji repozytorium. Każdy programista będący członkiem zespołu może pobrać najnowszą wersję projektu ze zdalnego repozytorium co w terminologii nazywa się pull. Klonowanie repozytorium oznacza pobraniem całego projektu ze zdalnego „repo” na komputer, na którym nie było żadnej wersji projektu. Należy zwrócić uwagę na trzy główne korzyści korzystania z systemu kontroli wersji. Najważniejszą zaletą jest regularne tworzenie kopii kodu na komputerze programisty, jak zarówno na zdalnym serwerze. Kolejną zaletą jest możliwość tworzenia wersji, co znaczy, że specjaliści mają dostęp do nie tylko najnowszej wersji produktu jak  również do w całości oraz wybranych plikach. Wersjonowanie powiązane jest z taką zaletą, że każda zmiana w zdalnym repozytorium jest podpisana członkiem zespołu dokonującym zmiany. To wszystko pozwala na skuteczne zarządzaniem projektu, śledzenie jego rozwoju bez konieczności osobistego monitorowania go na każdym stanowisku pracy oraz efektywne zarządzanie zespołem pracującym nad projektem dzięki posiadaniu wiarygodnych danych o jego postępie.[[26]](#footnote-26)

Git jest narzędziem wiersza poleceń służącym do tworzenia własnego serwera kontroli wersji. Istnieje wiele publicznie dostępnych platform umożliwiających łatwe utworzenie zdalnego repozytorium bez potrzeby konfiguracji. Platformy te oferują interfejs internetowy do przeglądania repozytoriów, które posiadają wiele dodatkowych funkcji na  przykład zarządzanie uprawnieniami dostępu do gałęzi. Najpopularniejsze serwisy, które oferują trzymanie repozytoriów to na przykład GitHub, GitLab, Azure DevOps Services (kiedyś Visual Studio Team Foundation Services) czy Bitbucket. Atlassian, który jest właścicielem Bitbucket, dostarcza popularny klient Git nazwany SourceTree, posiadający graficzny interfejs użytkownika, który dziła nie tylko z Bitbucket, ale również z innymi usługami Git. SourceTree jest często wykorzystywany obok wbudowanego klienta w  nowszych wersjach Visual Studio. Wiele operacji dokonywanych prze Gita takich jak  akceptowanie czy wysyłanie commitów, można wykonywać bezpośrednio z Visual Studio, kiedy to SourceTree jest używany do bardziej zaawansowanych działań lub  rozwiązywania problemów.[[27]](#footnote-27)

## Modelowanie systemów

Unified Modeling Language czyli w skrócie UML jest Zunifikowanym językiem Modelowania, który jest graficzny i służy do specyfikowania, wizualizowania, konstrukcji oraz dokumentacji fragmentów w systemach informatycznych. To umożliwia standaryzację procesu tworzenia wycinków systemu, które obejmują różnorodne elementy, takie jak funkcję systemowe lub schematy baz danych. Język ten integruje zalety z różnych obszarów, takich jak modelowanie danych poprzez diagram związków encji, modelowanie przepływów danych za pomocą diagramów przepływu, analiza obiektowa związana z  modelowaniem obiektowym oraz zarządzanie złożonością przy użyciu komponentów. Początkowo, Zjednoczony Język Modelowania (UML) był używany gównie do tworzenia systemów informatycznych. Jednakże, ze względu na jego efektywność, jako narzędzia komunikacji, zaczął być stosowy równocześnie w innych dziedzinach takich jak  telekomunikacja, transport, przemysł obronny i lotniczy czy usługi bankowe. Zazwyczaj UML wykorzystywany jest wraz z jego reprezentacją graficzną gdzie jego elementy są  reprezentowane przez symbole, które są używane do tworzenia diagramów. Dzięki temu stworzone schematy umożliwiają precyzyjne śledzenie procesów. Od wersji UML 2.5.1 wyróżnia się trzynaście głównych typów diagramów oraz trzy rodzaje interakcji: diagramy struktury, takie jak diagramy klas, obiektów, struktur złożonych, komponentów, pakietów oraz wdrożeń. Diagramy zachowania takie jak diagramy przypadków użycia, maszyn stanowych oraz czynności oraz diagramy interakcji, takie jak diagramy przeglądu interakcji, sekwencji, komunikacji i czasowe. UML nie jest narzędziem programowania wizualnego, lecz modele stworzone w tym języku, mogą być powiązane z różnymi językami programowania. Istotnym faktem jest to, że model stworzony w UML może być przekształcony, na przykład, w tabele relacyjnej bazy danych, a także na podstawie istniejącej implementacji można stworzyć model graficzny. Warto zaznaczyć, że poziom szczegółowości diagramów zależy od fazy projektowania oprogramowania. Ogólnego diagramu można użyć na etapie analizy problemu, jednakże w trakcie dokumentacji technicznej konieczne jest użycie bardziej szczegółowych diagramów.[[28]](#footnote-28)

W standardzie UML każdy obiekt reprezentowany jest przez prostokąt, który zawiera jego elementy składowe. Prostokąt służy do umieszczania nazw elementów, nazw z polami i metodami lub samych nazw z ich polami. Można oraz zalecane jest dołączenie informacji o typie, typie zwracanym lub argumentach do składników elementów. W dodatku, składniki klasy mogą być oznaczone różnymi modyfikacjami dostępu takimi jak: „+” oznaczający składnik publiczny, „#” oznaczający składnik chroniony, „-” oznaczający składnik prywatny oraz „~” oznaczający składnik dostępny w obrębie projektu (package).[[29]](#footnote-29)

Oznaczenia metod abstrakcyjnych w klasie abstrakcyjnej powinny być wyróżnione kursywą oraz podkreśleniem. Interfejsy są przedstawione podobnie jak klasy, lecz nazwy interfejsów muszą być poprzedzone słowem interface. Implementacja interfejsu w danej klasie jest oznaczona pustym białym grotem strzałki na końcu przerywanej linii, a klasa, które implementuje interfejs musi zaimplementować jego metody. Rozważając klasy abstrakcyjne, klasa, która dziedziczy musi implementować abstrakcyjne metody. Diagramy klas UML mają zaletę opisywania związków między klasami. Relację pomiędzy nimi mogą być określone z uwzględnieniem cech krotności, takich jak jeden obiekt, od zera do trzech obiektów, „\*” - dowolna ilość obiektów, „3 - \*” - od trzech do dowolnej ilości obiektów. Ważnym jest umieszczenie krotności po obu stronach zależności. Jeśli krotność nie została podana zakłada się, że wynosi jeden. Związki między klasami mogą być zależnościami, agregacjami częściowymi, asocjacjami, agregacjami całkowitymi i dziedziczeniem. Zależność informuje, że klasa musi korzystać z informacji o drugiej i zmiana w jednej klasie może wymagać zmiany w drugiej. Oznaczenia używane w trzech typach „<<call>> dla  operacji wywołanych w jednej klasie przez drugą, „<<create>>” oznacza, że klasa A jest odpowiedzialna za tworzenie instancji klasy B, „<<instantiate>>” oznacza, że obiekt A jest reprezentacją klasy B, „<<use>>” oznacza, że klasa A wykorzystuje klasę B do realizacji swoich funkcjonalności. Zależność jest najbardziej podstawową relacją, w której jedna klasa czasowo wykorzystuje inną i wie o jej istnieniu. Zmiana w jednej klasie, może powodować wymóg zmiany w drugiej, lecz nie zawsze. Projektowanie diagramu z jak najmniejszą liczbą zależności ułatwia rozbudowę projektu.[[30]](#footnote-30)

Diagram klas obejmuje zbiór interfejsów, klas i kooperacji oraz powiązania między nimi. Diagram ten służy do opisywania struktury systemu, skupiając się głownie na relacjach między klasami z pominięciem innych szczegółów. Rozważając bardziej skomplikowane systemy, nie wszystkie elementy muszą być przedstawione na jednym diagramie. Kompletny model systemu jest efektem połączenia wszystkich diagramów, wraz z ich elementami oraz relacjami. Wybranie konkretnych klas do uwzględnienia na diagramie jest świadomym procesem, zależnym od analizy i projektowania systemu.[[31]](#footnote-31)

Diagram przypadków użycia (z ang. use case diagram) jest narzędziem służącym do  przedstawienia funkcjonalności systemu oraz jego relacji z otoczeniem, zawierając interakcje między użytkownikami korzystającymi z systemu oraz usług widocznych z  zewnątrz. Pełni kluczową rolę w projektowaniu systemu składając się z różnych elementów. Opisuje funkcjonalności oraz otoczenie systemu. Przypadki użycia stanowią scenariusze związane z celami użytkowników, definiując oczekiwane zachowania systemu. Istotne są dla analizy systemu i służy jako podstawa do komunikacji między uczestnikami projektu. Poszczególny przypadek użycia posiada określone cechy: nazwa, opis, zależności i relacje, przepływ zdarzeń, wymagania specjalne oraz początkowe i końcowe, diagramy aktywności. Kluczowym elementem jest przepływ zdarzeń to znaczy sekwencja czynności potrzebnych do osiągnięcia funkcjonalności opisanej w danym przypadku użycia.[[32]](#footnote-32)

Podsumowując tworzenie modeli UML służy temu, by lepiej zrozumieć potrzeby klienta i przedstawić działanie programu w sposób zrozumiały dla wszystkich zaangażowanych. Diagramy umożliwiając opisanie zasada funkcjonowania aplikacji oraz  prezentując aplikacje graficznie co pomaga w analizie systemu z różnych perspektyw. Modelowanie w języku UML pozwala projektowanie aplikacji już we wczesnych fazach tworzenia projektu, co umożliwia łatwe rozszerzenie jej o nowe funkcjonalności w  przyszłości. Model UML jest przydatny przy większych projektach, pozwalając programistom na zrozumienie sposobu implementacji oraz działania systemu. Mimo iż że  nie jest to język programowania graficznego, modele UML mogą być powiązane z  różnymi językami programowania, co ułatwia przeniesienie projektu do konkretnej technologii, na przykład w formie tabel w bazie danych.[[33]](#footnote-33)

## Opis przedsiębiorstwa

Największym celem funkcjonowania przedsiębiorstwa jest zaspokojenie potrzeb.

Przedsiębiorstwo na którym skupia się praca dzieli się na dwa obszary z którymi powiązani są interesariusze: zamówienie towaru z hurtowni przez zamawiających, odbiór towaru z hurtowni oraz dystrybucję do klientów na terenie Warszawy. Za ostatnie dwa są  odpowiedzialni kierowcy. Efektywne funkcjonowanie systemów logistycznych powinno być ściśle powiązane z głównymi celami organizacji. Implementacja logistyki w  przedsiębiorstwie z uwzględnieniem tych celów, umożliwia skuteczne planowanie, organizację i kontrolę poprzez szczegółowe wykonywanie zadań, obejmujących wszystkie procesy fizycznego przepływu towarów.[[34]](#footnote-34)

Wraz z rozwojem technologii wzrasta również złożoność procesów gospodarczych co wpływa na ważna rolę informacji w zarządzaniu przedsiębiorstwem. Zarządzanie informacjami obejmuje szereg działań, które tworzą funkcję informacyjną w  przedsiębiorstwie, włączając w to pozyskiwanie, przewarzanie, przechowywanie, wykorzystywanie oraz udostępnianie informacji. Największym wyzwaniem w zarządzaniu informacjami jest ciągłe dostosowywanie się do ciągle zmieniających się technologii informatycznych do sprostania zmieniającym się potrzeba organizacji. Wykorzystywanie informacji staje się istotnym czynnikiem efektywnego funkcjonowania przedsiębiorstwa, zwłaszcza w kontekście zarządzania łańcuchem dostaw. Nowoczesne technologie w tym internet oraz zintegrowane systemy informatyczne znacznie ułatwiają efektywne gromadzenie i zarządzanie zasobami informacyjnymi.[[35]](#footnote-35)

Największym problemem przedsiębiorstwa, którego praca dotyczy jest brak jednolitego systemu informatycznego do wspomagania pracy kierowców którzy są głównym filarem przedsiębiorstwa bo to od nich zależy cały odbiór, przywóz i dystrybucja sprzedawanych towarów. Między kierowcami, a osobami odpowiedzialnymi za zamawianie występują problemy komunikacyjne, czego komunikatory dostępne za darmo nie są w stanie rozwiązać. Jak można zauważyć występują dwa rodzaje komunikacji: bezpośrednia jak na przykład rozmowa czy pośrednia jak na przykład rozmowa przez telefon czyli medium.[[36]](#footnote-36) Wiec właśnie przedsiębiorstwo powinno skupić się na stworzeniu aplikacji ułatwiającą komunikację pośrednią gdyż kierowcy są tylko przez krótki okres czasowy rozważając ramy dnia pracy w biurze, gdzie pracują osoby zamawiające.

Identyfikacja interesariuszy jest ważnym punktem do określenia problemów, potrzeb oraz oczekiwań związanym z zaprojektowaniem systemu wspomagającego komunikację. Interesariuszy, których obejmie projekt można podzielić na dwie grupy: kierowcy oraz  osoby zamawiające.[[37]](#footnote-37) Rolą kierowców jest odebranie informacji od osób zamawiających odnośnie towaru zamówionego na dany dzień, pojechaniem po odpowiedni towar oraz dowóz go do klienta. Czas w tym wszystkim ma znaczenie ponieważ ważnym celem przedsiębiorstwa jest dowóz do 24 godzin od zamówienia. Osoby zamawiające odpowiedzialne są za otrzymywanie zamówień od klienta, zamówienie ich w odpowiednich hurtowniach na dowóz bezpośredni do firmy, a jeśli jest to niemożliwe to komunikowanie się z kierowcą w celu potrzeby odebrania towaru z hurtowni.

* 1. **Technologie do stworzenia dobrej aplikacji internetowej**

Dobrym wyborem w nowoczesnym tworzeniu aplikacji internetowych jest język TypeScript. Jest to dobry wybór ze względu na jego elastyczność tzn. można tworzyć w nim zarówno front-end aplikacji oraz back-end. Do tego istnieje również wiele ciekawych szkieletów programowania z których nowocześnie korzysta coraz więcej softwarehousów czy też dużych przedsiębiorstw. Jest to język, który jak wyżej napisane wymusza statyczne typowanie dzięki temu podczas pisania wiemy od razu błąd wynikający z błędnego przekazania typów, a więc kod jest łatwy w debugowaniu oraz naprawianiu błędów.

Next.js jest popularnym frameworkiem, szczególnie w świecie aplikacji internetowych opartych na React.js. Jego główną zaletą jest możliwość tworzenia zarówno front-endu, jak i back-endu aplikacji w jednym miejscu, co zwiększa wydajność i ułatwia utrzymanie kodu. Dodatkowo, jego integracja z językiem TypeScript zapewnia większą pewność i bezpieczeństwo podczas pisania kodu. Next.js oferuje również wiele zaawansowanych funkcji, takich jak generowanie statycznych stron, dynamiczne routowanie czy obsługa danych statycznych i dynamicznych. Dzięki tym cechom, jest popularnym wyborem wśród programistów, zarówno dla małych projektów startupowych, jak i dla dużych aplikacji korporacyjnych.

Prisma ORM jest świetnym rozwiązaniem do zarządzania danymi w aplikacji opartych na Node.js i TypeScript. Posiada intuicyjny interfejs oraz bogaty zestaw funkcji, które umożliwiają programistom łatwe i efektywne tworzenie, manipulację i odczytywanie baz danych. Prisma ORM obsługuje wiele popularnych systemów bazodanowych takich jak PostgrsSQL, MySQL czy mongoDB co sprawia, że jest elastycznym rozwiązaniem dla różnorodnych projektów. Jego zalety to także automatyczne generowanie silnych typów TypeScript na podstawie schematu bazy danych, co zapewnia większą pewność i bezpieczeństwo podczas korzystania z danych. Dzięki swojej wydajności i funkcjonalności, Prisma jest często wybieranym narzędziem przez programistów, którzy cenią sobie prostotę i efektywność w zarządzaniu danymi w aplikacjach.

MongoDB charakteryzuje się szybkim czasem odpowiedzi i łatwą skalownością co sprawia, że ta baza jest wybierana najczęściej do projektów, które wymagają dużego przepływu danych i szybkiego dostępu do nich. Ponadto, MongoDB oferuje wiele zaawansowanych funkcji, takich jak indeksowanie, agregacje danych i replikacja, co czyni ją wszechstronnym narzędziem dla różnorodnych zastosowań. Dzięki swojej popularności i wsparciu społeczności, MongoDB jest powszechnie stosowanym rozwiązaniem w świecie aplikacji internetowych i big data.

Wybrane technologię są bardzo dobrym wyborem do stworzenia aplikacji internetowej, ponieważ nie są one przestarzałe, a ciągle rozwijające się poprzez społeczność programistyczną czy też przedsiębiorstwa. Mają bardzo duże społeczności organizujące niesamowite wydarzenia na żywo, na których też są przedstawiane najnowsze ulepszenia oraz przyszłość rozwoju poszczególnych technologii. Dzięki dobrze sprecyzowanym i opisanym dokumentacją korzystanie z nich jest bardzo łatwe jeśli zna się język przez który jest obsługiwana. Sam TypeScript jest bardzo dobrym rozwiązaniem dla JavaScript deweloperów dzięki typom statycznym co powoduje, że pracuje się szybciej poprzez eliminację błędów oraz mniej chaotycznie. A więc można śmiało stwierdzić, że istnieją technologię dzięki którym uda się stworzyć dobra i wydajną aplikację internetową.

# ROZDZIAŁ 2 ANALIZA OBSZARU APLIKACJI INTERNETOWYCH WSPOMAGAJĄCYCH PRACĘ KIEROWCÓW

## Kontekst biznesowy oraz technologiczny



Przedsiębiorstwo opisywane w pracy jest zarejestrowane na rynku od 2015 roku co  pozwoliło na ugruntowanie sobie pozycji oraz zdobycie zaufania i pozyskania stałych klientów. W przedsiębiorstwie ważnymi aspektami są jakość świadczonych usług, jakość sprzedawanych towarów, komunikacja z klientem oraz czas dostawy na terenie warszawy, który oscylowano na 24 godziny po złożeniu oraz zaakceptowaniu zamówienia.

Dostawa jest jednym z ważniejszych elementów na rynku e-commerce. Tak więc czas świadczonej usługi jest bardzo ważnym czynnikiem przy ocenie zadowolenia klientów. Określany jest jako okres, który mija pomiędzy złożeniem zamówienia, aż do momentu jego dostarczenia.[[38]](#footnote-38)

Jako, że firma posiada własnych kierowców dzięki czemu nie musi korzystać zewnętrznych firm przewozowych co przekłada się na kontrolowany czas dostaw. Stosunek kierowców do osób zamawiających wynosi siedem do trzech co sprawia, że każdy kierowca wypełniający swoje zadania musi skontaktować się z osobą zamawiająca. Przedsiębiorstwo nie posiada własnego magazynu co sprawia, że codziennie towar jest zamawiany z hurtowni zamiast magazynowany na miejscu. Na artykuły sprzedawane w przedsiębiorstwie składają się artykuły biurowe takie jak długopisy, papier ksero czy segregatory i wiele innych, artykuły spożywcze jak mleko czy woda, artykuły chemiczne jak papier toaletowy czy środki chemiczne do czyszczenia oraz elektronika jak nawet monitory, myszki czy kable. Taka różnorodność artykułów powoduje potrzebę zamawiania z różnorodnych hurtowni, czy to ze względu na markę. Hurtownie z których korzysta przedsiębiorstwo to PBS, Allmix, Helkra, Selgros, Merida, Neopak, Debiut plus, Polcan, Rafcom, DAP fruits. Tylko trzy pierwsze hurtownie oferują dowóz raz dziennie w godzinach porannych co powoduje, że  kierowcy muszą rozdzielić między sobą pozostałe w celu odebrania z nich niezbędnych artykułów. Po przywiezieniu wszystkich artykułów do firmie następuje ich segregacja i  dopakowywanie do towaru spakowanego na wieczór dnia poprzedniego z tego względu, że hurtownie Allmix, helkra oraz PBS są przywożone rano przez hurtownie oraz odbierane po dostarczeniu towaru do klienta wieczorem oraz wieczorem wstępnie pakowane. Po segregacji oraz pakowaniu rozpoczyna się dostarczanie zamówień do firm, które zamówiły w dzień poprzedzający.

Kluczowym elementem działania każdej firmy jest dobrze zorganizowane biuro, które wymaga odpowiedniego wyposażenia. Zaopatrzenie biura w niezbędne artykuły to  kluczowy czynnik wpływający na efektywność pracy. W dzisiejszych czasach przez artykuły biurowe rozumie się nie tylko długopisy, zeszyty czy ołówki, ale też większe technologie jak gilotyny, monitory czy tusze do drukarek. Branża artykułów biurowych często doświadcza trudności związanych z kryzysami ekonomicznymi, ponieważ firmy ograniczają swoje wydatki, zwłaszcza na administrację. Jednakże rynek ten może być stabilny lub nawet rosnący z uwagi na wzrost zapotrzebowania na papier i tekturę, co wynika z danych Konfederacji Europejskiego Przemysłu Papierniczego (CEPI). Mimo rosnącej cyfryzacji, rynek artykułów biurowych nadal ma potencjał na rozwój. Firmy powstające codziennie w Polsce stają się potencjalnymi klientami dla przedsiębiorstw zajmujących miejsce w branży biurowej. Producenci artykułów biurowych non stop wprowadzają innowacje, tworząc coraz bardziej nowoczesne produkty, by przyciągać klientów. Istnieją różne targi papiernicze na świecie. Jednym z nich jest Paperworld we Frankfurcie. Odwiedzenie go jest skutecznym sposobem na pozyskanie informacji o produktach, konkurencji oraz trendach rynkowych. W Polsce funkcjonuje ponad 200 firm zajmujących się dystrybucją artykułów biurowych, co świadczy o dużej konkurencji w tej branży. Podsumowując, przedsiębiorstwa zajmujące się handlem biurowo-papierniczym muszą codziennie rozwiązywać wiele problemów logistycznych i dostosować się do zmieniającego się otoczenia konkurencyjnego, gdyż rynek ten nadal ma potencjał na wzrost.[[39]](#footnote-39)

Codziennie kierowcy borykają się z problemami logistycznymi, czy to objawiające się przez przestoje w pracy ze względu na źle podany towar w jednej z darmowych aplikacji, z których korzysta firma, czy też nieekonomiczną jazda to znaczy po parę razy w jeden punkt znów ze względu na brak komunikacji co do odbioru bądź przewozu towaru co przekłada się również na zużyte paliwo czy ogólne zużywanie się samochodów czyli głównego narzędzia pracy kierowcy. Dlatego też zarządowi powinno zależeć na wprowadzeniu odpowiednich narzędzi informatycznych, które pomogą zaoszczędzić pieniądze długoterminowo.

Przedsiębiorstwo korzysta z wyszukanych darmowych aplikacji internetowych czy  to mobilnych do wspomagania działań logistycznych. Jedną z takich aplikacji jest Listonic, który działa jak listy zakupów. O ile w życiu codziennym funkcjonowaniu ludzi do robienia zakupów, może się sprawdzać to do wspomagania logistycznego, wyznaczania zadań, szczegółowego wpisywaniu towarów brakuje niektórych funkcjonalności, które by  się przydały. Następną aplikacją wykorzystywaną przez firmę jest komunikator Messenger. O ile do komunikacji ze znajomymi spełnia swoją rolę, to w codziennym funkcjonowaniu firmy dzieje się duży chaos przez wysyłanie zdjęć w-z czy zadawanie pytań na temat zamówień. Zarząd prawdopodobnie wolałby trzymać sprawy firmowe na własnym serwerze oraz we własnej bazie danych zamiast wysyłać je do niekontrolowanych przez przedsiębiorstwo. Brak jest jednolitego systemu gdzie znalazłyby się wszystkie niezbędne informacje potrzebne na konkretny dzień.

Na rynku nie ma aplikacji spersonalizowanych pod przedsiębiorstwo zajmujące się sprzedażą artykułów biurowych. Oczywiście są firmy, które zajmują się tworzeniem takich aplikacji na zamówienie ale wiąże się to z bardzo dużymi kosztami. Istnieją jednak aplikacje, które w kontekście ogólnym mają wspomagać kierowców dokonujących dostaw. Aplikacją taką jest na przykład Circuit jest to aplikacja, która po wpisaniu punktów dostaw ma pomóc w optymalizacji trasy uwzględniając korki czy drogi zamknięte. W darmowej wersji może przybrać do dziesięciu punktów plus miejsce zaczęcia oraz zakończenia trasy. Można również zaznaczyć miejsce w samochodzie w którym stoi towar oraz sprecyzować ile będzie trwać dany postój co pozwala określać dokładniej godzinę skończenia. Kolejna aplikacją jest GSM tasks, które ma takie funkcjonalności jak: efektywne zarządzanie czasem, śledzenie dostaw na żywo, przejrzysta wizualizacja danych i intuicyjna obsługa, bez papierowa obsługa klienta poprzez podpis elektroniczny, płynne integracje. Da się również połączyć aplikację ze wewnętrznymi systemami jak: integracja gps, poczta elektroniczna oraz platforma zakupowa. Jest to jednak aplikacja, która tylko jako okres próby na  czternaście dni jest darmowa potem wiąże się z kosztami korzystania. Następną aplikacją jest Track-POD, która podobnie jak Circuit pomaga zoptymalizować trasę dla kierowcy dodatkowo pozwala to klientom śledzić dokładną lokalizację kierowcy. Jeśli kierowca potrzebuje informacji o dostawie dostanie je również w tej aplikacji jak i może skontaktować się z klientem bezpośrednio z aplikacji. Niestety korzystanie z takiej aplikacji również jest drogie bo miesięczna subskrypcja na wszystkie funkcjonalności może wynosić nawet 69 euro oraz nie jest bezpośrednio skonfigurowana pod wymagania przedsiębiorstwa.

## Analiza procesów przedsiębiorstwa



Obecny stan przedsiębiorstwa składa się z następującego procesu. Zamówienia składane przez klienta od poniedziałku do piątku do godziny szesnastej są dowożone na adres dostawy podany przez klienta na następny dzień roboczy. W czasie zaakceptowania zamówienia osoby odpowiedzialne za zamówienia wysyłają odpowiednie zamówienia do odpowiednich hurtowni. Spowodowane jest to, iż na przedsiębiorstwo korzysta z łącznie z ośmiu hurtowni w zależności od rodzaju towaru. Zamówienia na hurtowniach składane są na ten sam dzień do godziny trzynastej, a zamówienia składane po tej godzinie są przywożone przez hurtownie osobiście do przedsiębiorstwa. Zamówienia, które zdążą zostać zebrane są odbierane przez jednego z kierowców z dwóch hurtowni z artykułami biurowymi. Hurtownie spożywcze, z elektroniką, z artykułami chemicznymi czy papierowymi są odbierane przez kierowców od godziny ósmej, każdego dnia roboczego jeśli wymaga tego zamówienie. Wszystkie informacje dotyczące zamówień, które trzeba odebrać z hurtowni kierowcy dostają na aplikacji Listonic. Jest to aplikacja do tworzenia list zakupów, a więc nie jest dedykowana dla przedsiębiorstw biurowych co sprawia, że nie ma czegoś takiego jak dostosowanie jednostek lub priorytetyzacji zadań. Kierowcy i osoby zamawiające widzą na aplikacji to samo i niestety brakuje rozdzielenie widoków w zależności od roli, aby tylko osoby zamawiające miały możliwość dodania zadań. Po skompletowaniu rannego towaru i dowiezieniu przez hurtownie jest faza pakowania, które odbywa się przez papierowe dokumenty w-z i stawianiu tzw. „ptaszków” jeśli towar jest spakowany. Następnie podczas rozwozu towarów do klienta po zostawieniu towaru kierowca jest odpowiedzialny za uzyskaniu podpisu osoby odbierającej towar w celu potwierdzenia odbioru. Po trasie jeden z kierowców jest odpowiedzialny za odbiór towaru z hurtowni ze skompletowanym towarem jak napisane powyżej. Kierowca ten musi przywieźć towar do osoby, która pakuje towar na następny dzień roboczy z towaru, który został zebrany. Pakowanie wieczorne oraz odbiór wieczornych hurtowni są to obowiązki dodatkowo płatne, a więc kierowcy, którzy pracują na umowę zlecenie muszą odnotować czas pracy oraz odbyte obowiązki dodatkowe w papierowej liście obecności. Rozmowy wewnątrz firmy odbywają się przez aplikację Messenger.

Planem jak powinien wyglądać, że wszystkie obecne czynności przeprowadzane papierowo bądź przez aplikacje niededykowane mają być przeniesione do urządzeń elektronicznych. Kierowcy powinni dostawać informację o towaru do odebrania w sposób czytelny i jasny w taki sposób, że kierowca ma pojęcie jaki dokładnie towar i w jakiej ilość dla przykładu jeden karton lub jedną bindę. Usprawniłoby to proces logistyczny przedsiębiorstwa. Pakowanie towaru poprzez kartki powoduje iż firma dużo go zużywa co generuje koszty i jest to mało ekologiczne. Przeniesienie tego do aplikacji zaoszczędziłoby pieniędzy firmie. Lista obecności również jest w formie papierowej i poza czasem spędzanym na uzupełnianiu jej, wykorzystywany jest również papier, a dałoby się to przenieść do aplikacji i zautomatyzować. Warto również dodać, że chat mógłby zostać przeniesiony do własnej aplikacji co zredukuje możliwość wypływu danych.

## Analiza potrzebnych funkcjonalności



Jedną z niezbędnych funkcjonalności do dobrej komunikacji jest priorytetyzacja zadań. Dzięki specjalnemu wyróżnieniu zadań z etykietą ustawioną jako ważne kierowcą trudniej byłoby nie zauważyć i nie wykonać danego zadania. Z powodu chęci utrzymania dobrych relacji z klientami, niekiedy firma musi sprostać nietypowym wymaganiom na przykład firma ma spotkanie o godzinie dwunastej i chcieliby otrzymać niezbędne artykuły do godziny jedenastej. Wiec aplikacja mogła by wyróżnić w jakiś sposób na przykład kolorem priorytet takiej dostawy oraz możliwość zostawienia notatki. Ważnym aspektem są również zamówienia tak zwane ekologiczne, które nie musza zostać dostarczone do 24 godzin od zamówienia tylko klient zamawiający w poniedziałek może ustawić swoje zamówienie jako zamówienie ekologiczne i dać termin dostawy do na przykład maksymalnie do piątku.

Następną funkcjonalnością jest specjalne dostosowanie jednostek ze względu na  zróżnicowanie w towarze biurowym. Na przykład papier ksero liczy się w ryzach czy w  kartonach, gdzie wodę do picia można liczyć w zgrzewkach czy butelkach. Jest to jeden z problemów, który można zauważyć w przedsiębiorstwie codziennie. Specjalne dostosowanie jednostek zaoszczędzi wielu pytań i przyśpieszy wybór artykułów zamówionych przez klientów na hurtowniach szczególnie problem ten występuje na  hurtowni Selgros gdzie kupowane są głównie artkuły spożywcze czy też chemiczne w  największej ilości z wszystkich hurtowni.

Koleją potrzebną oraz brakującą funkcjonalnością jest dostosowanie widoku i  ograniczenie tylko do rzeczy niezbędnych dla danego kierowcy. Dla przykładu, kierowca A jeżdżący na hurtownie Merida niech widzi tylko rzeczy do odbioru z Meridy zamiast widzieć na przykład rzeczy z Selgrosa z który jest odpowiedzialny kierowca B. To sprawi skupienie się tylko i wyłącznie na zadaniach dla danego użytkownika zalogowanego na  swoje konto. Wyświetlane zamówienia będą tylko dla regionu dostosowanego do kierowcy, a w przypadku konta z rolą osoby zamawiającej wyświetlane będą wszystkie zamówienia. Kierowca widząc swoje zamówienia będzie mógł odhaczać artykuły spakowane bądź zostawiać artykuły, których brakuje. Dzięki temu osoba zamawiająca mająca dostęp do wszystkich zamówień będzie miała wgląd również w artykuły wymagające ponownego zamówienia czy też złożenia reklamacji.

Prywatny chat sprawdził by się lepiej z tego względu, że dostosowanie interfejsów mogłoby opierać się na wymaganiach kierowców. Wszystkie zdjęcia wiadomości będą trzymane w bazie danych przedsiębiorstwa co może skutkować zmniejszonym ryzykiem wypływu danych wrażliwych dla przedsiębiorstwa. W tej funkcjonalności można rozważyć dwa aspekty: chat prywatny (osoba do osoby) oraz chat publiczny (grupowy). Jako, że  najczęściej korzysta się z chatu publicznego, a ze sprawami prywatnymi nawiązuje się kontakt telefoniczny to aplikacja skupiłaby się właśnie wokół chatu publicznego, gdzie można będzie wysyłać zdjęcia oraz zadawać pytania do ogółu.

Messenger w przedsiębiorstwie służy również do kontroli czasu pracy jako, że  kierowcy pracują na umowę zlecenie. Można byłoby zrobić kalendarz miesięczny w  którym kierowca zaznaczałby czas pracy, numer rejestracyjny, którym jeździł oraz inne dodatkowe informacje, które są zawarte w umowie oznaczone jako dodatkowo płatne jak na przykład wieczorny odbiór towaru czy wieczorne pakowanie. Dane te również zapisywane będą w bazie danych w razie chęci skontrolowania kierowcy przez zarząd. Ograniczeniem tutaj będzie co do dnia. Kierowca będzie musiał i mógł zaznaczyć tylko ten dzień który jest aktualnie co zniweluje oszustwa bo jak wiadomo „papier wszystko przyjmie”.

W miarę rozwoju aplikacji dobrym pomysłem byłoby połączenie aplikacji z  platformą zakupową na której zamawiają klienci dzięki czemu uniknie się ręcznie przepisywania zamówień i każdy będzie miał na swoim urządzeniu mobilnym czy też komputerze aktualne zamówienia. Wypadałoby tylko dostosować format zamówień by  odpowiadał aplikacji internetowej.

Przy rozwoju aplikacji i połączeniu jej z platformą tak, żeby zamówienia pojawiały się automatycznie, będzie można przejść z papierowych w-z na zamówienia w formie elektronicznej i brać elektroniczny podpis od klienta. Mimo iż będzie potrzeba zainwestowania w służbowe telefony to długoterminowo oszczędność na papierze będzie powodować zwrot tej inwestycji. W dodatku nie będzie problemu z gubiącymi się kartkami bo wszystko będzie zapisane w bazie danych.

Listy zadań czy zakupów świetnie sprawdzą się w aplikacji ze względu na codzienne spersonalizowane zakupy przez klienta. Jako, że ilość hurtowni odwiedzanych codziennie przez kierowcę jest dosyć spora to dobrze działająca lista wraz z przyjemnym dla oka interfejsem pozwoli w przyjemny sposób pomóc w pracy kierowcy. Przedsiębiorstwo pracuje już z darmową aplikacją do wpisywania listy zakupów, lecz działa to średnio i  najlepiej jakby wszystko było połączone w jedną aplikację, a tak nie jest.

## Aspekty prawne i regulacyjne

Jako, że przy rejestracji aplikacji będzie przymus podanych danych osobowych trzeba zastosować się do ustawy RODO, która mówi o ochronie osób fizycznych, w  kontekście przetwarzania danych osobowych określonych w przepisach art. 2 i art. 3 rozporządzenia Parlamentu Europejskiego i Rady UE z dnia 27 kwietnia 2016r. dotyczącego ochrony danych osobowych oraz swobodnego przepływu takich danych, a także znoszącego dyrektywę 95/46/WE.[[40]](#footnote-40) Ponadto trzeba uwzględnić artykuł 25 który mówi o aktualnym poziomie technologicznym, kosztach implementacji oraz charakterze, zakresie, kontekście i  celach przetwarzania danych osobowych, stosowaniu odpowiednich środków technicznych oraz organizacyjnych. Obejmują one m.in. pseudonimizację danych, które są  zaprojektowane w taki sposób, aby zapewnić minimalizację danych i niezbędne zabezpieczenia. Celem jest zgodność z zasadami ochrony danych oraz ochrona osób fizycznych. Administrator powinien również stosować odpowiednie środki, aby  automatycznie przetwarzane były tylko te dane osobowe, które są niezbędne do  osiągnięcia określonego celu przetwarzania. Dotyczy to zarówno ilości zbieranych danych, zakresu przetwarzania, okresu przechowywania, jak i dostępności danych. Zapewnienie tego wymaga, aby dane osobowe nie były domyślnie udostępniane bez zgody osoby, które one dotyczą. W celu spełnienia wymogów administrator może zastosować zatwierdzony mechanizm certyfikacji.[[41]](#footnote-41)

Jako aplikacja informatyczna jest usługą świadczoną drogą elektroniczną, warto zapoznać się z tą ustawą, która mówi, że usługodawca który dostarcza usługi drogą elektroniczną, takie jak przekazywanie danych przez sieć telekomunikacyjną lub  zapewnienie dostępu do sieci telekomunikacyjnej, nie ponosi odpowiedzialności za treść tych danych, pod warunkiem że: nie jest inicjatorem przekazu danych, nie wybiera odbiorcy przekazu danych oraz nie dokonuje wyboru ani modyfikacji informacji zawartych w przekazie. Dodatkowo, wyłączenie odpowiedzialności obejmuje również tymczasowe przechowywanie danych w celu przeprowadzania transmisji o ile dane te nie są  przechowywane dłużej niż koniecznie dla zrealizowania transmisji. Podmiot nie będzie odpowiadał za przechowywane dane jeśli: Przekazując dane oraz zapewniając ich tymczasowe przechowywanie, nie dokonuje modyfikacji danych. Stosuje uznaną i  powszechnie stosowaną w tej dziedzinie technikę informatyczną ustalającą parametry techniczne dostępu do danych. Nie zakłóca normalnego funkcjonowania technik informatycznych stosowanych do zbierania informacji o korzystaniu z tych danych. Ponadto, jeśli podmiot niezwłocznie usunie dane lub niemożliwi dostęp do nich po  otrzymaniu informacji o ich usunięciu z pierwotnego źródła transmisji lub na żądanie są lub innego organu, nie będzie ponosił odpowiedzialności za przechowywane dane. W  przypadku udostępniania zasobów systemu teleinformatycznego w celu przechowywania danych przez usługobiorcę, podmiot nie będzie ponosił odpowiedzialności, jeśli nie ma wiedzy o bezprawnym charakterze danych lub działalności z nimi związanej. Jeśli jednak otrzyma oficjalne zawiadomienie lub wiarygodną wiadomość o bezprawnym charakterze danych, niezwłocznie uniemożliwi dostęp do nich. Ponadto usługodawca może przetwarzać pewne informacje osobowe usługobiorcy, które są  niezbędne do ustanowienia, dostosowania, zmiany lub rozwiązania umowy między nimi. Te informacje obejmują nazwisko, imiona, numer PESEL lub inny dokument tożsamości, adres do korespondencji oraz adres zamieszkania stałego, dane do zweryfikowania elektronicznego podpisu oraz adresy e-mail usługobiorcy. Dodatkowo, w celu realizacji umowy lub innych czynności prawnych, usługodawca może przetwarzać inne informacje niezbędne ze względu na rodzaj świadczonej usługi lub sposobu jej rozliczenia. Dane te, których podanie jest niezbędne do świadczenia usług drogą elektroniczną, są wyróżniane i  oznaczane przez usługodawcę. Usługodawca może również przetwarzać inne dane, takie jak dane reklamowe, badania rynku oraz preferencje usługobiorców, za zgodą usługobiorcy. Ponadto, dane dotyczące sposobu korzystania przez usługobiorcę z usługi drogą elektroniczną, takie jak identyfikatory użytkownika, informacje o sieci telekomunikacyjnej oraz historii korzystania z usługi, mogą być przetwarzane przez usługodawcę. Usługodawca jest również zobowiązany do udostępnienia tych danych organom państwa na ich żądanie zgodnie z obowiązującymi przepisami prawa.[[42]](#footnote-42)

Ze względu na przekazywanie zdjęć faktur i tym podobnych rzeczy należących do  przedsiębiorstwa oraz zawierających poufne informacje, które są własnością firmy należy rozważyć ustawę o prawie autorskim i prawach pokrewnych, które mówi o tym, że prawo autorskie obejmuje każdą indywidualnie zmaterializowanej działalności twórczej, niezależnie od jej wartości, przeznaczenia czy sposobu wyrażania, co jest określane jako utwór. W zakresie prawa autorskiego znajdują się między innymi utwory wyrażone w formie słowa pisanego, symboli matematycznych, grafik, dzieła plastyczne, fotografie, wzory przemysłowe, projekty architektoniczne i urbanistyczne, utwory muzyczne, teatralne, choreograficzne, pantomimiczne oraz działa audiowizualne, w tym filmy. Prawo autorskie ochrania wyłącznie sposób, w jaki dany utwór jest wyrażony, nie obejmuje ochroną samych pomysłów, koncepcji, procedur czy idei, a także odkryć czy zasad działania. Twórcy przysługuje ochrona autorska od momentu stworzenia utworu, nawet jeśli jest on niekompletny, oraz nie wymaga ona spełnienia żadnych formalności. Ponadto jeśli w  umowie nie ma odmiennych postanowień, kiedy pracownik tworzy utwór w ramach wykonywanych obowiązków związanych ze stosunkiem pracy, pracodawca zyskuje prawa autorskie majątkowe w zakresie określonym prze cel umowy o pracę i zgodny z intencją stron po przyjęciu tego utworu. Jeśli jednak w ciągu dwóch lat od daty przyjęcia utworu pracodawca nie podejmie działań mających na celu jego rozpowszechnienie, autor ma prawo ustalić na piśmie odpowiedni termin dla pracodawcy. Jeśli termin ten upłynie bezskutecznie, prawa przekazane wcześniej pracodawcy, razem z własnością nośnika utworu, wracają do  autora, chyba że umowa stanowi inaczej. Możliwe jest również ustalenie innego terminu na rozpowszechnienie utworu w umowie o pracę. Jeśli umowa o pracę nie określa inaczej, po przyjęciu utworu pracodawca uzyskuje również własność nośnika, na którym utwór został utrwalony.[[43]](#footnote-43)

Jako, że aplikacja musi korzystać z bazy danych, ważne jest zapoznanie się z ustawą o ochronie bazy danych, która mówi, że prawo zapewnia ochronę bazom danych, niezależnie od ochrony przewidzianej dla utworów autorskich. Baza danych definiowana jest jako zbiór danych lub materiałów zorganizowanych według określonej struktury, wymagający istotnego nakładu inwestycyjnego. Pobieranie danych to przeniesienie ich na inny nośnik, a  wtórne wykorzystanie to ich publiczne udostępnienie. Wypożyczenie bazy danych nie jest traktowane jako pobieranie danych ani wtórne ich wykorzystanie. Ochrona nie obejmuje programów komputerowych użytych do tworzenia bazy danych. Bazy danych są chronione, jeśli producent jest obywatelem państwa UE lub EFTA, osobą prawną z siedzibą na ich  terytorium, lub jeśli ochrona wynika z umów międzynarodowych. Producentowi bazy danych przysługuje wyłączne prawo pobierania danych i ich wtórnego wykorzystania. Uważa się, że producentem jest osoba, której nazwisko lub firma jest powiązana z bazą danych w sposób widoczny dla publiczności. Ponadto jeśli rozważyć czas trwania to ochrona bazy danych trwa przez piętnaście lat od momentu jej stworzenia. Jeśli baza danych zostanie publicznie udostępniona w tym czasie, ochrona wygasa po kolejnych piętnastu latach od  pierwszego udostępnienia. W przypadku istotnej zmiany treści bazy danych, jak dodanie, zmiana lub usunięcie części, wymagająca nowego istotnego nakładu pracy, okres ochrony jest ponownie liczony.[[44]](#footnote-44)

## Sformułowanie problemu

Problemem rozdziału było sprawdzenie czy istnieją na rynku już takie aplikacje, które mogą wspomóc kierowców w przedsiębiorstwie zajmującym się sprzedażą artykułami biurowymi. Niestety na rynku są aplikacje dla kurierów, które pracują dla firm przewozowych, których towar w paczce nie robi różnicy. Dla przedsiębiorstwa, które same musi skompletować towar, jest ważne co jest wyświetlane w aplikacji, a w takich aplikacjach nie ma tak sprecyzowanych informacji na temat towaru. A więc także hipotezę można obalić ze względu na mało sprecyzowane funkcjonalności, których wymaga przedsiębiorstwo.

# ROZDZIAŁ 3 OPIS SYSTEMU ORAZ BADANIA



## Start projektu

Cała aplikacja postawiona będzie na szkielecie next.js. Do utworzenia projektu wykorzystany zostanie npm (node package menager). Jest to największe repozytorium oprogramowania, które umożliwia programistom open source z każdego miejsca na ziemi na dzielenie się oraz korzystanie z pakietów, a również organizacjom na zarządzanie prywatnymi projektami. Jej głównymi elementami są strony internetowe, interfejs wiersza poleceń (CLI) oraz rejestru.[[45]](#footnote-45) Zacząć należy od instalacji node, w tym celu należy wejść na  stronę nodejs.org w celu pobrania pliku instalacyjnego. Node jest to środowisko, które uruchamia JavaScript zbudowane na silniku JavaScript V8[[46]](#footnote-46). Node package manager powinien się zainstalować wraz z zainstalowaniem się Node. W celu sprawdzenia poprawności zainstalowania się Node oraz NPM można wpisać następujące komendy wiersza poleceń „npm -v” oraz „node -v”. Literka „-v” jest to flaga, która oznacza version czyli wersję na początku npm potem node.

rys. 1 wersja node

Obraz zawierający tekst, Czcionka, zrzut ekranu, czarne

Opis wygenerowany automatycznie

rys. 2 Wersja npm

Obraz zawierający tekst, Czcionka, zrzut ekranu, czarne

Opis wygenerowany automatycznie

Tak samo jak na rysunku 1 jak i 2 widać, że ukazuje się wersja co znaczy, że node jak i npm zostały poprawnie zainstalowane. Następnie, można przystąpić do utworzenia projektu next.js. W tym celu należy utworzyć folder w prawidłowej ścieżce w której chcemy trzymać pliki. Można zrobić to na dwa sposoby przez wiersz poleceń i wpisanie komendy „mkdir [nazwa]”, lub przez prawy przycisk myszy następnie nowy i folder. Warto zaznaczyć, że wszystkie wymienione przykłady są prezentowane na systemie operacyjnym windows 11. Na Linuxie czy też MacOS, może się to nieco różnić. Jeśli folder już istnieje należy do niego wejść „cd [nazwa folderu]” i wpisać komendę, którą podaje oficjalna strona next.js „npx create-next-app@latest .”. Kropka na końcu oznacza, że projekt zostanie utworzony w aktualnej ścieżce w której tworzący projekt się znajduje. Następnie wyskoczy pytanie o paczkę należy wpisać „y” od yes czyli tak.

rys. 3 Tworzenie aplikacji w next.js

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

Następnie wyskoczą inne pytania konfiguracyjne jak preferowany język JavaScript/TypeScript czy sposób pisania stylów co pokazuje rysunek numer 3. Po poprawnie zainstalowanym projekcie na samym końcu wyskakuje zielony napis „Success!” co oznacza sukces. Do wystartowania projektu w środowisku lokalnym wystarczy użyć komendy w folderze w którym został zainstalowany projekt „npm run dev”.

rys. 4 Uruchomienie aplikacji

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Widać, że projekt uruchomił się na porcie 3000 po wejściu na stronę zostały wykonane pierwsze zapytania typu GET.

Projekt przyjmie podstawową architekturę plików, którą zaleca producent.

rys. 5 Architektura plików

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

Foldery takie jak .idea, .next czy node\_modules są to foldery konfiguracyjne szkieletu wiec one nie muszą zostać omówione, ponieważ w każdym projekcie są takie same w zależności oczywiście od wersji. Następnie folder app jest to routing aplikacji w którym zostaną utworzone foldery z nazwami podstron jak na przykład „about” w którym umieścić należy plik page.tsx z odpowiednim kodem TypeScriptowym. W takim przypadku endpoint „/about” (na przykład <http://localhost:3000/about>) powinien przenieść na odpowiednia podstronę. Folder „Components” zawiera wszystkie reużywalne komponenty które można następnie używać w całych featurach. Wiec odpowiednio folder features składa się na pliki zawierające połączenie wszystkich niezbędnych komponentów do utworzenia widoku. Folder Context zawiera wszystkie contexty czyli takie mechanizmy w react, które umożliwiają przekazywanie danych przez drzewo komponentów, bez konieczności ręcznego przekazywania propsów na każdym poziomie. Gdyby nie kontekst przekazywanie danych wyglądałoby w następujący sposób od rodzica do dziecka czyli w patrząc na drzewo komponentów od góry do dołu. Jednak by ułatwić przekaz niewygodnych danych jak motyw interfejsu, które są wymagane przez wiele komponentów w aplikacji stworzono kontekst. Zapewnia on również mechanizm dzielenia się wartościami między komponentami bez konieczności jawnego przekazywania danych przez każdy poziom drzewa.[[47]](#footnote-47) Hooks jest to  folder zawierający wszystkie hooki, które służą do przetrzymywania stanu aplikacji bez konieczności pisania klas.[[48]](#footnote-48) Folder lib zawiera wszystkie konfiguracje bibliotek jak na przykład utworzenie klienta prisma czy opcje autoryzacji do biblioteki next auth. Folder messages zawiera jedynie pliki typu json, które zawierają tłumaczenia stron jeśli takie by wystąpiły. Prisma zawiera wszystkie pliki, użyte do obsługi bazy danych. Public zwiera zdjęcia ikony i tym podobne. Na końcu utils zawiera funkcję, które mogą być użyte w wielu komponentach jak na przykład przeliczenie jednostek. Z dalszych plików powinien interesować tylko plik tailwind.config.ts w którym zawarte są wartości styli nadawanych później w projekcie, .env w który zawarte są zawarte zmienne środowiskowe jak np. adres bazy danych czy domeny aplikacji oraz .gitignore w którym zapisuje się czego nie wysyłać do repozytorium z powodu za dużej ilości plików takich jak pliki konfiguracyjne.

Tworząc projekt warto skorzystać z darmowych bibliotek ułatwiających i  przyśpieszających pracę. Jedną z takich biblioteka jest biblioteka dzięki której możemy korzystać z gotowych komponentów i przerabiać je pod daną aplikację i dany wygląd. Biblioteka ta nazywa się NextUI. Do instalacji tej biblioteki jak i każdej zawartej w tym projekcie należy użyć node package managera wpisując komendę „npm install @nextui-org/react framer-motion”. Nastepnie wystarczy importować dany komponent „import {Button, ButtonGroup} from "@nextui-org/react";” i można go użyć w następujący sposób na przykładzie guzika co pokazane na rysunku 6.

rys. 6 Przykład użycia Buttona z NextUI

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Dodatkowo można dodać jako props (np. color=”primary”), takie własności jak rozmiar, wariant promień zakrzywienia krawędzi, wstawienie ikony i wiele innych o których mówi dokumentacja. Ponadto możemy zmieniać wartości nadane już przez NextUI jak na  przykład kolor primary może być kolorem który nadpiszemy w pliku tailwind.config.ts. Komponent ten funkcjonuje jak normalny tag HTML <buton/> czyli możemy przekazywać funkcję onClick, onChange itd. Biblioteką użytą do obsługi wysyłania na serwer zapytań klienta będzie axios. Jest to klient HTTP opartym na obietnicach (z ang. Promise-based) przeznaczonym do pracy zarówno w node.js, jak i w przeglądarce. Jest narzędziem izomorficznym, co znaczy, że może być używane w obu środowiskach za pomocą tego samego kodu. Na serwerze wykorzystuje natywny moduł http w node.js, podczas gdy w  przeglądarce korzysta z żądań XMLHttpRequest.[[49]](#footnote-49) Do zainstalowania wystarczy użyć komendy „npm install axios”. Nastepnie można przejść już do użycia axiosa. Na przykładzie zapytania typu POST

rys. 7 Przykład użycia axios

Obraz zawierający tekst, zrzut ekranu, Czcionka, design

Opis wygenerowany automatycznie

W tym przykładzie z rysunku została użyta metoda POST do wysłania danych określonych w  data na backend znajdujący się pod danym endpointem. W tym przypadku musiała zostać utworzona instancja axiosa w której można określić stały adres i wpisywać tylko endpointy. W takim podstawowym użyciem można dodać jeszcze takie parametry jak headers czy parametry. Biblioteką użytą do warunkowego stylowania w pracy, został wykorzystana bilbioteka clsx. Do instalacji wystarczy użyć komendy „npm install clsx”. Stylowanie warunkowe jest bardzo przydatne do na przykład nadawania kolorów komponentów, które są odpowiedzialne za zadania, jak nadanie priorytetu zadaniu, gdzie zadanie z mniejszym priorytetem ma kolor zielony, a z większym priorytetem kolor czerwony. Biblioteka bcrypt została użyta do zaszyfrowania haseł trzymanych w bazie danych. Do obsługi i walidacji formularzy zostały użyte biblioteki zod oraz react hook form. Pozwalają one na pobranie odpowiednich typów danych z formularzy deklarując je szybciej. Po walidacji zwracane, jeśli występują błędy na przykład zły format emailu czy niespełnienie warunków hasła typu wielka litera, znak specjalny itd. Do zarządzania stanem całej aplikacji czyli pisaniem odpowiednich hooków, które pomagają pobierać dane i przechowywać użyto biblioteki react query. Później we wszystkich komponentach wystarczy użyć hooka, który dostarcza biblioteka, do użycia danych. W projekcie zastosowane jest również tłumaczenie, a do tego została użyta biblioteka next-intl. Tutaj należy ustawić w odpowiedni sposób plik layout, który definiuje RootLayout czyli główny układ strony co znaczy, że należy nadać odpowiedni kontekst stronie zależnie jaki stan języka strony jest obecnie aktywny. Opisany wyżej katalog messages uczestniczy w tworzeniu tłumaczenia gdyż w nim zawarte są pliki JSON z tłumaczeniami na wybrane języki, których potrzebuje aplikacja. W tym przypadku jest to angielski oraz polski czyli pliki kolejno nazywać się będą en.json oraz pl.json. Należy również ustawić middleware, który odpowiedzialny jest za obsługę tłumaczenia w tym przypadku jest połączony z autoryzacją wiec więcej o nim w kolejnym podrozdziale.

Autoryzacja jest bardzo ważną częścią projektu. Już przy rejestracji użytkownik musi wpisać odpowiednie dane jak: imię, nazwisko, email, hasło, zaakceptować warunki korzystania oraz wybrać jedną z dwóch opcji, czy jest kierowcą czy osobą zamawiającą. Jest to o tyle ważne, że widok i pokazywane dane będą pokazywane w odpowiedniej ilość i  sposób dla danej kategorii. To autoryzacji została użyta odpowiednia biblioteka, która pozwala zrobić autoryzacje we szkielecie next.js. Pozwala na obsługę dowolnej usługi OAuth oraz obsługuje OAuth 1.0, 1.0A, 2.0 oraz OpenID Connect. Next auth posiada na  dodanie odpowiednich providerów logowania takich jak github czy google. Wystarczy na podstawie githuba zrobić OAuth aplikacje w ustawieniach dla deweloperów, podać odpowiedni adres URL i można cieszyć się logowaniem przy użyciu github, oczywiście przez dodanie odpowiedniego providera w kodzie co pokazuje rysunek niżej.

rys. 8 GithubProvider

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

NextAuth obsługuje również uwierzytelnianie za pomocą adresu e-mail bez hasła. Biblioteka wspiera uwierzytelnianie bezstanowe z dowolnym backendem jak Active Directory czy LDAP oraz innych. Obsługuje tokeny JWT oraz sesje w bazie danych. Stworzony pod kątem Serverless, lecz działa wszędzie (AWS Lambda, Docker, itd.). Autoryzacja przy użyciu nextAuth, może być używana z użyciem bazy danych lub bez jej użycia. Stosuje koncept BYOD (z ang. Bring Your Own Database) co znaczy, że może zostać użyty z dowolną bazą danych. Posiada wbudowaną obsługę MySQL, Postgres, MariaDB, SQL Server, SQLite oraz MongoDB. Wykorzystuje tokeny Cross-Site Request Forgery (CSRF) do zapytań typu POST. Polityka plików cookie domyślnie dązy do  stosowania polityki, która jest najbardziej restrykcyjna odpowiedniej dla każdego pliku cookie. Podczas gdy JSON Web Tokens są włączone, domyślnie zostają szyfrowane (JWE) przy pomocy A256GCM. Dla ułatwienia deweloperom automatycznie generuje symetryczne klucze do podpisywania i szyfrowania. Platforma oferuje synchronizację kart oraz wiadomości keepalive, aby wspierać krótkotrwałe sesje. Stosowane są najnowsze wytyczne publikowane przez Open Web Application Security Project. Zaawansowane opcje pozwalają na definiowanie własnych procedur obsługi kontroli, które konta mogą się logować, kodować i dekodować tokeny JSON Web Tokens oraz ustawić niestandardowe zasady bezpieczeństwa plików cookie i właściwości sesji, co pozwala kontrolować dostęp i  częstotliwość walidacji sesji.[[50]](#footnote-50) By wprowadzić autoryzację NextAuth do aplikacji należy w pierwszym kroku po zainstalowaniu biblioteki, dodać API route pod ścieżką „/app/api/auth/[…nextauth]”, plik route.ts w którym zawarta będzie inicjalizacja NextAuth.

rys. 9 Inizacjilacja Next auth

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

Jak widać na rysunku jest definiowana funkcja o nazwie „handler”, która jest wynikiem wywołania fukcji NextAuth z argumentem authOptions, który został stworzony w innym katalogu dla utrzymania czystości kodu. Następnie zostaje wyeksportowane jako GET oraz POST co znaczy, że funkcja obsługuje zarówno żądania typu GET jak i również POST. Do  prawidłowego skonfigurowania opcji autoryzacji należy podać strategię sesji w tym przypadku będzie to JWT. Następnie będzie to podanie odpowiedniego klucza, który pozwoli aplikacji rozpoznać zautoryzowaną sesję. Dobry klucz można wygenerować przy użyciu wiersza poleceń, przy pomocy komendy „openssl rand -base64 32”. Później w kolei jest zdefiniowanie callbacków. Callback session, który jest wywoływany poczas tworzenia sesji i w przypadku ten aplikacji będzie zwracał całą sesję rozszerzoną o dane usera oraz callback jwt, który jest wywoływany podczas generowania JWT i zwraca całe dane o tokenie rozszerzone i id usera, jego adres e-mail oraz imię. Następnie na samym końcu należy sprecyzować providery jak wspomniane szybciej można użyć githuba lub googla lecz w tej aplikacji użyty będzie tylko ten który będzie stworzony w formularzu czyli logowanie przez podanie adresu email oraz hasła. Należy również stworzyć asynchroniczną funkcję do  autoryzacji która sprawdzi czy hasło i email zostały podane, następnie jeśli warunek się  spełni, sprawdzi czy użytkownik o podanym adresie e-mail istnieje w bazie, kolejnym krokiem jest sprawdzenie czy hasło użytkownika się zgadza jeśli tak sesja zostanie zwrócona, a jeśli nie można zwrócić błąd, który następnie zostanie wyświetlony użytkownikowi. Gdy wszystkie te rzeczy zostały wykonane, można teraz rozszerzyć architekturę plików o foldery private, gdzie zostaną utrzymywane podstrony, które dostępne będą tylko po autoryzacji oraz public, gdzie zawarte będą podstrony do których niewymagana jest autoryzacja w tym przypadku strona logowania oraz rejestracji. Na rysunku przedstawiony jest widok z visual studio code.

rys. 10 Rozszerzony folder app

Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie

Gdy wszystkie kroki zostaną wykonane, można w prawidłowy sposób użyć funkcji signIn, która dostarcza producent NextAuth, która odwoła się do funkcji autoryzującej i  podczas gdy wszystko będzie się zgadzać przeniesie do odpowiedniej postronny z katalogu private. Na końcu zostaje skonfigurować middleware.

rys. 11 MIddleware

Obraz zawierający tekst, zrzut ekranu, oprogramowanie

Opis wygenerowany automatycznie

W tym przypadku jak wspomniane było w podrozdziale wyżej odpowiedziany on jest za obsługę tłumaczenia oraz przekierowywanie ze strony prywatnych gdy nie uzyskana jest autoryzacja. W przypadku tej aplikacji jeśli w url bez autoryzacji, ktoś będzie chciał przenieść się na endpoint na przykład „/orders”, gdzie ta podstrona jest prywatna, zostanie przekierowany na podstronę logowania. Rysunek przedstawia całość kodu, który potrzebny jest do wykonania tego zabezpieczenia.

Ważnymi funkcjonalnościami są te co za pewne pojawią się w aplikacji jako pierwsze. Jest to związane ze słabymi rozwiązaniami obecnie panującymi na rynku. Ale odpowiedzieć należy na pytanie „Która z funkcjonalności jest najważniejsza dla aplikacji oraz przedsiębiorstwa?”. A więc biorą pod uwagę braki w płynności oraz ogólne słabe działanie aplikacji z których przedsiębiorstwo korzystało dotychczas oraz charakter w jaki przedsiębiorstwo funkcjonuje, można stwierdzić, że jedną z najważniejszych funkcjonalności będzie ta w której dodawane są towary do odebrana w dany dzień. Kierowcy oraz osoby zamawiające będą miały płynie działające narzędzie do wymiany informacji co  przyśpieszy pracę, zwiększy efektywność oraz ekonomiczność jazdy kierowców, a to  wszystko związane jest z zmniejszeniem prawdopodobieństwa niezadziałania aplikacji i  spowodowania, że kierowca będzie musiał się wracać na daną hurtownie z której towar został już odebrany.

## System na diagramach

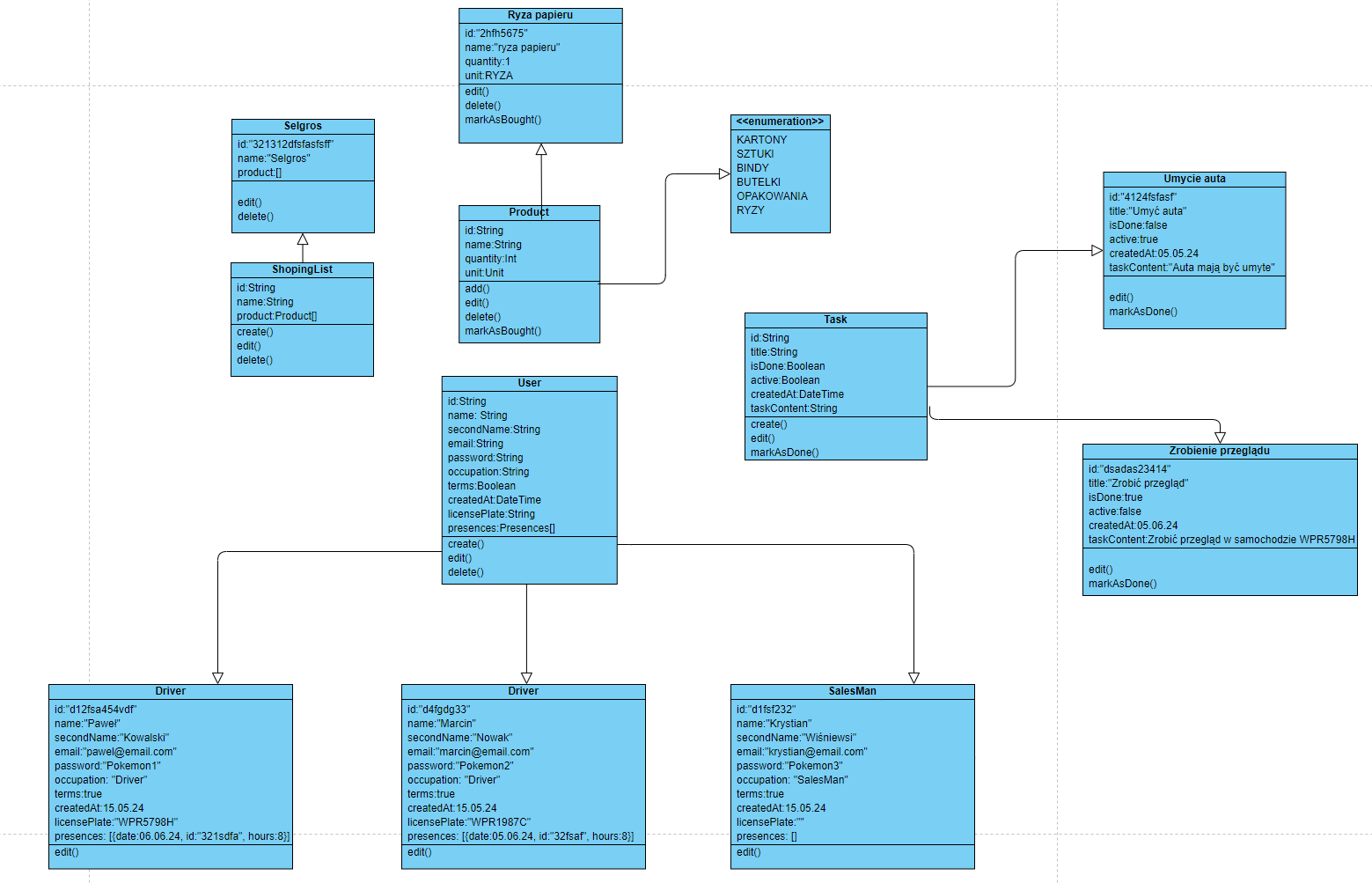
rys. 12 Diagram przypadków użycia

Obraz zawierający tekst, diagram, linia, zrzut ekranu

Opis wygenerowany automatycznie

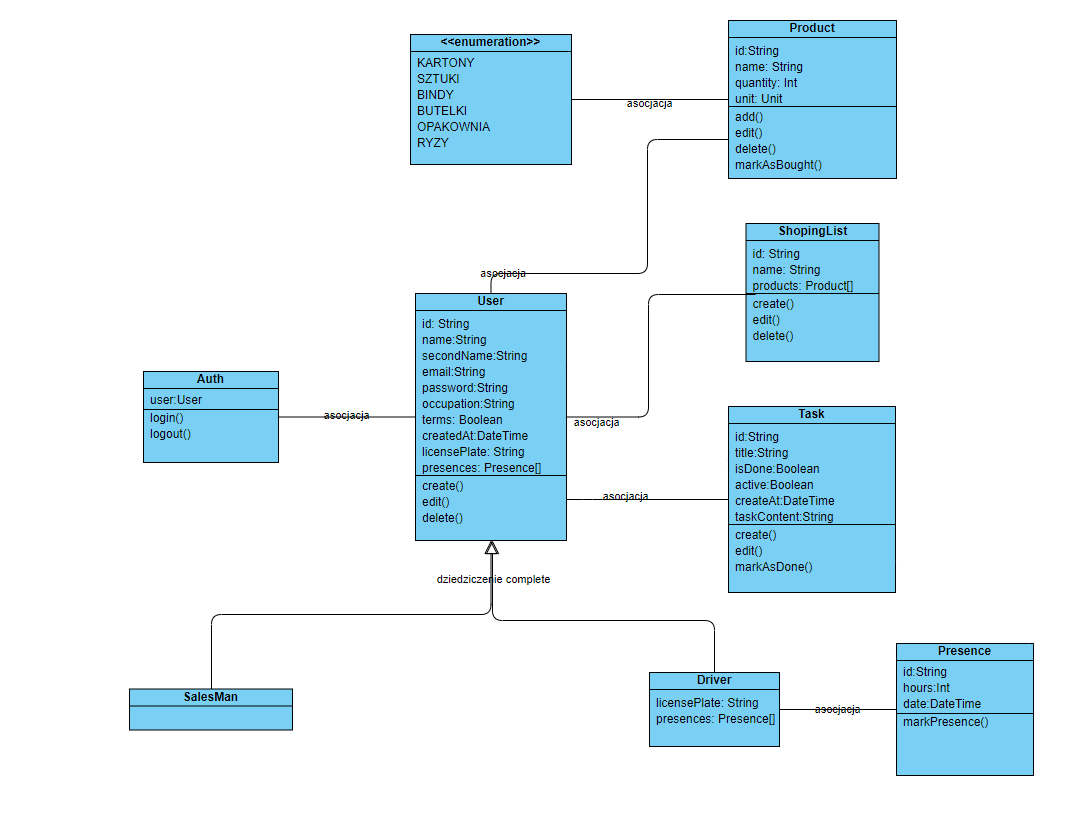
W tym systemie bierze udział trzech aktorów, dwóch interesariuszy: kierowcy oraz osoby zamawiające, jak i platforma zakupowa przedsiębiorstwa, na której osoby bądź firmy zamawiający towar składają swoje zamówienia. Połączenie z platforma w dalszej możliwej do osiągnięcia części rozwoju systemu będzie miało znaczący wpływ na dodawanie zamówień na konkretny dzień. Osoby zamawiające będą miały wgląd na całość zamówień, gdyż będą mogły dzięki temu w łatwy sposób zamawiać towar z dalszych hurtowni, a  kierowcy potrzebują tylko widoku swoich zamówień, które będą posegregowane w taki sposób, że dostosują się do wybranego przez kierowcę rejonu w którym jeździ w dany dzień. Zazwyczaj jest to obszar na stałe wiec nie w ustawieniach będzie jedynie możliwość zmiany jeśli przyjdzie taka konieczność, żeby system nie musiał pytać przy każdym logowaniu. Kolejnym przypadkiem użycia jest lista zakupów, do której będą mieli dostęp interesariusze, w taki sposób, że kierowcy będą mogli przeglądać i zaznaczyć jako kupione, a osoby zamawiające dodawać oraz edytować listy zakupowe. Kalendarz czyli na diagramie przypadek użycia dodaj obecność, będzie tylko dostępny dla kierowców ze względu na ich umowy i to, że najzwyczajniej osoby zamawiające tej funkcjonalności nie potrzebują do  swojej pracy. Zostają jeszcze listy zadań do wykonania i tutaj podobnie jak listy zakupowe, osoby odpowiedzialne za zamawianie będą miały dostęp do edycji, dodawania czy napisania notatki do zadań oraz nadawania priorytetu, a kierowcą potrzebny jest jedynie dostęp do  przeglądania zadań oraz oznaczenia wykonanych jako zrobione.

rys. 13 Diagram związków encji



Jak widać na diagramie stworzone są przykładowe instancje obiektów z poszczególnych klas. Tak więc przykładowo z klasy USER zostało stworzonych dwóch kierowców oraz jeden salesMan. Można zauważyć, że SalesMan jako iż nie jeździ samochodem na pozycji tablicy rejestracyjnej ma pusty string. Można by również dać typ jako string lub null w takim przypadku dobrym wyborem było by dać wartość null dla osoby która jest osobą zamawiającą. Nie posiada on również obecności ze względu na typ umowy, a wiec i w tym przypadku można by było zastosować wartość null. Z klasy TASK zostały utworzone dwie instancje odpowiadający tytułowi zadań. Od klasy PRODUCT została utworzona jedna instancja tak samo jak zostało to zrobione w SHOPINGLIST. Jest to tylko przykład jak mogłyby wyglądać instancje od klas. W użyciu użytkowników, zadań, produktów oraz list zakupowych będzie o wiele więcej lecz działanie będzie takie same.

rys. 14 Diagram klas



Jak widać klasy są bardzo podobne do modeli schematu prisma. Kluczowe klasy obejmują: Auth, User, Task, ShoppingList, Product, Presence, SalesMan oraz Driver, które pełnią określoną funkcje. Moduł autoryzacji (Auth) umożliwia uwierzytelnianie użytkowników i przechowuje zalogowanego użytkownika w sesji. Klasa User reprezentuje użytkowników systemu, z którymi związane są zadania, listy zakupów oraz obecności. Zadania są opisane przez klasę Task, a listy zakupów przez ShoppingList, z możliwością dodawania, edycji i usuwania zarówno zadań, jak i produktów. Klasa Product reprezentuje produkty w listach zakupów, z możliwością dodawania, edycji, usuwania oraz oznaczania jako kupione. Obecności pracowników są rejestrowane przez klasę Presence, a kierowcy są  reprezentowani przez klasę Driver, związaną z rekordami obecności.

## Badania do aplikacji

W aplikacji internetowej wspomagającej pracę kierowców w przedsiębiorstwie zajmującym się sprzedażą artykułów biurowych, głównymi podmiotami są osoby korzystające z aplikacji czyli kierowcy oraz osoby zamawiające. Należy spojrzeć na rynek takich aplikacji oraz na główne wymagania interesariuszy i porównać je ze sobą. Następnym podmiotem jest sam proces logistyczny, który polega na odbiorze towaru, kupnie, segregacji oraz dowozu do klienta. Przedsiębiorstwo posiada swoją platformę zakupową, którą można połączyć z aplikacją. Kierowcy zaczynają pracę koło godziny ósmej. Pierwszym zadaniem jest odbiór towaru z hurtowni, następnie wszyscy kierowcy spotykają się w firmie, gdzie wymieniają się towarami, dopakowują zamówienia z dnia poprzedniego oraz pakują swoje samochody. Koło godziny jedenastej kierowcy wyjeżdżają już na swoją trasę. Natomiast osoby odpowiedzialne za zamawianie od godziny ósmej zaczynają zamawiać zamówienia, gdzie od godziny ósmej do dziesiątej, odpowiedzialni są za sprawdzenie towaru oraz dopisanie odpowiednich pozycji do kupienia. Aktualna efektywność procesów logistycznych zależna jest od ilości dziennych zamówień. Firma dowozi na następny dzień zamówienia złożone do godziny szesnastej. Jeśli dużo zamówień się uzbiera pojawia się chaos towarowy oraz efektywność jazdy kierowców jak i ekonomiczność jest na niskim poziomie. Zauważyć trzeba, że występuje różnica urządzeń na których pracują poszczególne osoby. Kierowcy głównie czerpią informacje o towarze z telefonów, a osoby zamawiające z  komputerów.

Celami obserwacji jest zebranie opinii o korzystaniu z aplikacji z których korzysta aktualnie przedsiębiorstwo oraz czego oczekiwali by interesariusze od nowo powstałej aplikacji dostosowanej do charakteru przedsiębiorstwa. Należy również zaobserwować w  jaki sposób użytkownicy korzystają z aplikacji dotychczasowych, żeby dostosować aplikację w taki sposób, żeby przejście i dostosowanie się do nowej aplikacji było jak najłatwiejsze oraz trwało jak najkrótszy okres czasu. Obserwację również powinny przynieść dokładny wgląd w proces logistyczny co pozwoli na początkowej fazie tworzenia aplikacji oraz podczas jej rozwijania na dostosowywanie funkcjonalności w taki sposób by  były skierowane do przedsiębiorstwa w którym aplikacja będzie wspomagać.

Obserwacje można dokonywać w każdy dzień w którym przedsiębiorstwo pracuje tzn. od poniedziałku do piątku w godzina ósma szesnasta. Należy wziąć również pod uwagę dni w których jest mniej zamówień jak i takie w którym zamówień jest sporo, żeby zobaczyć czy do przedsiębiorstwa wkrada się chaos. Obserwacji dokonywać również należy w  poszczególne okresy czasu dla przykładu od ósmej do dziesiątej gdy towary z zamówień są dopiero odbierane z hurtowni bądź kupowane, żeby sprawdzić czy proces logistyczny jest efektywny.

Obserwacje trzeba gromadzić w postaci tabeli w której należy wziąć pod uwagę każdego interesariusza aplikacji, zdając odpowiednie pytania jak na przykład „Czy jesteś zadowolony z rozwiązań obecnie na rynku? Oceń od 1 do 5 gdzie 1 to niezadowolenie, a 5 to duże zadowolenie”. Przykładowa tabela może wyglądać następująco.

Tabela 4 Przykładowa tabela do zbierania wyników

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Numer pracownika | Zadowolenie z obecnych aplikacji | | | | |
| 1 | 2 | 3 | 4 | 5 |
| 1  2  3  4  5 |  |  |  |  |  |
| Razem |  |  |  |  |  |

Jak widać obserwacja może przybrać anonimowości najważniejsze jest zebranie ogólnego rezultatu obserwacji dostosowując pytania do celów obserwacyjnych.

## Rozważanie hipotezy i problemu badawczego

Rozważając problem badawczy jakim jest określenie czy stworzenie systemu opartym na szkielecie next.js jest trudne oraz hipotezy, która można podtrzymać, śmiało można stwierdzić, że stworzenie takiego systemu po rozrysowaniu go na diagramach oraz zapoznaniu się z dokumentacją jest zadaniem łatwym. Należy również pamiętać o podążanie z dokumentacją co w znacznym stopniu przyśpieszy pisanie systemu.

# ROZDZIAŁ 4 WYNIKI BADAŃ DOTYCZĄCE ISTOTNYCH FUNKCJONALNOŚCI



## Wyniki badań dotyczące aplikacji obecnie używanych

Pierwszym wariantem na którym skupiły się badania jest to wygląd aplikacji czyli interfejs. Jak wiadomo dla użytkowników korzystających z jakichkolwiek aplikacji czy stron internetowych pierwszym co się widzi to wygląd. Najczęściej zależy użytkownikom na jak najbardziej intuicyjnym korzystaniu oraz schludnym wyglądzie. Obserwacja skupiała się na aplikacji Listonic i brały udział dwie strony interesariuszy: osoby zamawiające oraz kierowcy. Oceniane jest w skali od jedynki do piątki, gdzie jedynka to bardzo słabo, natomiast piątka bardzo dobrze.

Tabela 5 Tabela obserwacji satysfakcji z interfejsu obecnie używanych aplikacji

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Numer Pracownika | Wygląd aplikacji | | | | |
| 1 | 2 | 3 | 4 | 5 |
| 1. Kierowca  2. Kierowca  3. Kierowca  4. Kierowca  5. Kierowca  6. Zamawiający  7. Zamawiający |  | X  X  X | X  X  X | X |  |
| Razem | 0 | 3 | 3 | 1 | 0 |

Jak wykazuje tabela odczucia interesariuszy nowej aplikacji względem starego interfejsu są raczej negatywne. Pytania dotyczące takich ocen wśród kierowców, gdzie wystąpiła jedna czwórka są różne. Między innymi negatywne oceny spowodowane są  częstymi aktualizacjami, które mają wpływ na wygląd interfejsu, zmieniając go co powoduje ciągłą potrzebę dostosowywania się do nowego wyglądu. Kierowcy oraz osoby zamawiające zwrócili jeszcze uwagę na brak dostosowania jednostek do charakteru przedsiębiorstwa. Co  prawda można dopisać ręcznie wpisując w rubrykę listy zakupów, że potrzebna jest na przykład jedna binda, lecz w późniejszym rozrachunku dla kierowców jest to nieczytelne. Przez obu interesariuszy uwaga skierowana była również na zbyt dużo, rzeczy wyświetlających się, a nie są używane przez przedsiębiorstwo. Jeden z kierowców dał ocenę cztery, może być to spowodowane korzystaniem przez niego z telefonu marki iphone z systemem operacyjnym ios, gdzie wygląd trochę się różnił niż na urządzeniach z  systemem operacyjnym android. Zamawiający dali zgodnie ocenę dwa, gdzie należy zwrócić uwagę, że obaj pracują na komputerach, a kierowcy na telefonach, tak więc dostosowując funkcjonalności należy przy projektowaniu interfejsu zwrócić uwagę również na opinię osób zamawiających.

Płynność działania jako kolejny wariant obserwacji będzie miał istotny wpływ na  wybór technologii aplikacji by dostosowywać technologie w taki sposób by aplikacja działała szybko i płynnie. W dalszym rozwoju aplikacji będzie można przejść na przykład na GraphQL zamiast używać Rest API czy możliwość przepisania backendu aplikacji z api routes, które są wykorzystywane w next.js na dedykowany szkielet dla aplikacji backendowych. W tej obserwacji wyniki również są przedstawione w formie tabeli i skali od jednego do pięciu, gdzie jeden oznacza niepłynność, a pięć oznacza brak uprzedzeń co  do  płynności działania.

Tabela 6 Tabela obserwacji płynności aplikacji

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Numer Pracownika | Płynność aplikacji | | | | |
| 1 | 2 | 3 | 4 | 5 |
| 1. Kierowca  2. Kierowca  3. Kierowca  4. Kierowca  5. Kierowca  6. Zamawiający  7. Zamawiający |  |  | X  X  X | X  X  X | X |
| Razem | 0 | 0 | 3 | 3 | 1 |

Jak widać w tej obserwacji wyniki wypadły nieco lepiej. Zauważyć należy, że  kierowca obsługujący aplikację na urządzeniu z systemem operacyjnym ios znów dał najwyższą z ocen więc można założyć, że jego spostrzeżenia opierają się o sprzęt, który używa. Pozostali kierowcy mniej więcej jednogłośnie oceniają płynność aplikacji na dobrą. Jednakże osoby zamawiające ocenili wydajność i płynność aplikacji najsłabiej. Warte zaznaczenia jest tutaj również urządzenie z którego pracują, a jest to komputer z systemem operacyjnym windows. Należy więc zwrócić uwagę również na tych interesariuszy, przeprowadzić z nimi wywiad dlaczego uważają, że obecna aplikacja nie spełnia ich wymagań dotyczących płynności działania. A może tak jak i w przypadku kierowcy z innym systemem operacyjnym, odczucia dotyczące płynność działania związane są ze sprzętem, a  więc w takim przypadku będzie trzeba rozwijać aplikację w taki sposób, żeby była jak najbardziej wydajna.

Częstotliwość aktualizacji jest bardzo ważnym czynnikiem, szczególnie jeśli chodzi o aplikacje na której opiera się przedsiębiorstwo i one właśnie nie ma wpływu na ich częstotliwość czy czas występowania. Wraz z aktualizacjami mogą wystąpić również prace serwisowe, które jeśli odbędą się w czasie gdy przedsiębiorstwo pracuje, może znacznie sparaliżować komunikację i procesy logistyczne przedsiębiorstwa. Tabela przedstawia wyniki obserwacji, gdzie jeden oznacza, że aktualizacje pojawiają się rzadko, a pięć oznacza, że aktualizacje występują często.

Tabela 7 Wyniki obserwacji częstotliwości występowania aktualizacji

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Numer Pracownika | Płynność aplikacji | | | | |
| 1 | 2 | 3 | 4 | 5 |
| 1. Kierowca  2. Kierowca  3. Kierowca  4. Kierowca  5. Kierowca  6. Zamawiający  7. Zamawiający | X  X  X  X  X | X  X |  |  |  |
| Razem | 5 | 2 | 0 | 0 | 0 |

Obserwacje pokazują, że częstotliwość aktualizacji jest bardzo niska i jest to wynik dotyczący interesariuszy obu grup. Tak dobre w tym przypadku wyniki wymagają zadania następnego pytania czy te aktualizacje, które występują są szkodliwe dla odczuć korzystających z niej lub czy jest wręcz odwrotnie. Tutaj zgodnie obie grupy interesariuszy zgadzają się ze sobą, że przy aktualizacjach zmieniających interfejs trzeba poświecić odpowiednią ilość czasu na przyzwyczajenie się do korzystania z nowego wyglądu. Badani zaznaczyli również, że jedna aktualizacja szczególnie przyczyniła się do negatywnych odczuć związanym z korzystaniem z aplikacji. Wprowadziła on pola tekstowe, które zamiast przy dużej ilości tekstu powiększać się to pozostawały takie same dodając na końcu trzy kropki i to wszystko sprawiło, że na ekranach dużych typu monitor wszystko było widoczne lecz listy zakupów na aplikacji na urządzenia mobilne były nieczytelne dla kierowców oraz niewygodnie do korzystania bo do zobaczenia pełnej pozycji trzeba było kliknąć w  przedmiot który interesuje kierowcę i to ukazywało dopiero pełny opis co przy dla przykładu pięćdziesięciu pozycjach do kupienia jest strasznie niewygodne. Wyróżnić tutaj można jest pytanie o prace serwisowe i tutaj odpowiedź brzmi, że zdarzyło się kilka które szczególnie utrudniły pracę danego dnia ponieważ nie było połączenia z bazami danych co  spowodowało brak możliwości wstawiania przedmiotów na listy. Pozostałe formy prowadzenia prac serwisowych nie utrudniały pracy przedsiębiorstwa.

Ilość funkcjonalności w aktualnie korzystanej aplikacji jest następnym kryterium obserwacji. Przedsiębiorstwo nie korzysta z dedykowanej aplikacji, która służy do  komunikowania się pomiędzy interesariuszy, jest to bardziej lista zakupów dla gospodarstw domowych lecz niektóre funkcjonalności, które w niej występują sprawdzają się codziennej pracy. Tabela przestawia wyniki obserwacji lecz tutaj odpowiedzieć można było tylko twierdząco lub przecząco w celu zweryfikowania potrzeby tworzenia nowych funkcjonalności.

Tabela 8 Wyniki obserwacji zadowolenia dotyczącego ilości funkcjonalności w obecnie korzystanej aplikacji

|  |  |  |
| --- | --- | --- |
| Numer pracownika | Czy ilość funkcjonalności jest wystarczająca w obecnie używanej aplikacji? | |
| TAK | NIE |
| 1. Kierowca  2. Kierowca  3. Kierowca  4. Kierowca  5. Kierowca  6. Zamawiający  7. Zamawiający |  | X  X  X  X  X  X  X |
| Razem | 0 | 7 |

W tej obserwacji jest jednoznaczne, że aplikacja z której obecnie korzysta przedsiębiorstwo nie zaspokaja wszystkich potrzeb, które wymagane są przez charakter pracy przedsiębiorstwa. Wśród obecnie wykorzystywanych funkcjonalności w aplikacji największym zgłaszanym zastrzeżeniem jest brak dostosowywania jednostek do charakteru przedsiębiorstwa. Wiadomo że zwykła lista zakupowa posiada tylko coś takiego jak ilość na  przykład „Kisiel ilość 3” i w aktualny sposób dodawanie odpowiedniej pozycji do nabycia z hurtowni wygląda w następujący sposób: „serwetki białe pakowane 100szt – 1”. W tym przykładzie nie wiadomo czy chodzi o jedną bindę serwetek czy o jeden cały karton, gdzie występuje o wiele więcej bind. Konflikt ten sprawia, że marnuje się czas na zbędne telefony wykonywane przez kierowcę do biura w celu ustalenia ilości lub jeśli ktoś tego nie wykona dokonanie złego zakupu co następnie powoduje dodatkową pracę w celu zwrócenia zbędnego towaru, gdyż przedsiębiorstwo unika magazynowania zbędnego towaru z powodu małej ilości przestrzeni magazynowej. W przypadku kontaktu z biurem występują problemy, gdyż trzeba sprawdzić raporty z dnia poprzedniego by znaleźć pozycję, która interesuje kierowcę, a to również pochłania czas. A więc przypadku artykułów biurowych występuje tych jednostek znacznie więcej jak właśnie bindy, kartony czy ryzy papieru, które uzupełniane regularnie zaoszczędzą dużo błędów oraz czasu upłynniając procesy logistyczne firmy. Następnie chęci interesantów były zwrócone w stronę przekazywania zamówień do telefonów, a co za tym idzie zostawianie tam wyraźnych notatek, które na  kartce są często pomijane oraz ustawianie priorytetu zamówienia jako ważne, gdy  dotychczas odbywało się to po prostu za pośrednictwem komunikacji na Messengerze co często również jest nieodczytywane i później klient jest niezadowolony.

Procesy logistyczne czyli odbiór, segregacja oraz dowód towaru do klienta są  podstawą działalności tego przedsiębiorstwa. Tak więc oczywistym jest, żeby  zaobserwować nastroje panujące wśród interesantów dotyczących właśnie tego aspektu. Tabela przedstawia wyniki, gdzie pięć to całkowite zadowolenia z procesów logistycznych przy korzystaniu z obecnych aplikacji komunikacyjny, a jeden jest to chęć całkowitej zmiany.

Tabela 9 Wynik obserwacji procesów logistycznych przy pracy z aplikacjami obecnie wykorzystywanymi

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Numer Pracownika | Procesy logistyczne | | | | |
| 1 | 2 | 3 | 4 | 5 |
| 1. Kierowca  2. Kierowca  3. Kierowca  4. Kierowca  5. Kierowca  6.Zamawiający  7. Zamawiający |  | X  X  X | X  X | X  X |  |
| Razem | 0 | 3 | 2 | 2 | 0 |

Większość kierowców nie jest usatysfakcjonowana z obecnych systemów komunikacyjnych i argumentują to strasznym zamieszaniem pomiędzy aplikacjami, gdzie  czasem by informacje dotyczące zamówienia są przekazywane na chatach zewnętrznych typu Messenger i jest to mało wygodne i czytelne dla nich. Osoby zamawiające z drugiej strony są bardziej usatysfakcjonowane, gdyż ich praca kończy się na wstawieniu odpowiedniej pozycji na daną listę. Nie dali najwyższej oceny, ponieważ narzekają na liczne telefony z pytaniami o dokładną ilość czy daną pozycję o kierowców. A  wiec najlepszym rozwiązaniem byłoby przeniesienie wszystkich najważniejszych informacji dotyczących czy to produktów do nabycia czy zamówień w jedno miejsce, którym była by aplikacja.

W dynamicznie rozwijającym się świecie pod względem rozwoju technologii coraz więcej rzeczy można zrobić przez aplikację bądź przeglądarkę, dlatego też warto byłoby zaobserwować czy w przedsiębiorstwie zajmującym się artykułami biurowymi też  wystąpiłaby chęć przejścia z papieru na elektronikę. Tabela przedstawia odpowiedzi jedynie zero jedynkowo, albo tak, albo nie.

Tabela 10 Tabela chęci przejścia z papieru na elektronikę w przedsiębiorstwie

|  |  |  |
| --- | --- | --- |
| Numer pracownika | Chęć przejścia z papieru na elektronikę | |
| TAK | NIE |
| 1. Kierowca  2. Kierowca  3. Kierowca  4. Kierowca  5. Kierowca  6. Zamawiający  7. Zamawiający | X  X  X  X  X  X | X |
| Razem | 6 | 1 |

Jak widać zdecydowana większość wyraża chęć przejście na elektronikę uzasadniając to tym, że jako osobą młodym łatwiej korzystać z telefonu komórkowego, a  niżeli zaznaczać zamówienia na kartkach. Dodatkowo jak wiadomo kartki z fakturami czy w-z zdecydowanie częściej się gubią niż telefony. Warte zaznaczenia jest to, że osoba, która nie wyraziła chęci na przejście z papieru na elektronikę jest zdecydowanie starsza od całej reszty interesariuszy, a więc uzasadnia to tym, że tak naprawdę nie przeszkadza mu korzystanie z papierów. Dodatkowo interesariusze zainteresowani zaznaczyli, że  są  zainteresowani korzystaniem z miesięcznych generowanych raportów dla na przykład listy obecności.

Pod obserwację został również wzięty chat grupowy, który aktualnie odbywa się przez komunikator Messenger. Przy tej obserwacji również odpowiedzi będą podzielone na zero jedynkowe, gdzie badana będzie chęć na przejście na chat do aplikacji.

Tabela 11 Tabela obserwacji chęci przeniesienia chatu grupowego do aplikacji

|  |  |  |
| --- | --- | --- |
| Numer pracownika | Chęć przeniesienia chatu grupowego do aplikacji | |
| TAK | NIE |
| 1. Kierowca  2. Kierowca  3. Kierowca  4. Kierowca  5. Kierowca  6. Zamawiający  7. Zamawiający | X  X  X  X | X  X  X |
| Razem | 4 | 3 |

W tej obserwacji zdania są dosyć podzielone, osoby, które nie wyraziły takiej chęci uzasadniały to dużym korzystaniem z aplikacji Messenger wiec nie robiłoby to dla nich większej różnic, lecz osoby wyrażający chęć takiego przejścia deklarowały, że byłby to ciekawy sposób na segregację rozmów i odłączenie życia prywatnego od spraw związanych z pracą.

## Pożądane funkcjonalności wśród interesariuszy

Wśród pożądanych funkcjonalności wynikających z obserwacji najczęściej występuje lista zakupów, gdyż dużym problemem w aplikacjach z których aktualnie korzysta przedsiębiorstwo jest niestabilne działanie takie jak: niepłynne działanie czy problemy z połączeniem się z bazą danych. Problemy występują również w pracach serwisowych nad aplikacją czy aktualizacją, które zmieniają funkcjonalności czy też sam wygląd aplikacji co powoduje zamieszania w działaniu przedsiębiorstwa oraz procesach logistycznych. Ponadto aplikacja posiada dużo reklam ze względu na sposób zarobkowy tej aplikacji. Aby reklamy zlikwidować należałoby wykupić subskrypcję, na którą przedsiębiorstwo nie chce inwestować pieniędzy. A więc stworzenie takiej funkcjonalności na długi okres czasu pozwoli usprawnić procesy logistyczne poprzez intuicyjne działanie. Mimo, iż funkcjonalność ze względu na sam kod czy wygląd wydaje się bardzo łatwy, ponieważ na front-endzie należy napisać parę komponentów, a na back-endzie prosty CRUD (Create, Read, Update, Delete) to jest bardzo dopasowana i pożądana dla przedsiębiorstwa zajmującego się sprzedażą artykułów biurowych.

Kolejnym pożądaną funkcjonalnością jest lista zamówień. W tej podstronie widok powinien być na dwa widoki zależny od roli. Jedną z roli jest kierowca, a więc kierowca powinien widzieć jedynie swoje zamówienia, które wywozi i dostosowane będzie to rejonu w którym jeździ. Osoby odpowiedzialne za zamawianie będą natomiast miały dostęp do  wszystkich zamówień. Po kliknięciu w dane zamówienie strona powinna przenosić pod  endpoint z id zamówienia z widokiem dokumentu w-z. Lista zamówień może zostać podłączona do platformy zakupowej co sprawi że zamówienia będą od razu widoczne z  poziomu aplikacji. Osoby zamawiające będą mogły dodawać priorytet też dla danego zamówienia dla przykładu firma zamyka się szybciej lub mają spotkanie i potrzebują artykułów biurowych, a więc kierowca od razu będzie mógł dostosować swój przejazd pod daną firmę, żeby być w niej do konkretnej godziny. Widoczne będą również komentarze zostawiane przez klientów w zamówieniach lub wystąpi możliwość dodawania notatek przez osoby zamawiające.

Obserwacje wskazują również na potrzebę kontroli obecności w pracy przez kierowców. Jako, że wszyscy kierowcy są zatrudnieni na umowę zlecenie występuje tutaj obowiązek kontroli godzinowej nakładanej przez zarząd przedsiębiorstwa. Także funkcjonalność ta może w fazie rozwoju aplikacji przyjąć formę fizycznego kalendarza, a  na  początku może być również jako lista, gdzie zawarte będą daty pobierane z systemu z  zablokowaniem na dany dzień i przedział godzinowy, aby nie dało się uzupełniać obecności w inne dni co pomoże w sporządzaniu raportów miesięcznych. Będzie również obowiązek wprowadzenie liczby godzin licząc od godziny ósmej. Numer rejestracyjny przypisany na dany dzień będzie pobierany z bazy danych z ustawień profilu. Możliwość będzie również zaznaczenia czy dany kierowca brał udział w wieczornej segregacji towaru, które jest dodatkowo płatne.

Zadania to następna funkcjonalność, którą kierowcy jak i osoby zajmujące się zamówieniem towaru wykazały zainteresowanie w tej funkcjonalności gdyż karta ta może oszczędzić kontakt telefoniczny, który dla kierowców może być niekomfortowy, a w ciągu dnia mogą wydarzyć się różne wydarzenia, które będą wpisywane jako zadania do zrobienia. Przykładami takich zadań mogą być czynności dotyczące pojazdów, gdyż kierowcy mają samochody „pod dom”, czy też zadania dotyczące obsługi klienta jak odebranie zwrotu towaru czy ten nawiązanie kontaktu z klientem w celu ustalenia godziny dostawy.

## Implementacja

Dotychczas z aplikacji wspomagających pracę korzystano z telefonów z systemem android oraz ios jak i z komputerów stacjonarnych czy laptopów z systemem operacyjnym windows. Tworzona aplikacja obsługiwana będzie przez przeglądarkę internetową, a wiec i  w tym przypadku jest możliwość z każdego urządzenia odsługujące przeglądarkę internetową na przykład chrome. Jako, iż pierwotnie nie są planowane takie funkcjonalności jak elektroniczny podpis to też nie ma potrzeby używania telefonów z rysikiem. W aplikacji może wystąpić funkcja druku, a więc do wymagań dojdzie również posiadanie drukarki, aby  w pełni korzystać z aplikacji. Zalecaną przeglądarką do obsługi jest chrome z tego względu, że pisane style są oparte właśnie o tą przeglądarkę, ponieważ dla firefox kaskadowy arkusz styli czasem może się różnić.

Nowa aplikacja to też nowy sposób funkcjonowania przedsiębiorstwa, a wiec najlepiej jest wprowadzić nową aplikację jak każdy będzie w firmie, aby móc przeprowadzić krótkie spotkanie zaznajamiające z aplikacją. Dobrą propozycją wdrożenia aplikacji jeśli chodzi o czas jest początek miesiąca ze względu też na kalendarz, gdzie wypłaty będą już wypłacone i listę obecności będzie uzupełniać się od początku. Miejscem wprowadzenia jest oczywiście część firmy, która odpowiedzialna jest za zamawianie oraz ta część firmy, która odpowiedzialna jest za dowóz.

Problemami oraz wyzwaniami może być przyzwyczajenie się do interfejsu aplikacji, gdyż może zająć to trochę czasu lecz powinno być dosyć łatwe. Problemem może być też brak internetu w firmie, gdyż nie będzie możliwości połączenia się z aplikacją, ale też nie będzie wtedy możliwości składania zamówień i dostępu do platformy zakupowej, a wiec problem ten będzie nierozwiązywalny przez osoby odpowiedzialne za aplikację. Wyzwaniami będzie również generowanie raportów przez osoby korzystające z aplikacji, gdyż aplikacja nie będzie miała takiej funkcjonalności na samym początku aplikacji i będzie potrzeba kontaktu z programistą, który ma dostęp do bazy danych.

Integracja z platformą zakupową przyjdzie w trakcie rozwoju aplikacji. Wystąpi jednak potrzeba do napisania od początku lub przerobienia API dotychczasowego, aby móc pobierać oraz wyświetlać dane na front-endzie aplikacji. Będzie też potrzeba dostosowania rejonów zamówienia, aby zwracała odpowiednie wartości. To wszystko pomogłoby przejść z obsługi zamówień w formie papierowej na elektroniczną. Wystąpiłaby też konieczność z  programistami, którzy byli odpowiedzialni za napisanie platformy.

Należałoby do prawidłowego działania przedsiębiorstwa dać czas na testy działania, można przyjąć, że przez pierwszy miesiąc korzystania z aplikacji należy przeprowadzać obserwację: czy procesy logistyczne są bardziej płynne, czy interesariusze są zadowoleni z  korzystania aplikacji, czy wpływa to na wykorzystanie papieru ksero, a jeśli tak i  zapotrzebowanie w przedsiębiorstwie się zmniejsza to czy przynosi to oszczędności dla przedsiębiorstwa, czy aplikacja działa płynnie podczas prawdziwych wyzwań jak ilość dziennych zamówień oraz czy występują błędy i awarie podczas dziennego korzystania z  aplikacji. Następnie na koniec miesiąca należałoby przeprowadzić ankietę z pytaniami zamkniętymi, gdzie pytania mogłyby wyglądać w następujący sposób: czy korzystanie było przyjemniejsze, niż używanie papieru, czy według Ciebie aplikacja upłynniła procesy logistyczne, czy interfejs aplikacji jest orientacyjny oraz z jednym przykładowym pytaniem otwartym: czego oczekiwałbyś od aplikacji w przyszłości? To pytanie pomogłoby nadać kierunek rozwoju aplikacji. Gdyby większość aplikacji była negatywna można by  przeprowadzić wywiad grupowy odnośnie zmian w aplikacji, które są niezbędne do  usprawnienia jej działania czy też zmiany interfejsu by był bardziej wygodny i  orientacyjny dla interesariuszy.

## Rozwój aplikacji

W trakcie rozwoju projektu dobrym pomysłem byłoby przepisać całą aplikację na  framework dla urządzeń mobilnych takich jak flutter czy react native, aby móc obsługiwać ją na android oraz ios. Gdyby to nastąpiło trzeba byłoby napisać backend aplikacji również od początku bo żadne z tych frameworków nie ma możliwości takich jak next.js poprzez api routes, gdzie można napisać od razu backend. Do tego można by  wykorzystać nest.js, aby zostać w technologiach opartych na języku javascript. Aplikacja możliwa byłaby do pobrania poprzez sklep play w android oraz App store w ios. Pomogłoby to też na rozwój aplikacji w taki sposób, aby inne małe przedsiębiorstwa mogły korzystać z  aplikacji, jeśli byłaby możliwość rozpowszechnienia.

Gdyby wprowadzić aplikację dedykowaną dobrym pomysłem byłoby skorzystanie z  headless cms. Jest to system do zarządzania treścią sprawi to że front-end będzie całkowicie odizolowany od backendu. Korzystać z headless cms będą osoby odpowiedzialne za  zamawianie co sprawi, że nie będą musieli przyzwyczajać się do nowego interfejsu jakby takie były wprowadzone. Jest to bardzo łatwy sposób do dodawania nowych treści jak  zadania czy notatki. Wprowadzenia handless cms spowoduje skrócenie kodu na front-endu bo nie będzie trzeba generować dwóch widoków dla osoby zamawiającej oraz  dla  kierowców. Technologiami możliwymi do użycia były by Contento czy strapi, które są  jedne z najlepszych obecnie headless cmsów wykorzystywanych w web developmencie.

Elektroniczny podpis jest używany przez wiele firm kurierskich czy też pocztę polską. Wprowadzenie tej technologii wymagałoby zaznajomienie się z dodatkowymi aktami prawnymi. Obecnie za tą kwestię odpowiedzialne są dwa główne akty prawne rozporządzenie UE nr 910/2014 z 23.7.2014 r. w sprawie identyfikacji elektronicznej i usług zaufania w odniesieniu do transakcji elektronicznych na rynku wewnętrznym oraz ustawa z 05.09.2016 r. o usługach zaufania oraz identyfikacji elektronicznej. Wprowadzenie elektronicznego podpisu miałoby sens, gdy aplikacja byłaby w takiej wersji rozwoju, że  przedsiębiorstwo nie korzystałoby już do obsługi zamówień z papieru ksero. Spowodowałoby to, że przedsiębiorstwo byłoby bardziej przyjazne środowiskowo, gdyż  papier ksero zużywany byłby w znacznie mniejszej ilości.

Gdyby w firmie była chęć i opcja wprowadzenia zamawiającego towaru do systemu poprzez aplikację to istniałaby możliwość generowania raportów oraz faktur drogą elektroniczną bez konieczności wpisywania z papieru do systemu. Na takie wprowadzenie musiałby też się zdecydować zarząd, który trzyma dużą ilość faktur papierowych, które są  zarchiwizowane oraz zespół odpowiedzialny za fakturowanie i wprowadzanie towaru do  systemu. Możliwe, że spowodowałoby to, że niektóre osoby w tym zespole niebyły by  konieczne co zaoszczędziłoby pieniędzy dla przedsiębiorstwa.

Przedsiębiorstwo korzysta aktualnie z komunikatora Messenger, co powoduje kolejną aplikację do obsługi przedsiębiorstwa. W przyszłości rozwoju aplikacji dobrym pomysłem na funkcjonalność byłoby dodanie wewnętrznego chatu grupowego, gdzie połączony byłby z bazą danych przedsiębiorstwa co spowoduje, że byłaby mniejsza szansa na wykradzenie wrażliwych danych z tego względu, że często na chat grupowy są wysyłane zdjęcia faktur czy też w-z. Na rynku są fajne technologie backendowe do tworzenia tego typu rzeczy jak np. socket.io. Warte dodania było by oznaczenie danego użytkownika coś na przykładzie Messengera, gdzie „@Adrian Kowalski” sprawia, że użytkownikowi o tym imieniu i nazwisku wyskakuje powiadomienie i szybciej jest możliwość, że odpowie na dany temat.

## Analiza problemu badawczego oraz hipotezy

Jako, że w czwartym rozdziale hipotezą postawiona została aby sprawdzić czy interesariusze nowej aplikacji są niezadowoleni z dotychczasowych rozwiązań można podtrzymać tą hipotezę ze względu na podzielone zdania w zależności jaki aspekt był rozpatrywany. Problem badawczy dotyczący tego rozdziału należy wziąć pod uwagę z jakich funkcjonalności interesariusze są najbardziej niezadowoleni ze względu na możliwość wprowadzania funkcjonalności po kolei do aplikacji, a więc istnieje również możliwość korzystania z dotychczasowych rozwiązań w równi z nową aplikacją.

# ZAKOŃCZENIE

Głównym celem pracy dyplomowej oraz pokazanie w jaki sposób można usprawnić komunikację między osobami odpowiedzialnymi za zamawianie towarów z hurtowni zewnętrznych oraz kierowcami odpowiedzialnymi za dowóz towaru do klienta w przedsiębiorstwie zajmującym się sprzedażą artykułów biurowych za pomocą prostego systemu informatycznego. Dlatego też praca przyjęła hipotezę główną w  następującej postaci „Wprowadzenie aplikacji internetowej wspomagającej pracę kierowców w firmie zajmującej się sprzedażą artykułów biurowych, która przyczyni się do  zwiększenia efektywności działań logistycznych oraz poprawy obsługi klienta”. W pracy zawarte zostały hipotezy poboczne odpowiednio do każdego rozdziału. Przedstawiony został również główny problem badawczy „Jakie funkcjonalności aplikacji mogłyby efektywnie wspomagać pracę kierowców w firmie zajmującej się sprzedażą artykułów biurkowych, mając na uwadze optymalizację procesów logistycznych?”

W pierwszym rozdziale jako, że był to rozdział teoretyczny to zostały przeanalizowane różne technologie pod wieloma względami jak na przykład wydajność. Przedstawiony został język w którym projekt zostanie napisany oraz przedstawione szkielety oraz biblioteki z których aplikacja będzie korzystać. Wybranymi technologiami były: język javaScript z nakładką typeScript, który umożliwia statyczne typowanie oraz szkielet next.js w którym można pisać front-end jak i wydajny back-end aplikacji poprzez API routes. Technologie te zostały przyrównane do innych popularnych rozwiązań i  szkieletów takich jak Angular czy Vue.js. Biblioteki opierały się głównie na wyglądzie komponentów aplikacji, wykorzystana została biblioteka Nextui, która dostarcza bardzo ładnych i modyfikowalnych komponentów aplikacji przy bardzo prostym użyciu. Została również wybrana baza danych analizując wszystkie dostępne możliwości na rynku oraz technologie SQL jak i NoSQL i tutaj ze względu, iż aplikacja nie będzie duża wybrano rozwiązania NoSQL i bazę mongoDb. Rozwiązania które oferuje next.js są renderowanie serwerowe oraz klineta co sprawia, że aplikacja działa wydajniej niż w innych szkieletach. Przedstawione zostały również diagramy UML takie jak diagram przypadków użycia, diagram klas czy diagram ERD czyli diagram związków encji. Pomagają one w znacznym stopniu zaprojektować oraz udokumentować działanie aplikacji. W projekcie został również użyty system kontroli wersji i w tym rozdziale również były opisywane i porównywane różne systemy jednakże został wybrany Git i platforma Github. Na końcu rozdziału przedstawione zostały technologie wykorzystywane w przedsiębiorstwach takie jak  rozwiązania informatyczne w obszarze logistyki oraz zostali przedstawieni interesariusze aplikacji.

W drugim rozdziale została przedstawiona charakterystyka przedsiębiorstwa jak i analizę rynku artykułów biurowych w Polsce jak i  problemy logistyczne w pracy codziennej, aby można było ustawić na jakie funkcjonalności są najbardziej pożądane przez interesariuszy. Zostały również przedstawione aplikacje, które aktualnie są używane przez przedsiębiorstwo oraz aplikacje, które są obecnie na rynku. Została przedstawiona charakterystyka przedsiębiorstwa oraz analiza procesów przedsiębiorstwa co pozwoliło kolejnych częściach rozdziału zostały przedstawić propozycje niezbędnych funkcjonalności, które powinny pojawić się w aplikacji, aby usprawnić procesy logistyczne. Na końcu rozdziału zostały także przedstawione aspekty prawne i regulacyjne takie jak: ustawa o ochronie danych osobowych, ustawę o świadczeniu usług drugą elektroniczną, ustawa o prawie autorskim i prawach pokrewnych, ustawa o  ochronie baz danych.

W trzecim rozdziale został przeanalizowany został system oraz techniki badawcze pod względem zaczęcia dedykowanej aplikacji, przedstawiona została architektura plików w projekcie oraz dokładny opis z każdego folderu jaki występuje w  projekcie. Jeśli chodzi o biblioteki zostały przedstawione oraz opisane jak i pokazane na rysunku dokładne użycie oraz wykorzystanie. Wspomniane zostało również o procesie autoryzacji, który jest ważnym aspektem projektu, gdyż aplikacja oparta jest na rolach po  których dostosowywane są widoki. Role odzwierciedlają interesariuszy czyli osoby odpowiedzialne za zamawianie oraz kierowców. Dzięki autoryzacji również blokuje się  dostęp do podstron do których osoby nieupoważnione nie powinny mieć dostępu. W trzecim rozdziale został przedstawiony system na diagramach wcześniej opisywanych w rozdziale pierwszym. Zostały wstawione rysunki diagramów oraz ich opisy. Na końcu zostały przedstawione badania do aplikacji czyli obserwacji takie jak podmiot obserwacji, cele obserwacji, miejsce i czas dokonywania obserwacji jak i sposób dokonywania i  dokumentowania obserwacji.

Czwarty rozdział na samym początku przedstawia wyniki badań, które zostały przeprowadzone na terenie firmy takie jak: wygląd dotychczasowej aplikacji, płynność jej działania, częstotliwość aktualizacji, ilość funkcjonalności jak i procesy logistyczne podczas korzystania z obecnie używanych aplikacji. Zbadane została również chęć przejścia z papieru na elektronikę oraz chęć przejścia przez interesariuszy na chat grupowy w aplikacji zamiast korzystać z Messengera. W dalszej część pracy zostały przedstawione pożądane wśród interesariuszy funkcjonalności jak lista zakupów czy też elektroniczna lista obecności. Implementacja to kolejny podrozdział. Został rozpatrzony niezbędny sprzęt do użytkowania aplikacji, czas i miejsce implementacji, problemy i wyzwania, integracja z platformą zakupową przedsiębiorstwa oraz czas który należy przeznaczyć na testy. Na samym końcu zostały przedstawione pomysły na dalszy rozwój aplikacji pod względem nowych funkcjonalności, które mogłyby pojawić się w aplikacji.

Dalszymi krokami w rozwoju aplikacji powinno być zebranie danych dotyczących wymagań interesariuszy. Po danym okresie używania aplikacji każdy z interesantów powinien mieć swoje przemyślenia dotyczące co zmienić, co dodać do aplikacji. Można posłużyć się wywiadem, aby zebrać te informacje. Aby dotrzeć do finalnej wersji aplikacji należałby przepisać aplikację na szkielet dla aplikacji na telefon jak na przykład react native czy flutter, tak aby famework obsługiwał dwa rodzaje systemów operacyjnym ios oraz android. To ułatwiłoby większości interesariuszy obsługiwać aplikację, ponieważ nie musieliby wchodzić na przeglądarkę tylko posiadaliby aplikację wgraną na swój telefon. Dla osób zamawiających trzeba by napisać headless cms do zarządzania treścią w aplikacji. Na rynku jest wiele darmowych szkieletów takich jak payload cms czy strapi. Zaoszczędziłoby to tworzenia różnych widoków gdyż, kierowcy nie mieliby dostępu do cms więc nie mogliby dodawać pozycji do list, a osoby zamawiające z łatwością mogliby wykonywać wszystkie aktywności CRUD.

Rozważając wszystkie hipotezy i problemy badawcze można podtrzymać hipotezę główną, a wynika to głównie z obserwacji oraz przeglądu najnowszych technologii. Dało się dostrzec, że stworzenie takiej aplikacji w współpracy z dokumentacją do odpowiednich technologii jest zadaniem do wykonania wystarczy tylko odpowiednia ilość czasu. Badania oraz opisy przedsiębiorstwa oraz analizy pokazują, że funkcjonalności sprecyzowane do przedsiębiorstwa zajmującego się sprzedażą artykułów biurowych są niezbędne i przyczynią się do zwiększenia efektywności działań logistycznych oraz obsługi klienta.

# BIBLIOGRAFIA

**Pozycje zwarte:**

1. Adeel Ehsan, Mohammed Ahmad M. E. Abuhaliqa, Cagatay Catal, Deepti Mishra, \*RESTful API Testing Methodologies: Rationale, Challenges, and Solution Directions\*, Department of Computer Science & Engineering, Qatar University, Doha 2713, Qatar, 2022.
2. Bazy danych NoSQL, Wyższa Szkoła Bankowa w Gdańsku 2021, s. 3-5.
3. Cieplelewska M., Fundusze strukturalne i polityka regionalna [w:] Unia Europejska. Integracja Polski z Unią Europejską, red. E. Kadecka-Wyrzykowska i E. Synowiec, PWN, Warszawa 2023.
4. Dr Charles R. Severance, Python for Everybody. Exploring Data in Python 3, 2023r., s. 9.
5. Interfejsy komunikacyjne baz danych, Wyższa Szkoła Bankowa w Gdańsku 2021, s. 5.
6. Joanna Woźniak, Identyfikacja kanałów dystrybucji i otoczenia konkurencyjnego w branży papierniczo-biurowej na przykładzie wybranego przedsiębiorstwa, Społeczna Akademia Nauk, Rzeszów, 2018r., s. 235-236.
7. Jacek Matulewski, Wersjonowanie i backup kodu, Toruń, 2020r., s. 1-2.
8. Keynes J.M. (red.), Ogólna teoria zatrudnienia, procentu i pieniądza, PWN, Warszawa 2022.
9. Konrad Bielak, Bartłomiej Borek, Małgorzata Plechawska-Wójcik, Analiza obciążeniowa aplikacji internetowych z użyciem szkieletów Angular, React i Vue, Department of Computer Science, Lublin University of Technology, Lublin, 2022r., s. 77, 79, 80-82.
10. Konrad Szynalski, Dawid Różański, Wprowadzenie do modelowania w języku UML, Wrocławska Wyższa Szkoła Informatyki Stosowanej "Horyzont", Wrocław, 2022r., s. 31-32, 33-34, 36.
11. Małgorzata Kamińska, Iwona Szczęsna, Agnieszka Wielgus, Robert Żak, Praca zespołowa jako kompetencja XXI wieku, Novum, Płock, 2021r., s. 31.
12. Mariusz Śliwa, Beata Pańczyk, Porównanie wydajności interfejsów programistycznych na przykładzie REST API, GraphQL i gRPC, Department of Computer Science, Lublin, 2021, s. 360-361.
13. Marta Niewiadomska, Wykorzystanie systemów informatycznych w logistycznej działalności przedsiębiorstw, Wydawnictwo Naukowe Uniwersytetu Szczecińskiego, Szczecin, 2019r., s. 106, 108.
14. Nate Black, Boris Cherny on TypeScript, IEEE SOFTWARE, 2020r., s. 98-99.
15. Pawelec Kamil, Kopniak Piotr, Porównanie szkieletów do wytwarzania aplikacji internetowych dla języka PHP, Wydawnictwo Politechniki Lubelskiej, 2022r., s. 26-27.
16. Robert Ryszard Paciorek, Matematyka dla Ciekawych Świata, 2023r., s. 1.
17. Tursunbek Sadriddinovich Jalolov, PROGRAMMING LANGUAGES, THEIR TYPES AND BASICS, Uzbekistan, 2023r., s. 147.
18. Urinboev Abdushukur Abdurakhimovich, THE FUTURE OF JAVASCRIPT: EMERGING TRENDS AND TECHNOLOGIES, Włochy, 2023r., s. 12-14.

**Pozycje internetowe:**

1. Dane za stroną: encyklopedia.pwn.pl (dostęp 03.06.24r.).
2. Dane za stroną: mfiles.pl (dostęp 03.06.24r.).
3. Dane za stroną: typeorm.io (dostęp 03.05.24r.).
4. Dane za stroną internetową: https://www.prisma.io/ (dostępne 07.04.2024r.).
5. Dane za stroną internetową: https://axios-http.com (dostępne 23.04.2024 r.).
6. Dane za stroną internetową: https://docs.npmjs.com (dostęp 23.04.2024 r.).
7. Dane za stroną internetową: https://legacy.reactjs.org/ (dostęp 23.04.2024 r.).
8. Dane za stroną internetową: https://next-auth.js.org/ (dostęp 25.04.2024 r.).
9. Dane za stroną internetową: https://nodejs.org (dostęp 23.04.2024 r.).

# AKTY NORMATYWNE

1. Ustawa o ochronie danych osobowych z dnia 10 maja 2018r. art. 1, Dz.U.2018 poz. 1000
2. ROZPORZĄDZENIE PARLAMENTU EUROPEJSKIEGO I RADY (UE) 2016/679 z dnia 27 kwietnia 2016 r. w sprawie ochrony osób fizycznych w związku z przetwarzaniem danych osobowych i w sprawie swobodnego przepływu takich danych oraz uchylenia dyrektywy 95/36/WE art.25
3. Ustawa z dnia 18 lipca 2002 r. o świadczeniu usług drogą elektroniczną, art. 12, 13.1, 14.1, 18.1, Dz. U. 2002 Nr 144 poz. 1204
4. Ustawa z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych, art. 1, 12, Dz.U. z 2022 r. poz. 2509

# SPIS RYSUNKÓW

[rys. 1 wersja node 38](#_Toc169955372)

[rys. 2 Wersja npm 38](#_Toc169955373)

[rys. 3 Tworzenie aplikacji w next.js 39](#_Toc169955374)

[rys. 4 Uruchomienie aplikacji 39](#_Toc169955375)

[rys. 5 Architektura plików 40](#_Toc169955376)

[rys. 6 Przykład użycia Buttona z NextUI 41](#_Toc169955377)

[rys. 7 Przykład użycia axios 42](#_Toc169955378)

[rys. 8 GithubProvider 43](#_Toc169955379)

[rys. 9 Inizacjilacja Next auth 44](#_Toc169955380)

[rys. 10 Rozszerzony folder app 45](#_Toc169955381)

[rys. 11 MIddleware 45](#_Toc169955382)

[rys. 12 Diagram przypadków użycia 46](#_Toc169955383)

[rys. 13 Diagram związków encji 47](#_Toc169955384)

[rys. 14 Diagram klas 48](#_Toc169955385)

# SPIS TABEL

[Tabela 1 Telefony\_komórkowe 11](#_Toc169955386)

[Tabela 2 Pracownicy 12](#_Toc169955387)

[Tabela 3 Wszystkie\_dane 12](#_Toc169955388)

[Tabela 4 Przykładowa tabela do zbierania wyników 50](#_Toc169955389)

[Tabela 5 Tabela obserwacji satysfakcji z interfejsu obecnie używanych aplikacji 51](#_Toc169955390)

[Tabela 6 Tabela obserwacji płynności aplikacji 52](#_Toc169955391)

[Tabela 7 Wyniki obserwacji częstotliwości występowania aktualizacji 53](#_Toc169955392)

[Tabela 8 Wyniki obserwacji zadowolenia dotyczącego ilości funkcjonalności w obecnie korzystanej aplikacji 54](#_Toc169955393)

[Tabela 9 Wynik obserwacji procesów logistycznych przy pracy z aplikacjami obecnie wykorzystywanymi 55](#_Toc169955394)

[Tabela 10 Tabela chęci przejścia z papieru na elektronikę w przedsiębiorstwie 56](#_Toc169955395)

[Tabela 11 Tabela obserwacji chęci przeniesienia chatu grupowego do aplikacji 56](#_Toc169955396)

# STRESZCZENIE

Wyższa Szkoła Bankowa w Poznaniu Warszawa, lipiec 2024

Specjalność: Programowanie

**Streszczenie pracy dyplomowej**

Aplikacja internetowa wspomagająca pracę dostawców w przedsiębiorstwie zajmującym się sprzedażą artykułów biurowych

Autor: Paweł Jaruszewski

Promotor: Dr. Inż. Katarzyna Skroban

Słowa kluczowe: Aplikacja internetowa, next.js, react, javascript

Praca dyplomowa rozważa stworzenie aplikacji internetowej, która wspomaga komunikację pomiędzy kierowcami, a osobami zamawiającymi w przedsiębiorstwie zajmującym się sprzedażą artykułów biurowych. Omawia najnowsze technologie javascriptowe takie jak next.js czy szkielety front-endowe. Wyjaśnia użycie poszczególnej bazy danych oraz technologii ORM do połączenia z bazą. Przedstawione są także, diagramy UML takich jak diagram erd czyli diagram encji, diagram klas oraz diagram przypadków użycia, które przedstawiają cały system na diagramach. Praca bada jakie funkcjonalności są najbardziej potrzebne oraz pożądane przez interesantów oraz analizuje poszczególne ustawy i akty prawne dotyczących wytworzenia takiej aplikacji. Analizuje również dotychczasowe rozwiązania stosowane przez przedsiębiorstwo oraz aktualne aplikacje występujące na rynku. Przedstawia krok po kroku jak w łatwy sposób można stworzyć projekt, przedstawia dokładną architekturę plików oraz wyjaśnia biblioteki używane do tworzenia interfejsu. Na samym końcu analizuje wyniki badań dokonanych na przedsiębiorstwie w celu zbadania aktualnego stanu zadowolenia z bieżących aplikacji z których korzysta przedsiębiorstwo oraz wymagania interesariuszy.

WSB University in Poznań Warszawa, July 2024

Specialization: Programming

**Diploma Thesis Abstract**

A web application supporting the work of suppliers in a company selling office supplies

Supervisor: Dr. Inż. Katarzyna Skroban

Keywords: Web application, next.js, react, javascript

The thesis considers the creation of a web application that supports communication between drivers and customers in a company dealing with the sale of office supplies. It discusses the latest JavaScript technologies such as next.js or front-end frameworks. Explains the use of a particular database and ORM technology to connect to the database. Also presented are UML diagrams such as the Erd diagram (entity diagram), class diagram and use case diagram, which represent the entire system in diagrams. The paper examines what functionalities are most needed and desired by the interested parties and analyzes individual acts and legal acts concerning the development of such an application. It also analyzes the current solutions used by the company and the current applications on the market. It shows step by step how to easily create a project, presents the exact architecture of the files and explains the libraries used to create the interface. At the very end, it analyzes the results of research conducted on the company in order to examine the current state of satisfaction with the current applications used by the company and the requirements of stakeholders.

1. Dane za stroną: encyklopedia.pwn.pl (dostęp 03.06.24r) [↑](#footnote-ref-1)
2. Dane za stroną: mfiles.pl (dostęp 03.06.24r.) [↑](#footnote-ref-2)
3. *Bazy danych NoSQL*, Wyższa Szkoła Bankowa w Gdańsku 2021, s.4 [↑](#footnote-ref-3)
4. Ibidem, s.3 [↑](#footnote-ref-4)
5. *Bazy danych NoSQL*, Wyższa Szkoła Bankowa w Gdańsku 2021, s.4-5 [↑](#footnote-ref-5)
6. *Interfejsy komunikacyjne baz danych,* Wyższa Szkoła Bankowa w Gdańsku 2021, s.5 [↑](#footnote-ref-6)
7. Dane za stroną internetową: <https://www.prisma.io/> (dostępne 07.04.2024r.). [↑](#footnote-ref-7)
8. Dane za stroną: typeorm.io (dostęp 03.05.24r.) [↑](#footnote-ref-8)
9. Adeel Ehsan, Mohammed Ahmad M. E. Abuhaliqa, Cagatay Catal, Deepti Mishra, *RESTful API Testing Methodologies: Rationale, Challenges, and Solution Directions*

   *Department of Computer Science & Engineering, Qatar University, Doha 2713, Qatar, 2022* [↑](#footnote-ref-9)
10. Mariusz Śliwa, Beata Pańczyk, *Porównanie wydajności interfejsów programistycznych na przykładzie REST API, GraphQL i gRPC,* Department of Computer Science, Lublin, 2021, s. 360-361 [↑](#footnote-ref-10)
11. Urinboev Abdushukur Abdurakhimovich, *THE FUTURE OF JAVASCRIPT: EMERGING TRENDS AND TECHNOLOGIES*,Włochy, 2023, s. 12-14 [↑](#footnote-ref-11)
12. Nate Black, *Boris Cherny on TypeScript,* IEEE SOFTWARE, 2020r., s. 98-99 [↑](#footnote-ref-12)
13. Pawelec Kamil , Kopniak Piotr, *Porównanie szkieletów do wytwarzania aplikacji internetowych dla języka PHP,* Wydawnictwo Politechniki Lubelskiej, 2022r., s. 26-27 [↑](#footnote-ref-13)
14. Dr Charles R. Severance, *Python for Everybody. Exploring Data in Python* 3,2023r.,s. 9 [↑](#footnote-ref-14)
15. Robert Ryszard Paciorek, *Matematyka dla Ciekawych Świata,* 2023r., s. 1 [↑](#footnote-ref-15)
16. Tursunbek Sadriddinovich Jalolov, *PROGRAMMING LANGUAGES, THEIR TYPES AND BASICS, Uzbekistan, 2023r., s. 147* [↑](#footnote-ref-16)
17. Ibidem *s. 147* [↑](#footnote-ref-17)
18. Konrad Bielak, Bartłomiej Borek, Małgorzata Plechawska-Wójcik, *Analiza obciążeniowa aplikacji internetowych z użyciem szkieletów Angular, React i Vue,* Department of Computer Science, Lublin University of Technology, Lublin, 2022r. s. 77 [↑](#footnote-ref-18)
19. Ibidem, s. 79 [↑](#footnote-ref-19)
20. Konrad Bielak, Bartłomiej Borek, Małgorzata Plechawska-Wójcik, *Analiza obciążeniowa aplikacji internetowych z użyciem szkieletów Angular, React i Vue,* Department of Computer Science, Lublin University of Technology, Lublin, 2022r. s. 79 [↑](#footnote-ref-20)
21. Ibidem, s. 80 [↑](#footnote-ref-21)
22. Konrad Bielak, Bartłomiej Borek, Małgorzata Plechawska-Wójcik, *Analiza obciążeniowa aplikacji internetowych z użyciem szkieletów Angular, React i Vue,* Department of Computer Science, Lublin University of Technology, Lublin, 2022r. s. 80 [↑](#footnote-ref-22)
23. Ibidem, s. 80-82 [↑](#footnote-ref-23)
24. Ibidem, s. 82 [↑](#footnote-ref-24)
25. Jacek Matulewski, *Wersjonowanie i backup kodu*, Toruń, 2020r., s.1 [↑](#footnote-ref-25)
26. Ibidem, s. 1-2 [↑](#footnote-ref-26)
27. Ibidem, s. 2 [↑](#footnote-ref-27)
28. Konrad Szynalski, Dawid Różański, *Wprowadzenie do modelowania w języku UML,* Wrocławska Wyższa Szkoła Informatyki Stosowanej "Horyzont", Wrocław, 2022r., s. 31 [↑](#footnote-ref-28)
29. Ibidem, s. 32 [↑](#footnote-ref-29)
30. Ibidem, s. 32 [↑](#footnote-ref-30)
31. Ibidem, s.34 [↑](#footnote-ref-31)
32. Ibidem, s. 33-34 [↑](#footnote-ref-32)
33. Ibidem, s. 36 [↑](#footnote-ref-33)
34. Marta Niewiadomska, *Wykorzystanie systemów informatycznych w logistycznej działalności przedsiębiorstw,* Wydawnictwo Naukowe Uniwersytetu Szczecińskiego, Szczecin, 2019r., s.106 [↑](#footnote-ref-34)
35. Ibidem, s.108 [↑](#footnote-ref-35)
36. Małgorzata Kamińska, Iwona Szczęsna, Agnieszka Wielgus, Robert Żak, *Praca zespołowa jako kompetencja XXI wieku,* Novum, Płock, 2021r. s.31 [↑](#footnote-ref-36)
37. Aleksander Sobota, Ryszard Janecki, Grzegorz Karoń, Piotr Soczówka, *ROLA INTERESARIUSZY W PROCESIE PROJEKTOWANIA KONCEPCJI KOLEI METROPOLITALNEJ NA PRZYKŁADZIE GÓRNOŚLĄSKO-ZAGŁĘBIOWSKIEJ METROPOLII,* Politechnika Śląska, 2019r., s.110 [↑](#footnote-ref-37)
38. Anna Rybak, *Ocena jakości dostawy a zakupy w e-commerce,* Politechnika Częstochowska, 2019r. s.27 [↑](#footnote-ref-38)
39. Joanna Woźniak, *Identyfikacja kanałów dystrybucji i otoczenia konkurencyjnego w branży papierniczo-biurowej na przykładzie wybranego przedsiębiorstwa*, Społeczna Akademia Nauk, Rzeszów, 2018r., s.235-236 [↑](#footnote-ref-39)
40. Ustawa o ochronie danych osobowych z dnia 10 maja 2018r. art. 1, Dz.U.2018 poz. 1000 [↑](#footnote-ref-40)
41. ROZPORZĄDZENIE PARLAMENTU EUROPEJSKIEGO I RADY (UE) 2016/679 z dnia 27 kwietnia 2016 r. w sprawie ochrony osób fizycznych w związku z przetwarzaniem danych osobowych i w sprawie swobodnego przepływu takich danych oraz uchylenia dyrektywy 95/36/WE art.25 [↑](#footnote-ref-41)
42. Ustawa z dnia 18 lipca 2002 r. o świadczeniu usług drogą elektroniczną, art. 12, 13.1, 14.1, 18.1, Dz. U. 2002 Nr 144 poz. 1204 [↑](#footnote-ref-42)
43. Ustawa z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych, art. 1, 12, Dz.U. z 2022 r. poz. 2509 [↑](#footnote-ref-43)
44. Ustawa z dnia 27 lipca 2001 r. o ochronie baz danych, art.1-8, art. 10 [↑](#footnote-ref-44)
45. Dane za stroną internetową: <https://docs.npmjs.com> (dostęp 23.04.2024 r.) [↑](#footnote-ref-45)
46. Dane za stroną internetową: https://nodejs.org (dostęp 23.04.2024 r.) [↑](#footnote-ref-46)
47. Dane za stroną internetową: https://legacy.reactjs.org/ (dostęp 23.04.2024 r.) [↑](#footnote-ref-47)
48. Ibidem. (dostęp 23.04.2024 r.) [↑](#footnote-ref-48)
49. Dane za stroną: [https://axios-http.com](https://axios-http.com/) (dostępne 23.04.2024 r.) [↑](#footnote-ref-49)
50. Dane za stroną: <https://next-auth.js.org/> (dostęp 25.04.2024 r.) [↑](#footnote-ref-50)