



AKADEMIA MARYNARKI WOJENNEJ

im. BOHATERÓW WESTERPLATTE

WYDZIAŁ MECHANICZNO - ELEKTRYCZNY

KATEDRA INFORMATYKI

PRACA INŻYNIERSKA

System zarządzania zasobami teleinformatycznymi

Wykonawca:

Mateusz Wedel

Praca wykonana pod kierunkiem:

dra inż. Artura Zacniewskiego

Spis treści

Wykaz skrótów i oznaczeń	5
Wprowadzenie	8
1.1. Przedstawienie wybranego problemu technicznego	8
1.2. Przedstawienie celu pracy	9
2. Dokumentacja projektu	9
2.1. Opis typu aplikacji	9
2.1.1. Podział ze względu na funkcjonalność	9
2.1.2. Podział ze względu na architekturę wraz ze wstępem teoretycznym.	9
2.1.3. Podział ze względu na infrastrukturę wraz z uwzględnieniem sposobu komunikacji	12
2.2. Koncepcja oraz techniczne aspekty projektu.	14
2.2.1. Cel implementacji oraz grupa docelowa	14
2.2.2. Wymagania funkcyjne	14
2.2.3. Moduły funkcjonalne aplikacji	15
2.2.4. Spis aktorów	16
2.2.5. Diagramy przypadków użycia	17
2.2.6. Przebieg scenariuszy	20
2.2.7. Platforma docelowa	35
2.3. Struktura Aplikacji	35
2.3.1. Warstwa aplikacji – część Backendowa	36
2.3.1.1. Podział warstwy internetowej	36
❖ Kontrolery	36
❖ DTO	37
❖ Walidatory	38
❖ Mappery	39
❖ Obsługa błędów	40
❖ Zabezpieczenia aplikacji	41
2.3.1.2. Warstwa serwisowa	43
2.3.1.3. Warstwa repozytorium	45
2.3.2. Warstwa prezentacji - część Frontendowa	47
2.3.2.3. Komponenty	47
2.3.2.4. Serwisy	49
2.3.2.5. Interceptory oraz dodatkowe klasy	49
2.3.2.6. Modele	51

2.3.2.5.	Routing	51
2.4.	Struktura bazy danych	52
2.5.	Testowalność	52
2.6.	Konfiguracja	54
2.7.	Wdrożenie aplikacji z wykorzystaniem platformy Docker	56
2.7.1.	Spis obrazów Docker wraz z przeznaczeniem	56
2.7.2.	Kontenery, ich połączenie oraz volume	60
2.7.1.	Docker Compose	60
2.7.2.	Wdrożenie aplikacji z najbardziej przydatnymi komendami do zarządzania kontenerami	62
3.	Wyniki	64
3.1.	Prezentacja aplikacji	64
3.1.1.	Strona logowania	64
3.1.2.	Strona główna	64
3.1.3.	Lista sprzętu	65
3.1.4.	Dodanie sprzętu	65
3.1.5.	Szczegóły sprzętu	66
3.1.6.	Usunięcie sprzętu	67
3.1.7.	Lista miejsc	67
3.1.8.	Dodanie/edycja miejsca	68
3.1.9.	Dodanie/edycja miejsca – podanie błędnych danych	68
3.1.10.	Szczegóły miejsca	69
3.1.11.	Usunięcie miejsca	69
3.1.12.	Lista osób	70
3.1.13.	Dodanie/edycja osoby	70
3.1.14.	Szczegóły osoby	71
3.1.15.	Tworzenie wypożyczenia	71
3.1.16.	Eksportowanie sprzętu	72
3.1.17.	Protokół zniszczenia sprzętu	72
3.1.18.	Raport wypożyczonego sprzętu	73
3.1.19.	Lista hostów	74
3.1.20.	Dodanie/edycja hostów	74
3.1.21.	Szczegóły hosta	75
3.1.22.	Dodanie/edycja przyłączenia sieciowego	75
4.	Podsumowanie	76

4.7.	Wnioski	76
4.8.	Późniejszy rozwój	76
Spis literatury		78
Spis rysunków		80
Spis tabel		82
Spis załączników		83
Streszczenia		84
	Streszczenie po polsku	84
	Summary in English	84

Wykaz skrótów i oznaczeń

Host – „dowolna maszyna (komputer, karta sieciowa, modem itp.) uczestnicząca w wymianie danych lub udostępniająca usługi sieciowe poprzez sieć komputerową za pomocą protokołu komunikacyjnego TCP/IP oraz posiadająca własny adres IP” [1].

PDF – „**Portable Document Format** – format plików służący do prezentacji, przenoszenia i drukowania treści tekstowo-graficznych, stworzony przez firmę Adobe Systems. Obecnie rozwijany i utrzymywany przez Międzynarodową Organizację Normalizacyjną” [2].

JavaScript - „(JS) to skryptowy (interpretowany lub kompilowany metodą JIT) język programowania, w którym funkcje są "obywatelami pierwszej kategorii" - obiektami, które można przechowywać w zmiennych jako referencje i przekazywać jak każde inne obiekty” [3].

Java Persistence API – „JPA jest tylko specyfikacją. Opisuje ona interfejs, z którymi współpracuje klient wraz z standardowymi metadanymi mapowania obiektowo-relacyjnego. (Wykorzystując adnotacje Javy lub pliki opisowe XML). Poza definicją API, JPA również wyjaśnia (jednakże nie wyczerpująco) jak te specyfikacje powinny być zaimplementowane przez biblioteki wyższego poziomu” [4].

JSON – „**JavaScript Object Notation** – jest prostym formatem wymiany danych. Zapis i odczyt danych w tym formacie jest łatwy do opanowania przez ludzi. Jednocześnie, z łatwością odczytują go i generują komputery” [5].

Docker – „Docker to platforma dla deweloperów oraz administratorów systemowych do budowania, udostępniania i uruchamiania aplikacji z wykorzystaniem kontenerów. Użycie kontenerów w celu wdrożenia aplikacji nazywane jest konteneryzacją” [6].

NGINX - „serwer WWW (HTTP) oraz serwer proxy dla HTTP i IMAP/POP. Zaprojektowany z myślą o wysokiej dostępności i silnie obciążonych serwisach (nacisk na skalowalność i niską zajętość zasobów)” [7].

REST – „REST jest akronimem od Representational State Transfer. Jest to styl architektoniczny dla rozproszonych systemów hipermedialnych oraz został pierwszy raz zaprezentowany przez Roy’a Fieldinga w 2000 roku w jego sławnej rozprawie doktorskiej” [8].

POJO – „W inżynierii oprogramowania, POJO jest zwyczajnym obiektem Javy, nie podlegającym żadnym specjalnym ograniczeniom innymi niż te wymuszone przez specyfikację języka Java” [9].

DTO – „Obiekt, który przenosi dane pomiędzy procesami/systemami...” [10].

JWT – „to pewien ciąg znaków w formacie JSON ([https:// www.json.org/](https://www.json.org/)) zakodowany w strukturę JWS (JSON Web Signature) lub JWE (JSON Web Encryption). Dodatkowo każda z tych opcji musi być zserializowana w sposób kompaktowy (ang. compact serialization – to jedna z dwóch serializacji wyliczanych w JWS i JWE)” [11].

UUID – „Jest to 128-bitowy numer wykorzystywane w celu identyfikacji informacji w systemach komputerowych” [12].

Obraz (docker) – „Obraz jest to uporządkowana kolekcja zmian w systemie plików roota oraz odpowiednie parametry uruchomieniowe do użytku w środowisku wykonawczym kontenera. Obraz zazwyczaj zawiera kombinację warstwowych systemów plików ułożone jeden na drugim. Obraz nie zawiera stanu oraz nigdy się nie zmienia” [13].

Kontener (docker) – „Kontener to uruchomiona instancja obrazu dockerowego” [14].

YAML – „uniwersalny język formalny przeznaczony do reprezentowania różnych danych w ustrukturalizowany sposób” [15].

Minifikacja - „proces mający na celu zmniejszenie kodu źródłowego poprzez usunięcie niepotrzebnych znaków bez zmieniania jego funkcjonalności” [16].

Wprowadzenie

1.1. Przedstawienie wybranego problemu technicznego

„Wszystko jest teoretycznie niemożliwe, dopóki nie zostanie zrobione.”

cyt. Robert A. Heinlein

W czasach buma technologicznego większość dobrze prosperujących instytucji zawierzyło organizację oraz swoje działania na systemach komputerowych. Większość z nich wypracowało własne rozwiązania, które są gwarantem bezpieczeństwa przetwarzanych danych. Jak zauważono, wprowadzenie systemu usprawnia placówki nie tylko pod kątem osiąganych wyników, ale również pod względem organizacyjnym.

Realizacja projektu „Systemu Zarządzania Zasobami Teleinformatycznym ” niesie ze sobą prosty przekaz. Pożądanym efektem jest uzyskanie zaplanowanej koordynacji poruszania się środków trwałych w placówce, co pomoże usprawnić proces wypożyczenia potrzebnego sprzętu. Wartością dodaną jest to, że korzystanie z powyższego rozwiązania zwiększa świadomość administratora o posiadanych zasobach. Jako następny walor można dodać, że zastosowanie zcentralizowanej bazy danych wyklucza konieczność korzystania z wielu plików CSV oraz daje możliwość wykonania kopii zapasowej. Dodatkowo wykluczona jest konieczność synchronizacji tych plików, gdyż aby zobaczyć wprowadzone zmiany w danych wystarczy odświeżyć przeglądarkę.

Jak wcześniej wspomniano, w czasach rozwoju technologicznego mnogość dostępnych narzędzi do realizacji projektu jest dość pokaźna. Wykorzystane technologie zostały wytypowane świadomie, ze względu na panujące trendy obecnie na rynku. Wykorzystanie Angulara pomogło uzyskać płynność działania aplikacji. Natomiast jako narzędzie wdrażające użyto Dockera, gdyż posiada dobrze rozbudowaną dokumentację wraz z przykładami. Język Java umożliwił napisanie zoptymalizowanej części serwerowej.

1.2 Przedstawienie celu pracy

Celem pracy jest projekt, implementacja oraz wdrożenie systemu do zarządzania zasobami teleinformatycznymi. System będzie podzielony na dwie główne części - wypożyczanie sprzętu oraz zarządzanie hostami. Projekt oraz implementacja obejmuje zebranie wymagań od grupy docelowej, ustalenie najważniejszych funkcjonalności, wybranie technologii jak i napisanie samej aplikacji. Należy skupić dużą uwagę również na procesie wdrożenia systemu i wybrać takie rozwiązania, dzięki którym proces ten w prosty sposób będzie mógł zostać powtórzony w przyszłości przez grupę docelową.

2. Dokumentacja projektu

2.1. Opis typu aplikacji

2.1.1. Podział ze względu na funkcjonalność

System zarządzania zasobami teleinformatycznymi można uznać jako aplikację oferującą konkretne usługi. Użytkownik systemu może w prosty i przejrzysty sposób zarządzać danymi - dodawać, usuwać oraz edytować zasoby korzystając z interfejsu graficznego. Oprócz tego, administrator może wypożyczać sprzęt interesantom, a także wygenerować kilka rodzajów raportów w formacie PDF.

2.1.2. Podział ze względu na architekturę wraz ze wstępem teoretycznym.

Architektura aplikacji podzielona jest na dwa, główne moduły:

- Część „Front-endowa”, inaczej UI (User Interface). Jest to część systemu, dzięki której użytkownik wchodzi w interakcję z systemem. Moduł ten komunikuje się dzięki protokołowi http z częścią serwerową aplikacji, pozyskuje i wysyła dane

oraz prezentuje je w przystępny sposób użytkownikowi. Do implementacji tej części pracy zostały wykorzystane następujące technologie:

- Typescript – Jest to język programowania, który umożliwia dużo więcej niż sam JavaScript, który w procesie kompilacji transpilowany jest(?) to pliku wynikowego z rozszerzeniem `.js``, dzięki czemu może być bezproblemowo wykorzystywany przez wszystkie przeglądarki, które obsługują JavaScript. Głównymi zaletami i przewagami języka Typescript nad JavaScriptem jest statyczne oraz silne typowanie, dzięki którym wiele pomyłek może być wykrytych już na poziomie kompilacji kodu, do czego mogli już przyzwyczaić się programiści Javy.
- Angular w wersji 8 – JavaScriptowy framework, który korzysta z TypeScripta do tworzenia aplikacji typu Single Page Application, inaczej nazywane dynamicznymi aplikacjami internetowymi. Dzięki temu rozwiązaniu można wyeliminować generowanie statycznych plików HTML po stronie serwera i zastąpić je zestawem kilku plików, które są pobierane przy pierwszym kontakcie użytkownika z aplikacją.

Dalej, wszystkie informacje oraz dane są pobierane asynchronicznie, z wykorzystaniem reaktywnego podejścia, co sprawia wrażenie, że aplikacja działa dużo płynniej.

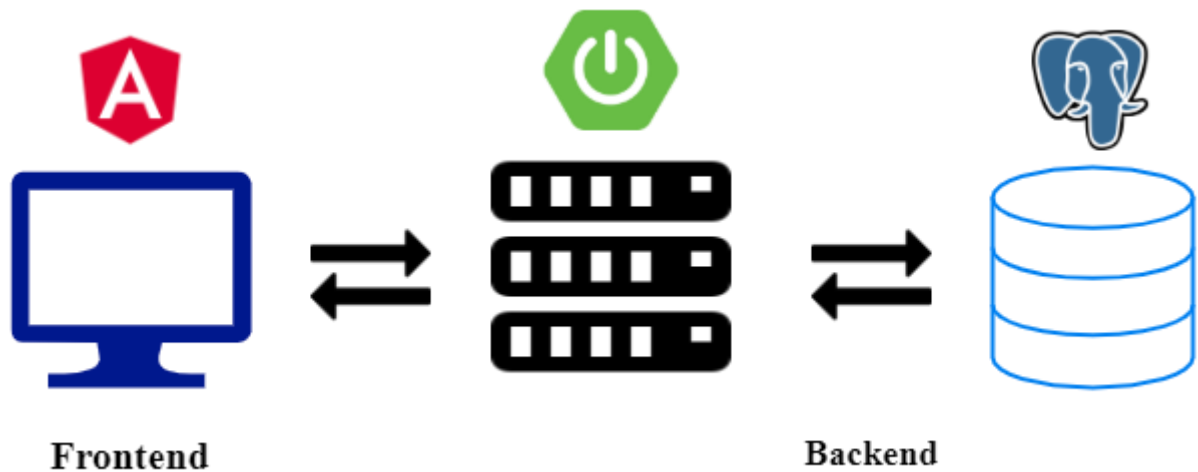
- Angular Material Design – zestaw uzupełnień do frameworka Angular, który udostępnia gotowe komponenty do wykorzystania np. takie jak formularze i tabele.

➤ Część "Back-endowa", inaczej część serwerowa. Moduł ten odpowiada za przetwarzanie oraz udostępnianie danych, które są potrzebne modułowi UI do poprawnego działania. Moduł ten odpowiada również za komunikację z bazą

danych. Do implementacji tej części pracy zostały wykorzystane następujące technologie:

- Java w wersji 11 (początkowo w wersji 8) – język programowania implementujący takie paradygmaty, jak obiektowy oraz funkcyjny, używany najczęściej w serwerowych częściach systemów informatycznych. Według danych portalu Github, drugi najpopularniejszy język programowania.
- Spring Framework – zaawansowany framework łączący zalety programowania deklaratywnego, aspektowego oraz zdarzeniowego, pozwalający tworzyć rozbudowane, webowe (lecz nie tylko) systemy informatyczne. Dodatkowo, w celu usprawnienia procesu konfiguracji, zostało użyte rozwiązanie Spring Boot, które dostarcza domyślne ustawienia aplikacji, które w prosty sposób możemy nadpisać.
- Hibernate – framework, który implementuje standard Java Persistence API oraz pozwala zrealizować warstwę dostępu do relacyjnej bazy danych.
- Postgresql – system do zarządzania relacyjną bazą danych, który pozwala zarządzać danymi przechowywanymi przez aplikację.

Na rysunku nr 1 przedstawiono podział systemu na Back-end oraz Front-end.



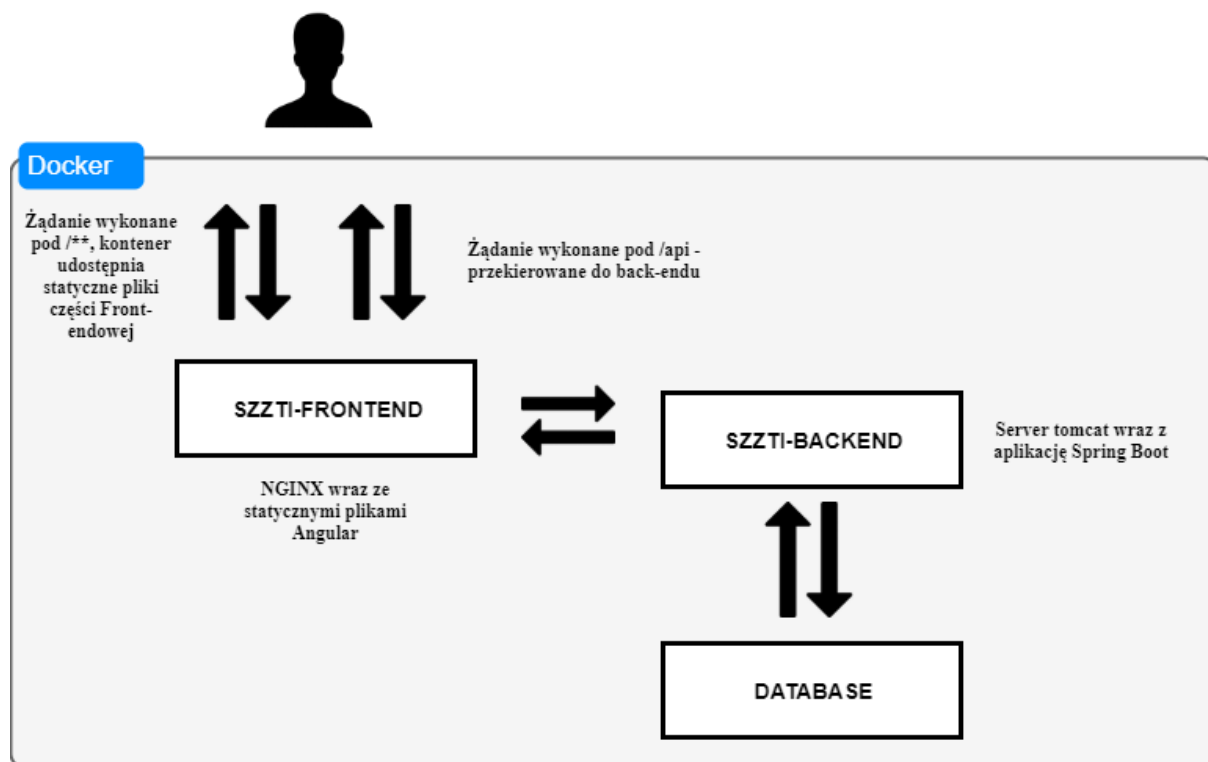
Rysunek 1 Podział na Front-end oraz Back-end

Źródło: Opracowanie własne

2.1.3. Podział ze względu na infrastrukturę wraz z uwzględnieniem sposobu komunikacji

Każda część systemu wymaga do działania pewnej części procesora oraz pamięci ram. Dodatkowo, musi zostać zapewniona komunikacja pomiędzy bazą danych oraz poszczególnymi modułami. Komunikacja pomiędzy modułami odbywa się z wykorzystaniem protokołu http. Wymiana informacji odbywa się z użyciem formatu JSON.

Do połączenia modułów aplikacji oraz bazy danych została wykorzystana platforma Docker. Narzędzie to pozwala stworzyć dedykowane pod własne potrzeby obrazy aplikacji, które możemy uruchomić na każdym systemie, który obsługuje platformę Docker. Poniższy obrazek nr 2 pokazuje wymianę informacji pomiędzy kontenerami.



Rysunek 2 Sposób komunikacji i podział z poziomu infrastruktury

Źródło: Opracowanie własne

Serwer NGINX udostępnia statyczne, skompilowane oraz zminifikowane pliki Front-endowego modułu oraz przekazuje żądania do Back-endowej części aplikacji z wykorzystaniem reverse-proxy. Serwer backendowy to REST-owy webservice, procesujący żądania od klienckiej części aplikacji, dzięki czemu udostępnia albo tworzy nowe zasoby w systemie.

2.2. Koncepcja oraz techniczne aspekty projektu

2.2.1. Cel implementacji oraz grupa docelowa

Celem implementacji Systemu Zarządzania Zasobami Teleinformatycznymi jest ułatwienie oraz scentralizowanie wszelakich informacji o szeroko rozumianych przedmiotach i sprzętach na obszarze Katedry Informatyki Akademii Marynarki Wojennej.

Grupą docelową tego systemu będą technicy katedry, którzy zamienią nieporządek w plikach arkuszowych na rzecz aplikacji webowej, która pomoże zarządzać inwentaryzacją oraz przedmiotami, a także pracownicy katedry, którzy będą mogli sprawdzić, jakie obecnie sprzęty są im wypożyczone.

2.2.2. Wymagania funkcyjne

Technik powinien posiadać konto administratora w systemie, które umożliwia dodawanie, edytowanie usuwanie oraz wyświetlanie wszystkich przedmiotów oraz sprzętów. Technik ma mieć również możliwość dodawania nowych pracowników oraz miejsc, którym dany sprzęt może zostać wypożyczony.

Główną funkcjonalnością systemu ma być wypożyczanie sprzętu. Każdy sprzęt może zostać przypisany do osoby oraz do miejsca. Po wejściu w szczegóły wypożyczenia można je usunąć. System ma umożliwić wyszukiwanie miejsc po nazwie oraz osób po kodach. Po wejściu w szczegóły powinien pojawić się dokładny opis wraz ze przypisanym sprzętem.

Administrator systemu ma mieć możliwość generowania raportów w formacie pdf:

- protokół uszkodzenia sprzętu - specjalny raport możliwy do wygenerowania na stronie ze szczegółami sprzętu.
- protokół wypożyczenia – lista wszystkich przedmiotów przypisana do osoby lub miejsca.

Dodatkowo, system ma umożliwić wyeksportowanie wybranych sprzętów do pliku w formacie .csv.

Kolejnym wymaganiem funkcjonalnym jest możliwość zarządzania hostami. System powinien umożliwić dodawanie, edytowanie oraz usuwanie maszyn, uwzględniając ich kod inwentarzowy oraz miejsce, gdzie się znajdują. Do każdego hosta powinna być możliwość przypisania dowolnej ilości przyłączy sieciowych, które definiują adres IP, VLAN, adres MAC, numer przyłącza i nazwę przyłącza. System ma umożliwić łatwe wyszukiwanie hostów, pozwalając podać jego nazwę, kod inwentarzowy, miejsce gdzie się znajduje, a także pozwala uwzględnić wszystkie pola przyłączenia sieciowego.

2.2.3. Moduły funkcjonalne aplikacji

Moduł logowania:

- Zalogowanie do systemu za pomocą istniejącego konta użytkownika

Moduł sprzętu:

- Dodawanie sprzętu
- Usuwanie sprzętu
- Wyświetlanie listy sprzętów wraz z możliwością filtrowania
- Wyświetlenie szczegółów sprzętu
- Generowanie pliku csv wraz z wybranym sprzętem

Moduł miejsc:

- Dodawanie miejsc oraz ich edytowanie
- Usuwanie miejsc
- Lista miejsc wraz z możliwością filtrowania
- Wyświetlenie szczegółów miejsca.

Moduł osób:

- Wyświetlanie listy osób

- Edycja, dodawanie oraz usuwanie osób
- Wyświetlenie szczegółów osób

Moduł wypożyczeń:

- Dodawanie nowych wypożyczeń
- Usuwanie wypożyczeń
- Informacje o wypożyczeniach

Moduł raportów:

- Generowanie raportu uszkodzenia sprzętu
- Generowanie raportu wypożyczenia sprzętu dla miejsc
- Generowanie raportu wypożyczenia sprzętu dla osób

Moduł hostów:

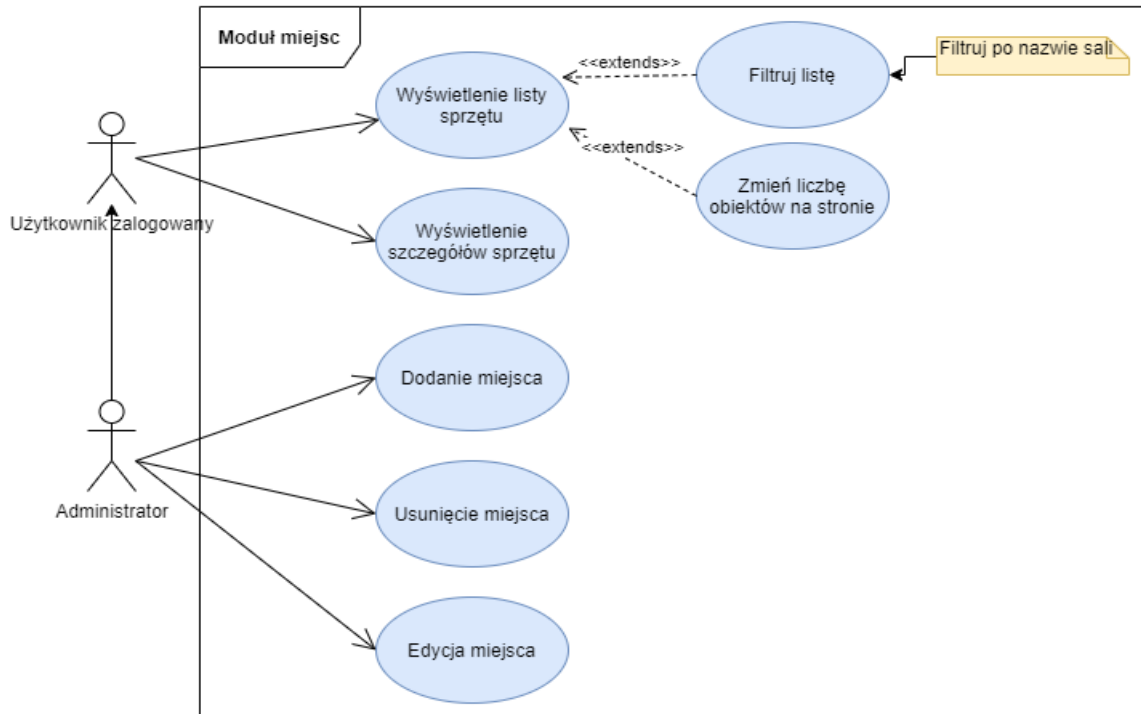
- Dodawanie i edytowanie hostów
- Usuwanie hostów
- Wyświetlanie szczegółów hostów
- Wyświetlanie listy hostów wraz z możliwością filtrowania
- Przypisanie przyłączy sieciowych do hostów.

2.2.4. Spis aktorów

- Użytkownik niezalogowany - brak dostępu do aplikacji, ma jedynie dostęp do strony logowania.
- Użytkownik zalogowany - osoba, która ma uprawnienia do przeglądania zawartości serwisu po zalogowaniu.
- Administrator systemu - osoba, która oprócz uprawnień do przeglądania zawartości serwisu ma również uprawnienia do dodawania, usuwania oraz edycji.

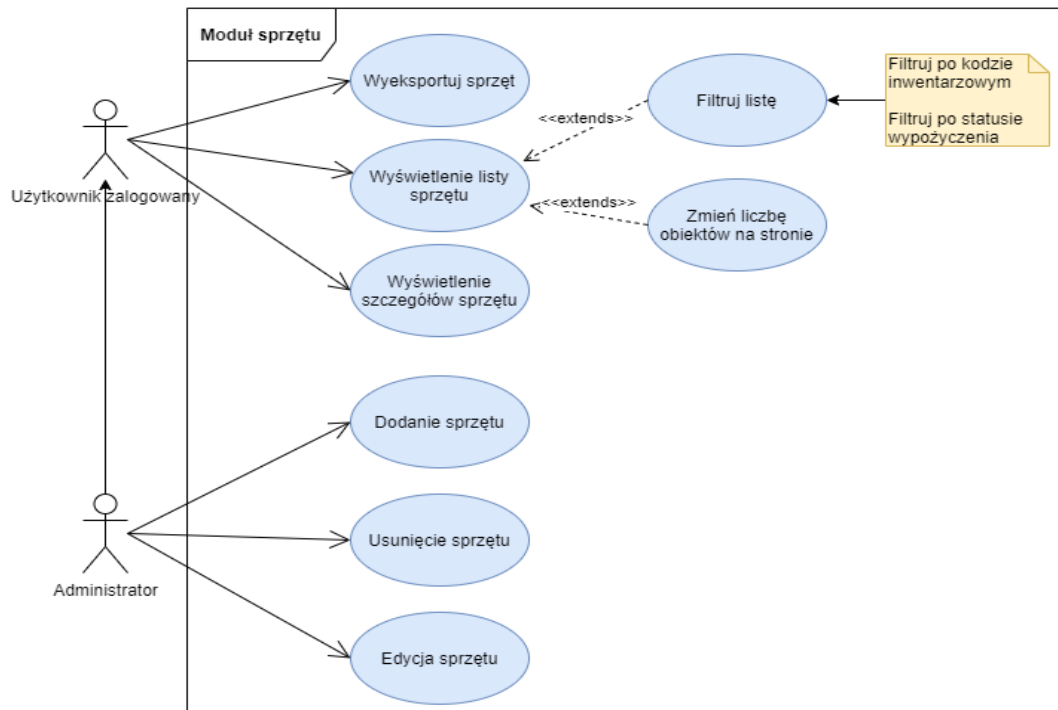
2.2.5. Diagramy przypadków użycia

Na rysunkach od nr 3 do nr 8 przedstawiono diagramy przypadków użycia, które są nieodłączną częścią fazy projektowania systemu informatycznego.



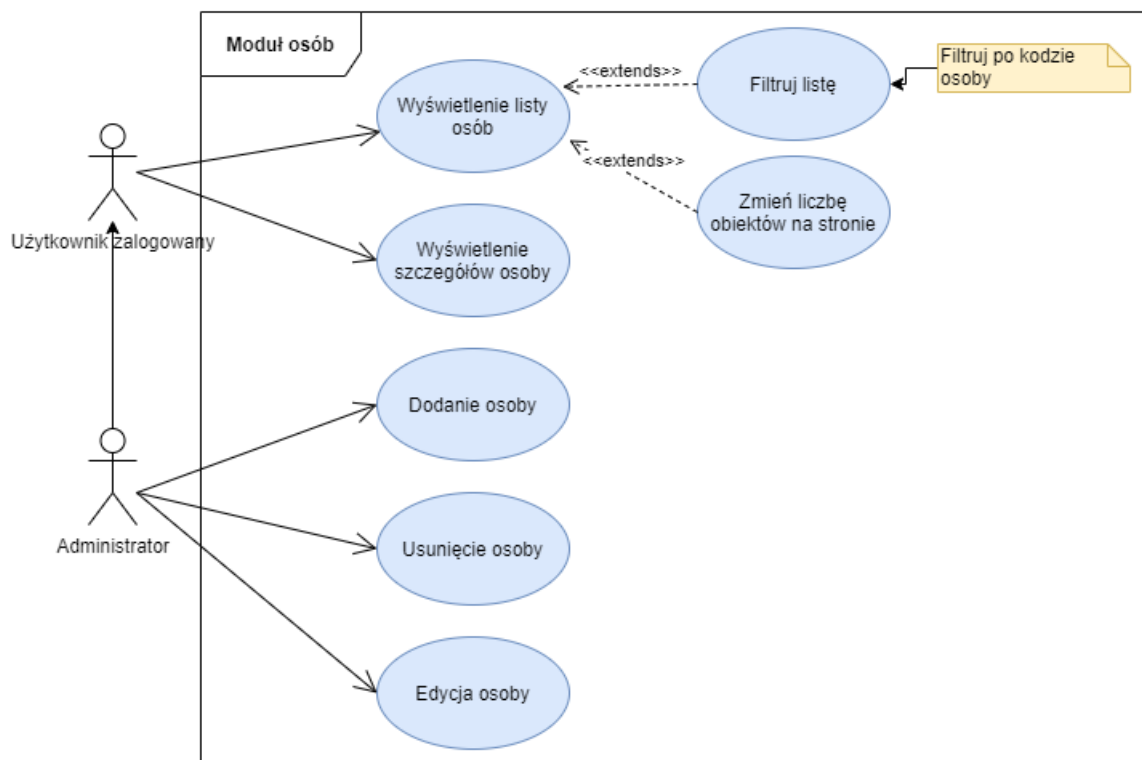
Rysunek 3 Diagram przypadków użycia - Moduł miejsc

Źródło: Opracowanie własne



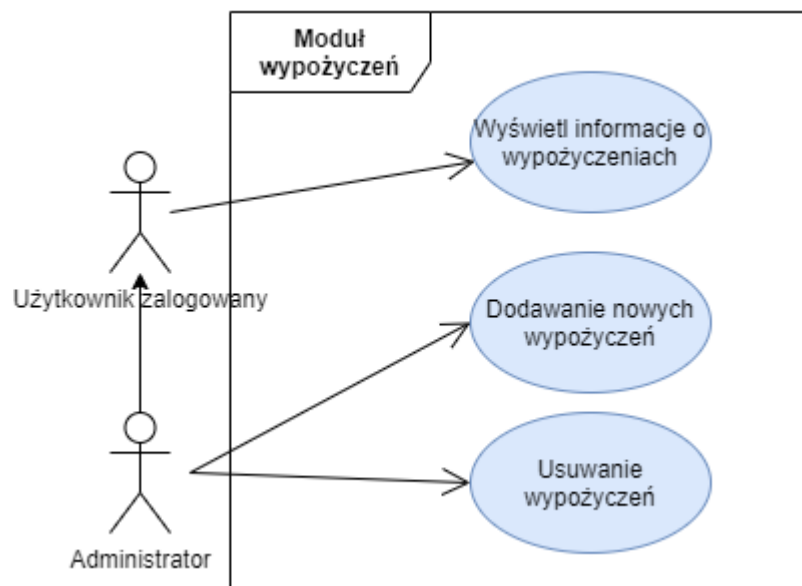
Rysunek 4 Diagram przypadków użycia - Moduł sprzętu

Źródło: Opracowanie własne



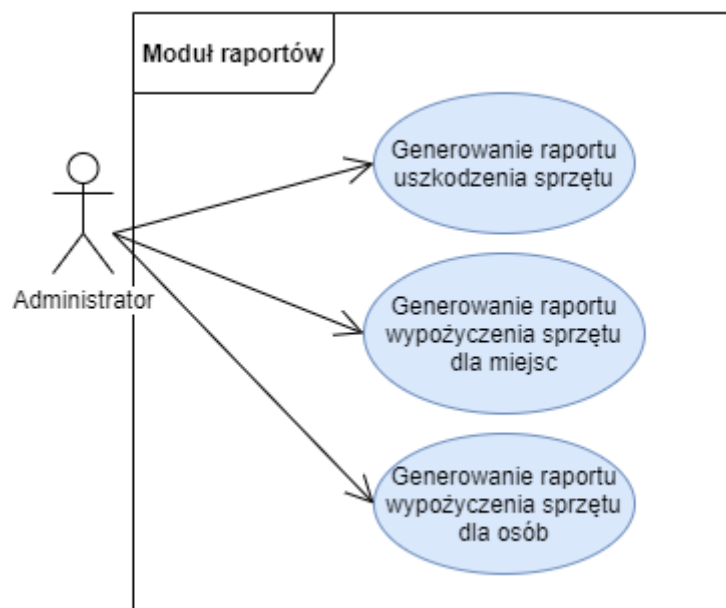
Rysunek 5 Diagram przypadków użycia - Moduł osób

Źródło: Opracowanie własne



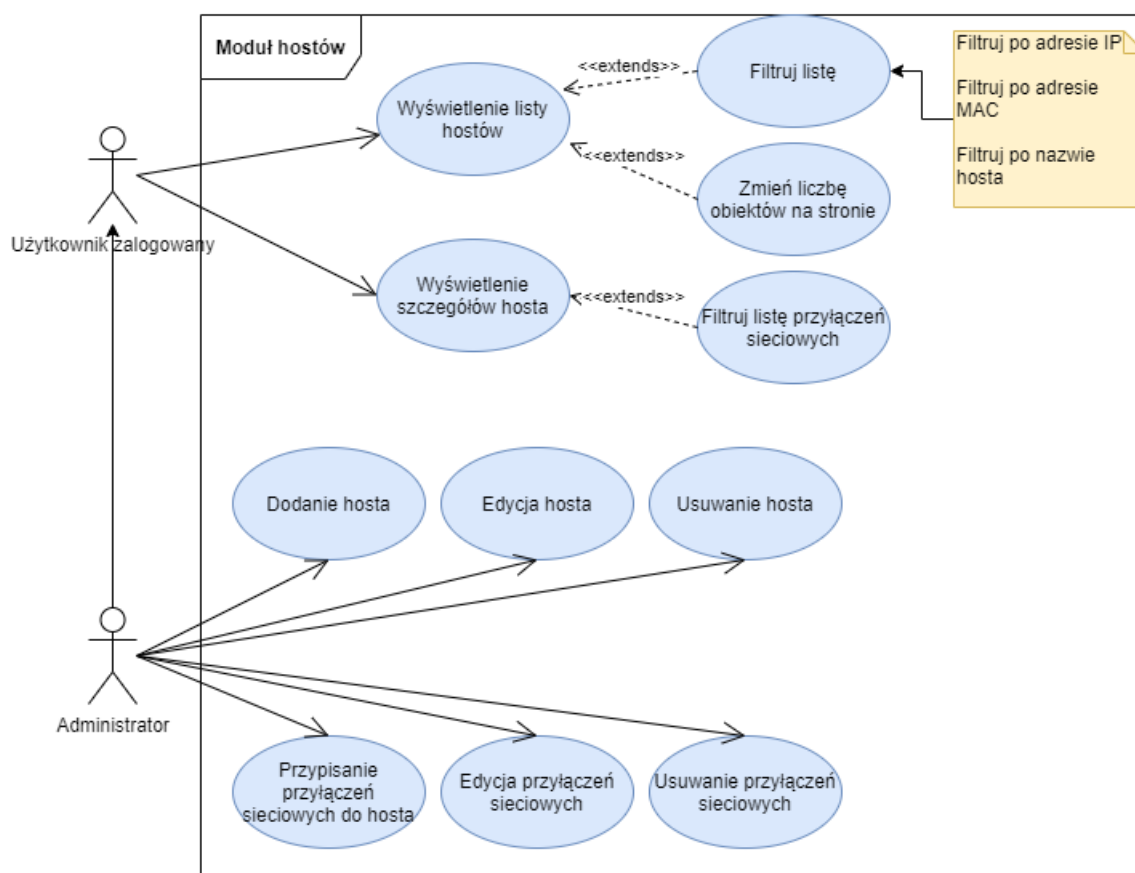
Rysunek 6 Diagram przypadków użycia - Moduł wypożyczeń

Źródło: Opracowanie własne



Rysunek 7 Diagram przypadków użycia - Moduł raportów

Źródło: Opracowanie własne



Rysunek 8 Diagram przypadków użycia - Moduł hostów

Źródło: Opracowanie własne

2.2.6. Przebieg scenariuszy

Scenariusze przedstawione w tabelach od nr 1 do nr 27 są uzupełnieniem do diagramów przypadków użycia.

Tabela 1 Wyświetlenie listy sprzętu

Nazwa UC	Wyświetlenie listy sprzętu
Skrócony opis UC	Zalogowany użytkownik ma możliwość sprawdzenia listy z całym dostępnym i zarejestrowanym sprzętem
Aktor/Aktorzy	Użytkownik zalogowany
Warunki początkowe	Użytkownik musi być zalogowany
Przebieg	<ol style="list-style-type: none">1. Użytkownik klika na "Lista przedmiotów" dostępne w górnej nawigacji2. Wybiera interesujące go filtry3. Wyświetlenie listy przedmiotów podzielonych na strony, domyślnie po 10 elementów na każdą
Przebiegi alternatywne	<i>"Brak przedmiotów o wybranych filtrach"</i> <ol style="list-style-type: none">3. Wyświetlenie pustej tabeli.
Warunki końcowe	Użytkownik widzi przefiltrowaną listę przedmiotów

Źródło: Opracowanie własne

Tabela 2 Dodanie sprzętu

Nazwa UC	Dodanie sprzętu
Skrócony opis UC	Administrator systemu ma możliwość dodania nowego sprzętu
Aktor/Aktorzy	Administrator systemu.
Warunki początkowe	Użytkownik jest zalogowany do systemu z uprawnieniami administratora.
Przebieg	<ol style="list-style-type: none">1. Administrator przechodzi na podstronę "Dodaj przedmiot"2. Uzupełnia przygotowany formularz.3. Klika przycisk zapisz.
Przebiegi	<i>"Niepoprawnie wprowadzone dane"</i>

alternatywne	3. Nieprawidłowo wypełnione pole zostanie podświetlone na czerwono
Warunki końcowe	Użytkownik dodał pożądaną liczbę przedmiotów oraz został przekierowany na podstronę z listę sprzętu.

Źródło: Opracowanie własne

Tabela 3 Logowanie

Nazwa UC	Logowanie
Skrócony opis UC	Niezałogowany użytkownik ma możliwość zalogowania się do systemu
Aktor/Aktorzy	Użytkownik niezałogowany
Warunki początkowe	Brak
Przebieg	<ol style="list-style-type: none"> 1. Niezałogowany użytkownik klika na opcję Zaloguj w górnej nawigacji 2. Wpisuje login i hasło w formularzu i klika zaloguj.. 3. Zostaje przekierowany na stronę główną.
Przebiegi alternatywne	<p><i>"Nieprawidłowy login bądź hasło"</i></p> <ol style="list-style-type: none"> 3. Wyświetlenie komunikatu o błędzie
Warunki końcowe	Użytkownik ma większy dostęp do systemu, zamiast przycisku Zaloguj pojawia się wyloguj.

Źródło: Opracowanie własne

Tabela 4 Edycja sprzętu

Nazwa UC	Edycja sprzętu
Skrócony opis UC	Administrator systemu ma możliwość edycji sprzętu
Aktor/Aktorzy	Administrator systemu
Warunki początkowe	Użytkownik zalogowany do serwisu z uprawnieniami do edytowania przedmiotów. Sprzęt musi istnieć w bazie.
Przebieg	<ol style="list-style-type: none"> 1. Użytkownik przechodzi na podstronę "Lista przedmiotów" 2. Użytkownik klika na ikonkę długopisu.

	3. Edytuje wybrane pola w formularzu i naciska zatwierdź.
Przebiegi alternatywne	<i>"Brak uprawnień do edytowania przedmiotów"</i> 3. Wyświetlenie komunikatu o błędzie.
Warunki końcowe	Przedmiot zostaje edytowany i zapisany do bazy danych.

Źródło: Opracowanie własne

Tabela 5 Usunięcie sprzętu

Nazwa UC	Usunięcie sprzętu
Skrócony opis UC	Zalogowany użytkownik ma możliwość sprawdzenia szczegółów wybranego sprzętu.
Aktor/Aktorzy	Administrator systemu
Warunki początkowe	- Użytkownik zalogowany do serwisu z uprawnieniami do usuwania przedmiotów
Przebieg	1. Użytkownik przechodzi na podstronę "Lista przedmiotów" 2. Użytkownik klika na ikonkę śmietnika. 3. Użytkownik zatwierdza przycisk naciskając Tak na modalu.
Przebiegi alternatywne	<i>"Brak uprawnień do usuwania przedmiotów"</i> 3. Wyświetlenie komunikatu o błędzie <i>"Użytkownik klika nie w okienku"</i> 3. Przedmiot nie zostaje usunięty, ponownie pojawia się tabela z przedmiotami.
Warunki końcowe	Przedmiot zostaje usunięty, wypożyczenie powiązane z tym przedmiotem zostają usunięte, tabela zostaje przeładowana.

Źródło: Opracowanie własne

Tabela 6 Wyświetl szczegóły sprzętu

Nazwa UC	Wyświetl szczegóły sprzętu
Skrócony opis UC	Zalogowany użytkownik ma możliwość sprawdzenia szczegółów wybranego sprzętu.

Aktor/Aktorzy	Zalogowany użytkownik
Warunki początkowe	Użytkownik musi być zalogowany.
Przebieg	<ol style="list-style-type: none"> 1. Użytkownik przechodzi na podstronę "Lista przedmiotów" 2. Użytkownik wybiera ikonę szczegóły 3. Pojawia się strona ze szczegółami przedmiotu i informacją o wypożyczeniu.
Przebiegi alternatywne	Brak.
Warunki końcowe	Użytkownik ma możliwość przejścia do innych podstron.

Źródło: Opracowanie własne

Tabela 7 Wyeksportuj sprzęt

Nazwa UC	Wyeksportuj sprzęt
Skrócony opis UC	Zalogowany użytkownik ma możliwość wyeksportowania zaznaczonych sprzętów do pliku w formacie .csv
Aktor/Aktorzy	Zalogowany użytkownik
Warunki początkowe	Użytkownik musi być zalogowany
Przebieg	<ol style="list-style-type: none"> 1. Użytkownik przechodzi na podstronę „Lista sprzętu” 2. Użytkownik wybiera produkty za pomocą zaznaczone znajdującego się po lewej stronie. 3. Użytkownik klika przycisk Eksportuj zaznaczone
Przebiegi alternatywne	<p>“Użytkownik nie wybrał żadnego sprzętu”</p> <p>3. Pojawia się komunikat o niewybraniu żadnego sprzętu do wyeksportowania.</p>
Warunki końcowe	W przeglądarce użytkownika pojawia się plik z rozszerzeniem .csv zawierający wybrany sprzęt

Tabela 8 Wyświetlenie listy osób

Nazwa UC	Wyświetlenie listy osób
Skrócony opis UC	Zalogowany użytkownik ma możliwość sprawdzenia listy ze wszystkimi osobami zapisanymi w systemie, którym można wypożyczyć sprzęt.
Aktor/Aktorzy	Użytkownik zalogowany
Warunki początkowe	Użytkownik musi być zalogowany
Przebieg	<ol style="list-style-type: none"> 1. Użytkownik klika na "Lista osób" dostępne w górnej nawigacji 2. Wybiera interesujące go filtry 3. Wyświetlenie listy osób podzielonych na strony, domyślnie po 10 elementów na każdą
Przebiegi alternatywne	<p><i>"Brak przedmiotów o wybranych filtrach"</i></p> <ol style="list-style-type: none"> 3. Wyświetlenie pustej tabeli.
Warunki końcowe	Użytkownik widzi przefiltrowaną listę osób

Źródło: Opracowanie własne

Tabela 9 Dodanie osoby

Nazwa UC	Dodanie osoby
Skrócony opis UC	Administrator systemu ma możliwość dodania nowej osoby, której będzie można przypisać sprzęt
Aktor/Aktorzy	Administrator
Warunki początkowe	Użytkownik musi być zalogowany i posiadać uprawnienia do dodawania osób
Przebieg	<ol style="list-style-type: none"> 1. Użytkownik wybiera ikonę plusika na stronie z listą osób 2. Wypełnia formularz oraz klika zatwierdź. 3. Użytkownik otrzymuje informacje o poprawnym dodaniu osoby.
Przebiegi	<i>"Kod osoby już istnieje"</i>

alternatywne	3. Zostaje wyświetlony komunikat o duplikacji kodu osoby.
Warunki końcowe	Nowa osoba została pomyślnie dodana do systemu, system przekierowuje użytkownika na listę wszystkich osób.

Źródło: Opracowanie własne

Tabela 10 Edycja osoby

Nazwa UC	Edycja osoby
Skrócony opis UC	Administrator systemu ma możliwość edytowania wybranej osoby
Aktor/Aktorzy	Administrator systemu
Warunki początkowe	Użytkownik musi być zalogowany i posiadać uprawnienia do edycji osób.
Przebieg	<ol style="list-style-type: none"> 1. Użytkownik wybiera ikonę plusika na stronie z listą osób 2. Wypełnia formularz oraz klika zatwierdź. 3. Użytkownik otrzymuje informacje o poprawnym zapisaniu edytowanej osoby.
Przebiegi alternatywne	<p>“Kod osoby już istnieje”</p> <p>3. Zostaje wyświetlony komunikat o duplikacji kodu osoby.</p>
Warunki końcowe	Nowa osoba została pomyślnie dodana do systemu, system przekieruje użytkownika na listę wszystkich osób.

Źródło: Opracowanie własne

Tabela 11 Wyświetlenie szczegółów osoby

Nazwa UC	Wyświetlenie szczegółów osoby
Skrócony opis UC	Zalogowany użytkownik ma możliwość sprawdzenia dodatkowych informacji o osobie, a także sprawdzenia listy wszystkich przedmiotów do niej przypisanych.
Aktor/Aktorzy	Zalogowany użytkownik
Warunki początkowe	Użytkownik musi być zalogowany.

Przebieg	<ol style="list-style-type: none"> 1. Użytkownik wybiera ikonę szczegóły na liście osób przy interesującym go rekordzie. 2. Wyświetlona zostaje strona ze szczegółami osoby wraz z listą sprzętu do niej przypisanych.
Przebiegi alternatywne	<p>“Brak sprzętu przypisanego do danej osoby”</p> <ol style="list-style-type: none"> 2. Pojawia się komunikat o braku sprzętu przypisanych do wybranej osoby
Warunki końcowe	Użytkownik uzyskuje informacje o wypożyczeniach.

Źródło: Opracowanie własne

Tabela 12 Wyświetlenie szczegółów miejsca

Nazwa UC	Wyświetlenie szczegółów miejsca
Skrócony opis UC	Zalogowany użytkownik ma możliwość sprawdzenia szczegółów miejsca wraz z listą sprzętu tam wypożyczonymi
Aktor/Aktorzy	Zalogowany użytkownik
Warunki początkowe	Użytkownik musi być zalogowany
Przebieg	<ol style="list-style-type: none"> 1. Użytkownik wybiera ikonę szczegóły na liście miejsc przy interesującym go rekordzie. 2. System wyświetla informacje o miejscu wraz z listą sprzętu tu wypożyczoną.
Przebiegi alternatywne	<p>“Brak sprzętu przypisanego do danego miejsca”</p> <ol style="list-style-type: none"> 2. Pojawia się komunikat o braku sprzętu przypisanych do wybranego miejsca
Warunki końcowe	Użytkownik uzyskuje informacje o wypożyczeniach dla miejsca.

Źródło: Opracowanie własne

Tabela 13 Dodanie miejsca

Nazwa UC	Dodanie miejsca
Skrócony opis UC	Administrator systemu ma możliwość dodania nowego miejsca, do którego będzie można przypisać sprzęt
Aktor/Aktorzy	Administrator
Warunki początkowe	Użytkownik musi być zalogowany i posiadać uprawnienia do dodawania miejsc
Przebieg	<ol style="list-style-type: none"> 1. Użytkownik wybiera ikonę plusika na stronie z listą miejsc 2. Wypełnia formularz oraz klika zatwierdź. 3. Użytkownik otrzymuje informacje o poprawnym dodaniu miejsca.
Przebiegi alternatywne	<p>“Nazwa miejsca jest już zajęta”</p> <ol style="list-style-type: none"> 3. Zostaje wyświetlony komunikat o duplikacji nazwy miejsca.
Warunki końcowe	Nowe miejsce zostało pomyślnie dodane do systemu, system przekieruje użytkownika na listę wszystkich miejsc

Źródło: Opracowanie własne

Tabela 14 Edycja miejsca

Nazwa UC	Edycja miejsca
Skrócony opis UC	Administrator systemu ma możliwość edytowania wybranego miejsca
Aktor/Aktorzy	Administrator systemu
Warunki początkowe	Użytkownik musi być zalogowany i posiadać uprawnienia do edycji miejsc.
Przebieg	<ol style="list-style-type: none"> 1. Użytkownik wybiera ikonę długopisu na stronie z listą miejsc 2. Wypełnia formularz oraz klika zatwierdź. 3. Użytkownik otrzymuje informacje o poprawnym zapisaniu edytowanego miejsca.

Przebiegi alternatywne	"Nazwa osoby jest już zajęta" 3. Zostaje wyświetlony komunikat o duplikacji nazwy miejsca.
Warunki końcowe	Miejsca zostało zaktualizowane w systemie, system przekieruje użytkownika na listę wszystkich miejsc.

Źródło: Opracowanie własne

Tabela 15 Generowanie raportu uszkodzenia sprzętu

Nazwa UC	Generowanie raportu uszkodzenia sprzętu.
Skrócony opis UC	Użytkownik ma możliwość wygenerowania raportu zniszczenia sprzętu.
Aktor/Aktorzy	Zalogowany użytkownik
Warunki początkowe	Użytkownik musi być zalogowany.
Przebieg	1. Użytkownik przechodzi w szczegóły wybranego przedmiotu 2. Użytkownik klika przycisk generuj protokół zniszczenia sprzętu
Przebiegi alternatywne	Brak.
Warunki końcowe	Użytkownik otrzymuje wygenerowany raport o zniszczeniu sprzętu w formacie pdf.

Źródło: Opracowanie własne

Tabela 16 Generowanie raportu wypożyczenia dla osoby

Nazwa UC	Generowanie raportu wypożyczenia dla osoby
Skrócony opis UC	Użytkownik ma możliwość stworzenia listy przypisanych do siebie przedmiotów w formacie pdf.
Aktor/Aktorzy	Zalogowany użytkownik
Warunki początkowe	Użytkownik musi być zalogowany
Przebieg	1. Użytkownik przechodzi w szczegóły wybranej osoby

	2. Pod listą wypożyczonych przedmiotów klika przycisk Generuj raport dla osoby
Przebiegi alternatywne	“Brak przedmiotów przypisanych do osoby” 2. Brak przycisku do wygenerowania raportu przez co nie ma możliwości wygenerowania raportu.
Warunki końcowe	Użytkownik otrzymuje wygenerowany raport z listą wypożyczonych przedmiotów.

Źródło: Opracowanie własne

Tabela 17 Dodawanie nowych wypożyczeń

Nazwa UC	Dodawanie nowych wypożyczeń
Skrócony opis UC	Administrator systemu ma możliwość przypisania sprzętu do miejsca oraz do osoby.
Aktor/Aktorzy	Administrator systemu
Warunki początkowe	Użytkownik musi być zalogowany jako administrator.
Przebieg	1. Administrator przechodzi na stronę z listą sprzętu 2. Wybiera interesujące go przedmioty poprzez zaznaczenie kwadraciku po lewej stronie. 3. Klika przycisk Wypożycz zaznaczone 4. Na nowej stronie wybiera kolejno miejsce wypożyczenia oraz osobę. 5. Klika przycisk zapisz.
Przebiegi alternatywne	“Wybrano już wypożyczony sprzęt” 3. Przycisk wypożycz zaznaczone zostaje szary, pojawia się komunikat informujący o błędzie. “Nie wybrano ani miejsca ani osoby” 5. Pojawia się komunikat o błędzie, informujący o konieczności wybrania osoby lub miejsca.
Warunki końcowe	Wypożyczenie zostało przypisane, użytkownik został przekierowany na stronę ze szczegółami wypożyczenia.

Źródło: Opracowanie własne

Tabela 18 Usuwanie wypożyczeń

Nazwa UC	Usuwanie wypożyczeń
Skrócony opis UC	Administrator ma możliwość odpisania sprzętu od miejsca czy osoby.
Aktor/Aktorzy	Administrator systemu
Warunki początkowe	Użytkownik musi być zalogowany jako administrator
Przebieg	<ol style="list-style-type: none"> 1. Administrator przechodzi na stronę ze szczegółami miejsca lub osoby. 2. Na liście przypisanego sprzętu klika ikonę śmietnika przy interesującym go sprzęcie. 3. Wciska tak przy wyświetlonym modalu.
Przebiegi alternatywne	<p>“Użytkownik klika nie w okienku”</p> <p>Wypożyczenie nie zostaje usunięte, ponownie pojawia się strona ze szczegółami wypożyczenia.</p>
Warunki końcowe	Sprzęt dostaje odpisany, lista jest przeładowywana.

Źródło: Opracowanie własne

Tabela 19 Wyświetl informacje o wypożyczeniach

Nazwa UC	Wyświetl informacje o wypożyczeniach
Skrócony opis UC	Użytkownik ma możliwość sprawdzenia listy przypisanego sprzętu do miejsca lub osoby.
Aktor/Aktorzy	Zalogowany użytkownik
Warunki początkowe	Użytkownik musi być zalogowany.
Przebieg	<ol style="list-style-type: none"> 1. Użytkownik przechodzi na szczegóły wybranego miejsca lub osoby. 2. W dolnej części ekranu pojawia się paginowana tabela z przypisanym sprzętem.
Przebiegi alternatywne	<p>“Brak sprzętu przypisanego do wybranej osoby lub miejsca”</p> <p>2. Pojawia się napis Brak wypożyczeń</p>

Warunki końcowe	Użytkownik uzyskuje informacje o statusie wypożyczeń.
-----------------	---

Źródło: Opracowanie własne

Tabela 20 Dodanie hosta

Nazwa UC	Dodanie hosta
Skrócony opis UC	Administrator systemu ma możliwość dodania nowego hosta do systemu.
Aktor/Aktorzy	Administrator systemu
Warunki początkowe	Użytkownik musi być zalogowany jako administrator.
Przebieg	<ol style="list-style-type: none"> 1. Administrator przechodzi na stronę Lista hostów 2. Wciska plusik na górze tabeli 3. Wypełnia formularz i wciska zatwierdź.
Przebiegi alternatywne	<p>“Nazwa hosta jest już zajęta”</p> <ol style="list-style-type: none"> 3. Zostaje wyświetlony komunikat o duplikacji nazwy hosta.
Warunki końcowe	Host zostaje dodany, użytkownik zostaje przekierowany na listę hostów.

Źródło: Opracowanie własne

Tabela 21 Edycja hosta

Nazwa UC	Edycja hosta
Skrócony opis UC	Administrator ma możliwość edytowania wcześniej stworzonego hosta.
Aktor/Aktorzy	Administrator systemu.
Warunki początkowe	Użytkownik musi być zalogowany jako administrator.
Przebieg	<ol style="list-style-type: none"> 1. Administrator przechodzi na stronę Lista hostów 2. Wciska ikonkę długopisu po prawej stronie 3. Wypełnia formularz i wciska zatwierdź
Przebiegi alternatywne	<p>“Nazwa hosta jest już zajęta”</p> <ol style="list-style-type: none"> 3. Zostaje wyświetlony komunikat o duplikacji nazwy hosta.

Warunki końcowe	Host zostaje edytowany, użytkownik zostaje przekierowany na listę hostów.
-----------------	---

Źródło: Opracowanie własne

Tabela 22 Usuwanie hosta

Nazwa UC	Usuwanie hosta
Skrócony opis UC	Administrator ma możliwość usunięcia wcześniej stworzonego hosta
Aktor/Aktorzy	Administrator systemu.
Warunki początkowe	Użytkownik musi być zalogowany jako administrator.
Przebieg	<ol style="list-style-type: none"> 1. Administrator przechodzi na stronę Lista hostów 2. Wciska ikonę śmietnika po prawej stronie 3. W pojawiającym się komunikacie potwierdzającym klika Tak.
Przebiegi alternatywne	<p>“Użytkownik klika nie w okienku”</p> <p>Host nie zostaje usunięty, ponownie pojawia się tabela z hostami.</p>
Warunki końcowe	Host zostaje usunięty z bazy, wraz z nim usunięte zostają wszystkie przyłączenia sieciowe.

Źródło: Opracowanie własne

Tabela 23 Wyświetlenie szczegółów hosta

Nazwa UC	Wyświetlanie szczegółów hosta
Skrócony opis UC	Użytkownik ma możliwość sprawdzenia szczegółów hosta wraz z listą przypisanych do niego przyłączy.
Aktor/Aktorzy	Zalogowany użytkownik
Warunki początkowe	Użytkownik musi być zalogowany do systemu.
Przebieg	<ol style="list-style-type: none"> 1. Użytkownik przechodzi na stronę Lista hostów 2. Klika ikonę szczegółów przy interesującym go

	rekordzie 3. Pojawia się podstrona ze szczegółami hosta oraz tabela z przyłączeniami sieciowymi.
Przebiegi alternatywne	“Brak przyłączy sieciowych przypisanych do hosta” 3. Pojawia się pusta tabela.
Warunki końcowe	Użytkownik uzyskuje informacje o wybranym hoście.

Źródło: Opracowanie własne

Tabela 24 Wyświetlenie listy hostów

Nazwa UC	Wyświetlanie listy hostów
Skrócony opis UC	Użytkownik ma możliwość wyświetlenia listy hostów, a także dowolnego filtrowania tejże listy.
Aktor/Aktorzy	Zalogowany użytkownik
Warunki początkowe	Użytkownik musi być zalogowany
Przebieg	1. Użytkownik przechodzi na stronę Lista hostów
Przebiegi alternatywne	Brak.
Warunki końcowe	Użytkownik uzyskuje podgląd na listę wszystkich hostów.

Źródło: Opracowanie własne

Tabela 25 Przypisanie przyłączy sieciowych do hostów

Nazwa UC	Przypisanie przyłączy sieciowych do hostów
Skrócony opis UC	Administrator systemu ma możliwość przypisania przyłączenia sieciowego do hosta.
Aktor/Aktorzy	Administrator systemu.
Warunki początkowe	Użytkownik musi być zalogowany do serwisu i posiadać uprawnienia administratora.
Przebieg	1. Administrator przechodzi na listę hostów 2. Administrator wybiera szczegóły przy interesującym

	<p>go rekordzie w tabeli.</p> <p>3. Klika ikonę plusika na stronie ze szczegółami.</p> <p>4. Wypełnia formularz i wciska zatwierdź.</p>
Przebiegi alternatywne	<p>“Użytkownik niepoprawnie wypełnił formularz”</p> <p>Wyświetlony zostaje odpowiedni komunikat o błędzie, pola zawierające błędy podświetlone zostają na czerwono.</p>
Warunki końcowe	<p>Przyłącze sieciowe pomyślnie zostaje zapisane, użytkownik zostaje przekierowany na szczegóły hosta, do którego zostało dodane przyłącze.</p>

Źródło: Opracowanie własne

Tabela 26 Usunięcie przyłączenia sieciowego

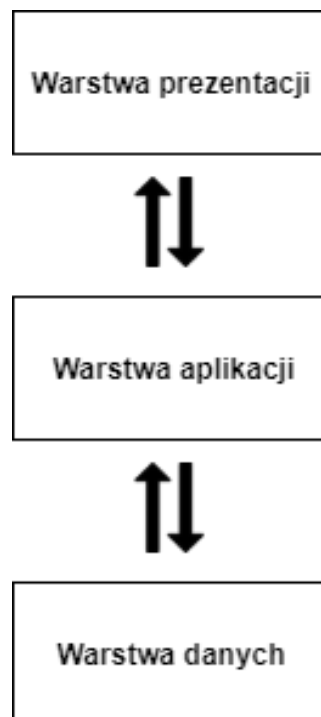
Nazwa UC	Usunięcie przyłączenia sieciowego
Skrócony opis UC	Administrator systemu ma możliwość usunięcia przyłączenia sieciowego.
Aktor/Aktorzy	Administrator systemu.
Warunki początkowe	Użytkownik musi być zalogowany do serwisu i posiadać uprawnienia administratora.
Przebieg	<ol style="list-style-type: none"> 1. Administrator przechodzi na listę hostów 2. Administrator wybiera szczegóły przy interesującym go rekordzie w tabeli. 3. Klika ikonę śmietnika przy interesującym go rekordzie z przyłączeniami sieciowymi 4. Klika tak w modalu potwierdzającym.
Przebiegi alternatywne	<p>“Użytkownik klika nie w okienku”</p> <p>Przyłącze nie zostaje usunięte, ponownie pojawia się podstrona z szczegółami hosta.</p>
Warunki końcowe	Przyłącze zostaje odpisane od hosta oraz usunięte.

2.2.7. Platforma docelowa

Aplikacja zostanie uruchomiona na sprzęcie SYNOLOGY. Częścią pracy było zdefiniowanie oraz wybranie odpowiedniego sposobu wdrożenia aplikacji. Zdecydowano się na wdrożenie wykorzystujące platformę Docker, dzięki czemu na serwerze wymagana jest jedynie zainstalowana ów platforma. Opis procesu wdrożenia zawarto w podrozdziale 2.7.4

2.3. Struktura Aplikacji

Architektura trójwarstwowa, ukazana na rysunku nr 9, to typ architektury klient-serwer, w której warstwa prezentacji, warstwa aplikacji oraz zarządzanie danymi są od siebie oddzielone. Podział ten pozwala stworzyć odrębne części systemu, gdzie częścią wspólną są jedynie medium oraz sposób komunikacji.



Rysunek 9 Architektura trójwarstwowa

Źródło: Opracowanie własne

2.3.1. Warstwa aplikacji – część Backendowa



Rysunek 10 Podział warstwy aplikacji

Źródło: „Petri Kainulainen Blog” [Online] Available:

<https://www.petrikainulainen.net/software-development/design/understanding-spring-web-application-architecture-the-classic-way/> [Data uzyskania dostępu: 13.01.2020]

Warstwa aplikacji przedstawiona została na rysunku 10. Odpowiada ona za przetwarzanie żądań użytkowników i jest swego rodzaju sercem systemu, gdyż to ona zajmuje się przekazywaniem danych pomiędzy warstwami.

2.3.1.1. Podział warstwy internetowej

Każdy kontroler wywołuje metody z odpowiadającego mu serwisu. Dodatkowo, w tej warstwie znajdują się klasy odpowiedzialne za obsługę błędów oraz zabezpieczenia części serwerowej aplikacji.

❖ Kontrolery

Każdy kontroler to komponent, który obsługuje żądania HTTP. Spring deleguje szczegóły żądania do odpowiednich klas oznaczonych adnotacjami `@Controller` oraz `@RequestMapping`. W tabeli nr 27 zostały opisane kontrolery zaimplementowane w systemie.

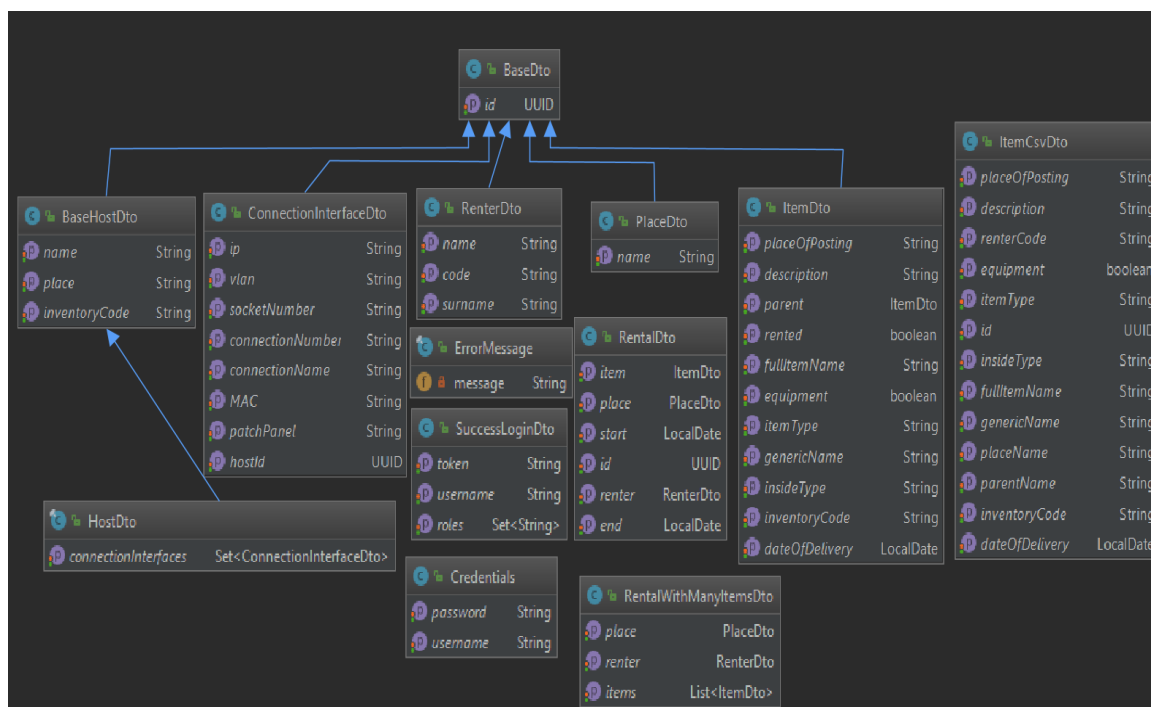
Tabela 27 Opis kontrolerów

ConnectionInterfaceController	Zawiera metody odpowiedzialne za przetwarzanie żądań tworzenia, usuwania i edytowania adresami przypisanymi do hostów.
CsvExportController	Klasa ta odpowiedzialna jest za generowanie plików csv dla obiektów domenowych.
HostController	Klasa procesująca żądania związane z hostami.
ItemController	W tej klasie umieszczone są metody związane z sprzętem. Oprócz tworzenia i usuwania, klasa ta umożliwia wyszukiwanie sprzętu po nazwie oraz kategorii, udostępniając dane w paginowanej formie.
PlaceController	Klasa procesująca żądania związane z miejscami, do których możemy przypisać sprzęt. Klasa ta zawiera metodę do wyszukiwania miejsca po nazwie.
RentalController	W tej klasie odbywa się zarządzanie wszystkimi wypożyczeniami. Dodatkowo, znajduje się tutaj metoda, która pozwala wyszukać wszystkie wypożyczenia przypisane do konkretnej osoby lub miejsca.
RenterController.	Klasa pozwalająca zarządzać osobami, do których można przypisać sprzęt.
ReportController	Klasa odpowiedzialna za generowanie raportów z przygotowanych wcześniej wzorów. Raporty generowane są w formacie PDF, a typ raportu zależy od wywołanej metody.

Źródło: Opracowanie własne

❖ DTO

Data transfer objects to obiekty, proste POJO, które są używane do enkapsulacji danych z obiektów domenowych. To te obiekty są udostępniane warstwie prezentacji (UI-owi), a także to ten rodzaj obiektów jest odbierany przez każdy kontroler w przypadku tworzenia czy aktualizowania zasobu. Poniższy rysunek nr 11 przedstawia diagram klas DTO.



Rysunek 11 Diagram klas obiektów DTO

Źródło: Opracowanie własne

❖ Walidatory

Dla każdej klasy DTO istnieje odpowiadająca mu klasa walidatora, która sprawdza poprawność i spójność danych na poziomie POJO – poprawność ID, brak pustych pól, poprawne typy danych. Na rysunku 12 przedstawiono implementację jednego z walidatorów.

```

14 @Service
15 @AllArgsConstructor
16 public class RentalDtoValidator {
17
18     private final RentalRepository rentalRepository;
19
20     public void validateRental(RentalDto rentalDto) {
21         validatePlaceAndRenter(rentalDto.getPlace(), rentalDto.getRenter());
22         validateItem(rentalDto.getItem());
23     }
24
25     public void validateRentalWithManyItems(RentalWithManyItemsDto rentalDto) {
26         validatePlaceAndRenter(rentalDto.getPlace(), rentalDto.getRenter());
27         rentalDto.getItems().forEach(this::validateItem);
28     }
29
30     private void validatePlaceAndRenter(PlaceDto placeDto, RenterDto renterDto) {
31         if (placeDto == null && renterDto == null) {
32             throw new ValidationException(
33                 new ErrorMessage("Renter id and place id are null. Provide at least one."));
34         }
35     }
36
37     private void validateItem(ItemDto itemDto) {
38         if (itemDto == null || itemDto.getId() == null) {
39             throw new ValidationException(new ErrorMessage("Item id cannot be null"));
40         }
41     }

```

Rysunek 12 Przykładowa implementacja walidatora

Źródło: Opracowanie własne

❖ Mappery

W celu skonwertowania obiektu typu DTO na obiekt domenowy – encję wykorzystujemy mappery. Operacja w drugą stronę również odbywa się w tej klasie. Klasy te nie zajmują się sprawdzaniem poprawności obiektów. Na rysunku 13 została przedstawiona przykładowa metoda mappera, która zajmuje się zamienianiem obiektu domenowego do DTO.

```

@Service
public class ItemMapper {

    public ItemDto toDto(Item item) {
        if (null == item) {
            return null;
        }
        ItemDto itemDto = ItemDto.builder()
            .placeOfPosting(item.getPlaceOfPosting())
            .insideType(item.getInsideType())
            .equipment(item.isEquipment())
            .inventoryCode(item.getInventoryCode())
            .itemType(item.getItemType())
            .fullItemName(item.getFullItemName())
            .description(item.getDescription())
            .genericName(item.getGenericName())
            .dateOfDelivery(item.getDateOfDelivery())
            .rented(item.getRental() != null)
            .build();
        itemDto.setParent(toDto(item.getParent()));
        itemDto.setId(item.getId());
        return itemDto;
    }
}

```

Rysunek 13 Przykładowa implementacja mappera.

Źródło: Opracowanie własne

❖ Obsługa błędów

Z uwagi na część serwerową napisaną w języku Java, najprostszym sposobem na obsługę błędów jest korzystanie z bloków try-catch w celu złapania rzuconych przez program wyjątków. Z praktycznego punktu widzenia bloki te jedynie zaśmiecają kod. Z tego powodu całkowicie zrezygnowano z takiego podejścia i zamieniono je na globalną obsługę błędów. Dzięki temu, każdy fragment kodu, który może rzucić wyjątek, jedynie deklaruje go w sygnaturze metody. W ten sposób odpowiednia klasa, `WebExceptionHandler` przechwytuje wyjątek, wyciąga z niego komunikat błędu i zwraca użytkownikowi informację w formacie json. Przykład ukazano na rysunku 13.


```
1 {"message": "Place with provided name already exists"}
```

Rysunek 14 Przykładowa odpowiedź z serwera informująca o błędzie.

Źródło: Opracowanie własne

W systemie, klasa `WebExceptionHandler`, zawiera trzy metody do obsługi błędów, opisane zostały w tabeli 28.

Tabela 28 Opis klas odpowiedzialnych za obsługę błędów

<code>handleNotFoundException</code>	Metoda odpowiedzialna za obsługę wyjątków <code>NotFoundException</code> , w przypadku gdy np. obiekt z podanym ID nie został znaleziony w bazie.
<code>handleValidationException</code>	Metoda odpowiedzialna za obsługę wyjątków <code>ValidationException</code> – wszelakich błędów związanych z obiektami DTO, poprawnościami pól.
<code>handleDataIntegrityException</code>	Metoda odpowiedzialna za obsługę wyjątków <code>DataIntegrityViolationException</code> – błędów, które zostały rzucone na poziomie bazy danych np. podczas próby zapisania miejsca z tą samą nazwą.

Źródło: Opracowanie własne

❖ Zabezpieczenia aplikacji

Wymiana informacji pomiędzy częścią serwerową oraz kliencką wymaga pewnego rodzaju sposobu, aby przyznać użytkownikowi dostęp do zasobu. Z racji skorzystania ze stylu REST, aplikacja serwerowa jest bezstanowa i nie zapisuje żadnych informacji o użytkowniku za pomocą sesji. Z tego względu wymaga dostarczenia wszystkich potrzebnych informacji, które są niezbędne do wykonania żądania. Jako rozwiązanie tego problemu zostały użyte tokeny JWT, które zawierają takie informacje jak nazwa użytkownika oraz jego rolę w systemie. Klasy odpowiedzialne za zabezpieczenia opisane zostały w tabeli nr 29, a przykład jednej z metod wykorzystywanych w procesie autentykacji ukazano na rysunku nr 15.

Tabela 29 Opis klas odpowiedzialnych za autoryzację i autentykację

JwtAuthenticationFilter	Klasa odpowiedzialna za obsługę żądania wysłanego pod ścieżkę /login. Znajdują się tutaj metody, odpowiedzialne za wyciągania loginu oraz hasła z żądania, delegowania operacji sprawdzenia, czy dany użytkownik znajduje się w bazie, a finalnie generowanie tokenu JWT, jeżeli proces autentykacji przebiegł pomyślnie.
JwtAuthorizationFilter	Klasa odpowiedzialna za wyciąganie tokenu z żądania, sprawdzenia czy został wygenerowany na pewno przez serwer oraz czy nie wygaśł. Po pomyślnym przeprowadzeniu tych operacji na czas żądania w kontekście aplikacji ustawiany jest użytkownik, by inne metody mogły bez problemu skorzystać z informacji o użytkowniku, bez potrzeby ponownego procesowania tokenu.
WebSecurityConfig	W klasie tej znajduje się główna konfiguracja zabezpieczeń części serwerowej systemu. To tutaj każdy z endpointów (pojęcia do wyjaśnienia) jest wylistowany wraz z wymaganą rolą, która pozwala na dostęp do konkretnych metod. Znajduje się tutaj także konfiguracja CORS (pojęcia do wyjaśnienia).
SecurityConstans	Klasa zawierająca wartości potrzebne do wygenerowania tokenu JWT jak i sam JWT secret, który służy do sprawdzania poprawności tokenu.

```

50 @Override
51 protected void successfulAuthentication(HttpServletRequest request, HttpServletResponse response,
52     FilterChain chain, Authentication authResult) throws IOException {
53     User user = (User) authResult.getPrincipal();
54
55     Set<String> roles = user.getAuthorities().stream()
56         .map(GrantedAuthority::getAuthority)
57         .collect(Collectors.toSet());
58     log.debug("Generating token.");
59     String token = Jwts.builder()
60         .signWith(SignatureAlgorithm.HS512, SecurityConstants.JWT_SECRET.getBytes())
61         .setIssuer(SecurityConstants.TOKEN_ISSUER)
62         .setSubject(user.getUsername())
63         .setExpiration(new Date(System.currentTimeMillis() + 8640000))
64         .claim("roles", roles)
65         .compact();
66
67     String responseLogin = objectMapper
68         .writeValueAsString(new SuccessLoginDto(user.getUsername(), token, roles));
69     // Mateusz Wedeł, 2 months ago • Add loading demo users and do some changes for frontend login f
70     PrintWriter out = response.getWriter();
71     response.setContentType("application/json");
72     response.setCharacterEncoding("UTF-8");
73     out.print(responseLogin);
74     out.flush();
75 }

```

Rysunek 15 Metoda odpowiedzialna za generowanie tokenu JWT

Źródło: Opracowanie własne

2.3.1.2. Warstwa serwisowa

Warstwa serwisowa jest to warstwa pośrednia pomiędzy warstwą internetową, a warstwą repozytorium. To tutaj wywoływane są metody repozytorium, które bezpośrednio operują na bazie danych. W tych klasach również wykonywane są różnego rodzaju operacje kwalifikujące się jako logika biznesowa. Warstwa ta operuje już na obiektach encji, a nie na DTO. Rysunek 16 przedstawia przykładową implementację serwisu w frameworka Spring, a tabela 30 omawia każdy z serwisów.

```

12 @Service
13 @AllArgsConstructor
14 @Slf4j
15 public class ConnectionInterfaceService {
16
17     private final ConnectionInterfaceRepository connectionInterfaceRepository;
18
19     public ConnectionInterface save(ConnectionInterface connectionInterface) {
20         log.debug("Saving connectionInterface");
21         return connectionInterfaceRepository.save(connectionInterface);
22     }
23
24     public ConnectionInterface update(ConnectionInterface connectionInterface) {
25         log.debug("Updating connectionInterface with id: {}", connectionInterface.getId());
26         return connectionInterfaceRepository.save(connectionInterface);
27     }
28
29     public void deleteById(UUID connectionInterfaceId) {
30         log.debug("Removing connectionInterface with id: {}", connectionInterfaceId);
31         this.connectionInterfaceRepository.deleteById(connectionInterfaceId);
32     }
33
34     public ConnectionInterface getById(UUID connectionInterfaceId) {
35         log.debug("Finding connectionInterface with id {}", connectionInterfaceId);
36         return connectionInterfaceRepository.findById(connectionInterfaceId)
37             .orElseThrow(() -> new NotFoundException(
38                 new ErrorMessage(String.format("Connection interface with id %s not found.",
39                     connectionInterfaceId.toString()))));
40     }
41 }
42

```

Rysunek 16 Przykładowa implementacja serwisu

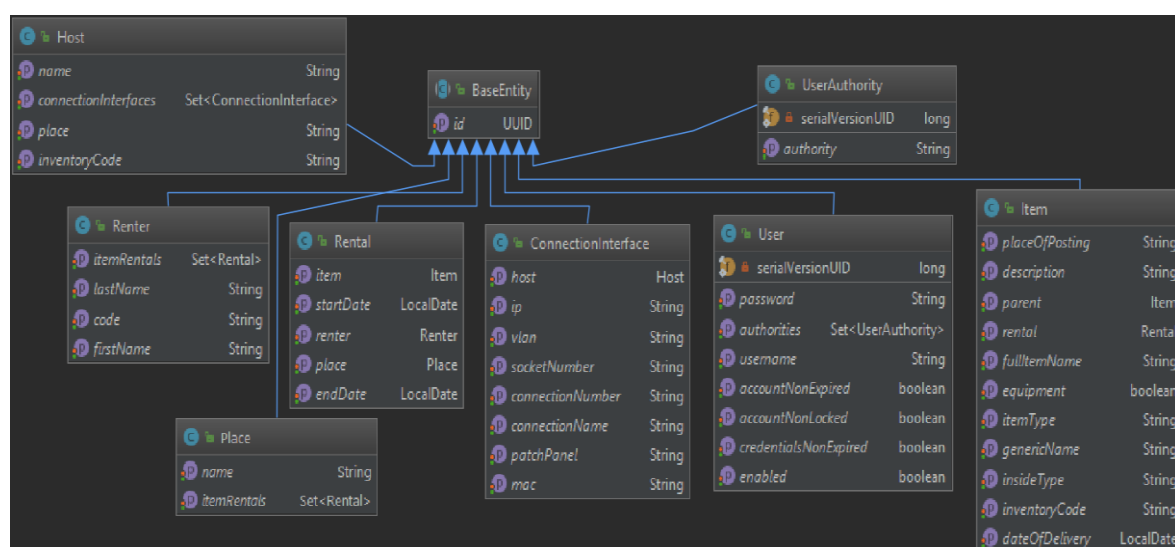
Źródło: Opracowanie własne

Tabela 30 Spis klas serwisowych

ConnectionInterfaceService	W tym serwisie znajdują się metody związane z przypisywanymi adresami do hostów
HostService	Serwis odpowiedzialny za zarządzanie hostami
ItemService	Tutaj znajdują się metody, które pozwalają zapisywać oraz odczytywać sprzęt.
RentalService	Metody do zarządzania wypożyczeniami oraz sprawdzania ich poprawności zostały umiejscowione w tej klasie.
RenterService	Serwis udostępniający metody związane z osobami, do których można przypisać wypożyczenie.
ReportService	Serwis odpowiedzialny za generowanie raportów. Klasa ta wczytuje wcześniej zapisany wzorec, kompiluje go oraz uzupełnia go danymi. W formie skompilowanego, uzupełnionego raportu, jest on zwracany do kontrolera a tam jest zamieniany do pliku z pożądanym formatem.

2.3.1.3. Warstwa repozytorium

Warstwa ta bezpośrednio wykonuje operacje na bazie danych. Biblioteka Spring Data dostarcza podstawowe operacje bazodanowe, a także pozwala definiować własne, bardziej rozbudowane zapytania. Każda encja dziedziczy po encji bazowej BaseEntity, która określa typ klucza głównego oraz sposób w jaki będzie on generowany. W tym systemie, kluczem głównym jest UUID, a procesem jego generowania zajmuje się wirtualna maszyna Javy. Poniższy rysunek nr 17 obrazuje diagram klas encji, a tabela 31 tłumaczy zastosowania każdej z nich.



Rysunek 17 Diagram klas dla encji

Źródło: Opracowanie własne

Tabela 31 Opis klas encji

Item	Główna klasa definiująca sprzęt w systemie.
Place	Definiuje miejsce, do którego można przypisać sprzęt.
Renter	Jest to osoba, której możemy przypisać sprzęt
Rentals	Encja ta definiuje połączenia pomiędzy sprzętem, osobami oraz miejscami.
Host	Hostem nazwać można sprzęt, do którego będzie można przypisać adresy IP (przyłączenia sieciowe)
ConnectionInterface	Przyłączenia sieciowe, przypisywane do hostów.
User	Encja użytkownika definiująca osoby, które mają dostęp do systemu. Zawiera ona m.in. login oraz hasło.
UserAuthority	Encja ta definiuje uprawnienia, które są wykorzystywane w procesie autoryzacji użytkownika.

Źródło: Opracowanie własne

Oprócz bazowej klasy Repozytorium dla każdej encji, w aplikacji znalazły się również dodatkowe implementacje opisane w tabeli 32. Kawałek metody odpowiedzialnej za wyciąganie danych z bazy danych przedstawia rysunek 18.

Tabela 32 Własne implementacje klas repozytorium

ItemRepositoryImpl	Klasa, która udostępnia własną implementację wyszukiwania sprzętu. Na podstawie przekazanych parametrów przez użytkownika, opakowanych klasą ItemSearchParameter, metoda konstruuje zapytanie do bazy.
RentalRepositoryImpl	Dzięki tej klasie możliwe jest wyszukiwanie w bazie wypożyczeń, korzystając z miejsc oraz osób, a także id czy nazw przedmiotów.
HostRepositoryImpl	Dzięki tej klasie możliwe jest wyszukiwanie wszystkich hostów w bazie. Filtrowanie odbywa się za pomocą stworzonego zapytania, które obejmuje pola z klasy ConnectionInterface.

```

@Slf4j
public class HostRepositoryImpl implements SearchHostRepository {

    private static final List<String> connectionInterfaceParameters = List.of(IP,
        MAC, PATCH_PANEL, CONNECTION_NAME, CONNECTION_NUMBER);

    private static final List<String> hostParameters = List.of(NAME, PLACE, INVENTORY_CODE);

    @PersistenceContext
    private EntityManager entityManager;

    @Override
    public Page<Host> search(HostSearchParameters searchParameters, Pageable pageable) {
        log.debug("Starting builder query");
        CriteriaBuilder builder = entityManager.getCriteriaBuilder();
        CriteriaQuery<Host> query = builder.createQuery(Host.class);
        Root<Host> root = query.from(Host.class);
        query.select(root);
        //root.fetch("connectionInterfaces", JoinType.LEFT);
        applyFilters(searchParameters, builder, query, root);

        List<Host> list = entityManager.createQuery(query)
            .setFirstResult(pageable.getPageNumber() * pageable
                .getPageSize())
            .setMaxResults(pageable.getPageSize()).getResultList();

        return PageableExecutionUtils
            .getPage(list, pageable, () -> getCountForRentals(searchParameters));
    }
}

```

Rysunek 18 Fragment klasy repozytorium odpowiedzialnej za wyszukiwanie hostów na podstawie podanych parametrów

Źródło: Opracowanie własne

2.3.2. Warstwa prezentacji - część Frontendowa

Warstwa prezentacji to aplikacja napisana z wykorzystaniem języka TypeScript wraz z frameworkiem Angular w wersji 8. Aplikacja jest podzielona na komponenty, gdzie każdy komponent zawiera kod oparty o znaczniki HTML, plik CSS oraz plik z kodem TypeScript. Dodatkowo, każdy komponent może wykorzystywać serwisy, które dostarczają szereg funkcjonalności, jak np. metody do pobierania danych z serwerowej części systemu. Dodatkowo, framework Angular używa RxJS, który pozwala korzystać z zalet programowania reaktywnego. Warto dodać, że warstwa prezentacji kwalifikuje się jako tak zwany cienki klient.

2.3.2.3. Komponenty

Komponenty są grupowane na podstawie obiektów, na których operują. Grupy te nazywają się modułami. Opis wszystkich komponentów został zawarty w tabeli nr 33.

Tabela 33 Spis wszystkich Angularowych komponentów

ConnectionInterfaceForm	Komponent, który odpowiada za dodawanie i edytowanie adresów przypisywanych do hostów.
HomeComponent	Odpowiedzialny za wyświetlanie głównej strony aplikacji.
HostForm	Komponent odpowiedzialny za edytowanie oraz dodawanie hostów. W przypadku edycji, konkretny host jest pobierany z backendu, dzięki czemu wszystkie jego pola są widoczne i dostępne do edytowania
HostList	Komponent, który pokazuje tabelę hostów, podzielony na strony po domyślne 10 elementów. Komponent ten pozwala również filtrować hosty po wszystkich dostępnych dla niego polach.
HostView	W tym komponencie wyświetlane są informacje o konkretnym hoście, miejscu jego przypisania, informacji, jaki sprzęt jest do niego przyłączony, a także listę wszystkich adresów sieciowych, które są przypisane do tego hosta.
ItemAddComponent	Komponent umożliwiający dodanie nowego sprzętu.
ItemListComponent	paginowana tabela, która umożliwia filtrowanie sprzętu na te, które są wypożyczone i te, które nie są. Komponent pozwala również generowanie plików csv z zaznaczonych elementów, a także filtrować sprzęt po nazwie oraz kodzie inwentarzowym.
ItemViewComponent	Komponent ze szczegółami wybranego sprzętu, a także informacjami o jego wypożyczeniu.
LoginComponent	Z wykorzystaniem tego komponentu, użytkownik może zalogować się do serwisu.
PlaceFormComponent	Komponent umożliwiający dodawanie i edytowanie miejsc.
PlaceListComponent	Komponent wyświetlający paginowaną tabelę ze wszystkimi miejscami, do których możemy przypisać wypożyczenia. Komponent ten również umożliwia wyszukiwanie miejsc po nazwie.
PlaceViewComponent	Komponent wyświetlający szczegółowe informacje o miejscu wraz z listą przedmiotów, które są do niego przypisane. Umożliwia odpisywanie sprzętu od miejsc.
RentalAddComponent	Progresywny komponent umożliwiający przypisać wybraną listę sprzętu do konkretnego miejsca i/oraz osoby. Każde pole zawiera autouzupełnianie o wszystkie możliwe opcje miejsc oraz osób.

RenterFormComponent	Komponent pozwalający edytować oraz dodawać nowe osoby.
RenterListComponent	Tabela wyświetlająca wszystkie osoby, podzielony na strony po minimalnie 10 obiektów na każdą. Umożliwia także wyszukiwanie osób po kodach.
RenterViewComponent	Udostępnia informację o osobie oraz listę przedmiotów, które są do niej przypisane.

Źródło: Opracowanie własne

2.3.2.4. Serwisy

Serwisy to specjalnie oznaczone klasy, które mogą być wstrzyknięte do poszczególnych komponentów, zapewniając szereg metod niezbędnych do ich poprawnego działania. Spis serwisów został przedstawiony w tabeli 34.

Tabela 34 Opis serwisów w Agnularze

AuthenticationService	Serwis udostępniający informacje o obecnie zalogowanym użytkowniku. Dodatkowo, serwis ten odpowiada za wykonywanie żądania do serwera z loginem oraz logoutem(?) z systemu.
ConnectionInterfaceService	W tym serwisie znajdują się metody, które realizują zapytania http do serwera na obiektach typu Connection Interface.
HostService	Serwis ten udostępnia szereg metod, które pozwalają dodawać oraz edytować hosty.
PlaceService	Serwis obsługujący żądania przeznaczone dla obiektów Place.
RentalService	W tym serwisie znajdują się metody, które pozwalają wysłać żądanie do backendu, związane z wypożyczeniami.
RenterService	Odpowiada za zarządzanie osobami, do których można przypisać sprzęt.

2.3.2.5. Interceptory oraz dodatkowe klasy

Oprócz komponentów oraz serwisów możemy także wyróżnić interceptory. Są to specjalne rodzaje klas, które są wywoływane w specjalnych okolicznościach, robiąc z nich swego rodzaju metody globalnego użytku. W systemie wyróżniamy dwa rodzaje

interceptorów, omówienie ich znajduje się w tabeli 35, a przykład implementacji `ErrorInterceptor` zamieszczono na rysunku nr 19.

Tabela 35 *Objaśnienie klas interceptorów*

ErrorInterceptor	Klasa odpowiedzialna za globalną obsługę błędów i wyświetlanie odpowiednich komunikatów użytkownikowi. Klasa ta sprawdza, status http żądania zwróconego przez serwer, a następnie podejmuje odpowiednie akcje.
JwtInterceptor	Klasa odpowiedzialna za wstrzykiwanie tokenu JWT do każdego żądania wysyłanego do serwera w sytuacji, gdy użytkownik został poprawnie zalogowany, a token znajduje się w pamięci lokalnej przeglądarki.

```

13 @Injectable()
14 export class ErrorInterceptor implements HttpInterceptor {
15     constructor(private authenticationService: AuthenticationService, private toastr: ToastrService) {}
16
17     intercept(
18         request: HttpRequest<any>,
19         next: HttpHandler
20     ): Observable<HttpEvent<any>> {
21         return next.handle(request).pipe(
22             catchError(error => {
23                 if (error.status === 0) {
24                     this.toastr.warning('Brak połączenia z serwerem backendowym.');
```

Rysunek 19 *Przykład interceptor - obsługa błędów*

Źródło: Opracowanie własne

Dodatkowo, system posiada klasę nazwaną AuthGuard. Klasa ta sprawdza, czy użytkownik ma dostęp do poszczególnych komponentów aplikacji, bazując na liście ról zwróconych przez serwer przy procesie logowania. Dzięki temu, przykładowo, niezalogowany użytkownik nie przejdzie na stronę z dodawaniem sprzętu.

2.3.2.6. Modele

Modele odpowiadają obiektom DTO z serwerowej części aplikacji. Każdy serwis oraz komponent operuje na tych obiektach aby udostępniać potrzebne informacje użytkownikowi. Oprócz tego, znajdują się tutaj różne opakowania, na przykład obiekt reprezentujący stronę (Page) z obiektami.

2.3.2.5. Routing

Użytkownicy jak i same przeglądarki są przyzwyczajone do kilku rodzajów nawigowania po aplikacji takie jak wpisanie adresu URL czy kliknięcie linku na stronie w celu przejścia na nową. Angular Router korzysta z tego sposobu, przystosowując go pod swoje własne potrzeby. Dzięki temu, w pliku konfiguracyjnym można zdefiniować pod jaką ścieżką będzie dostępny dany komponent. Poniższy rysunek 18 przedstawia fragment implementacji routingu.

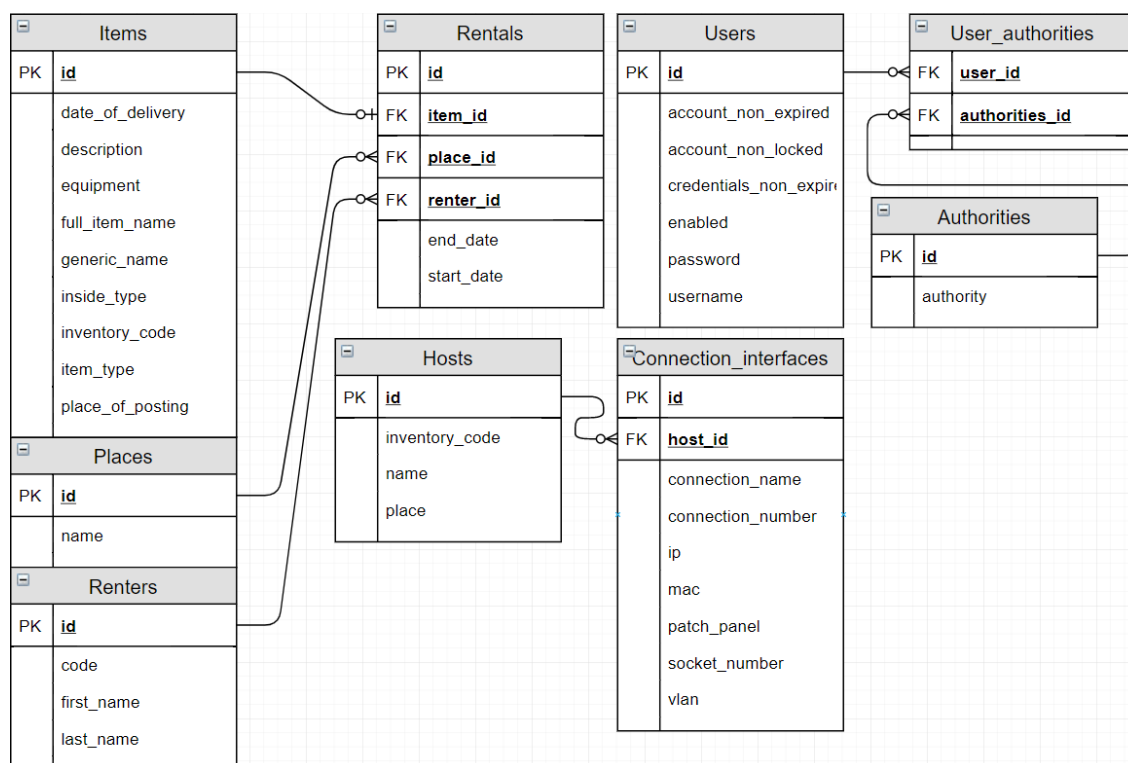
```
const routes: Routes = [
  {
    path: "",
    component: HomeComponent
  },
  {
    path: "items",
    component: ItemListComponent
  },
  {
    path: "login",
    component: LoginComponent
  },
  {
    path: "items/add",
    component: ItemAddComponent,
    canActivate: [AuthGuard],
    data: { roles: [Role.Admin] }
  },
]
```

Rysunek 20 Fragment klasy odpowiedzialnej za routing.

Źródło: Opracowanie własne

2.4. Struktura bazy danych

Baza danych to miejsce, gdzie składowane są główne dane aplikacji. Z uwagi na łatwość przemodelowania tych danych na obiekty, została wykorzystana relacyjna baza danych. Schemat został przedstawiony na rysunku 21.



Rysunek 21 Schemat bazy danych

Źródło: Opracowanie własne

2.5. Testowalność

Podczas tworzenia systemu ważnym aspektem było również testowanie aplikacji. Możemy je podzielić na kilka kategorii:

- Testy jednostkowe kodu backendowego – w celu ich przeprowadzenia zostało użyte popularne narzędzie JUnit, które pozwala tworzyć powtarzalne, automatyczne testy jednostkowe dla systemów napisanych w języku Java.

Dodatkowo zostało użyte Mockito, aby zasymulować niektóre elementy systemu podczas fazy testowania. Testy te testują tylko jedną klasę, dzięki czemu są one szybkie w wykonaniu i jest ich zazwyczaj najwięcej ze wszystkich. Przykład takiego testu obrazuje rysunek 20.

```
15 @RunWith(MockitoJUnitRunner.class)
16 public class JpaUserDetailsServiceTest {
17
18     @Mock
19     private UserRepository userRepository;
20
21     private JpaUserDetailsService userDetailsService;
22
23     @Before
24     public void setUp() throws Exception {
25         userDetailsService = new JpaUserDetailsService(userRepository);
26     }
27
28     @Test(expected = UsernameNotFoundException.class)
29     public void shouldThrowExceptionWhenUsernameNotFound() {
30         //given
31         when(userRepository.findByUsername(any(String.class))).thenReturn(Optional.empty());
32         String username = "testname";
33
34         //when
35         userDetailsService.loadUserByUsername(username);
36     }
37 }
38
```

Rysunek 22 Przykładowy test jednostkowy

Źródło: Opracowanie własne

- Testy integracyjne kody backendowego - zadaniem testów integracyjnych jest sprawdzenie powiązań pomiędzy klasami. Z tego też względu, w przypadku aplikacji w napisanej w Springu, wstaje również kontekst całej aplikacji, przez co testy te trwają znacznie dłużej w porównaniu do testów jednostkowych.
- Testy manualne – każda funkcjonalność po napisaniu i wprowadzeniu została przetestowana manualnie. Dodatkowo, w połowie etapu pisania systemu, została wdrożona developerska wersja aplikacji, która była możliwa do przetestowania również przez grupę docelową.

2.6. Konfiguracja

System pozwala na podstawową konfigurację za pomocą pliku `.env`. Jego zawartość została przedstawiona na rysunku 23, a opis został zawarty w tabeli 36.

```
1  ## DATABASE SETTINGS
2
3  DATABASE_URL=jdbc:postgresql://db:5432/szzti
4  POSTGRES_USER=postgres
5  POSTGRES_PASSWORD=psql
6  POSTGRES_DB=szzti
7
8  # APPLICATION PROFILE
9  # Define current application profile, set profile to dev to load init demo-data to application
10 # for testing.
11 SPRING_PROFILES_ACTIVE=dev
12
13 # Base application url used in cors and generated links. This url is also used for UI application
14 # to do request to the backend. Remember about adding port if it wont be the default 80.
15 BASE_APPLICATION_URL=http://192.168.0.220
16
17 # Application port, the default is :80.
18 BASE_APPLICATION_PORT=80
19
```

Rysunek 23 Konfiguracja aplikacji

Źródło: Opracowanie własne

Tabela 36 Opis zmiennych konfiguracyjnych systemu

DATABASE_URL	Używany przez aplikację serwerową, url ten definiuje adres, pod którym dostępna jest baza danych.
POSTGRES_USER	Nazwa użytkownika bazy danych
POSTGRES_PASSWORD	Hasło dostępowe do bazy danych
POSTGRES_DB	Nazwa bazy, która zostanie utworzona na potrzeby aplikacji.
SPRING_PROFILES_ACTIVE	Profil aplikacji serwerowej który definiuje kilka podstawowych zachowań aplikacji. Wyróżniamy: <ul style="list-style-type: none">• dev – system podczas startu wyczyści bazę danych oraz załaduje testowe dane do systemu. Dodatkowo, CORS będą domyślnie wyłączone.• production – system sprawdzi spójność bazy danych ze stanem encji w systemie oraz nie

	załaduje danych testowych. CORSy zostaną ustawione, aby akceptować żądania tylko z UI-owej części aplikacji.
BASE_APPLICATION_URL	Adres, który jest używany do generowanych linków w aplikacji, a także ten adres jest ustawiany jako Domyślnie używany przy ustawianiu zabezpieczeń CORS.
BASE_APPLICATION_PORT	Port pod którym będzie dostępny serwer nginx jak i cała aplikacja. Domyślnie to port 80.

Źródło: Opracowanie własne

W aspekcie konfigurowalności aplikacji pojawia się pewien problem. Podczas pisania kodu Front Endowej części aplikacji jesteśmy w stanie zdefiniować adres, pod który mają być wysyłane żądania HTTP. Po skompilowaniu plików do wersji produkcyjnej i stworzeniu obrazu Docker zmiana adresu wymaga ponownej kompilacji i stworzenia tegoż obrazu. W środowisku produkcyjnym domena, adres IP, czy Port systemu może się z czasem zmienić. Z tego względu został stworzony specjalny skrypt przedstawiony na rysunku 22.

```

1  #!/bin/sh
2  if [ ! -f template-es5.js ];then
3      main_es5=$(find /usr/share/nginx/html -iname 'main-es5*')
4      echo $main_es5
5      cat ${main_es5} > template-es5.js
6
7  fi
8  if [ ! -f template-es2015.js ];then
9      main_es2015=$(find /usr/share/nginx/html -iname 'main-es2015*')
10     echo $main_es2015
11     cat ${main_es2015} > template-es2015.js
12
13 fi
14
15 envsubst "`printf '%${s}' ' $(sh -c "env|cut -d=' ' -f1")`" < template-es5.js > ${main_es5}
16 envsubst "`printf '%${s}' ' $(sh -c "env|cut -d=' ' -f1")`" < template-es2015.js > ${main_es2015}
17
18 nginx -g 'daemon off;'
```

Rysunek 24 Skrypt ustawiający zmienne środowiskowe w kodzie w momencie startu kontenera

Źródło: Opracowanie własne

Dzięki temu skryptowi możemy używać zmiennych środowiskowych w kodzie. Podczas uruchamiania kontenera, skrypt ten podmienia zmienne na odpowiadające im wartości. Wszystkie te zmienne są zdefiniowane w pliku `.env`. Dzięki takiemu podejściu, w sytuacji gdy adres IP lub port aplikacji zmieni się, wystarczy stworzyć kontener na nowa, wcześniej wpisując nowy adres w pliku `.env`. Dzięki temu pomijamy konieczność ponownego kompilowania kodu i budowania obrazu.

2.7. Wdrożenie aplikacji z wykorzystaniem platformy Docker

Platforma Docker umożliwia łatwe wdrożenie i zarządzanie aplikacją. Jest to alternatywne podejście względem tworzenia wirtualnych maszyn. Głównymi bytami w dockerze z punktu widzenia aplikacji są obrazy oraz kontenery.

2.7.1. Spis obrazów Docker wraz z przeznaczeniem

Sposób tworzenia obrazu definiuje specjalny plik nazywany **Dockerfile**, który definiuje jak obraz ma zostać stworzony. Zawiera on szereg poleceń definiujących obraz bazowy, twórcę, wszelkie komendy, który mają zostać wykonane podczas tworzenia obrazu.

System wykorzystuje trzy obrazy docker'owe do swojego działania:

- `sztti-backend` - obraz o podstawie składającej się ze środowiska uruchomieniowego aplikacji napisanych w Javie (`openjdk:11.0.5-jre`) wraz ze skompilowaną i zbudowaną przy użyciu narzędzia `gradle` aplikacją. DockerFile tego obrazu przedstawiony jest na rysunku 25.


```
1 FROM openjdk:11.0.5-jre-stretch
2
3 COPY build/libs/*.jar /szzti.jar
4
5 EXPOSE 8080
6 CMD java $JAVA_OPTS -jar szzti.jar
```

Rysunek 25 Dockerfile dla szzti-backend

Źródło: Opracowanie własne

- postgres-db - oficjalny obraz bazy danych POSTGRESQL udostępniony na dockerhub.
- szzti-frontend - obraz stworzony na potrzeby aplikacji. Finalny obraz zawiera server nginx, który udostępnia użytkownikom statyczne pliki skompilowanego kodu Angular 8, a także służy jako reverse proxy żądań, które są wysyłane w stronę aplikacji back endowej. DockerFile tego obrazu przedstawia rysunek 26.

```

1  ### STAGE 1: Build ###
2
3  FROM node:10-alpine as builder
4
5  RUN mkdir /szzti
6
7  WORKDIR /szzti
8
9  COPY package.json package-lock.json ./
10
11 RUN npm install
12
13 COPY . .
14
15 RUN npm run build-prod
16
17
18 ### STAGE 2: Setup ###
19
20 FROM nginx:1.14.1-alpine
21
22 COPY nginx/default.conf /etc/nginx/conf.d/
23
24 COPY run.sh /run.sh
25
26 RUN rm -rf /usr/share/nginx/html/*
27
28 COPY --from=builder /szzti/dist/szzti-frontend /usr/share/nginx/html
29 |   Mateusz Wedeł, 2 months ago • Add dockerfile and nginx config.
30 RUN chmod +x run.sh
31
32 CMD ["/bin/sh", "run.sh"]
33

```

Rysunek 26 Dockerfile dla obrazu szzti-frontend

Źródło: Opracowanie własne

W pliku dockerfile przedstawionym na rysunku 26 możemy wyróżnić takie komendy jak:

- FROM - komenda ta definiuje obraz bazowy, od którego zaczęty zostanie proces budowania obrazu.
- RUN - wykonanie komendy z poziomu shella/konsoli bashowej.

- WORKDIR - definiuje bazowy katalog w którym będą wykonywane wszystkie kolejne operacje podczas budowania obrazu.
- COPY - dzięki tej komendzie możemy skopiować pliki bądź całe katalogi bezpośrednio do obrazu.
- CMD - domyślna komenda, która zostanie wywołana podczas startu kontenera.

Budowanie aplikacji Frontendowej w wersji produkcyjnej, a następnie wykorzystanie jej na wybranym serwerze wymaga innego podejścia niż w przypadku aplikacji Back-endowej. Budowanie tegoż obrazu podzielone jest na dwa etapy. Zadaniem pierwszego etapu jest skopiowaniu kodu aplikacji, a następnie skompilowanie go do wersji produkcyjnej, finalnie otrzymując 6 plików, które są efektem całego projektu. Drugi etap, nazwany Setup, korzysta tylko ze skompilowanych plików stworzonych w etapie pierwszym. Wszystkie zależności, które potrzebne były w procesie budowania, ale są zbędne później, zostają pominięte. Korzyści z tego płynące to zredukowanie wagi obrazu z ponad 1 GB do 22MB. Finalnie, wynikiem procesu budowy jest obraz z skonfigurowanym serwerem nginx.

Każdy ze stworzonych obrazów został umieszczony w rejestrze obrazów docker <https://hub.docker.com/u/wheeezyx> umożliwiając pobranie ich z każdego systemu, który posiada zainstalowaną platformę docker.

2.7.2. Kontenery, ich połączenie oraz volume

Każdy kontener jest domyślnie odizolowany od reszty ze względów bezpieczeństwa przez co nie mogą się one ze sobą komunikować. Aby połączyć ze sobą kilka kontenerów tak, aby były dla siebie widoczne z punktu widzenia sieciowego, Docker umożliwia stworzenie specjalnej, dedykowanej sieci (nazywanej network). Odpowiedzialność za stworzenie takiej sieci została przekazana narzędziu Compose.

Kontenery powinny być łatwe w restartowaniu oraz w ponownym tworzeniu. Problem, jaki się z tym pojawia to kontener, który będzie służył za bazę danych. W tym celu został wykorzystany volume, który daje możliwość zapisywania danych w zorganizowany sposób poza kontenerem. Dzięki temu, jeden bądź więcej kontenerów może korzystać z tych samych plików, dane nie usuwają się po usunięciu kontenera, a nowo stworzone kontenery mogą być automatycznie przyłączane do tego volume.

2.7.1. Docker Compose

W celu uproszczenia procesu wdrożenia, tworzenia networków czy kontenerów oraz wykorzystania volume zostało wykorzystane narzędzie Compose. Narzędzie to korzysta z pliku YAML, które w deklaratywny sposób definiuje całą konfigurację systemu z punktu widzenia platformy Docker.

```

1 version: '3'
2 services:
3   db:
4     container_name: 'postgres_db'
5     image: postgres:9.6
6     env_file: .env
7     logging:
8       driver: json-file
9     volumes:
10      - szzti-db:/var/lib/postgresql/data
11
12   backend:
13     image: wheeezyx/szzti-backend
14     env_file: .env
15     command: bash -c 'while !</dev/tcp/db/5432; do sleep 1; done; java $JAVA_OPTS -jar szzti.jar'
16     logging:
17       driver: json-file
18     depends_on:
19      - db
20
21   frontend:
22     image: wheeezyx/szzti-frontend
23     env_file: .env
24     ports:
25      - "${BASE_APPLICATION_PORT:-80}:80"
26     logging:
27       driver: json-file
28     depends_on:
29      - backend
30
31 volumes:
32   szzti-db:
33     external: false

```

Rysunek 27 Deklaratywna konfiguracja infrastruktury z wykorzystaniem narzędzia Docker Compose

Źródło: Opracowanie własne

Plik ten korzysta z 3 wersji narzędzia. Definiuje trzy serwisy (trzy kontenery): **bazę danych** (db), **backend** (część serwerowa) oraz **frontend** (część UI). Każdy kontener korzysta z pliku .env, który dostarcza mu konfigurację znajdującą się w nim części systemu.

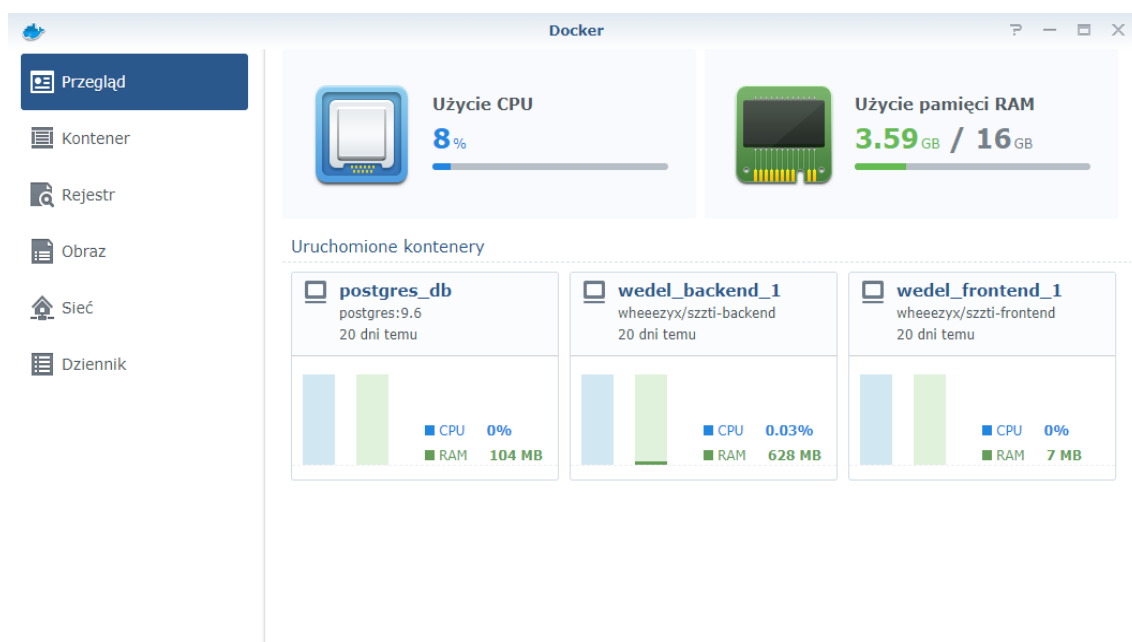
2.7.2. Wdrożenie aplikacji z najbardziej przydatnymi komendami do zarządzania kontenerami

Mając na uwadze konfigurowalność systemu, wykorzystanie platformy docker wraz z udostępnieniem obrazów do specjalnego rejestru do wdrożenia aplikacji potrzeba tylko dwóch plików: plik `.env` wraz z konfiguracją oraz plik `docker-compose.yml` definiujący ustawienia kontenerów.

Gdy pliki te zostaną stworzone, a na systemie docelowym jest już zainstalowana platforma Docker należy:

1. Wykonać polecenie **docker-compose pull**, które ściągnie obrazy z rejestru.
2. Wykonać polecenie **docker volume create szzti-db** - stworzyć volume, który będzie używany przez kontener z bazą danych.
3. Wykonać polecenie **docker-compose up -d**, które uruchomi wszystkie kontenery w trybie detached.

Po krótkiej chwili, gdy system prawidłowo się uruchomi, będzie dostępny pod adresem i portem zdefiniowanym w pliku `.env`. Po wykonaniu tych komend, w graficznym interfejsie Dockera dla systemu SYNOLOGY powinny pojawić się trzy kontenery wraz z kilkoma przydatnymi informacjami.



Rysunek 28 Statystyki uruchomionych kontenerów z Synology docker tool

Źródło: Opracowanie własne

Inne przydatne komendy do zarządzania kontenerami opisano w tabeli nr 37.

Tabela 37 Spis najbardziej przydatnych komend do zarządzania kontenerami.

docker-compose down	Wyłącza kontenery, usuwa je razem z networkiem.
docker container ls	Spis wszystkich uruchomionych kontenerów
docker exec -it ID <COMMAND>	Polecenie pozwalające podłączyć się do działającego kontenera oraz wykonać w nim pożądaną komendę, np.: uruchomienie konsoli bashowej.
docker stats	Wyświetla statystyki obecnie uruchomionych kontenerów - procent zużywanego procesora i ramu
docker-compose logs	Wyświetla logi podanego serwisu, w przypadku tego systemu dozwolonymi opcjami są db , backend , frontend .

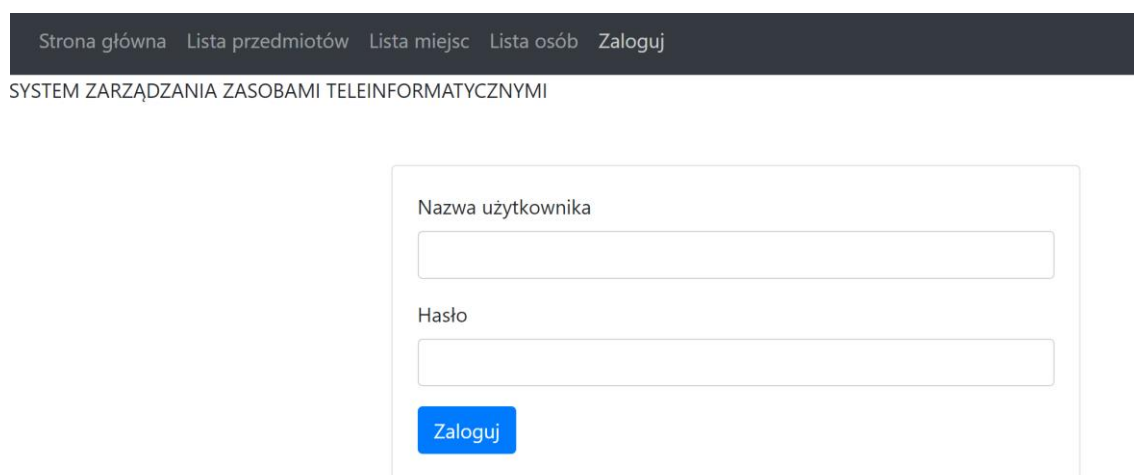
3. Wyniki

3.1. Prezentacja aplikacji

Zasoby w aplikacji dostępne są po zalogowaniu. Z uwagi na dwa konta użytkowników, prezentacja systemu zostanie przeprowadzana z konta administratora. Prezentacja aplikacji została przedstawiona na rysunkach od nr 29 do nr 50.

3.1.1. Strona logowania

Strona logowania do aplikacji.



Strona główna Lista przedmiotów Lista miejsc Lista osób Zaloguj

SYSTEM ZARZĄDZANIA ZASOBAMI TELEINFORMATYCZNYMI

Nazwa użytkownika

Hasło

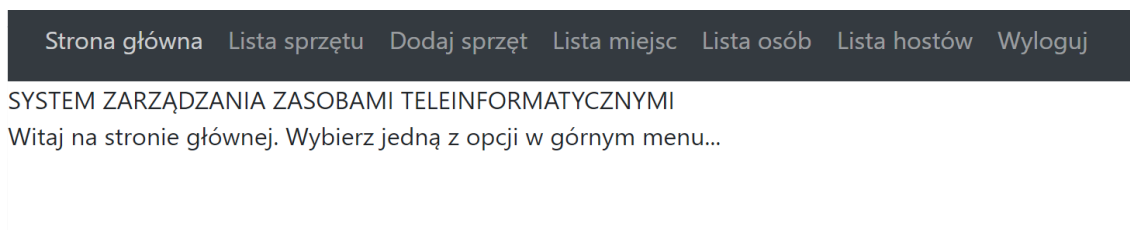
Zaloguj

Rysunek 29 Prezentacja strony logowania.

Źródło: Opracowanie własne

3.1.2. Strona główna

Główna strona aplikacji, która widoczna jest po zalogowaniu. Użytkownik systemu widzi wiadomość powitalną oraz może przemieścić się na inną podstronę wykorzystując nawigację znajdującą się u góry ekranu.



Rysunek 30 Prezentacja strony głównej

Źródło: Opracowanie własne

3.1.3. Lista sprzętu

Lista sprzętu dostępna po wybraniu „Lista sprzętu” w panelu nawigacji.

	Nazwa przedmiotu	Kod inwentarzowy	Typ	Typ środka	Akcje
<input type="checkbox"/>	Computer1	CODE1	EQUIPMENT	NN	
<input type="checkbox"/>	Office 1234	CODE2	SOFTWARE	T	
<input type="checkbox"/>	Krzeslo z oparciem	CODE3	FURNITURE	M	
<input type="checkbox"/>	Długopis	CODE4	OFFICE	N	

Rysunek 31 Prezentacja - lista sprzętu

Źródło: Opracowanie własne

3.1.4. Dodanie sprzętu


Formularz umożliwiający dodanie nowego sprzętu. Dodatkowo, możemy zdefiniować ilość, która doda się do systemu.

Nazwa sprzętu	Data dostarczenia
Kod inwentarzowy	Nazwa zwyczajowa
Nazwa środka	Typ przedmiotu
Miejsce zaksięgowania	Opis
<input type="checkbox"/> Ile dodać? <input type="checkbox"/> Inwentarz1	
<input type="button" value="Zapisz"/>	

Rysunek 32 Formularz dodania nowego sprzętu

Źródło: Opracowanie własne

3.1.5. Szczegóły sprzętu


Po wybraniu konkretnego sprzętu za pomocą ikony , pojawiają się szczegółowe informacje. Dodatkowo, na tej podstronie, znajduje się przycisk do wygenerowania protokołu zniszczenia sprzętu oraz informacje o wypożyczeniu.

Strona główna	Lista sprzętu	Dodaj sprzęt	Lista miejsc	Lista osób	Lista hostów	Wyloguj
---------------	---------------	--------------	--------------	------------	--------------	---------

SYSTEM ZARZĄDZANIA ZASOBAMI TELEINFORMATYCZNYMI

Kod: CODE1
id: fffa0207-abfe-4289-8f5e-5462d18cce43


Pełna nazwa **Computer1** Nazwa zwyczajowa **TEST** Typ środka **NN** Typ przedmiotu **EQUIPMENT** Data zaksięgowania **2019-01-01** Miejsce zaksięgowania **TEST**

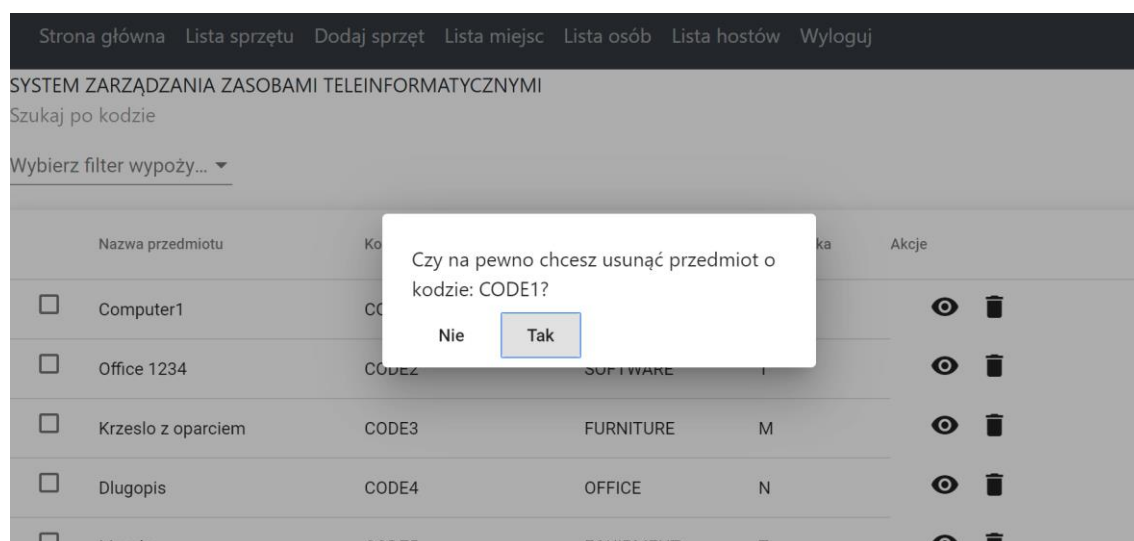
Szczegóły wypożyczenia:
Sala: 129/353 

Rysunek 33 Prezentacja - szczegóły sprzętu

Źródło: Opracowanie własne

3.1.6. Usunięcie sprzętu

Po wybraniu ikony  w liście sprzętu można usunąć dany sprzęt.

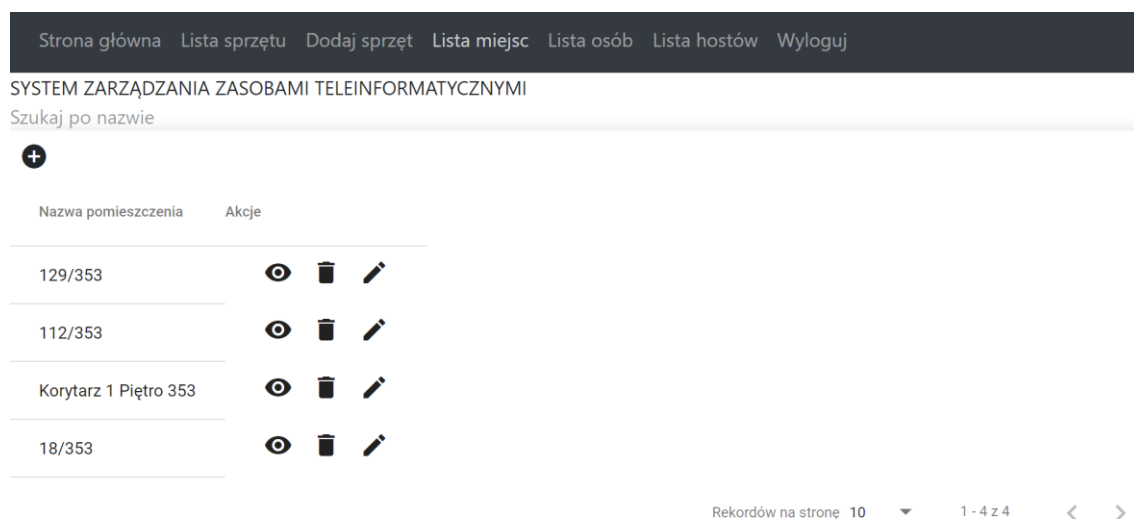


Rysunek 34 Prezentacja - usunięcie sprzętu

Źródło: Opracowanie własne

3.1.7. Lista miejsc



Po wybraniu „Lista miejsc” w górnej nawigacji, pojawia się lista miejsc, którym można przypisać sprzęt.



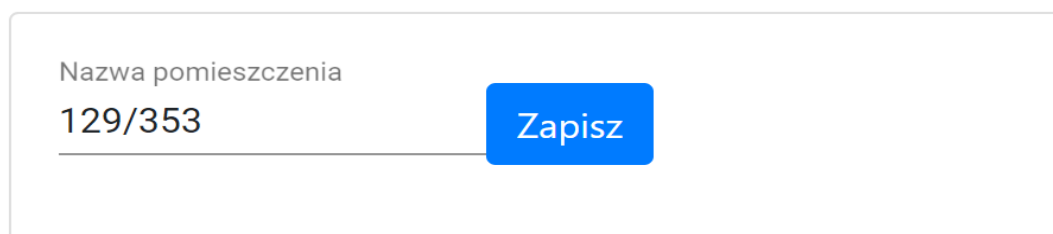
Rysunek 35 Prezentacja - lista miejsc

Źródło: Opracowanie własne

3.1.8. Dodanie/edycja miejsca

Po kliknięciu na  lub  możemy dodać nowe miejsce bądź edytować istniejące. Formularz do tej czynności wygląda następująco. W przypadku dodania nowego miejsca, formularz ten byłby pusty.

Edytuj miejsce



Rysunek 36 Prezentacja - edycja pomieszczenia.

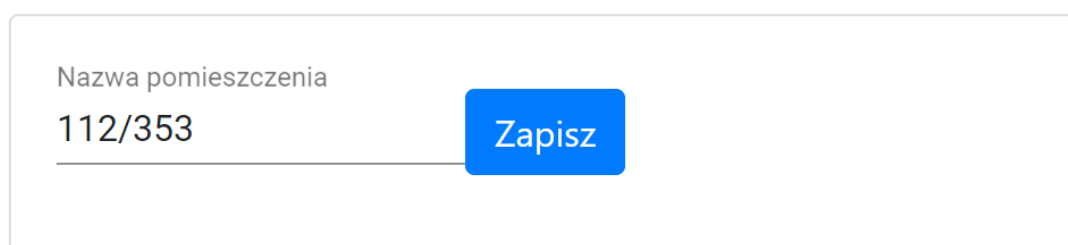
Źródło: Opracowanie własne

3.1.9. Dodanie/edycja miejsca – podanie błędnych danych

Przykładowy komunikat informujący o błędnym wypełnieniu formularza. W tym przypadku, nazwa miejsca jest już zajęta.




Edytuj miejsce



Rysunek 37 Prezentacja - komunikat błędu

Źródło: Opracowanie własne




3.1.10. Szczegóły miejsca

Ikona  przenosi nas do szczegółów konkretnego miejsca, skąd możemy zobaczyć listę przypisanego do niego sprzętu oraz wygenerować raport dla pomieszczenia.

SYSTEM ZARZĄDZANIA ZASOBAMI TELEINFORMATYCZNYMI

Nazwa: 129/353
id: **d4a20291-f7a1-468a-99f6-39e7d35b5bfc**

Lista przedmiotów przypisanych do tej sali:

Nazwa przedmiotu	Nazwa sali	Kod osoby wypożyczającej
<input type="checkbox"/> Computer1 	129/353 	Brak 

Rekordów na stron

Generuj raport dla pomieszczenia


Rysunek 38 Prezentacja - szczegóły miejsca













Źródło: Opracowanie własne

3.1.11. Usunięcie miejsca

SYSTEM ZARZĄDZANIA ZASOBAMI TELEINFORMATYCZNYMI

Szukaj po nazwie



Nazwa pomieszczenia	Akcje
129/353	  
112/353	  
Korytarz 1 Piętro 353	  
18/353	  

Czy na pewno chcesz usunąć miejsce:
129/353? Wszystkie wypożyczone tutaj
przedmioty zostaną odpisane.

Nie **Tak**

Rekordów na stronę 10 1 - 4 z 4 < >

Rysunek 39 Prezentacja - usunięcie miejsca

Źródło: Opracowanie własne

69

3.1.12. Lista osób

Po wybraniu w górnym zakładki Lista osób, pojawia się paginowana lista wszystkich osób w systemie. Widok jest standardowy jak dla wszystkich list tego typu w aplikacji.

SYSTEM ZARZĄDZANIA ZASOBAMI TELEINFORMATYCZNYMI

Szukaj po kodzie



			
Imię	Nazwisko	Kod	Akcje
Jan	Kowalski	3312AA	  
MateuszWedel		11231	  
Tomasz Kowalski		113123	  
John	Smith	1135CC	  

Rekordów na stronę 10 1 - 4 z 4 < >

Rysunek 40 Prezentacja - lista osób

Źródło: Opracowanie własne

3.1.13. Dodanie/edycja osoby

Po wybraniu ikony  lub  możemy dodać nową osobę bądź edytować istniejącą. Formularz do tej czynności wygląda następująco.

Edytuj wypożyczającego

Imię	Nazwisko
Jan	Kowalski
Kod	
3312AA	<input type="button" value="Zapisz"/>

Rysunek 41 Prezentacja - edycja osoby

Źródło: Opracowanie własne

3.1.14. Szczegóły osoby

Po wybraniu szczegółów osoby pojawia się lista sprzętu do niej przypisanych, a także przycisk do wygenerowania raportu w formacie PDF.

SYSTEM ZARZĄDZANIA ZASOBAMI TELEINFORMATYCZNYMI

Kod: **11231** Imię: **Mateusz** Nazwisko: **Wedel**
Id: **0bb60e7f-d3d8-431c-8278-55d233fb656e**

Lista przedmiotów przypisanych do tej osoby:

Nazwa przedmiotu	Nazwa sali	Kod osoby wypożyczającej
<input type="checkbox"/> Office 1234	<input type="checkbox"/> Brak	11231 <input type="checkbox"/>

Rekordów na stronę 10 1 - 1 z 1

Generuj raport dla osoby

Rysunek 42 Prezentacja - szczegóły osoby

Źródło: Opracowanie własne

3.1.15. Tworzenie wypożyczenia

Po wybraniu listy sprzętów gotowych do wypożyczenia i kliknięciu przycisku „Wypożycz zaznaczone” na podstronie z listą sprzętu, pojawia się strona ze szczegółami tworzenia wypożyczenia. Po wybraniu miejsca wypożyczenia oraz/lub osoby, możliwe jest zatwierdzenie wypożyczenia.

SYSTEM ZARZĄDZANIA ZASOBAMI TELEINFORMATYCZNYMI

1 Wybierz miejsce wypożyczenia

2 Wybierz osobę wypożyczającą

3 Podsumowanie

Lista przedmiotów:
CODE5 - Myszka

W sumie: 1
112/353
1135CC

Cofnij Zapisz

Rysunek 43 Prezentacja - tworzenie nowego wypożyczenia.

Źródło: Opracowanie własne

3.1.16. Eksportowanie sprzętu

Po wybraniu z listy interesujących nas sprzętów i kliknięciu na opcję „Eksportuj zaznaczone”, wygenerowany zostanie plik csv zawierający listę sprzętów.

1	id	Pełna nazwa produktu	Kod inwentarzowy	Data dostarczenia	Miejsce zaksięgowania
2	fffa0207-abfe-4289-8f5e-5462d18cce43	Computer1	CODE1	01.01.2019	TEST
3	11dae30a-e24a-4646-8700-0c03d9c62d86	Office 1234	CODE2	01.01.2019	TEST3

Rysunek 44 Prezentacja - wynik eksportowania sprzętu

Źródło: Opracowanie własne

3.1.17. Protokół zniszczenia sprzętu

Tak wygląda przykładowy protokół zniszczenia wybranego z listy sprzętu.

PROTOKÓŁ USZKODZENIA SPRZĘTU TECHNICZNEGO NR	
Sporządzony w Pracowni Informatyki	Dnia: 2020-01-22
Przez komisję w składzie:	Przewodniczący:
	Członek:
Po dokonaniu przeglądu uszkodzonego sprzętu (maszyny, aparatu):	
Pełna nazwa:	Computer1
Typ:	EQUIPMENT
Numer inwentarzowy:	CODE1
Data otrzymania:	2019-01-01
Rodzaj uszkodzenia:
Przyczyny uszkodzenia:
Wnioski: przekazać do naprawy
Zatwierdzam:	Podpisy komisji:
.....
Przekazano do naprawy zam. nr dnia	

Rysunek 45 Prezentacja - protokół uszkodzenia sprzętu

Źródło: Opracowanie własne

3.1.18. Raport wypożyczonego sprzętu

Na podstronie ze szczegółami osoby, po wciśnięciu przycisku „Generuj raport dla osoby” zostanie wygenerowany raport wypożyczenia sprzętu. Zaprezentowany został na poniższym rysunku.

Formularz przekazania sprzętu		
Formularz przekazania sprzętu dla:		Imię: Mateusz Nazwisko: Wedel Kod: 11231
Nazwa sprzętu	Kod inwentarzowy	Data dostarczenia
Office 1234	CODE2	2019-01-01

22/01/2020

Miejsce na podpis













Strona 1 z 1

Rysunek 46 Prezentacja - formularz przekazania sprzętu

Źródło: Opracowanie własne

3.1.19. Lista hostów

Po wybraniu opcji „Lista hostów” z górnego menu, pojawia się tabela ze wszystkimi hostami. Tabela ta zawiera dużo więcej filtrowań niż inne, głównie ze względu na możliwość filtrowania hostów względem przyłączy sieciowych.

SYSTEM ZARZĄDZANIA ZASOBAMI TELEINFORMATYCZNYMI		
Szukaj po nazwie hosta		
Szukaj po nazwie miejsca		
Szukaj po kodzie inwentarzowym		
Szukaj po adresie IP		
Szukaj po adresie MAC		
Szukaj po patch panelu		
Szukaj po nazwie przyłącza		
Szukaj po numerze przyłącza		
+		
Nazwa hosta	Nazwa pomieszczenia	Akcje
C125	353/3	  
C126	353/3	  
C127	353/3	  
C128	353/3	  

Rysunek 47 Prezentacja - lista hostów

Źródło: Opracowanie własne

3.1.20. Dodanie/edycja hostów

Dodanie oraz edycja hosta wygląda bardzo podobnie jak w przypadku innych formularzy.

Dodaj nowego hosta

Nazwa hosta	Nazwa pomieszczenia
Kod inwentarzowy	<input type="button" value="Zapisz"/>

Rysunek 48 Prezentacja - dodanie hostów

Źródło: Opracowanie własne

3.1.21. Szczegóły hosta

Po wybraniu hosta z tabeli pojawiają się szczegółowe informacje na temat sprzętu do którego został przypisany ten host oraz listę przyłączy sieciowych przypisanych do wybranego hosta.

SYSTEM ZARZĄDZANIA ZASOBAMI TELEINFORMATYCZNYMI

Nazwa hosta: **C125**

Numer inwentarzowy: **CODE1**





Pełna nazwa przedmiotu **Computer1**

Nazwa zwyczajowa **TEST**

Typ środka **NN**

Typ

Filtruj po dowolnej kolu... 

IP	MAC	Numer gniazda	VLAN	Patch Panel	Nazwa przyłącza	Numer przyłącza	Akcje
128.121.12.13	12AAC	AKCCD					 
128.121.12.11	12AAC	AKCCD					 

Rekordów na stronę 10 1 - 2 z 2 < >


Rysunek 49 Prezentacja - szczegóły hosta

Źródło: Opracowanie własne

3.1.22. Dodanie/edycja przyłączenia sieciowego

Z poziomu szczegółów hosta możemy mu przypisać przyłączenie sieciowe albo edytować istniejące.

Edytuj interfejs hosta C125

Adres IP	Adres MAC
128.121.12.13	12AAC
VLAN	Numer gniazda
	AKCCD
Numer Patch panelu	Nazwa przyłącza
Numer przyłącza	

Rysunek 50 Prezentacja - edycja przyłączenia sieciowego

Źródło: Opracowanie własne

4. Podsumowanie

4.7. Wnioski

Celem pracy było zaprojektowanie, zaimplementowanie oraz wdrożenie aplikacji webowej umożliwiającej w prosty sposób zarządzanie sprzętem oraz hostami. System przeszedł kilka zmian w fazie projektowania oraz implementacji, lecz finalnie krytyczne funkcjonalności zostały zaimplementowane.

Proces tworzenia aplikacji w dużej części przypominał proces występujący w firmach: ciągły kontakt z klientem (w przypadku tego projektu, grupą docelową), reagowanie na uwagi oraz implementowanie poprawek do systemu. System może być prosto rozwijalny w przyszłości dzięki zastosowaniu architektury klient-serwer, przez co całość składa się tak naprawdę z dwóch aplikacji, gdzie dowolnie możemy podmienić którąś z nich na implementację w innym języku, na przykład przepisać UI używając popularnego frameworka React czy wykorzystać Pythona do części Back-endowej.

Do wdrożenia systemu wykorzystano Docker'a - platformę, dzięki której wdrażanie aplikacji jest dużo prostsze. System został tak napisany od strony infrastruktury, aby w przyszłości grupa docelowa mogłaby w prosty sposób przenieść aplikację na inny serwer, czy nawet wykorzystać jedną z popularnych chmur.

4.8. Późniejszy rozwój

Głównym elementem, który można rozwinąć w aplikacji jest wygląd. System został napisany z wykorzystaniem gotowych komponentów, które nie były zbyt stylizowane. Aplikacja dzięki temu zyskała by na pewno na pierwszym odbiorze, gdyż zwykle to wygląd to determinuje. Dodatkowo, część Front-endowa mogłaby zostać przepisania z wykorzystaniem responsywnego podejścia do stron internetowych, aby być odpowiednio skalowana na wielu rozdzielczościach.

System jest przystosowany, aby można było dodać do niego nowe moduły funkcjonalne, które mogą w przyszłości być przydatne z punktu widzenia grupy docelowej. Popularnym podejściem do realizowania wielomodułowych aplikacji jest

podejście wykorzystujące mikroserwisy. Z uwagi na wykorzystanie kontenerów, system ten po kilku modyfikacjach można by skalować horyzontalnie.

Spis literatury

- [1] „Wikipedia.” [Online] Available: <https://pl.wikipedia.org/wiki/Host>. [Data uzyskania dostępu: 15.01.2020]
- [2] „Wikipedia.” [Online] Available: https://pl.wikipedia.org/wiki/Portable_Document_Format. [Data uzyskania dostępu: 15.01.2020]
- [3] „MDN web docs” [Online] Available: <https://developer.mozilla.org/pl/docs/Web/JavaScript> [Data uzyskania dostępu: 14.01.2020]
- [4] Vlad Mihalcea: High-Performance Java Persistence, 2016.
- [5] „Json” [Online] Available: <https://www.json.org/json-pl.html> [Data uzyskania dostępu 12.01.2019]
- [6] „Oficjalna dokumentacja Docker” [Online] Available: <https://docs.docker.com/get-started/> [Data uzyskania dostępu: 12.01.2020]
- [7] „Wikipedia” [Online] Available: <https://pl.wikipedia.org/wiki/Nginx> [Data uzyskania dostępu: 15.01.2020]
- [8] „REST API Tutorial” [Online] Available: <https://restfulapi.net> [Data uzyskania dostępu 16.01.2020]
- [9] „Wikipedia” [Online] Available: https://en.wikipedia.org/wiki/Plain_old_Java_object#cite_note-bliki-1 [Data uzyskania dostępu: 16.01.2020]
- [10] „MartinFowler” [Online] Available: <https://martinfowler.com/eaCatalog/dataTransferObject.html> [Data uzyskania dostępu: 11.01.2020]
- [11] „Sekurak” [Online] Available: <https://sekurak.pl/jwt-security-ebook.pdf> [Data uzyskania dostępu 13.01.2020]
- [12] „Wikipedia” [Online] Available: https://en.wikipedia.org/wiki/Universally_unique_identifier [Data uzyskania dostępu 14.01.2020]
- [13] „Oficjalna dokumentacja Docker” [Online] Available: <https://docs.docker.com/glossary> [Data uzyskania dostępu 15.01.2020]
- [14] „Oficjalna dokumentacja Docker” [Online] Available: <https://docs.docker.com/glossary> [Data uzyskania dostępu 15.01.2020]

[15] „Wikipedia” [Online] Available: <https://pl.wikipedia.org/wiki/YAML> [Data uzyskania dostępu 16.01.2020]

[16] „Wikipedia” [Online] Available: <https://pl.wikipedia.org/wiki/Minifikacja> [Data uzyskania dostępu 12.01.2020]

Spis rysunków

Rysunek 1 Podział na Front-end oraz Back-end	12
Rysunek 2 Sposób komunikacji i podział z poziomu infrastruktury	13
Rysunek 3 Diagram przypadków użycia - Moduł miejsc	17
Rysunek 4 Diagram przypadków użycia - Moduł sprzętu	17
Rysunek 5 Diagram przypadków użycia - Moduł osób.....	18
Rysunek 6 Diagram przypadków użycia - Moduł wypożyczeń.....	18
Rysunek 7 Diagram przypadków użycia - Moduł raportów	19
Rysunek 8 Diagram przypadków użycia - Moduł hostów	19
Rysunek 9 Architektura trójwarstwowa.....	35
Rysunek 10 Podział warstwy aplikacji.....	36
Rysunek 11 Diagram klas obiektów DTO	38
Rysunek 12 Przykładowa implementacja walidatora	39
Rysunek 13 Przykładowa implementacja mappera	40
Rysunek 14 Przykładowa odpowiedź z serwera informująca o błędzie.....	41
Rysunek 15 Metoda odpowiedzialna za generowanie tokenu JWT	43
Rysunek 16 Przykładowa implementacja serwisu	44
Rysunek 17 Diagram klas dla encji	45
Rysunek 18 Fragment klasy repozytorium odpowiedzialnej za wyszukiwanie hostów na podstawie podanych parametrów	47
Rysunek 19 Przykład interceptora - obsługa błędów.....	50
Rysunek 20 Fragment klasy odpowiedzialnej za routing.	51
Rysunek 21 Schemat bazy danych	52
Rysunek 22 Przykładowy test jednostkowy	53
Rysunek 23 Konfiguracja aplikacji	54
Rysunek 24 Skrypt ustawiający zmienne środowiskowe w kodzie w momencie startu kontenera	55
Rysunek 25 Dockerfile dla szty-backend	57
Rysunek 26 Dockerfile dla obrazu szty-frontend	58
Rysunek 27 Deklaratywna konfiguracja infrastruktury z wykorzystaniem narzędzia Docker Compose	61
Rysunek 28 Statystyki uruchomionych kontenerów z Synology docker tool	63
Rysunek 29 Prezentacja strony logowania.....	64
Rysunek 30 Prezentacja strony głównej	65
Rysunek 31 Prezentacja - lista sprzętu.....	65
Rysunek 32 Formularz dodania nowego sprzętu	66
Rysunek 33 Prezentacja - szczegóły sprzętu	66
Rysunek 34 Prezentacja - usunięcie sprzętu	67
Rysunek 35 Prezentacja - lista miejsc	67
Rysunek 36 Prezentacja - edycja pomieszczenia.	68
Rysunek 37 Prezentacja - komunikat błędu.....	68
Rysunek 38 Prezentacja - szczegóły miejsca	69
Rysunek 39 Prezentacja - usunięcie miejsca	69
Rysunek 40 Prezentacja - lista osób.....	70
Rysunek 41 Prezentacja - edycja osoby	70
Rysunek 42 Prezentacja - szczegóły osoby.....	71

Rysunek 43 Prezentacja - tworzenie nowego wypożyczenia	71
Rysunek 44 Prezentacja - wynik eksportowania sprzętu	72
Rysunek 45 Prezentacja - protokół uszkodzenia sprzętu	72
Rysunek 46 Prezentacja - formularz przekazania sprzętu.....	73
Rysunek 47 Prezentacja - lista hostów.....	74
Rysunek 48 Prezentacja - dodanie hostów	74
Rysunek 49 Prezentacja - szczegóły hosta	75
Rysunek 50 Prezentacja - edycja przyłączenia sieciowego	75

Spis tabel

Tabela 1 Wyświetlenie listy sprzętu.....	20
Tabela 2 Dodanie sprzętu.....	20
Tabela 3 Logowanie.....	21
Tabela 4 Edycja sprzętu.....	21
Tabela 5 Usunięcie sprzętu	22
Tabela 6 Wyświetl szczegóły sprzętu	22
Tabela 7 Wyeksportuj sprzęt	23
Tabela 8 Wyświetlenie listy osób	24
Tabela 9 Dodanie osoby	24
Tabela 10 Edycja osoby	25
Tabela 11 Wyświetlenie szczegółów osoby	25
Tabela 12 Wyświetlenie szczegółów miejsca.....	26
Tabela 13 Dodanie miejsca	27
Tabela 14 Edycja miejsca	27
Tabela 15 Generowanie raportu uszkodzenia sprzętu	28
Tabela 16 Generowanie raportu wypożyczenia dla osoby	28
Tabela 17 Dodawanie nowych wypożyczeń.....	29
Tabela 18 Usuwanie wypożyczeń.....	30
Tabela 19 Wyświetl informacje o wypożyczeniach.....	30
Tabela 20 Dodanie hosta.....	31
Tabela 21 Edycja hosta.....	31
Tabela 22 Usuwanie hosta	32
Tabela 23 Wyświetlenie szczegółów hosta	32
Tabela 24 Wyświetlenie listy hostów	33
Tabela 25 Przypisanie przyłączeń sieciowych do hostów	33
Tabela 26 Usunięcie przyłączenia sieciowego	34
Tabela 27 Opis kontrolerów.....	37
Tabela 28 Opis klas odpowiedzialnych za obsługę błędów.....	41
Tabela 29 Opis klas odpowiedzialnych za autoryzację i autentykację	42
Tabela 30 Spis klas serwisowych.....	44
Tabela 31 Opis klas encji	46
Tabela 32 Własne implementacje klas repozytorium.....	46
Tabela 33 Spis wszystkich Angularowych komponentów	48
Tabela 34 Opis serwisów w Angularze	49
Tabela 35 Objasnienie klas interceptorów.....	50
Tabela 36 Opis zmiennych konfiguracyjnych systemu.....	54
Tabela 37 Spis najbardziej przydatnych komend do zarządzania kontenerami.	63

Spis załączników

Streszczenia

Streszczenie po polsku

Tematem niniejszej pracy inżynierskiej jest projekt, implementacja oraz wdrożenie systemu zarządzania zasobami teleinformatycznymi.

Aplikacja jest zaprojektowana z wykorzystaniem trójwarstwowej architektury, gdzie głównymi technologiami użytymi w projekcie są język Java, język TypeScript, frameworka takie jak Spring Boot oraz Angular. Jako bazę danych wykorzystano PostgreSQL.

System ma umożliwić zarządzanie sprzętem i ich wypożyczaniem, a także hostami. Dodatkowymi funkcjonalnościami są generowane w formacie PDF raporty czy eksportowanie samego sprzętu do plików .csv.

Ważnym aspektem pracy jest również sposób wdrożenia, by był on prosty i możliwie jak najprostszy dla późniejszych użytkowników systemu. Do tego celu, wykorzystano platformę Docker.

Summary in English

The subject of this engineering thesis is the design, implementation, and deployment of ICT resource management system.

The application is designed with the use of three-layer architecture, where the main technologies used in the project are Java, TypeScript, frameworks like Spring Boot and Angular. As the database, PostgreSQL was used.

The system is to enable easy management of equipment and their rentals, as well as hosts. Additional functionalities are reports generated in PDF format or exporting the hardware itself to .csv files.

An important aspect of the engineering thesis is also the way of the deployment that it was simple and as simple as possible for later system users. For this purpose, the Docker platform was used.