



POLSKO-JAPONŃSKA
AKADEMIA TECHNIK
KOMPUTEROWYCH

Wydział Informatyki

Katedra Sieci Komputerowych
Programowanie Systemowe i Sieciowe

Autor

Mateusz Dylewski

Numer indexu

S18737

OMS - Office Management System

Praca inżynierska napisana pod
kierunkiem:
Michail Mokkas

Warszawa, lipiec 2022 r.

Streszczenie

W dzisiejszych czasach technologia jest nieodłączną częścią każdego aspektu życia człowieka. Jest wykorzystywana zarówno w celach prywatnych jak i służbowych. Firmy chcącą nadążyć za konkurencją, muszą wprowadzać coraz to nowsze rozwiązania i technologie wspomagające pracowników w codziennych zadaniach.

Wychodząc na przeciw stosunkowo młodemu problemowi cyfryzacji i pracy zdalnej w małych przedsiębiorstwach, powstała koncepcja systemu umożliwiającego digitalizację pewnych procesów administracyjnych. W trakcie analizy i projektowania aplikacji, powstały diagramy przedstawiające funkcjonalności systemu oraz dane przechowywane na potrzeby różny modułów.

Rozwiązanie ma formę aplikacji przeglądarkowej. Zostało zaimplementowane przy użyciu technologii Java oraz Vue.js. Ciekawym aspektem projektu jest wykorzystanie WebRTC do przesyłania strumieni wideo oraz dźwięku bezpośrednio między przeglądarkami. Aplikacja wykorzystuje relacyjną bazę danych MySQL do zapewnienia trwałości danych. Natomiast bezpieczeństwa danych pilnuje bibliotek Spring Security wraz z tokenami JWT.

Słowa kluczowe

administracja, digitalizacja, cyfryzacja, aplikacja przeglądarkowa, kalendarz, spotkania, spotkanie, rezerwacja, rezerwacja miejsc, miejsce, miejsca, wideokonferencja, miejsce parkingowe, sala konferencyjna, biurko, hotdesk, hierarchia, użytkownik, vuex, vie router, v-calendar, bootstrap-vue, vie, agora-rtc-sdk-ng, java, spring, spring boot, spring boot data jpa, jpa, web starter, spring starter security, jwt, jjwt, intellij, visual studio code, postman, mysql, baza danych, mysql workbench

Spis Treści

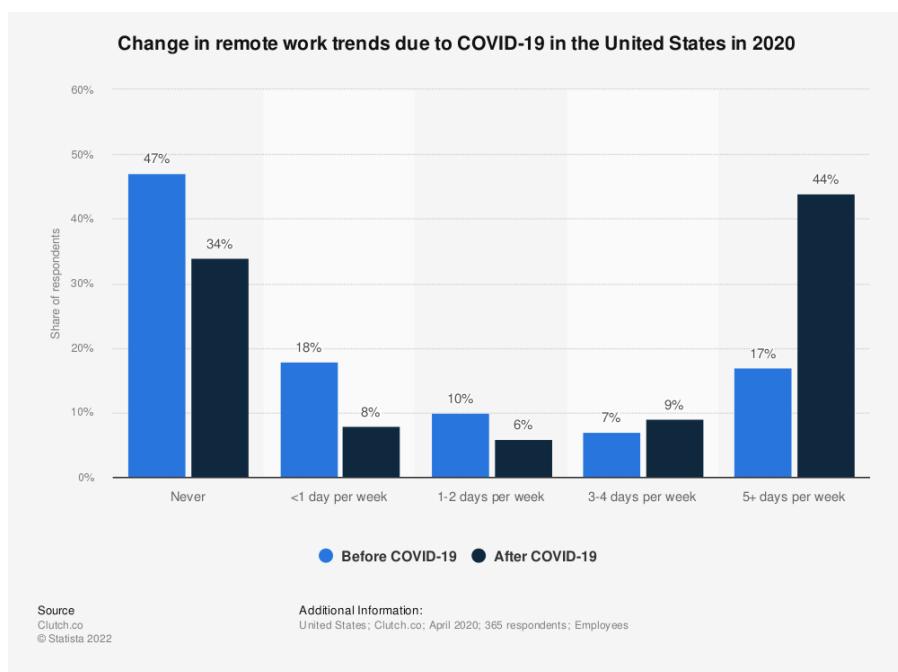
Streszczenie.....	1
Słowa kluczowe	1
Spis Treści.....	2
1.Wprowadzenie	4
1.1.Cel pracy.....	5
2.Analiza istniejących rozwiązań.....	6
2.1.Workday	6
2.2.SaldeoSMART	8
2.3.SYSTE0360	9
2.4.Flowscape	11
3.Opis proponowanego rozwiązania.....	13
3.1.Moduły aplikacji.....	13
3.2.Wymagania funkcjonalne	13
3.2.1.Moduł rezerwacji miejsc	13
3.2.2.Moduł spotkań.....	14
3.2.3.Moduł zarządzania hierarchią i użytkownikami	15
3.3.Wymagania niefunkcjonalne	17
3.3.1.Wymagania wydajności	17
3.3.2.Wymagania bezpieczeństwa	17
3.3.3.Wymagania dostępności	17
3.4.Aktorzy	18
3.5.Przypadki użycia	19
3.6.Diagram klas.....	20
3.7.Wykorzystana technologia	21
3.7.1.Front end	21
3.7.2.Back end.....	22
3.7.3.Baza danych	23
4.Opis implementacji	25
4.1.Wykorzystane narzędzia	25
4.2.Front end	26
4.2.1.Vuex	28
4.2.2.V-calendar.....	29
4.2.3.Bootstrap-vue.....	31
4.2.4.Vue router	32
4.2.5.Agora-rtc-sdk-ng	33
4.3.Back end	35
4.3.1.Spring Boot Data JPA.....	35

4.3.1.Spring Web Starter	38
4.3.2.Spring Starter Security i JJWT	39
5.Testy aplikacji.....	41
5.1.Logowanie.....	41
5.1.Moduł moduł spotkań	43
5.2.Moduł rezerwacji miejsc.....	45
5.3.Moduł zarządzanie hierarchią i użytkownikami	47
6.Przyszłość projektu.....	48
7.Podsumowanie	49
8.Słownik.....	50
9.Tabele, diagramy	51
10.Rysunki, fragmenty kodu	52
11.Bibliografia	53

1. Wprowadzenie

W dzisiejszych czasach rozwiązania informatyczne są nieodłączną częścią każdego przedsiębiorstwa. Digitalizacja pewnych procesów biznesowych, zarządczych nie bez powodu jest tak popularna. Pozwala na zwiększenie oraz ułatwienie dostępu do informacji, a także na przyśpieszenie ich przepływu. Złożenie podania o urlop lub zaświadczenie nie musi już wymagać wysłania wielu maili czy wędrówki do menadżera i działu HR, gdy wystarczy kilka kliknięć myszką. Informacje o pracownikach mogą być przechowywane cyfrowo a nie papierowo, co znacznie ułatwia do nich dostęp oraz służy środowisku. Terminarz sal konferencyjnych czy prywatny kalendarz również mogą istnieć w sieci tak, aby każdy mógł w prosty sposób zaplanować spotkanie z klientem a także swój czas pracy. Ze względu na ciągle otaczającą nas technologie, które ułatwiają nam życie codzienne, oczekujemy takich rozwiązań również w pracy. Tak jak dobra strona internetowa jest wizytówką firmy i kluczem do pozyskania klienta, tak samo dobry system wewnętrzny może być kluczem do zatrzymania pracownika.

Ostatnie doświadczenia związane z funkcjonowaniem przedsiębiorstw w dobie pandemii, pokazały jak ważną rolę odgrywają współczesne technologie informatyczne. Potwierdziło się także, że niezależnie od branży, rodzaju czy wielkości działalności, cyfryzacja jest procesem, którego nie można odkładać na później. Według badania przeprowadzonego w kwietniu 2020 r. ilość osób pracujących zdalnie, 5 dni w tygodniu, wzrosła ponad dwukrotnie po wybuchu pandemii. [1] Co więcej, duża część pracowników deklaruje chęć kontynuacji tego sposobu pracy również po jej zakończeniu.



Zmiana trendu pracy zdalnej spowodowana pandemią COVID-19 w stanach zjednoczonych w 2020 r.
[Bibliografia 1]

Praca na odległość już od dawna była znana jednak nigdy nie była tak powszechna jak teraz. Dotychczas pracodawcy podchodzili do niej dość sceptycznie, lecz okres pandemii pokazał, że pracownik może być równie skuteczny pracując w domu. Jednakże, wiąże się to z pewnymi trudnościami jak np. brak kontaktu „twarzą w twarz”, składanie wniosków na odległość, organizacja pracy w biurze, bo jak wiadomo niektóre czynności nigdy nie zostaną przeniesione do sieci.

Istnieje wiele rozwiązań informatycznych odpowiadających na nowo powstałe potrzeby pracowników i pracodawców. Jednak ze względu na ich poziom skomplikowania oraz zakres usług, które oferują, nie są dostępne dla wszystkich. Wdrożenie takiego systemu wymaga specjalistycznej wiedzy lub dużych nakładów pieniężnych na które małe oraz średnie przedsiębiorstwa nie zawsze są gotowe.

1.1. Cel pracy

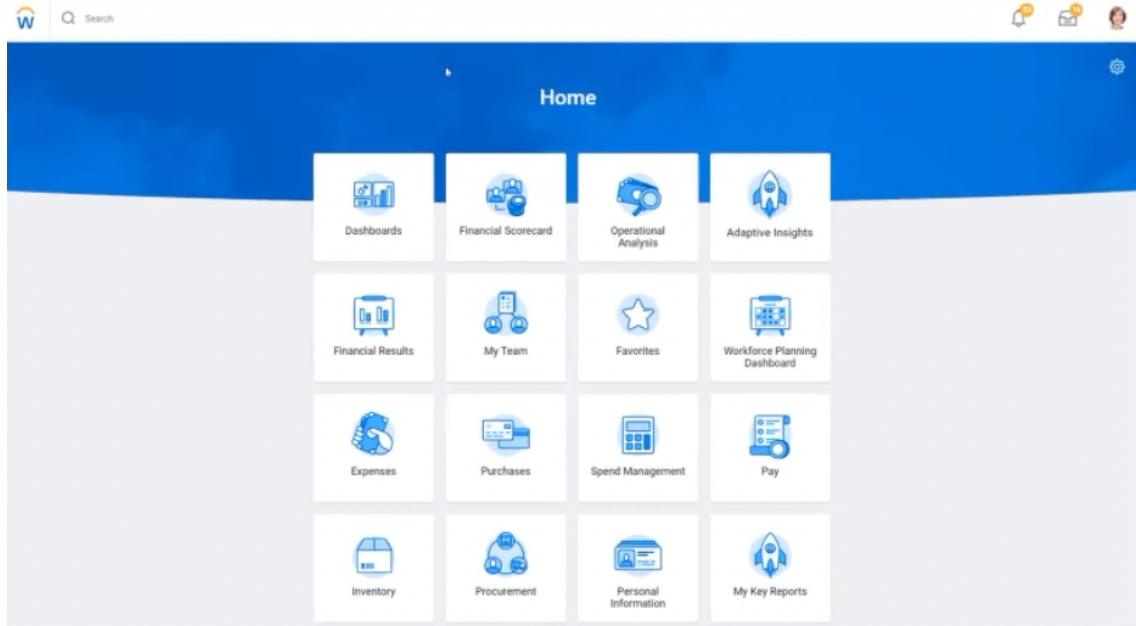
Celem niniejszej pracy dyplomowej jest stworzenie systemu zapewniającego możliwość digitalizacji podstawowych procesów biznesowych oraz zarządczych. Rozwiązanie powinno odpowiadać na większość problemów obecnych w różnych rodzajach działalności. Ze względu na wszechstronność oraz łatwą konfigurację, rozwiązanie będzie umożliwiało cyfryzację małym oraz średnim przedsiębiorstwom.

2. Analiza istniejących rozwiązań

W tej części pracy omówione zostaną już dostępne rozwiązania wspierające digitalizację procesów biznesowych i zarządczych. Ze względu na ciągły postęp technologiczny oraz rosnące zainteresowanie tym tematem jest ich wiele i wciąż pojawiają się nowe. Na potrzeby tego rozdziału zapoznałem się z czterema z nich oraz sprawdziłem, czy są one w stanie rozwiązać postawiony przeze mnie problem.

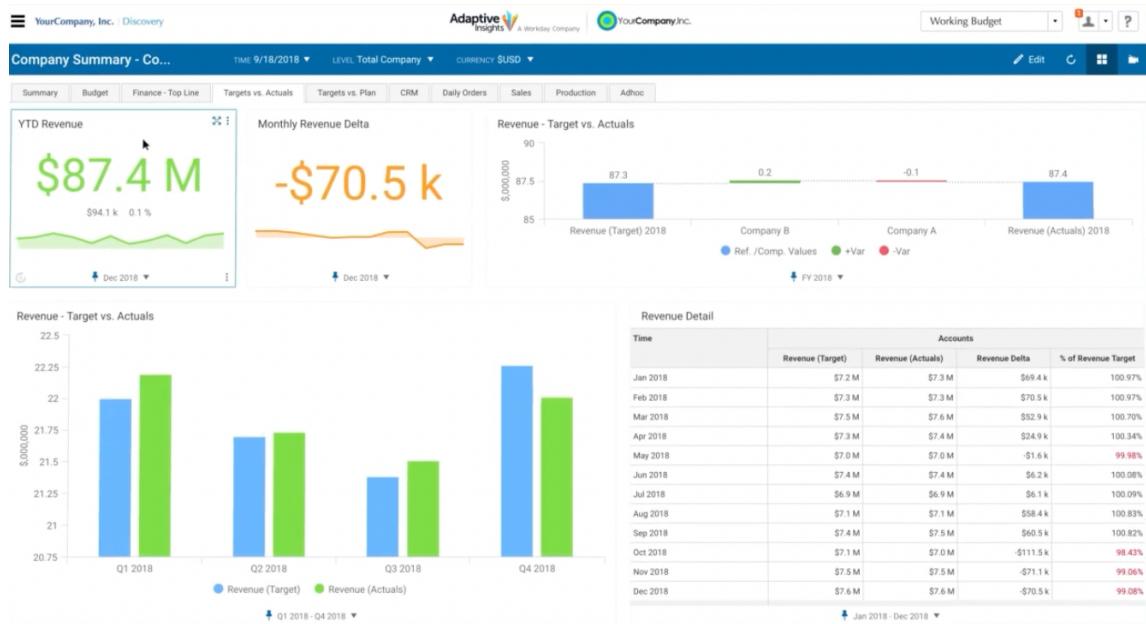
2.1. Workday

Pomysł na system powstał w 2005 roku. Już rok później została udostępniona część odpowiadająca za zarządzanie zasobami ludzkimi. W kolejnych latach aplikacja została rozbudowywana o dodatkowe moduły, takie jak: zarządzanie finansami firmy, kosztami, planem biznesowym, a także planowanie wydatków, tworzenie prognoz oraz wsparcie i rozwój talentów. [2] Dziś jest to czołowe rozwiązanie chmurowe wykorzystywane przez działy HR i Payroll. Natomiast zwykłym użytkownikom umożliwia dostęp do ich prywatnych danych, kalendarza z urlopami. Wspiera także proces ustalania i realizacji celów pracownika w danym okresie.



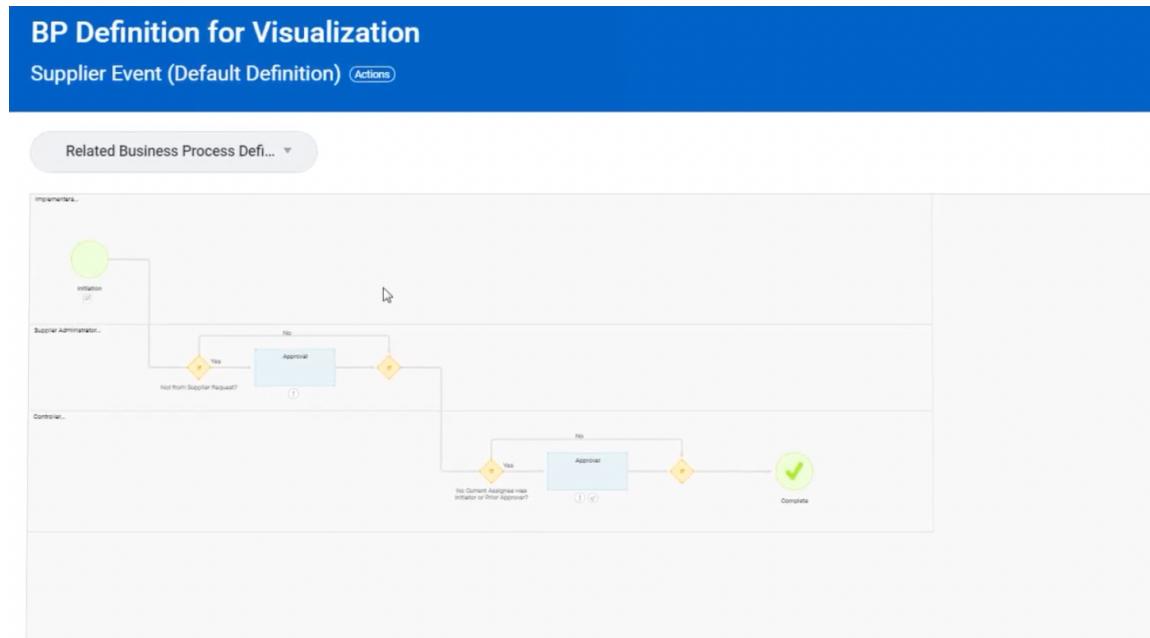
Panel główny systemu Workday.
[Bibliografia 2]

Główna zaletą systemu są wysoce rozbudowane raporty i analizy finansowe pokazujące całkowite przychody, ich zmianę na przestrzeni ostatnich lat oraz kluczowe segmenty. [3]



Przykładowy raport finansów firmy systemu Workday.
[Bibliografia 3]

Aplikacja umożliwia także, modyfikację przepływu procesów poprzez kafelkowy interface, dzięki czemu każdy może dostosować je do własnych potrzeb. [4]



Moduł edycji przepływu procesów biznesowych systemu Workday.
[Bibliografia 4]

System Workday jest bardzo rozbudowany jednak brakuje z nim kilku, kluczowych dla rozwiązania postawionego wcześniej problemu, modułów jak np. zarządzanie przestrzenią biurową, składanie wniosków o zaświadczenie, czy też możliwość rozmowy poprzez wideokonferencje. Co więcej, jak podaje producent, jest to rozwiązanie głównie dla średnich i dużych firm.

2.2. SaldeoSMART

Jak podaje producent, SaldeoSMART sprawdzi się wszędzie tam, gdzie potrzebne jest zapanowanie nad dokumentacją firmową. Rozwiązanie jest dostępne w dwóch rodzajach, dla firm oraz dla biura rachunkowego.

Dla biura rachunkowego [5]:

- Odczytywanie dokumentów - odczytywanie faktur, automatyzacja przelewów, generowanie raportów
- Obieg dokumentów - indywidualny obieg, historia edycji i opiniowanie dokumentów
- Komunikacja z klientem - wystawianie faktur, rozliczenia miesięczne, dokumenty kadrowe
- Zarządzanie biurem - zarządzanie zadaniami, logowanie czasu pracy, generowanie raportów, organizacja procesów
- Panel pracownika - zarządzanie zaliczkami, kontrolowanie wydatków, akceptacja zaliczek

Dla firm [6]:

- Odczytywanie dokumentów - odczytywanie faktur, automatyzacja przelewów, generowanie raportów
- Obieg dokumentów - indywidualny obieg, historia edycji i opiniowanie dokumentów
- Moja firma - wystawianie faktur, odczytywanie faktur, rozliczenia z kontrahentami, dokumenty kadrowe
- Panel pracownika - zarządzanie zaliczkami, kontrolowanie wydatków, akceptacja zaliczek
- Panel kontrahenta - odczytywanie faktur, generowanie przelewów, darmowe konta klientów, generowanie raportów

Jak widać różnica między dostępnymi rodzajami systemu jest nie wielka. Opcja dla firm posiada dodatkowo panel dla kontrahenta, natomiast brakuje w niej zarządzania zadaniami i logowania czasu pracy. Największą zaletą systemu jest podział na pakiety,

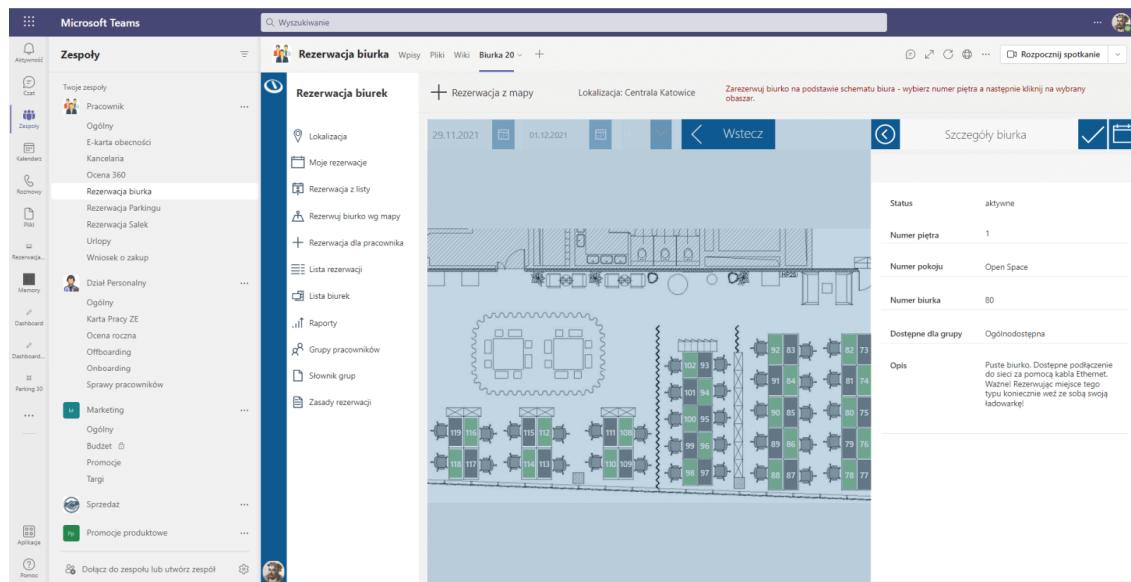
dzięki czemu interesant może wybrać, który z nich jest mu potrzebny. Rozwiążanie umożliwia cyfryzację ale brakuje mu wsparcia dla firm stosujących pracę zdalną np. zarządzania przestrzenią biurową, możliwości składania wniosków o zaświadczenie czy też kalendarza urlopowego.

2.3. SYSTE0360

Kompleksowe rozwiązanie wspierające firmę w digitalizacji procesów. Oferta składa się z czternastu rozszerzeń do aplikacji Microsoft Teams pakietu Office 360. Aplikacje podzielone są na trzy główne obszary:

Dla działu administracji:

- Rezerwacja biurek
- Rezerwacja miejsc parkingowych
- Rezerwacja salek konferencyjnych
- Wniosek o zakup
- Obieg dokumentów



Przykładowy wygląd mapy biura aplikacji SYSTE0360 - Rezerwacja biurek.
[Bibliografia 7]

Rezerwacja biurek, miejsc parkingowych i salek konferencyjnych odbywa się poprzez wybranie interesującego nas miejsca z listy dostępnych, a następnie dokonanie rezerwacji. Moduły dotyczące biurek oraz miejsc parkingowych są dodatkowo wyposażone w opcje „Rezerwacja z mapy”. [7] Aplikacja wyświetla mapę, na której

oznaczone są wszystkie miejsca oraz ich status, dostępność. Ułatwia to lokalizację miejsca oraz sprawia, że cała aplikacja staje się bardziej przyjazna użytkownikowi.

Dla działu personalnego:

- Elektroniczny wniosek urlopowy
- eKarta obecności
- Onboarding
- Offboarding
- Ocena 360 stopni
- Ocena okresowa pracownika

Pozostałe obszary

- CRM dla Teams
- Audyty wewnętrzne ISO
- Inwentaryzacja

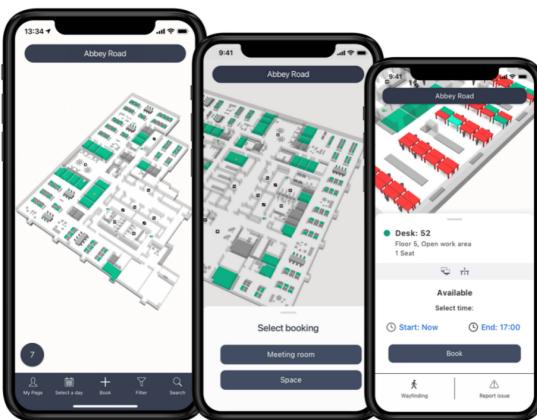
Aplikacja Microsoft Teams, na której bazuje system, jest bardzo rozbudowanym komunikatorem. Pozwala na tworzenie czatów zespołowych i prywatnych. Daje możliwość załączania dokumentów, wideokonferencji i wiele więcej. Wspiera także inne aplikacje pakietu Office 360 np. wspólną edycję dokumentów WORD i prezentacji PowerPoint w czasie rzeczywistym.

Połączenie Teams oraz SYSTE0360, wydaje się być idealnym rozwiązaniem dla firm chcących przenieść swoje procesy do cyfrowego świata oraz tych wdrażających możliwość pracy zdalnej. Jednakże jest ono w pełni zależne od innej aplikacji, co nie koniecznie będzie odpowiadało każdemu.

2.4. Flowscape

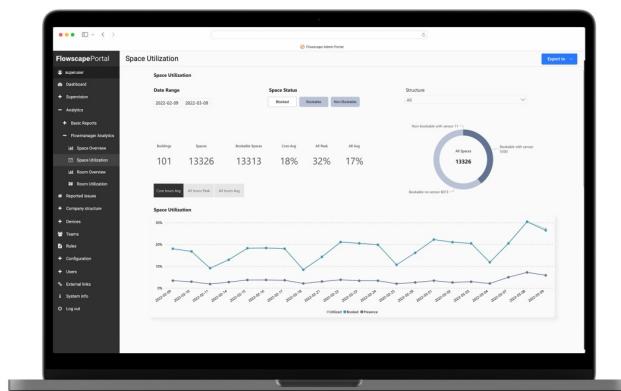
Rozwiązanie jest dostępne na rynku od 2011r. Wspiera przedsiębiorstwa oferujące hybrydowy tryb pracy. Oferta składa się z sześciu pakietów, jakimi są:

- Desk Management - opisany jako najbardziej pasujący dla firm składających się z 100 lub więcej pracowników, które podchodzą elastycznie do stanowisk pracy. W skład wchodzi: aplikacja mobilna, aplikacja webowa, zaawansowane zestawienia, mapa oraz wspomaganie szukania drogi.



Aplikacja mobilna aplikacji Desk Management.
[Bibliografia 8]

- Room Booking solution - jest podzielony na trzy typy. BASIC i PRO przeznaczone są dla firm składających się z 50 lub więcej pracowników. Dodatkowo zawierają wsparcie aplikacji Outlook Microsoft Office oraz podstawowe zestawienia. Pakiet PRO jest także wyposażony w aplikację mobilną oraz webową, mapę i wspomaganie szukania drogi. Ostatnia wersja, ENTERPRISE, rozszerza inne opcje o czujniki w pomieszczeniach i zaawansowane zestawienia. [9] Jednakże, jak podaje producent jest to pakiet dla firm zatrudniających 200 lub więcej pracowników.



Zaawansowany raport pakietu ENTERPRISE.
[Bibliografia 9]

- Room Panel solution - rozwiązanie rekomendowane jest dla firm zaczynających się już od 5 pracowników. Ma za zadanie ułatwić rezerwację pomieszczeń poprzez instalację tabletów przy wejściu. Na tablecie znajduje się harmonogram rezerwacji sali, dzięki czemu użytkownik jest w stanie szybko sprawdzić czy to pomieszczenie jest dostępne w czasie, który go interesuje.
- Workplace Analytics solution - pakiet składa się z czujników ruchu dla pomieszczeń oraz biurek. Zapewnia to najbardziej aktualne oraz pewne informacje na temat tego, które są wolne, a które zajęte. Jest to rozwiązanie przede wszystkim istotne dla małych pomieszczeń lub postawionych w przestrzeni wspólnej budek, których nie da się zarezerwować, gdyż służą do zapewnienia chwilowej prywatności np. gdy nagle zadzwoni klient.
- Building Messaging - jest to platforma pozwalająca na szybkie powiadomienia. Użytkownicy mogą otrzymywać ważne powiadomienia dotyczące np. zamknięcia biura ze względu na awarię prądu. W skład pakietu wchodzą aplikacja mobilna oraz webowa. Jak podaje producent jest to rozwiązanie przede wszystkim dla firm zatrudniających ponad 100 pracowników.
- Digital Workplace - najlepiej sprawdzi się w przedsiębiorstwach składających się z ponad 500 pracowników. Pakiet ten ma zapewnić kompleksową wirtualną przestrzeń biurową. W jego skład wchodzi większość z dotychczasowych pakietów. Jest to tak naprawdę bardzo elastyczne podejście, które pracodawca może dostosować wraz z ekspertem, tak aby spełniały jego potrzeby.

3. Opis proponowanego rozwiązania

Proponowane rozwiązanie ma formę strony internetowej, do której dostęp mają tylko pracownicy danego przedsiębiorstwa. Aplikacja została nazwana OMS (ang. Office Management System). Rozdział ten ma formę opisu faz analizy oraz projektowania.

W kolejnych podrozdziałach zostaną opisane poszczególne wymagania funkcjonalne, niefunkcjonalne oraz użytkownicy systemu. Dodatkowo, zostaną one poparte diagramami przypadków użycia i klas. Wizualizacje stworzono przy użyciu aplikacji Lucidchart.

3.1. Moduły aplikacji

Aplikacja została podzielona na kilka odrębnych modułów, z których każdy służy innemu celowi. Dostęp do modułów jest ograniczony przez grupy, które można przypisywać użytkownikom. Przewidziane są trzy grupy: pracownik, pracownik działu HR oraz administrator systemu.

- Moduł rezerwacji miejsc - dla wszystkich użytkowników aplikacji.
- Moduł spotkań - dla wszystkich użytkowników.
- Moduł zarządzania hierarchią i użytkownikami - dostępny tylko dla pracowników działu HR.

3.2. Wymagania funkcjonalne

3.2.1. Moduł rezerwacji miejsc

W systemie rozróżniane są trzy typy miejsc: biurko, miejsce parkingowe oraz pomieszczenie. Każde z nich posiada piętro, nazwę oraz opcjonalny opis. Miejsce parkingowe posiada dodatkowo rozmiar jeden z trzech: małe, zwykłe oraz duże. Dodatkowe informacje pomieszczenia to maksymalna ilość osób obecnych w jednym momencie oraz powierzchnia podana w metrach kwadratowych. [Tabela 1] W module rezerwacji miejsca są przedstawione w formie listy, którą użytkownik może filtrować. Miejsca zablokowane są oznaczone na kolor czerwony oraz są niemożliwe do rezerwacji. [Tabela 2]

Tabela 1. Informacje przechowywane na potrzeby moduły rezerwacji miejsc

Lp.	Nazwa	Informacje
1	Biurko	Piętro Nazwa Opcjonalny opis
2	Miejsce parkingowe	Piętro Nazwa Opcjonalny opis Rozmiar
3	Pomieszczenie	Piętro Nazwa Opcjonalny opis Powierzchnia kwadratowa

Źródło: opracowanie własne

Tabela 2. Wymagania funkcjonalne modułu rezerwacji miejsc

Lp.	Nazwa	Opis	Aktor
1	Zarezerwuj miejsce	Użytkownik może przejrzeć listę miejsc, wybrać datę i utworzyć rezerwację.	Pracownik Pracownik HR
2	Zwolnij miejsce	Użytkownik może przejrzeć listę swoich rezerwacji oraz zwolnić wybrane miejsce.	Pracownik Pracownik HR
3	Zablokuj miejsce	Użytkownik może przejrzeć listę wszystkich miejsc, następnie je zablokować.	Pracownik HR
4	Zarządzaj miejscami	Użytkownik może dodać, edytować oraz usunąć wybrane miejsce.	Pracownik HR

Źródło: opracowanie własne

3.2.2. Moduł spotkań

Przedstawiony jest w formie kalendarza na którym widnieją wszystkie wydarzenia zalogowanego użytkownika. [Tabela 3] Z jego poziomu można tworzyć spotkania, nadawać im czas oraz oznaczać jednym z dostępnych kolorów. Spotkanie może być prywatne lub można zaprosić także innych użytkowników systemu. [Tabela 4] Wydarzenie może mieć przypisaną wideokonferencję do której

dostęp mają tylko zaproszeni użytkownicy. Videokonferencja umożliwia spotkanie twarzą w twarz przy pomocy przesyłu strumienia wideo oraz dźwięku.

Tabela 3. Informacje przechowywane na potrzeby modułu spotkań

Lp.	Nazwa	Informacje
1	Wydarzenie	Tytuł Data Czas trwania w minutach Kolor oznaczenia Identyfikator videokonferencji Lista uczestników

Źródło: opracowanie własne

Tabela 4. Wymagania funkcjonalne modułu spotkań

Lp.	Nazwa	Opis	Aktor
1	Zarządzaj wydarzeniem	Użytkownik może stworzyć, edytować lub usunąć wydarzenie.	Pracownik Pracownik HR
3	Zarządzaj videokonferencją wydarzenia	Użytkownik może dodać, edytować lub usunąć videokonferencję do już stworzonego wydarzenia lub na etapie jego tworzenia.	Pracownik Pracownik HR

Źródło: opracowanie własne

3.2.3. Moduł zarządzania hierarchią i użytkownikami

Stworzony z myślą o zarządzaniu hierarchią panującą w firmie. Widok modułu przedstawia listę wszystkich użytkowników systemu oraz ich podwładnych. Użytkownik może wyszukać pracownika, którego przynależność do hierarchii chce zmienić, następnie wyświetli się lista pracowników, którzy znajdują się na niższym szczeblu oraz ich informacje. [Tabela 5] Do każdego rekordu przypisany jest przycisk umożliwiający edycję hierarchii pracownika. Po wybraniu zostaną wyświetcone aktualne informacje oraz pole do wyszukiwania nowego menadżera. Dodatkowo znajdują się tu opcje takie jak, edycja danych osobowych, usunięcie konta. [Tabela 6]

Tabela 5. Informacje przechowywane na potrzeby zarządzania użytkownikami

Lp.	Nazwa	Informacje
1	Użytkownik	Imię Nazwisko Email Hasło Stanowisko Menadżer Data urodzenia Rola użytkownika

Źródło: opracowanie własne

Tabela 6. Wymagania funkcjonalne modułu zarządzania hierarchią i użytkownikami

Lp.	Nazwa	Opis	Aktor
1	Wyświetl hierarchię	Użytkownik może wyszukać pracownika i wyświetlić listę osób do niego przypisanych.	Pracownik HR
2	Edytuj hierarchię	Po wyświetleniu listy pracowników i wybraniu rekordu, użytkownik może edytować pole zawierające menadżera.	Pracownik HR
3	Wyświetl informację o pracowniku	Po wyświetleniu listy pracowników i wybraniu rekordu, użytkownik może wyświetlić szczegóły wybranego pracownika.	Pracownik HR
4	Edytuj informację o pracowniku	Po wyświetleniu szczegółów wybranego pracownika, użytkownik może je edytować.	Pracownik HR

Źródło: opracowanie własne

3.3. Wymagania niefunkcjonalne

3.3.1. Wymagania wydajności

- Czas oczekiwania na odpowiedź aplikacji nie powinien przekraczać 2 sekund.
- Aplikacja powinna obsługiwać 200 użytkowników jednocześnie.
- Aplikacja powinna gwarantować brak redundancji w bazie danych, co pozytywnie wpłynie na jej wielkość.

3.3.2. Wymagania bezpieczeństwa

- Aplikacja powinna być dostępna tylko dla pracowników firmy.
- Użytkownik posiadający role pracownika nie powinien mieć dostępu do modułów innych ról i ich funkcjonalności.
- Aplikacja powinna zagwarantować trwałość oraz poprawność danych, oznacza to, że jeśli wystąpi nieoczekiwany błąd dane zawsze zostaną poprawne oraz poprawnie zapisane w bazie danych.
- Użytkownik niezalogowany nie powinien mieć dostępu do żadnych danych.
- Użytkownik powinien być w stanie przerwać edycje obiektów w dowolnym momencie bez naruszania danych.
- Hasła użytkowników powinny być od razu szyfrowane i przechowywane tylko i wyłącznie jako zaszyfrowane.

3.3.3. Wymagania dostępności

- Aplikacja powinna być dostępna całą dobę.
- Aktualizacja i konserwacja nie powinny trwać więcej niż 2 godziny i odbywać się tylko w godzinach
- Aplikacja powinna być schludna i przejrzysta, zbudowana ze stonowanych i znaczeniowych kolorów.

3.4. Aktorzy

System przewiduje obsługę dwóch aktorów, których różni rola oraz cel wykorzystania aplikacji. Sposób dziedziczenia przedstawiony jest na rysunku 1.

- Pracownik - wykorzystuje system do wsparcia codziennych zadań
- Pracownik HR - rozszerza cel pracownika o zarządzanie użytkownikami oraz hierarchią w firmie

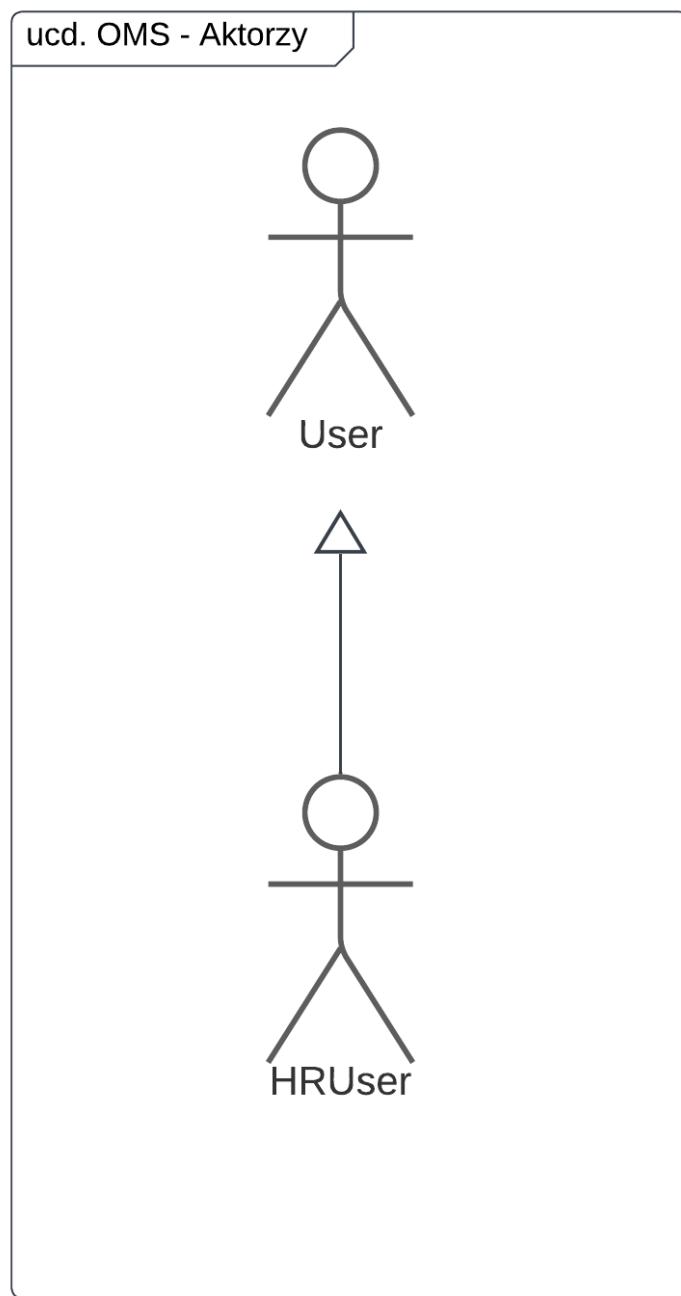


Diagram aktorów. Opracowanie własne.
[Tabele, diagramy 7]

3.5. Przypadki użycia

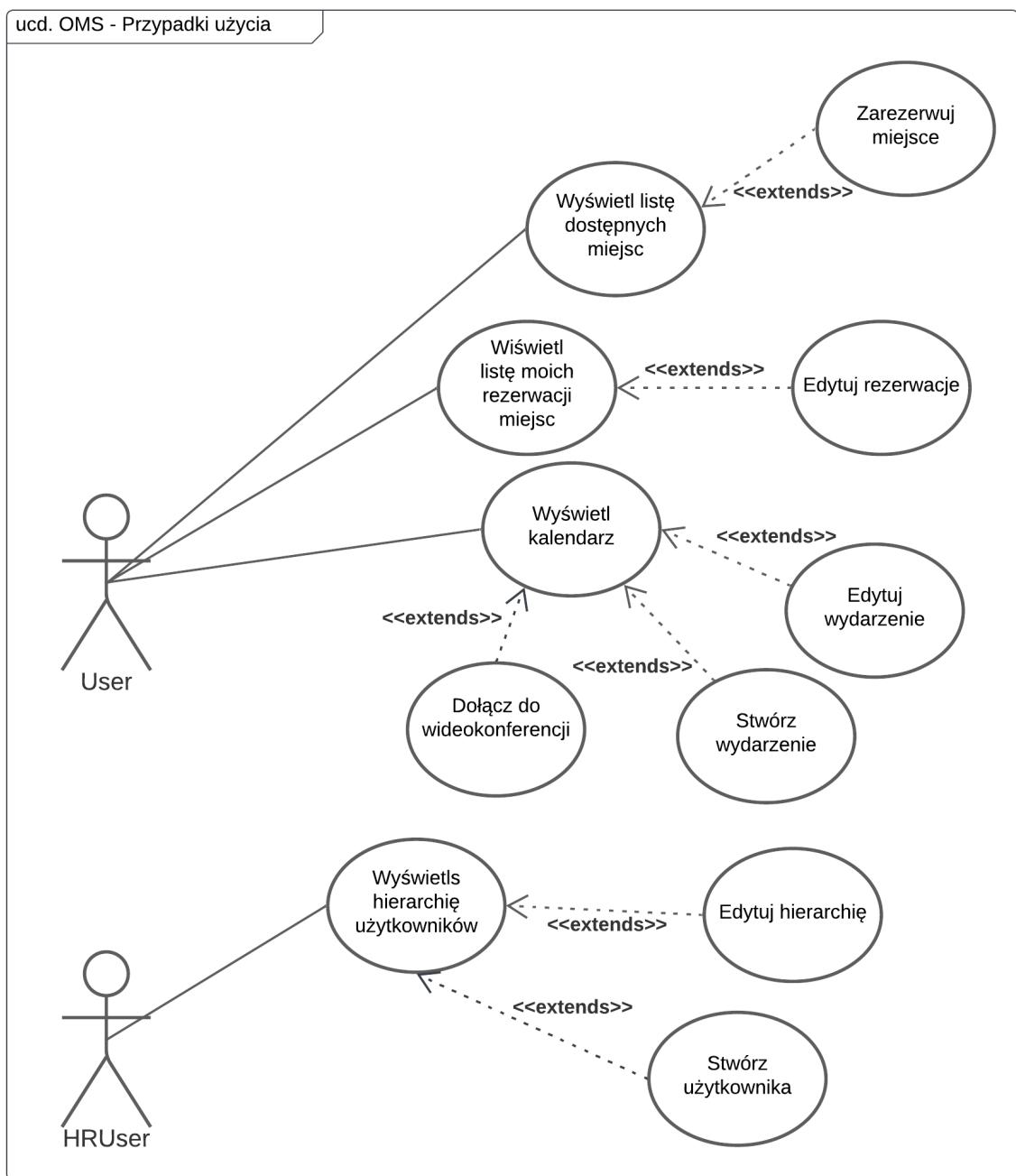


Diagram przypadków użycia. Opracowanie własne.
[Tabele, diagramy 8]

3.6. Diagram klas

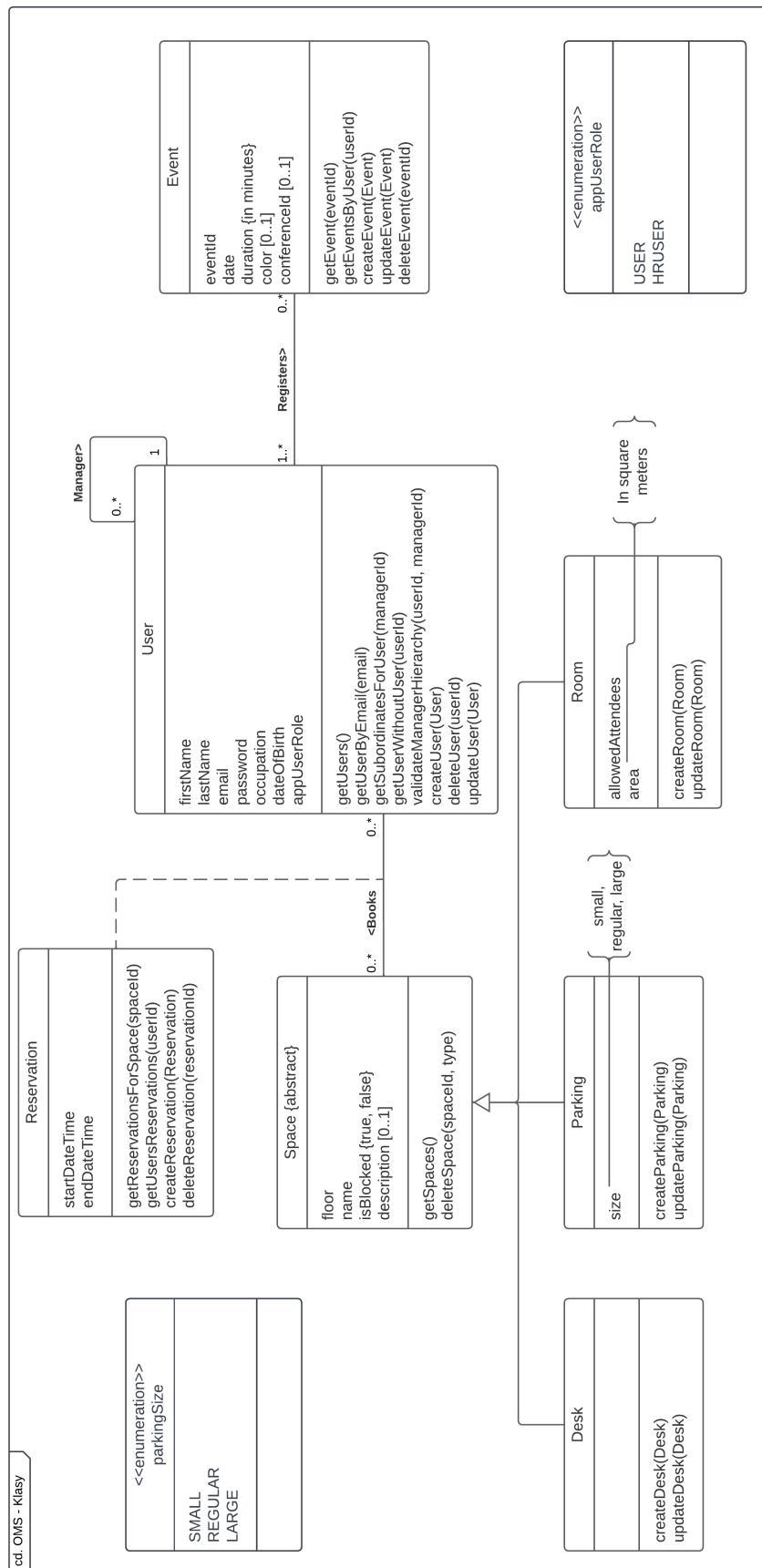


Diagram klas. Opracowanie własne.
[Tabele, diagramy 9]

3.7. Wykorzystana technologia

Nawiązując do wstępu tego rozdziału, rozwiązanie ma formę aplikacji webowej. Aplikacja została podzielona na trzy odrębne części z których każda ma inne zadanie. Na rynku dostępnych jest wiele narzędzi umożliwiających stworzenie takiej strony internetowej. W tym podrozdziale, zostaną omówione te z nich, które zostały użyte do implementacji.

3.7.1. Front end

„Jest to wszystko, co przeglądarka może odczytać, wyświetlić i uruchomić, czyli języki programowania takie jak: HTML, CSS i JavaScript.”¹. Zatem jest to także pierwsza linia łącząca użytkownika z aplikacją. Front end to nie tylko wygląd aplikacji ale także wszystkie przejścia, animacje, treści. Tworząc go, trzeba mieć na uwadze przede wszystkim dobro użytkownika. Każda z funkcji powinna być przezroczysta, dobrze opisana oraz responsywna, oznacza to, że czas odpowiedzi na każdą z czynności użytkownika musi być krótki, prawie nieauważalny.

HTML (ang. Hypertext Markup Language), jest wykorzystywany do tworzenia stron internetowych. Jednakże, nie jest to język programowania. Kod HTML wykorzystuje znaczniki (zwane tagami) do opisu elementów znajdujących się na stronie. Dzięki umieszczeniu treści między odpowiednimi tagami można przedstawić daną treść jako, między innymi, listę, nagłówek, link, obrazek a nawet wideo. Każdy z tagów, prócz nazwy, ma także opcjonalne atrybuty. Pozwalają one ustalić na przykład, źródło obrazu, link do przekierowania, jego klasę czy też unikatowy identyfikator. Podstawowy wygląd tagów jest bardzo prosty ale można go zmieniać przy pomocy atrybutu „style”. Nie jest to jednak optymalne rozwiązanie, ponieważ wymagałoby to wielokrotnego powtórzenia tego samego kodu. Z pomocą przychodzi CSS (ang. Cascading Style Sheet). Podobnie jak HTML, nie jest to język programowania. Nie jest także językiem znaczników. Jest to język arkuszy stylów. Pozwala selektywnie nadawać style elementom kodu HTML. W języku CSS, style przedstawione są w formie bloków składających się z nazwy tagu, klasy lub identyfikatora na podstawie którego styl przypisywany jest odpowiednim tagom. Kolejnym elementem bloku jest lista właściwości i przypisanych im wartości. Gotowy arkusz jest następnie przypisywany do dokumentu HTML.

¹ <https://kompan.pl/co-to-jest/front-end/>

Do stworzenia front end'u aplikacji wykorzystany został framework o nazwie Vue.js wersja 3. Został zbudowany na bazie języków HTML i CSS or przy użyciu JavaScript. Framework ten ma ogólną rzeszę zwolenników dzięki szczegółowej dokumentacji, jedno-plikowym komponentom oraz dlatego, że łączy najlepsze cechy dwóch konkurentów na rynku, którymi są React i Angular. Tak jak React składa się z komponentów i podobnie jak Angular posiada przystępne szablony. Jednakże, Vue jest dość nowym framework'iem, dlatego wymaga jeszcze wielu poprawek. Jeden plikowy komponent Vue oznacza zawarcie elementów HTML, CSS i JavaScript w jednym pliku co znacznie ułatwia proces tworzenia.

3.7.2. Back end

Back end to serwer a zarazem mózg całej aplikacji. Mimo, że jest niewidoczny dla użytkownika towarzyszy mu od samego zalogowania, przez każdą z funkcjonalności aż do momentu wylogowania. Jego podstawowymi funkcjami są, identyfikacja oraz autoryzacja, czyli udzielenie odpowiedzi na pytanie kto chce uzyskać dostęp do informacji i czy ma prawo je pozyskać. Gdy użytkownik zostanie już sprawdzony, serwer przetwarza zapytanie odebrane od front end'u aplikacji oraz zwraca odpowiedź, która jest następnie wyświetlana użytkownikowi.

Serwer aplikacji został stworzony przy użyciu języka Java wersja 11. Java to obiektowy język programowania, którego główną zaletą jest niezależność od architektury systemu, oznacza to, że kod napisany na jednym systemie operacyjnym może zostać uruchomiony na każdym innym. Jest to możliwe, ponieważ kod Javy komplikowany jest do kodu pośredniego, który potem jest uruchamiany przy pomocy wirtualnej maszyny Javy.

Jako wsparcie dla czystego języka programowania jakim jest Java, użyty został framework Spring Boot wersja 2.6.3. Jest to rozszerzenie framework'a Spring, który zapewnia takie technologie jak wstrzykiwanie zależności, walidacja i wspomaganie zarządzania danymi i wiele innych. Spring Boot zaś rozszerza podstawową wersję o automatyczną konfigurację, wbudowany serwer Tomcat czy też startowe zależności. Spring Boot został rozszerzony do dodatkowych zależności:

- **Spring Boot Data JPA** - są to zapytania wbudowane. Ułatwiają kontakt back end - baza danych poprzez wcześniej zdefiniowane zapytania. Programista nie musi pisać prostych zapytań SQL'owych aby otrzymać oczekiwany wynik z bazy danych, wystarczy znajomość słów kluczowych dostarczonych przez zależność. JPA samo zadba o poprawne przetłumaczenie nazwy metody na zapytanie. Tak jak na przedstawionym poniżej przykładzie.

- **Spring Web Starter** - zależność umożliwiająca stworzenie REST API. Zapewnia adnotacje takie jak `@RestController`, `@RequestMapping`, `@GetMapping` i wiele innych. Adnotacje te używane są do oznaczania końcówek po których następuje komunikacja z serwerem. `@RequestMapping` oznacza końówkę używaną do identyfikacji konkretnego kontrolera. Natomiast `@GetMapping` to identyfikator końówki odpowiadającej na zapytanie GET.
- **JWT** - Java JSON Web Token, jest to zależność pozwalająca tworzyć oraz weryfikować tokeny (JWT). Gdy użytkownik zaloguje się do aplikacji, jego email oraz hasło są weryfikowane, następnie na podstawie znalezionej rekordu tworzony jest token oraz przesyłany z powrotem do front end'u aplikacji. Tam jest przechowywany a następnie wykorzystywany jako potwierdzenie weryfikacji użytkownika przy kolejnych zapytaniach do serwera. Token JWT zawiera takie informacje jak: algorytm szyfrowania, email użytkownika, rola w aplikacji (wykorzystywana do autoryzacji) oraz data ważności. Poniżej przedstawiona jest konfiguracja oraz tworzenie tokenu.
- **Spring Starter Security** - aby zadbać o bezpieczeństwo aplikacji dodana została zależność Starter Security. Zawiera ona podstawowy plik konfiguracyjny dlatego od razu po załadowaniu nasza aplikacja posiada pierwsze zabezpieczenia jak na przykład strona logowania oraz domyślny użytkownik z hasłem generowanym co uruchomienie aplikacji. Jednakże, są to podstawowe zabezpieczenia. Przy większych projektach wymagana jest dodatkowa konfiguracja.

3.7.3. Baza danych

Trwałość danych oznacza, że wszystkie informacje jakie aplikacja akceptuje, przetwarza i dostarcza powinny być zapisane w jednym ustalonym miejscu, z którego w krótkim czasie mogą zostać odczytane. Baza danych jest to miejsce przechowujące dane w sposób określony i prosty do odczytania. W tym przypadku wykorzystany został relacyjny model bazy danych. Oznacza to, że dane przechowywane są w formie tabeli grupującej takie same obiekty. Każdy wiersz tabeli przedstawia jeden obiekt posiadający unikatowy klucz główny. Natomiast, każda kolumna to atrybut obiektu. Atrybuty powinny być unikatowe oraz współdzielone między rekordami tabeli aby nie przechowywać pustych wartości. Zależności między tabelami są wyrażone przy pomocy klucza obcego jako atrybutu tabeli. Dostępność do danych następuje przy pomocy języka zapytań SQL. Zapytania te dzielą się na:

- DDL (ang. Data Definition Language) - umożliwiające tworzenie, edycje oraz usuwanie struktury bazy danych
- DML (ang. Data Manipulation Language) - służące do manipulowania danymi w obrębie bazy danych. Dodawanie rekordów do tabel, usuwanie ich oraz edycja poszczególnych atrybutów lub cały wierszy.
- DQL (ang. Data Query Language) - najczęściej używane zapytania. Pozwalają na wyciągnięcie potrzebnych danych w sposób jaki oczekuje tego użytkownik.
- DCL (ang. Data Control Language) - rzadko kiedy używane. Głównie w momencie tworzenia bazy danych. Administrator używa ich do nadawania uprawnień użytkownikom bazy danych. Ogranicza w ten sposób dostęp do danych wrażliwych czy też poufnych.

Na potrzeby trwałości danych w aplikacji, wykorzystana została relacyjna baza danych MySQL. Jest to open source'owy system, aktualnie rozwijany przez firmę Oracle.

4. Opis implementacji

Kolejnym etapem procesu tworzenia aplikacji jest implementacja. Polega ona na przekształceniu koncepcji powstałej w trakcie fazy analizy i projektowania w rzeczywisty produkt. Wykorzystane zostaną do tego opisane wcześniej technologie, zaprojektowane tabele i diagramy. W tym rozdziale opisane zostaną kroki podjęte aby stworzyć pierwszą wersję gotowego produktu. Niektóre z nich zostały poparte wycinkami kodu stworzonymi przy pomocy darmowej wersji programu Snappify.

4.1. Wykorzystane narzędzia

W trakcie implementacji wykorzystane zostały specjalistyczne narzędzia wspomagające opisane w poprzednim rozdziale technologie. Nie są one konieczne jednak znacznie usprawniają proces tworzenia.

- **IntelliJ IDEA** - IDE wspierające języki programowania Java oraz Kotlin. Dostarcza takie funkcjonalności jak automatyczne kończenie składni, generowanie kodu, wykrywanie błędów pisowni czy podpowiadanie pisowni. Dodatkowymi atutami są wsparcie dla systemu kontroli wersji git oraz możliwość pracy z Docker.
- **Visual Studio Code** - uniwersalne IDE firmy Microsoft. Wsparcie dla języków programowania dodawane jest przy pomocy rozszerzeń. VSC zawiera dodatkowo wbudowany terminal. Z jego poziomu można w prosty sposób wykonywać polecenia na przykład startujące serwer JavaScript.
- **MySQL Workbench** - środowisko umożliwiające tworzenie, edycję oraz zarządzanie bazą danych MySQL. Niezbędne do pracy z bazą danych. Narzędzie posiada stosunkowo niewielką ilość funkcji co sprawia, że jest bardzo proste w użyciu a zarazem w pełni wystarczające.
- **Postman** - narzędzie służące do testowania REST API. Pozwala dowolnie modyfikować oraz wysyłać zapytania http. Po otrzymaniu odpowiedzi, wyświetla ją oraz wszystkie jej szczegóły. Posiada bardzo intuicyjny interfejs umożliwiający otwarcie wielu zakładek z różnymi zapytaniami. Znajduje się tu także historia wysłanych zapytań.

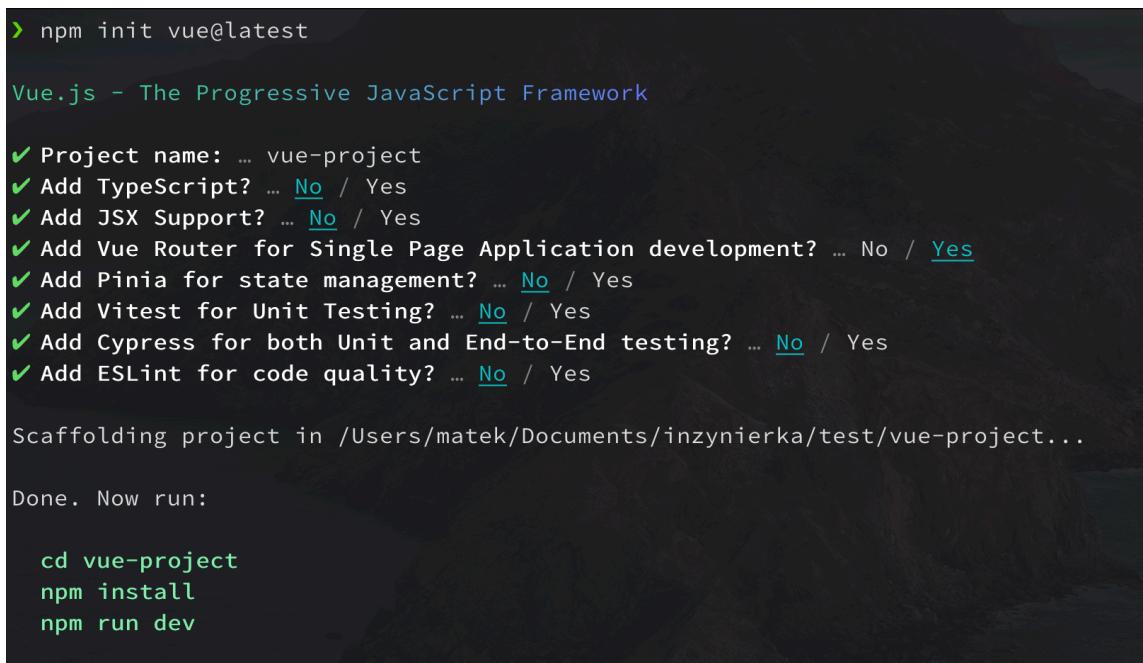
4.2. Front end

Pierwszym krokiem, po otwarciu folderu projektu w VSC, jest inicjalizacja projektu. Pozwala na to komenda:

```
npm init vue@latest
```

NPM (Node Package Manager) to manager pakietów dla aplikacji wykorzystującej Node.js. Pozwala swobodnie dodawać oraz usuwać pakiety, a także uruchamiać serwer i budować wersje gotową do wgrania na produkcję. W tym przypadku „init” służy do inicjalizacji nowego lub istniejącego projektu.

Wywołanie wyżej wskazanej komendy spowoduje rozpoczęcie procesu tworzenia projektu Vue. Pierwszym krokiem jest wybór nazwy projektu. Następnie initializer pyta o dodanie kilku sugerowanych pakietów. Poniższa ilustracja przedstawia widok terminala podczas inicjalizacji projektu.



```
> npm init vue@latest

Vue.js - The Progressive JavaScript Framework

✓ Project name: ... vue-project
✓ Add TypeScript? ... No / Yes
✓ Add JSX Support? ... No / Yes
✓ Add Vue Router for Single Page Application development? ... No / Yes
✓ Add Pinia for state management? ... No / Yes
✓ Add Vitest for Unit Testing? ... No / Yes
✓ Add Cypress for both Unit and End-to-End testing? ... No / Yes
✓ Add ESLint for code quality? ... No / Yes

Scaffolding project in /Users/matek/Documents/inzynierka/test/vue-project...

Done. Now run:

  cd vue-project
  npm install
  npm run dev
```

Widok terminala w trakcie inicjalizacji projektu. Opracowanie własne.
[Rysunki, diagramy, fragmenty kodu 1]

Na potrzeby aplikacji OMS dodano kilka pakietów, tylko jeden z nich znajduje się na liście sugerowanych.

- vuex - pozwala zarządzać stanem aplikacji pomiędzy różnymi komponentami
- uuid - używany do generowania unikatowych identyfikatorów alfanumerycznych
- v-calendar - dostarcza wcześniej zdefiniowane komponenty pozwalające zarządzać datami

- bootstrap-vue - pakiet dostarcza gotowe wyglądy dla tagów html. Nadanie wyglądu tagowi następuje poprzez atrybut „class”.
- vue-router - odpowiada za routing w obrębie aplikacji. Umożliwia przepisanie komponentów do konkretnych końcówek URL i przekierowania.
- vue-multiselect - dostarcza wcześniej zdefiniowane komponenty typu multiselect dropdown
- agora-rtc-sdk-ng - api serwera sygnalizującego. Pozwala na wymianę wiadomości między przeglądarkami. Informacje zawarte w wiadomości umożliwiają im zawarcie bezpośredniego kontaktu.

Kolejnym krokiem jest zainstalowanie pakietów (komenda „npm install”) i uruchomienie serwera (komenda „npm run dev”). W przypadku polecenia „npm run” słowo dev oznacza skrypt zdefiniowany w pliku konfiguracyjnym pakietów. Wszystkie komendy npm muszą być wywoływane z terminala, przy czym instancja ta musi znajdować się w katalogu głównym projektu. Po uruchomieniu serwera, zaczyna on nasłuchiwać na lokalnej maszynie na porcie 3000. Od tego momentu wystarczy w przeglądarce wprowadzić "<http://localhost:3000/>" aby zobaczyć aktualny stan projektu. Dodatkowym atutem jest dynamiczne zaciąganie zmian przez serwer. Oznacza to, że po wprowadzeniu zmian do pliku nie ma potrzeby restartowania całego serwera, wystarczy zapisanie zmian a serwer sam się odświeży.

W kolejnych podrozdziałach przedstawiono wykorzystanie najciekawszych bibliotek oraz rozwiążanie niektórych z napotkanych problemów.

4.2.1. Vuex

```
JS vuex.js

1 import Vuex from 'vuex';
2
3 const state = {
4     user: null
5 };
6
7 const store = new Vuex.Store({
8     state,
9     getters: {
10         user: (state) => {
11             return state.user;
12         }
13     },
14     actions: {
15         user(context, user) {
16             context.commit('user', user);
17         }
18     },
19     mutations: {
20         user(state, user) {
21             state.user = user;
22         }
23     }
24 });
25
26 export default store;
```

snappify.io

Użycie pakietu vuex. Opracowanie własne.
[Rysunki, diagramy, fragmenty kodu 2]

Vuex został wykorzystany do przechowywania informacji o aktualnie zalogowanym użytkowniku. Pozwala to na wyświetlenie odpowiednich funkcji dla odpowiednich ról. Stan (ang. state) służy do przechowywania obiektu jakim jest użytkownik (ang. user) prezentują to linijki od 3 do 5 na przedstawionej wyżej ilustracji. Następnie utworzony został magazyn (ang. store), służący do manipulowania stanem. Zastosowanie tej techniki pozwala na dostęp oraz modyfikację stanu z różnych komponentów projektu Vue.

4.2.2. V-calendar

Podstawowe komponenty dostarczane przez v-calendar to wcześniej stworzone komponenty Vue, które można dostosowywać do własnych potrzeb. Na rysunku numer 6, który znajduje się poniżej, przedstawiono wykorzystanie komponentu <v-calendar>. Pakiet oferuje dużą swobodę w personalizacji widoku i zachowania kalendarza. Wykorzystywane są do tego atrybuty tagu HTML jak na przykład:

- min-date - minimalna data
- max-date - maksymalna data
- is-range - jeśli obecny, to kalendarz przyjmuje zakres dat
- is-dark - ciemny wygląd kalendarza
- attributes - przyjmuje listę wydarzeń, które wyświetlanie są w kalendarzu

```
1  <template>
2    ...
3      <v-calendar
4        class="calendar w-100"
5        :attributes="attributes"
6        disable-page-swipe>
7      </v-calendar>
8    ...
9  </template>
```

Użycie komponentu dostarczonego przez v-calendar. Opracowanie własne.

[Rysunki, diagramy, fragmentu kodu 3]

Wymagane jest aby lista wydarzeń miała pewien format. Musi to być JSON, zawierać klucz (ang. key) oraz daty (ang. dates). Może także zawierać dodatkowe informacje jak customData, obiekt służący do przekazywanie własnych danych do komponentu. Poniższa ilustracja przedstawia fragment kodu JavaScript, który odpowiada za przygotowanie danych typu wydarzenie (ang. event) na potrzeby wyświetlania ich w kompaktowej wersji kalendarza. Dla każdego wydarzenia, które zwróci nam serwer back end, tworzony jest obiekt zawierający:

- daty (ang. dates) w formie tablicy składającej się z początkowej i końcowej daty - linijka 6

- klucz (ang. key) odpowiadającego kluczowi głównemu w bazie danych - linijka 7
- dane dodatkowe (ang. custom data) jak na przykład, tytuł spotkania oraz data w przystępnej dla użytkownika formie - linijki 8-21
- obiekt o nazwie popover pozwalający zmodyfikować element wyświetlający się po naciśnięciu dnia do którego przypisane jest jakieś wydarzenie - linijki 22-34
- obiekt bar, dla którego ustawiany jest jedynie kolor. - linijki 34-37

Ostatecznie w linijkach 39-46, dodawany jest obiekt zawierający klucz „today”, zakończenie (ang. highlight) w kolorze żółtym i bez wypełnienia, oraz datę ustawianą dynamicznie na aktualny dzień. Pozwala to na oznaczenie aktualnej daty w momencie wyświetlania kalendarza.

```

▼ CalendarCompactView.vue

1  <script>
2    computed: {
3      attributes() {
4        return [
5          ...this.events.map(event => ({
6            dates: [event.startDateTime, event.endDateTime],
7            key: event.id,
8            customData: {
9              title: event.title,
10             class: 'bg-' + event.color + ' text-white',
11             start: new Date(event.startDateTime).getHours()
12               + ':'
13               + (new Date(event.startDateTime).getMinutes() < 10 ? '0' : '') +
14               new Date(event.startDateTime).getMinutes(),
15             startDay: new Date(event.startDateTime).getDate(),
16             end: new Date(event.endDateTime).getHours()
17               + ':'
18               + (new Date(event.endDateTime).getMinutes() < 10 ? '0' : '') +
19               new Date(event.endDateTime).getMinutes(),
20             endDay: new Date(event.endDateTime).getDate()
21           },
22           popover: {
23             label: new Date(event.startDateTime).getHours()
24               + ':'
25               + (new Date(event.startDateTime).getMinutes() < 10 ? '0' : '') +
26               new Date(event.startDateTime).getMinutes()
27               + '-' +
28               new Date(event.endDateTime).getHours()
29               + ':'
30               + (new Date(event.endDateTime).getMinutes() < 10 ? '0' : '') +
31               new Date(event.endDateTime).getMinutes(),
32               + ' ' + event.title,
33             visibility: 'click'
34           },
35           bar: {
36             color: this.mapVCalColors[event.color]
37           }
38         )),
39         {
40           key: 'today',
41           highlight: {
42             color: 'yellow',
43             fillMode: 'outline'
44           },
45           dates: new Date()
46         }
47       ],
48     },
49   }
50 </script>
snappify.io

```

Przygotowanie wydarzeń na potrzeby komponentu v-calendar. Opracowanie własne.
[Rysunki, diagramy, fragmenty kodu 4]

4.2.3. Bootstrap-vue

Bootstrap-vue to zmodyfikowana biblioteka bootstrap. Tak samo jak jej pierwówzór dostarcza narzędzia pozwalające modyfikować wygląd elementów strony internetowej. Dzięki wcześniej zdefiniowanym klasom CSS, programista może w łatwy sposób ustalać wygląd oraz zachowanie tagów HTML. Dodatkowo, dostarcza wiele nowych komponentów i skryptów. Przykładowo są to:

- Przyciski
- Tabele
- Formularze

Aplikacja OMS używa bootstrap-vue przede wszystkim do tworzenia formularzy oraz wykorzystuje klasy CSS jakie ta biblioteka oferuje. Poniżej przedstawiono fragment kodu wykorzystującego tag <form> do utworzenia formularza dodania nowego miejsca w systemie.



V AddSpaceModal.vue

```
1 ...
2 <form class="requires-validation" @submit="submit" novalidate>
3   <div class="form-row">
4     <div class="form-group col">
5       <label for="spaceTypeSelect">Typ miejsca</label>
6       <select class="form-control"
7           id="spaceTypeSelect"
8           v-model="selectedType">
9         <option value="desk">Biurko</option>
10        <option value="parking">Miejsce parkingowe</option>
11        <option value="room">Salka konferencyjna</option>
12      </select>
13    </div>
14  ...
15 </form>
16 ...
```

snappify.io

Wykorzystanie klas „form” dostarczonych przez bibliotekę Bootstrap. Opracowanie własne.
[Rysunki, diagramy, fragmenty kodu 5]

Linijka numer 2 rozpoczęta formularz nadając mu klasę „requires-validation” wykorzystywaną do przeprowadzenia walidacji poprawności formularza oraz nadpisuje domyślną metodę wysyłającą formularz metodą „submit”. Atrybut „novalidate” wyłącza domyślną dla przeglądarki walidację.

Linijki o numerach 3,4 oraz 6 przedstawiają wykorzystanie klas biblioteki bootstrap. Klasa form-group nadaje tagowi <div> formę elementu grupującego nazwę pola oraz pole przyjmujące wartość. „col” jest natomiast skrótem od column (pl. kolumna) pozwala na oznaczenie tagu jako kolumny wiersza tego formularza wcześniej zdefiniowanego przy pomocy form-row.

4.2.4. Vue router

Pakiet Vue router odpowiada za nawigację w aplikacji OMS. Router należy najpierw zdefiniować. Polega to na zadeklarowaniu możliwych w projekcie ścieżek oraz nadanie im odpowiadających komponentów. Deklaracja ma formę tablicy obiektów JSON, posiadających ścieżkę (ang. path), nazwę (ang. name) oraz komponent (ang. component). Następnie przy pomocy metody createRouter, dostarczonej wraz z pakietem, tworzony jest obiekt wykorzystywany w innych komponentach do zarządzania nawigacją. Poniższy fragment kodu przedstawia implementację dla systemu OMS.

```
JS index.js

1 const routes = [
2   {
3     path: '/',
4     name: 'Home',
5     component: Home
6   },
7   {
8     path: '/users',
9     name: 'Users',
10    component: Users
11  },
12  ...
13  {
14    path: '/login',
15    name: 'Login',
16    component: Login
17  }
18 ]
19
20 const router = createRouter({
21   history: createWebHistory(process.env.BASE_URL),
22   routes
23 })
24
25 export default router;
```

snappify.io

Konfiguracja magazynu Vuex. Opracowanie własne.
[Rysunki, dogramy, fragmentu kodu 6]

4.2.5. Agora-rtc-sdk-ng

Agora IO to platforma umożliwiająca implementację wideo rozmowy w czasie rzeczywistym. Dostarcza bibliotekę zawierającą metody potrzebne do połączenia z serwerem zewnętrznym. Darmowa wersja udostępniana na potrzeby edukacji pozwala na 10 tysięcy minut rozmowy miesięcznie.

Implementację rozpoczyna się od stworzenia projektu w konsoli dostarczonej przez twórców rozwiązania. Na potrzeby aplikacji OMS utworzono projekt o nazwie OMS3, edukacja jako cel użycia oraz APP ID jako metoda autoryzacji. APP ID jest to identyfikator aplikacji ustalany automatycznie po stworzeniu projektu.

Kolejnym etapem jest implementacja po stronie aplikacji OMS. W tym celu stworzono folder o nazwie webRTC zawierający komponent Vue EventRoom oraz skrypt napisany w języku JavaScript.

EventRoom to podstawowy widok stworzony z jednego tagu <div> o atrybutie id streamsContainer. Komponent posiada metodę joinRoomInit inicjalizującą dołączenie do pokoju rozmowy. Metoda wywoływana jest w momencie powstania widoku. Pozwala na to metoda mounted dostarczona w frameworku Vue.

Skrypt „script.js” zawiera konfigurację połączenia, implementację metody joinRoomInit oraz dodatkowe metody. Pierwszym krokiem konfiguracji jest deklaracja zmiennej APP_ID oraz przypisanie do niej wcześniej utworzonego identyfikatora projektu. Następnie pobierane są parametry uid (identyfikator użytkownika) oraz roomId (identyfikator pokoju, do którego łączy się użytkownik). Informacje te są potrzebne do nawiązania połączenia. Poniższy fragment kodu przedstawia wykorzystanie wcześniej zebranych danych w metodzie joinRoomInit.



```
JS script.js

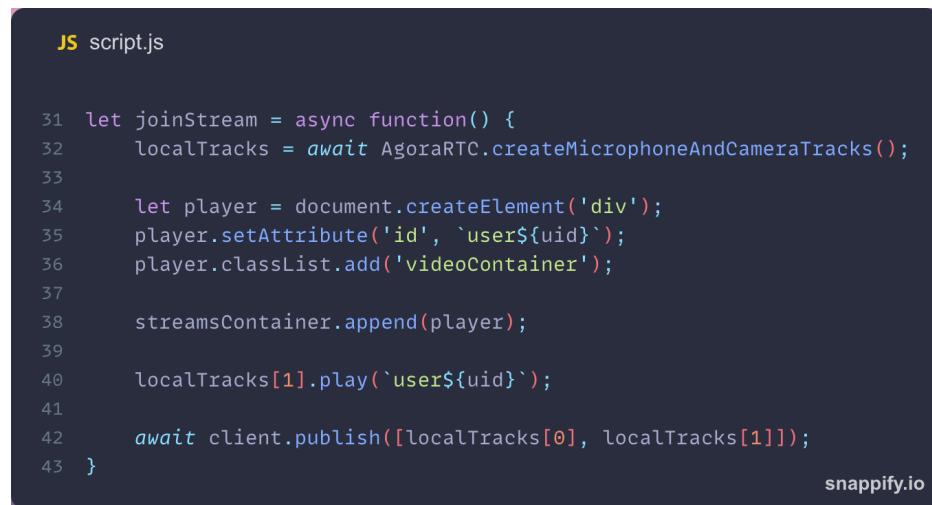
20  export let joinRoomInit = async function() {
21
22      client = AgoraRTC.createClient({mode: 'rtc', codec: 'vp8'});
23      await client.join(APP_ID, roomId, token, uid);
24
25      client.on('user-published', handleUserPublish);
26      client.on('user-left', handleUserLeft);
27
28      joinStream();
29 }
```

snappify.io

Metoda joinRoomInit inicjująca połączenie z Agora RTC. Opracowanie własne.
[Rysunki, diagramy, fragmentu kodu 7]

Linijka 22 tworzy obiekt klient (ang. client) korzystając z metody dostarczonej wraz z biblioteką agora-rtc-sdk-ng. W kolejnej linijce następuje połączenie klienta z projektem OMS3. Aby obsłużyć dołączenie innego użytkownika do pokoju dodano metodę handleUserPublish wywoływaną w momencie wystąpienie wydarzenia user-published. Analogicznie, wydarzenie wyjścia obsługiwane jest przy pomocy metody handleUserLeft przy wystąpieniu wydarzenia user-left. Ostatnią częścią inicjalizacji jest dołączenie strumienia video zalogowanego użytkownika przy pomocy metody joinStream.

Implementacja metody joinStream zaczyna się od pobrania lokalnego strumienia video. Strumień ten jest następnie dodawany jako tag <div> do wcześniej stworzonego tagu o id videoContainer. Ostatnim krokiem jest publikacja strumienia, dzięki temu każdy z użytkowników pokoju będzie mógł obsłużyć przychodzący obraz. Poniżej przedstawiony został opisany fragment kodu.



```
JS script.js

31 let joinStream = async function() {
32     localTracks = await AgoraRTC.createMicrophoneAndCameraTracks();
33
34     let player = document.createElement('div');
35     player.setAttribute('id', `user${uid}`);
36     player.classList.add('videoContainer');
37
38     streamsContainer.append(player);
39
40     localTracks[1].play(`user${uid}`);
41
42     await client.publish([localTracks[0], localTracks[1]]);
43 }
```

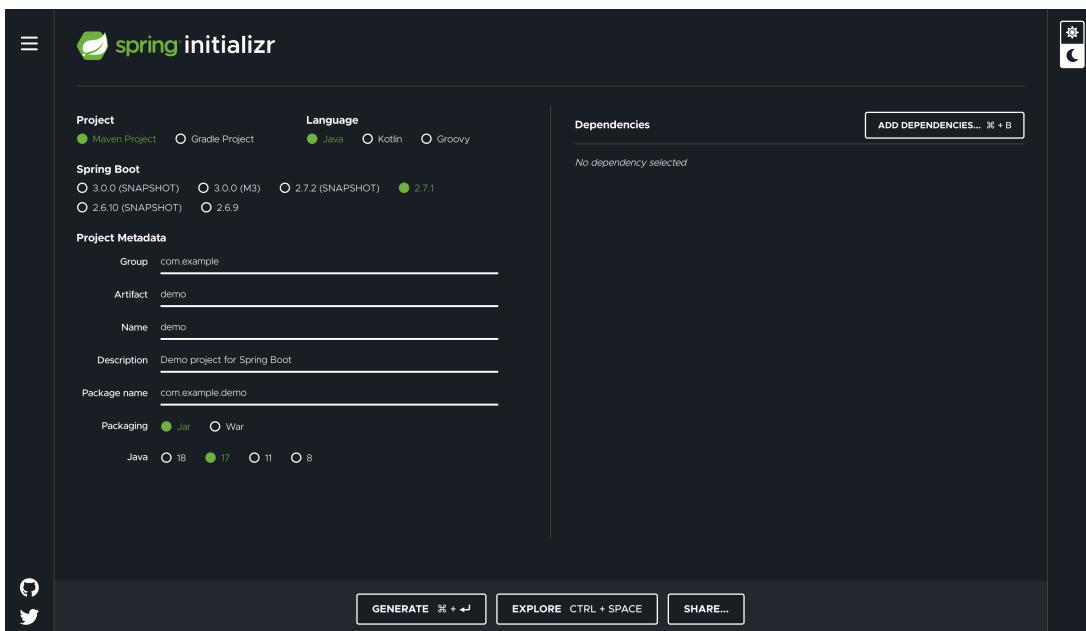
snappify.io

Metoda joinStream implementująca dołączenie użytkownika do pokoju.
[Rysunki, diagramy, fragmenty kodu 8]

Metoda handleUserPublish pobiera strumień przychodzący, tworzy tag <div>, nadaje mu wartość źródła strumienia, na końcu dodając go tagu videoContainer. handleUserLeft usuwa z głównego kontenera strumień odpowiadający użytkownikowi, który rozpoczął wydarzenie user-left.

4.3. Back end

Spring Initializr to narzędzie wykorzystywane do tworzenia projektów Spring. Pozwala na wybranie podstawowych szczegółów, takich jak język programowania, wersja Spring Boot, nazwa projektu czy zależności. Zależności można dodać także później. Wygenerowany projekt posiada podstawową konfigurację dzięki czemu można go uruchomić od razu po stworzeniu. Poniżej przedstawiono widok narzędzia.



Widok narzędzia Spring Initializr.
[Rysunki, diagramy, fragmenty kodu 9]

4.3.1. Spring Boot Data JPA

Java Persistence API został wykorzystany do zapewnienia trwałości danych. Poniższy fragment kodu przedstawia plik konfiguracyjny.

```
applaction.properties

31 spring.datasource.url=jdbc:mysql://localhost:3306/office_system?useSSL=false
32 spring.datasource.username=root
33 spring.datasource.password=password
34
35 spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5InnoDBDialect
36 spring.jpa.hibernate.ddl-auto=update
snappify.io
```

Spring - konfiguracja połączenia z bazą danych. Opracowanie własne.
[Rysunki, diagramy, fragmenty kodu 10]

- `spring.datasource.url` - wskazuje adres oraz port bazy danych

- username, password - to nazwa użytkownika oraz hasło umożliwiające połączenie z bazą
- spring.jpa.properties.hibernate.dialect - jest to określenie dialekta w jakim należy komunikować się z bazą
- spring.jpa.hibernate.ddl-auto - wartość update sprawia, że baza danych nie jest restartowana do pierwotnego stanu z przed uruchomieniem aplikacji

Na podstawie każdej encji, zaprojektowanej w trakcie fazy projektowania i analizy, utworzona została klasa Java. Klasę można interpretować jako tabelę w bazie danych a atrybuty klasy jako kolumny tej tabeli. Encję oznaczono adnotacją @Entity oraz @Table. Poniżej przedstawiono wykorzystanie tych adnotacji w przypadku encji użytkownika (ang. user).



```

31  @Entity
32  @Table(name = "User")
33  public class User {
34      private Long userId;
35      private String firstName;
36      private String lastName;
37      private String email;
38      private String password;
39      private String occupation;
40      private User manager;
41      private LocalDate dateOfBirth;
42      private UserRole userRole;
43      ...
44 }

```

snappify.io

Implementacja klasy User. Opracowanie własne.
[Rysunki, diagramy, fragmenty kodu 11]

Kolejnym krokiem przygotowania obiektu jest dodanie adnotacji do atrybutów. Dzięki temu JPA wie w jaki sposób przedstawić je w bazie danych. Zostały one dodane nad metodami zapewniającymi dostęp do atrybutów poza obiektem (tak zwane gettery).

Każda tabela powinna posiadać klucz główny. Do opisania tej kolumny użyta została adnotacja @Id oraz @GeneratedValue(strategy = GenerationType.IDENTITY) zapewniając tym samym automatyczne nadawanie unikatowych kluczy każdemu nowo utworzonemu obiekowi.

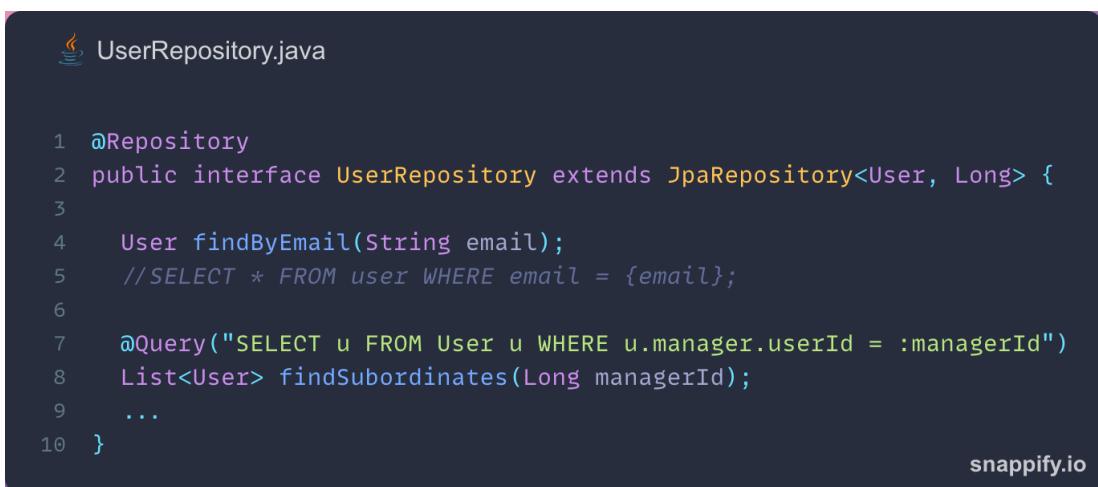
Większość z pozostałych atrybutów została oznaczona @Basic (pl. podstawowy). Oznacza to, że nie wymagają one specjalnego traktowania, są zwyczajnie

przepisywane do bazy danych. Wyjątkami są atrybuty przechowujący obiekt menadżera oraz role użytkownika.

Aby nie zapisywać całego obiektu typu użytkownik, pole manager oznaczono adnotacją @OneToOne. Pozwala to na zapisanie w bazie samego klucza głównego. Dzięki temu, gdy z bazy danych pobierany jest użytkownik JPA automatycznie wyszukuje odpowiedniego użytkownika i wstawia go do atrybutu manager.

Dostępne w aplikacji OMS role użytkownika są z góry ustalone, dlatego do ich przechowywania został utworzony enum. Jest to typ wyliczeniowy, który umożliwia zadeklarowanie ograniczonej liczby wartości. Takie pole wymaga specjalnej adnotacji JPA. Jest nią @Enumerated(EnumType.STRING). Dodatkowa konfiguracja EnumType.STRING oznacza, że w bazie danych przechowywane są role w formie wyrazów.

Do komunikacji z bazą danych wykorzystywane są repozytoria. Każda encja posiada własne repozytorium nazwane w formacie EncjaRepository, które dziedziczy po JpaRepository<User, Long>. „User” to klasa z której repozytorium jest związane a „Long” to typ atrybutu klucza głównego. Poniżej przedstawiono fragmentu kodu implementującego repozytorium dla obiektu użytkownik.



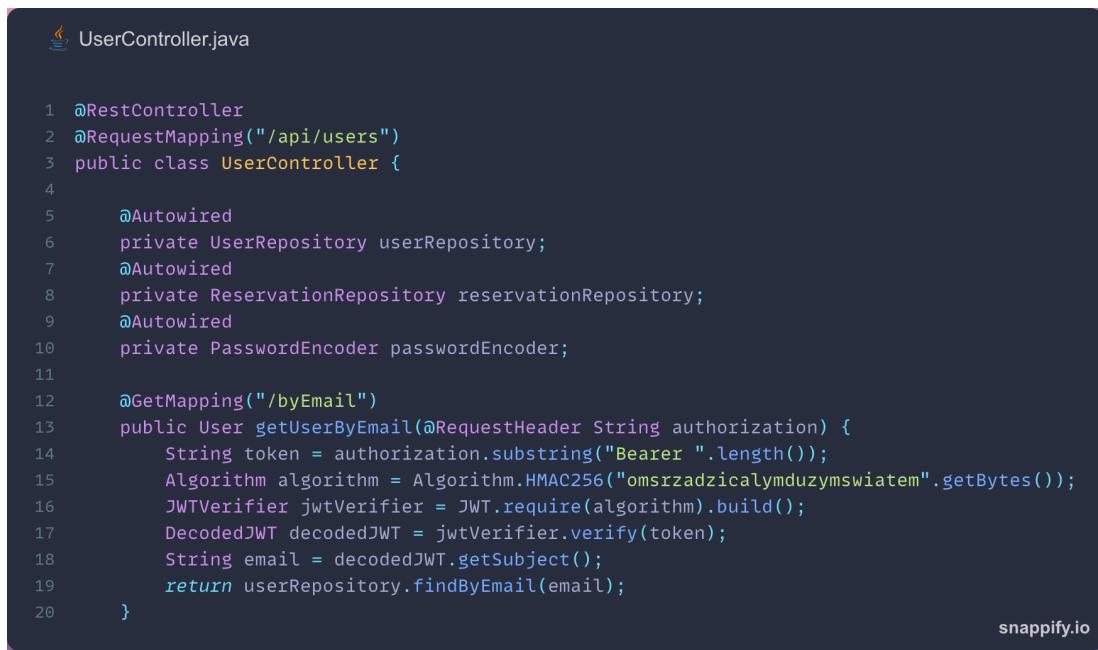
```
1  @Repository
2  public interface UserRepository extends JpaRepository<User, Long> {
3
4      User findByEmail(String email);
5      // SELECT * FROM user WHERE email = {email};
6
7      @Query("SELECT u FROM User u WHERE u.manager.userId = :managerId")
8      List<User> findSubordinates(Long managerId);
9
10 }
```

Implementacja interfejsu UserRepository. Opracowanie własne.
[Rysunki, diagramy, fragmenty kodu 12]

Przy pomocy adnotacji @Query, zdefiniowane zostało zapytanie SQL pobierające wszystkich użytkowników posiadających za menadżera użytkownika o numerze identyfikacyjnym „managerId”. W przypadku metody findByEmail wykorzystane zostało słowo kluczowe „findBy” dostarczone przez JPA. Pozwala to bibliotece automatycznie wygenerować zapytanie.

4.3.1. Spring Web Starter

Aplikacja OMS posiada pięć REST kontrolerów. Kontrolery grupują metody działające na tych samych obiektach. Poniżej omówiony został przypadek obiektu użytkownik (ang. user).



The screenshot shows a code editor window with the title "UserController.java". The code is as follows:

```
UserController.java

1  @RestController
2  @RequestMapping("/api/users")
3  public class UserController {
4
5      @Autowired
6      private UserRepository userRepository;
7
8      @Autowired
9      private ReservationRepository reservationRepository;
10
11     @Autowired
12     private PasswordEncoder passwordEncoder;
13
14     @GetMapping("/byEmail")
15     public User getUserByEmail(@RequestHeader String authorization) {
16         String token = authorization.substring("Bearer ".length());
17         Algorithm algorithm = Algorithm.HMAC256("omsrzadzicalynduzymswiatem".getBytes());
18         JWTVerifier jwtVerifier = JWT.require(algorithm).build();
19         DecodedJWT decodedJWT = jwtVerifier.verify(token);
20         String email = decodedJWT.getSubject();
21         return userRepository.findByEmail(email);
22     }
}
```

snappify.io

Implementacja kontrolera użytkownika. Opracowanie własne.
[Rysunki, diagramy, fragmenty kodu 13]

Klasa UserController została oznaczona adnotacjami @RestController oraz @RequestMapping. Pierwsza oznacza klasę jako kontroler pochodzący z biblioteki Spring Web Starter. Druga zaś wskazuje końcówkę, która odróżnia ją od innych kontrolerów.

Adnotacja @Autowired służy do wstrzykiwania ziarna (ang. bean) repozytorium. Jest to wymagane, ponieważ kontrolery to klasy, których instancje nie są tworzone w tradycyjny sposób. Nie posiadają konstruktorów, w których można by było zadeklarować wartość zmiennej przechowującej repozytorium.

Kolejne końcówki dostępne w obrębie danego kontrolera posiadają adnotację opisującą metodę HTTP jaką obsługują. Przypadek pokazany na rysunku 13 reaguje na zapytanie wysłane na końcówkę /api/users/byEmail z metodą GET (słującą do pobierania danych z serwera API). Aby poprawnie obsłużyć to zapytanie, musi ono posiadać nagłówek „authorization” zawierający token JWT. Token ten jest następnie dekodowany i pobierany jest z niego email użytkownika. Następnie wykorzystując metodę findByEmail dostępną, w repozytorium

użytkownika, pobierany jest z bazy obiekt o poszukiwanym adresie. Obiekt ten jest następnie zwracany w postaci obiektu JSON.

4.3.2. Spring Starter Security i JWT

Podstawowa konfiguracja dostarczona przez Spring Starter Security została nadpisana przy użyciu nowo stworzonej klasy SecurityConfig. Została ona oznaczona adnotacjami @Configuration oraz @EnableWebSecurity. Dodatkowo, klasa dziedziczy po WebSecurityConfigurerAdapter oraz nadpisuje metodę „configure”.



```
1 @Configuration
2 @EnableWebSecurity
3 public class SecutiryConfig extends WebSecurityConfigurerAdapter {
4     @Override
5     protected void configure(HttpSecurity http) throws Exception {
6         http.csrf().disable();
7         http.sessionManagement().sessionCreationPolicy(STATELESS);
8         http.addFilter(new AuthenticationFilter(authenticationManagerBean()));
9         http.addFilterBefore(new AutorizationFilter(), UsernamePasswordAuthenticationFilter.class);
10        http.authorizeRequests().antMatchers(GET, "/login", "/token/refresh").permitAll();
11        http.authorizeRequests().anyRequest().hasAnyAuthority("USER", "HRUSER", "ADMIN");
12    }
13    ...
14 }
```

snappify.io

Konfiguracja Spring Security. Opracowanie własne.
[Rysunki, diagramy, fragmenty kodu 14]

Najważniejszymi elementami konfiguracji jest ustawienie filtra uwierzytelnienia (linijka 8), oraz filtra autoryzacji (linijka 9). Klasy te jednak trzeba najpierw zaimplementować co zostanie opisane w kolejnych akapitach. Innym, równie istotnym ustawieniem są linijka 10 i 11. Przedstawiają definicje dostępności różnych elementów aplikacji zależnie od roli użytkownika. Metoda „antMatchers” przyjmuje jako parametry metodę HTTP oraz wiele obiektów typu String reprezentujących koncówki REST API. W przypadku aplikacji OMS została ona użyta do umożliwienia zalogowania się lub odświeżenia tokenu JWT wszystkim, gdyż skoro wywołują metodę autoryzacji prawdopodobnie nie posiadają jeszcze swojego tokenu lub jest on przeterminowany. Natomiast metoda „anyRequest” odpowiduje się do wszystkich pozostałych, wcześniej nie zdefiniowanych końcówek. W połączeniu z metodą „hasAnyAuthority” pozwala udzielić dostępu tylko użytkownikowi z rolą jedną z podanych jako parametry.

Filtr uwierzytelnienia odpowiada za to co się stanie gdy login oraz hasło podane przez użytkownika zostaną zaakceptowane. Z tego powodu utworzono nową klasę

AuthenticationFilter rozszerzającą UsernamePasswordAuthenticationFilter. Klasa ta nadpisuje dwie metody „attemptAuthentication” oraz „successfulAuthentication”. Pierwsza z nich weryfikuje tożsamość użytkownika sprawdzając poprawność loginu oraz hasła. Po udanej weryfikacji, wywoływana jest druga metoda, która generuje tokeny JWT oraz zapisuje je w plikach cookies odpowiedzi. Rysunek 15 przedstawia implementację metody „successfulAuthentication”.

```
AuthenticationFilter.java

43 ...
44 @Override
45 protected void successfulAuthentication(HttpServletRequest request, HttpServletResponse response,
46 FilterChain chain, Authentication authentication) throws IOException, ServletException {
47     User user = (User)authentication.getPrincipal();
48     Algorithm algorithm = Algorithm.HMAC256("omsrzadzialymduzymswiatem".getBytes());
49     String accessToken = JWT.create()
50         .withSubject(user.getUsername())
51         .withExpiresAt(new Date(System.currentTimeMillis() + 10*60*1000))
52         .withIssuer(request.getRequestURL().toString())
53         .withClaim("role", user.getAuthorities().stream().map(
54             GrantedAuthority::getAuthority).collect(Collectors.toList())
55         ))
56         .sign(algorithm);
57     String refreshToken = JWT.create()
58         .withSubject(user.getUsername())
59         .withExpiresAt(new Date(System.currentTimeMillis() + 60*60*1000))
60         .withIssuer(request.getRequestURL().toString())
61         .sign(algorithm);
62     Cookie accessTokenCookie = new Cookie("accessToken", accessToken);
63     Cookie refreshTokenCookie = new Cookie("refreshToken", refreshToken);
64     response.addCookie(accessTokenCookie);
65     response.addCookie(refreshTokenCookie);
66     Map<String, String> tokens = new HashMap<>();
67     tokens.put("accessToken", accessToken);
68     tokens.put("refreshToken", refreshToken);
69     response.setContentType(APPLICATION_JSON_VALUE);
70     new ObjectMapper().writeValue(response.getOutputStream(), tokens);
71 }
72 ...
```

snappify.io

Implementacja nadpisanej metody successfulAuthentication. Opracowanie własne.
[Rysunki, diagramy, fragmenty kodu 15]

Na potrzeby autoryzacji użytkownika generowane są dwa tokeny JWT: dostępu (ang. access token) oraz odświeżania (ang. refresh token). Tokeny szyfrowane są przy pomocy algorytmu HMAC256. Przechowuję informację o adresie email użytkownika, źródło oraz datę wygaśnięcia. Data wygaśnięcia tokenu dostępu to została ustawiona na 10 minut. Token odświeżania wygasza natomiast po 60 minutach.

5. Testy aplikacji

Kod źródłowy aplikacji został uruchomiony przy pomocy IDE na lokalnym urządzeniu. Serwer aplikacji (back end) domyślnie działa na porcie 8080. Natomiast warstwa widoku (front end) nasłuchiwa na porcie 3000. Podczas testów skupiono się na warstwie widoku. Ponieważ zawiera ona odniesienia do serwera, przetestowano w ten sposób całą aplikację OMS. Po uruchomieniu, dostęp do aplikacji następuje poprzez link <http://localhost:3000/>.

Gotowa aplikacja została przetestowana manualnie. Oznacza to, że osoba testująca wcieliła się w rolę użytkownika aby sprawdzić czy wszystkie funkcjonalności systemu działają poprawnie. Podczas testów korzystano z przeglądarki Google Chrome 103.0 (Oficjalna wersja) (arm64).

5.1. Logowanie

Pierwszym widokiem po nawiązaniu połączenia jest okno logowania. Służy ono do weryfikacji użytkownika przy pomocy adresu email oraz hasła. Gdy użytkownik wprowadzi już swoje dane, wybiera przycisk „ZALOGUJ”. Jeśli weryfikacja nie powiedzie się, wyświetlona zostanie informacja „Nieprawidłowy e-mail lub hasło.” a pola zostaną wyczyszczone. [Rysunek 16]

The screenshot shows a login form for the OMS application. The title 'OMS' is at the top. Below it are two input fields: 'E-mail' and 'Hasło'. Underneath the 'Hasło' field is a red error message: 'Nieprawidłowy e-mail lub hasło.' At the bottom is a large blue button labeled 'ZALOGUJ'.

Logowanie - nieprawidłowe dane.
[Rysunki, diagramy, fragmentu kodu 16]

Od tego momentu użytkownik może spróbować ponownie, aż do momentu wprowadzenia poprawnych danych. Gdy już to nastąpi, następuje przekierowanie do strony powitalnej aplikacji. [Rysunek 20]



Strona powitalna.
[Rysunki, diagramy, fragmenty kodu 17]

Nawigacja wewnętrz aplikacji odbywa się przy pomocy paska pobocznego (ang. sidebar), zwanego dalej sidebar. Sidebar jest widoczny przez cały czas, gdy użytkownik jest zalogowany. Składa się on z loga aplikacji, listy modułów oraz przycisku „Wyloguj”. Odpowiednio numery 1, 2 oraz 3. [Rysunek 17] Użytkownik może poruszać się między modułami wybierając odpowiedni z listy dostępnych.

5.1. Moduł moduł spotkań

Wewnątrz aplikacji, moduł spotkań, znajduje się pod nazwą „Kalendarz”. Z jego poziomu użytkownik jest w stanie zobaczyć spotkania przez niego utworzone lub te na które został zaproszony przez innych użytkowników. Dodatkowo znajduje się tu przycisk służący do tworzenia nowego spotkania.

Widok modułu spotkań.
[Rysunki, diagramy, fragmenty kodu 18]

Widok kalendarza różni się dla urządzeń mobilnych.
Aby zachować przejrzystość aplikacji, spotkania i informacje o nich zostały przeniesione do listy znajdującej się tuż pod widokiem kalendarza. Dla ułatwienia rozpoznania daty spotkania, kolor jakim spotkanie jest oznaczone znajduje się zarówno na liście jak i w kalendarzu.

Mobilny widok modułu spotkań.
[Rysunki, diagramy, fragmenty kodu 19]

Użytkownik wybierając przycisk „Dodaj wydarzenie” otwiera okno kreacji spotkania. Okno zawiera szczegółowy formularz. Po poprawnym wypełnieniu oraz zapisaniu, tworzy się nowe spotkanie, które jest natychmiast widoczne w kalendarzu.

The screenshot shows a modal dialog titled "Nowe wydarzenie" (New event). At the top, there are four color-coded radio buttons: blue, dark grey, teal, and red. Below them is a "Tytuł" (Title) input field, which is currently empty. Underneath the title is a section for "Czas trwania" (Duration) and "Data" (Date), both of which are set to "0" and "29.06.2022 23:05" respectively. The "Uczestnicy" (Participants) section contains a list of eight users: Adam Nowakowski - Kierownik, Admin Admin - Pracownik, Andrzej Bruk - Tester, Asia Sroka - HR Specialist, HRUser HRUser - Pracownik, Jan Dobry - Pracownik, Piotrek Kowalski - Pracownik Działu Marketingu, and Tomasz Myślik - Pracownik. At the bottom left is a checkbox labeled "Zaznacz, aby utworzyć wideokonferencje do spotkania." (Check to create a video conference for the meeting). On the right side, there is a blue "Zapisz" (Save) button.

Tworzenie nowego spotkania.
[Rysunki, diagramy, fragmenty kodu 20]

Jeżeli użytkownik zaznaczy checkbox „Zaznacz, aby utworzyć wideokonferencje do spotkania”, utworzony zostanie pokój. Zmieni się także widok szczegółów spotkania. Dodatkowo pojawi się przycisk „Otwórz okno spotkania”.

The screenshot shows a modal dialog titled "Szczegóły wydarzenia" (Event details). It displays the event information from the previous dialog: "Rozmowa z Adamem N." as the title, "User User - Pracownik" as the organizer, a duration of "60", and a date of "16.06.2022 11:30". The "Uczestnicy" section shows "Adam Nowakowski - Kierownik". At the bottom left is a blue "Otwórz okno spotkania" (Open meeting window) button, and at the bottom right is a teal "Edytuj" (Edit) button.

Widok szczegółów spotkania.
[Rysunki, diagramy, fragmenty kodu 21]

Gdy użytkownik wciśnie przycisk „Otwórz okno spotkania” otworzona zostanie nowa zakładka przeglądarki przedstawiająca widok pokoju wideokonferencji. Jeśli inny użytkownik także dołączy do tego spotkania, obraz jego kamery zostanie wyświetlony na ekranie użytkownika oraz będą mogli ze sobą rozmawiać przy pomocy mikrofonu.

5.2. Moduł rezerwacji miejsc

Miejsca					
Nazwa	Piętro	Opis	Rozmiar	Miejsca	Akcje
B123	1	Biurko z widokiem na morze.		8	<button>Rezerwuj</button>
B125	1	Blisko do kuchni.		8	<button>Rezerwuj</button>
P3	2	Rura wystająca ze ściany wchodzi na miejsca parkingowe, przez co jest mniejsze.	Duże	8	<button>Rezerwuj</button>
P5	-1	Blisko do windy towarowej.	Średnie	8	<button>Rezerwuj</button>
S57	1	Dodatkowo tablica interaktywna.	20m ²	8	<button>Rezerwuj</button>
S58	1	Projektor.	15m ²	8	<button>Rezerwuj</button>
S59	3	Ładny widok z okna.	25m ²	12	<button>Rezerwuj</button>
S60	2	Sala niebieska	32m ²	16	<button>Rezerwuj</button>

Moduł rezerwacji miejsc. [Rysunki, dogramy, fragmenty kodu 22]

Moduł rezerwacji miejsc jest przedstawiony w formie listy miejsc istniejących w aplikacji. [Rysunek 22] Z tego widoku użytkownik może wyszukiwać słów kluczowych we wszystkich rekordach, filtrować listę po nazwie miejsca, piętrze i opisie, a także reperować nie zablokowane miejsca. Widok różni się zależnie od roli użytkownika. [Rysunek 23] Administratorzy oraz pracownicy działu HR posiadają dodatkowe funkcje, takie jak: dodanie nowego miejsca, blokada, edycja oraz usunięcie istniejącego miejsca. Miejsca zablokowane są przekreślone kolorem czerwonym i nie można ich zarezerwować.

Rozszerzony moduł rezerwacji miejsc. [Rysunki, diagramy, fragmenty kodu 23]

Utworzenie nowej rezerwacji blokuje wybrane daty dla danego miejsca. Dodawany jest także nowy wpis do listy „Moje rezerwacje”. [Rysunek 24] Widok ten przedstawia listę rezerwacji jakich dokonał użytkownik. Dostępna jest tu także opcja anulowania rezerwacji.

Widok rezerwacji zalogowanego użytkownika.
[Rysunki, diagramy, fragmenty kodu 24]

5.3. Moduł zarządzanie hierarchią i użytkownikami

Moduł ten jest dostępny tylko dla użytkowników posiadających role administratora lub pracownika działu HR. Składa się z trzech sekcji. Pierwsza sekcja od lewej to lista wszystkich użytkowników systemu posortowana alfabetycznie. W środku znajduje się sekcja „Podwładni”, która domyślnie jest pusta. W momencie gdy z sekcji pierwszej zostanie wybrany użytkownik, to o ile posiada on jakiś podwładnych, zostaną oni wyświetleni w sekcji drugiej. Po prawej stronie modułu znajdują się szczegóły ostatnio wybranego użytkownika. Oznacza to, że korzystający z aplikacji może wybrać rekord zarówno z listy użytkowników jak i podwładnych, a szczegóły ostatnio wybranego, pojawią się w tej sekcji. Z poziomu tego modułu korzystający z aplikacji jest także w stanie stworzyć nowego użytkownika oraz edytować istniejącego. Widok modułu zarządzania hierarchią i użytkownikami nosi nazwę „Użytkownicy” a jego wygląd został przedstawiony na poniższym rysunku. [Rysunek 25]

Lista użytkowników		Podwładni		Szczegóły użytkownika	
Adam Nowakowski	p1	Admin Admin	p16	Email	jdobry@oms.oms
Admin Admin	p16	Asia Sroka	p7	Imię	Jan
Andrzej Bruk	p8	HRUser HRUser	p15	Nazwisko	Dobry
Asia Sroka	p7	Jan Dobry	p5	Data urodzenia	12.05.1982
HRUser HRUser	p15	User User	p14	Stanowisko	Pracownik
Jan Dobry	p5			Rola	User
Piotrek Kowalski	p2			Menadżer	1
Tomasz Myślik	p10				
User User	p14				

Moduł zarządzanie hierarchią i użytkownikami.
[Rysunki, diagramy, fragmenty kodu 25]

6. Przyszłość projektu

Aplikacja została stworzona w celu rozwiązymania konkretnych problemów. Mimo to, jest jeszcze wiele elementów, które można by było w przyszłości dodać lub zmienić. Ten rozdział przedstawia pomysły zrodzone w trakcie wytwarzania oprogramowania, jednakże ze względu na ograniczony czas i złożoność ich implementacji nie zostały wdrożone.

Moduł spotkań (kalendarz) jest w pełni sprawy, pozwala tworzyć, edytować oraz usuwać wydarzenia. Jednakże, w dzisiejszych czasach nie da się uniknąć stosowania innych aplikacji, jak na przykład OutLook. Jest to rozwiązanie dostarczające między innymi moduł prowadzenia kalendarza. Narzędzie te jest powszechnie używane w wielu firmach. Aplikacja OMS mogłaby obsługiwać synchronizację kalendarzy. Dzięki temu kalendarz byłby spójny między wszystkimi używanymi w firmie aplikacjami.

Spotkania aplikacji OMS obsługują także wideokonferencje. Umożliwia to kontakt twarzą w twarz z współpracownikami. Mimo, że większość firm zapewnia pracownikom kamerę oraz mikrofon, zawsze może zdarzyć się nieoczekiwany wyjątek. Aplikacja OMS mogłaby zostać rozszerzona o czat. Pozwoliło by to na swobodną rozmowę, także wtedy gdy sprzęt zawiedzie.

Według polskiego prawa, każdy pracodawca zatrudniający conajmniej jednego pracownika ma obowiązek ewidencjonowania czasu pracy. [9] Dlatego też, dużym krokiem w rozwoju aplikacji mógłby być moduł ewidencji czasu pracy. Umożliwiło by to na przydzielanie pracownikom projektów w ramach, których pracują. Następnie pracownicy mogli by co ustalony okres wypełniać formularz i wysyłać do akceptacji przełożonego.

7. Podsumowanie

Czynności wykonane w ramach tego projektu pozwoliły stworzyć aplikacje webową, która rozwiązuje problem digitalizacji procesów administracyjnych w małych oraz średnich firmach. Wynik końcowy dostarcza możliwość:

- zarządzania hierarchią firmy
- prowadzenia kalendarza prywatnego oraz wspólnego z innymi pracownikami
- rezerwacji materialnych miejsc znajdujących się w siedzibie biura, takich jak biurka, sale konferencyjne oraz miejsca parkingowe
- prowadzenia rozmów między użytkownikami z użyciem kamery oraz mikrofonu

Faza analizy i projektowania pozwoliła stworzyć szczegółową wizję aplikacji popartą diagramami. Pozwoliło to uniknąć czasochłonnych poprawek w kolejnych fazach procesu, a także znacznie przyczyniło się do połączenia modułów w jedną dobrze współpracującą całość.

Największymi wyzwaniami w trakcie tworzenia tego typu aplikacji są poprawność oraz trwałość danych przechowywanych w systemie. Aplikacja OMS zapewnia te cechy dzięki wykorzystaniu bibliotek specjalnie w tym celu stworzonych.

Kolejnym ważnym aspektem jest bezpieczeństwo. Dzięki zastosowaniu połączenia bibliotek Spring Starter Security i JWT, dostęp do aplikacji jest możliwy tylko po wcześniejszym zalogowaniu. Tokeny JWT umożliwiają także ograniczenie dostępu użytkownikom niższego stopnia do niektórych danych oraz funkcjonalności.

8. Słownik

Open source - używane w kontekście opisu dostępności oprogramowania, oznacza, że kod źródłowy aplikacji jest jawnym oraz ogólnodostępnym. Wykorzystanie takiego programu jest wciąż uregulowane licencją.

REST API - „Interfejs API, czyli *aplikacyjny interfejs programistyczny*, jest zestawem reguł definiujących sposób, w jaki aplikacje lub urządzenia mogą się ze sobą łączyć i komunikować.”²

IDE (ang. Integrated Development Environment) - program lub grupa programów wykorzystywanych do tworzenia, edycji, uruchamiania oraz testowania oprogramowania.

Dropdown - element aplikacji internetowej. Po wybraniu go wyświetlna zostaje lista opcji do wyboru. Multiselect dropdown pozwala wybrać kilka opcji.

URL - jest to adres strony internetowej. Wskazuje jej lokalizację w sieci.

JSON - tekstowy format zapisu danych.

² <https://www.ibm.com/pl-pl/cloud/learn/rest-apis>

9. Tabele, diagramy

[1]	Informacje przechowywane na potrzeby moduły rezerwacji miejsc
[2]	Wymagania funkcjonalne modułu rezerwacji miejsc
[3]	Informacje przechowywane na potrzeby modułu spotkań
[4]	Wymagania funkcjonalne modułu spotkań
[5]	Informacje przechowywane na potrzeby zarządzania użytkownikami
[6]	Wymagania funkcjonalne modułu zarządzania hierarchia i użytkownikami
[7]	Diagram aktorów. Opracowanie własne.
[8]	Diagram przypadków użycia. Opracowanie własne.
[9]	Diagram klas. Opracowanie własne.

10. Rysunki, fragmenty kodu

[1]	Widok terminala w trakcie inicjalizacji projektu. Opracowanie własne.
[2]	Użycie pakietu vuex. Opracowanie własne.
[3]	Użycie komponentu dostarczonego przez v-calendar. Opracowanie własne.
[4]	Przygotowanie wydarzeń na potrzeby komponentu v-calendar. Opracowanie własne.
[5]	Wykorzystanie klas „form” dostarczonych przez bibliotekę Bootstrap. Opracowanie własne.
[6]	Konfiguracja magazynu Vuex. Opracowanie własne.
[7]	Metoda joinRoomInit inicjalizująca połączenie z Agora RTC. Opracowanie własne.
[8]	Metoda joinStream implementująca dołączenie użytkownika do pokoju.
[9]	Widok narzędzia Spring Initializr. Data dostępu: 28.06.2022 https://start.spring.io/
[10]	Spring - konfiguracja połączenia z bazą danych. Opracowanie własne.
[11]	Implementacja klasy User. Opracowanie własne.
[12]	Implementacja interfejsu UserRepository. Opracowanie własne.
[13]	Implementacja kontrolera użytkownika. Opracowanie własne.
[14]	Konfiguracja Spring Security. Opracowanie własne.
[15]	Implementacja nadpisanej metody successfulAuthentication. Opracowanie własne.
[16]	Logowanie - nieprawidłowe dane.
[17]	Strona powitalna.
[18]	Widok modułu spotkań.
[19]	Mobilny widok modułu spotkań.
[20]	Tworzenie nowego spotkania.
[21]	Widok szczegółów spotkania.
[22]	Moduł rezerwacji miejsc.
[23]	Rozszerzony moduł rezerwacji miejsc.
[24]	Widok rezerwacji zalogowanego użytkownika.

11. Bibliografia

[1]	Data dostępu: 07.03.2022 https://www.statista.com/statistics/1122987/change-in-remote-work-trends-after-covid-in-usa/
[2] [3] [4]	Data dostępu: 16.03.2022 https://forms.workday.com/en-us/quick-demos/quick-demo-medium-enterprise/form.html
[5]	Data dostępu: 16.03.2022 https://www.saldeosmart.pl/biuro
[6]	Data dostępu 17.03.2022 https://www.saldeosmart.pl/firma
[7]	Data dostępu: 18.03.2022 https://www.systeo.pl/oferta-systeo/aplikacje-systeo365
[8]	Data dostępu: 29.03.2022 https://flowscapesolutions.com/solutions/desk-management
[9]	Data dostępu: 29.03.2022 https://flowscapesolutions.com/solutions/room-booking
[10]	Data dostępu: 30.06.2022 https://poradnikprzedsiebiorcy.pl/-ewidencja-czasu-pracy-nie-daj-sie-zaskoczyc-kontroli-pip