



# POLSKO-JAPOŃSKA AKADEMIA TECHNIK KOMPUTEROWYCH

Wydział Informatyki

Filia w Gdańsku

**Langmesser Adam**

Nr albumu s27119

Nazwa specjalizacji: Aplikacje Internetowe

**Redosz Mateusz**

Nr albumu s27094

Nazwa specjalizacji: Aplikacje Internetowe

**Oziemczuk Stanisław**

Nr albumu s26982

Nazwa specjalizacji: Aplikacje Internetowe

**Badek Kacper**

Nr albumu s29168

Nazwa specjalizacji: Aplikacje Internetowe

## **Aplikacja webowa: spoty-na-drony.pl**

Rodzaj pracy

inżynierska

Imię i nazwisko promotora

mgr Adam Urbanowicz

Gdańsk, miesiąc, 2100 obrony

**Streszczenie:** Celem niniejszej pracy było stworzenie w pełni funkcjonalnej i działającej aplikacji internetowej pozwalającej na szybkie wyszukiwanie spotów w okolicy oraz dzielenie się zdjęciami, filmami oraz doświadczeniem z innymi użytkownikami. W ramach pracy stworzono system składający się z trzech komponentów: [Frontendu](#), [Backendu](#) oraz bazy-danych. Aplikacja internetowa została wykonana przy pomocy [Frameworka](#) React w językach Javascript oraz Typescript, do stylu został użyty Tailwind. Serwis backendowy został stworzony w języku Java oraz biblioteki Spring Boot. Baza danych to PostgreSQL.

Komunikacja między komponentami odbywała się zgodnie ze standardem REST. Projekt został zrealizowany w podejściu ewolucyjno-przyrostowym z elementami Kanban.

**Słowa kluczowe:** — brak —



# POLSKO-JAPOŃSKA AKADEMIA TECHNIK KOMPUTEROWYCH

## Karta projektu

<b>Temat projektu:</b> Aplikacja webowa: spoty-na-drony.pl <b>Temat projektu po angielsku:</b> Web application: spoty-na-drony.pl	<b>Akronim:</b> Merkury <b>Data ustalenia tematu</b> 2023-10-10
<b>Promotor:</b>  mgr Adam Urbanowicz	<b>Konsultanci:</b>  1. — brak —
<b>Cele projektu:</b> Stworzenie w pełni funkcjonalnej aplikacji internetowej do rozwijania hobby (latania dronem).	
<b>Rezultaty projektu:</b> Aplikacja Internetowa, Dokumentacja Interaktywna mapa z wyświetlanymi spotami oraz pogodą. Zaawansowana wyszukiwarka spotów. Forum do dzielenia się informacjami na temat dronów. Chat jednoosobowy oraz grupowy. Konto użytkownika z możliwością zapisania ulubionych spotów.	
<b>Miary sukcesu:</b> Gotowa do wdrożenia aplikacja. Realizacja w terminie zgodnym z wymaganiami.	
<b>Ograniczenia:</b> Budżetowe: brak środków na wdrożenie. Zawodowe: brak doświadczenia. Czasowe: trzy semestry (09.2024 - 02.2026). Ludzkie: czteroosobowy zespół.	

Wykonawcy	Numer al- bumu	Specjalizacja	Tryb studiów
Langmesser Adam	s27119	Aplikacje Internetowe	Stacjonarny
Redosz Mateusz	s27094	Aplikacje Internetowe	Stacjonarny
Oziemczuk Stanisław	s26982	Aplikacje Internetowe	Stacjonarny
Badek Kacper	s29168	Aplikacje Internetowe	Stacjonarny

<b>Data ukończenia projektu:</b> 27 listopada 2025	<b>Recenzent:</b> dr Elżbieta Puźniakowska-Gałuch
---	--

# Spis treści

<b>1</b>	<b>Wstęp</b>	<b>6</b>
1.1	O projekcie . . . . .	6
1.2	Cel i zakres prac . . . . .	6
1.3	Geneza pomysłu . . . . .	6
<b>2</b>	<b>Opis problemu</b>	<b>7</b>
2.1	Rich picture . . . . .	7
2.2	Udziałowcy . . . . .	7
2.3	Istniejące rozwiązania . . . . .	9
2.4	Wizja rozwiązania . . . . .	9
2.5	Aspekty społeczne i biznesowe . . . . .	9
2.5.1	Aspekty społeczne . . . . .	9
2.5.2	Aspekty biznesowe . . . . .	9
<b>3</b>	<b>Planowanie</b>	<b>10</b>
3.1	Metodologia pracy . . . . .	10
3.1.1	Przegląd rozważanych podejść . . . . .	10
3.1.2	Odrzucone podejścia . . . . .	10
3.1.3	Wybrane podejście: Disciplined Agile Delivery (Lean Life Cycle) . . . . .	11
3.1.4	Narzędzia i komunikacja . . . . .	11
3.1.5	Podział ról w zespole . . . . .	12
3.2	Harmonogram projektu . . . . .	12
3.3	Technologie i narzędzia . . . . .	14
3.3.1	Technologie . . . . .	14

3.3.2	Narzędzia . . . . .	14
3.4	Zasoby i ograniczenia . . . . .	17
3.4.1	Zasoby . . . . .	17
3.4.2	Ograniczenia . . . . .	17
3.4.3	Usługi zewnętrzne . . . . .	17
3.5	Analiza ryzyka . . . . .	20
<b>4</b>	<b>Analiza wymagań</b>	<b>21</b>
4.1	Przypadki użycia . . . . .	22
4.1.1	Aktorzy . . . . .	22
4.1.2	Diagram przypadków użycia . . . . .	23
4.1.3	Scenariusze przypadków użycia . . . . .	23
4.2	Wymagania ogólne i dziedzinowe . . . . .	63
4.3	Wymagania funkcjonalne . . . . .	63
4.3.1	Funkcjonalności dla mapy . . . . .	63
4.3.2	Funkcjonalności dla chatu . . . . .	63
4.3.3	Funkcjonalności dla forum . . . . .	63
4.3.4	Funkcjonalności dla konta użytkownika . . . . .	63
4.3.5	Funkcjonalności dla logowania i rejestracji . . . . .	73
4.3.6	Funkcjonalności dla wyszukiwarki spotów . . . . .	74
4.3.7	Funkcjonalności dla motywu . . . . .	76
4.4	Wymagania pozafunkcjonalne . . . . .	78
4.5	Wymagania interfejs z otoczeniem . . . . .	78
4.6	Wymagania na środowisko docelowe . . . . .	78
<b>5</b>	<b>Projekt</b>	<b>79</b>
5.1	Wzorce projektowe . . . . .	79
5.2	Architektura systemu . . . . .	79
5.2.1	Diagram architektury . . . . .	79
5.2.2	Komponenty systemu . . . . .	79
5.3	Projekt bazy danych . . . . .	79
5.3.1	Model danych . . . . .	79

5.3.2	Diagram ERD . . . . .	79
5.4	Architektura interfejsu użytkownika . . . . .	79
5.4.1	Projekt strony głównej . . . . .	79
5.4.2	Projekt panelu logowania . . . . .	79
5.4.3	Projekt mapy . . . . .	79
5.4.4	Projekt chatu . . . . .	79
5.4.5	Projekt forum . . . . .	79
5.4.6	Projekt konta użytkownika . . . . .	79
<b>6</b>	<b>Przebieg realizacji projektu</b>	<b>80</b>
6.1	Sprint 1 . . . . .	80
6.2	Sprint 2 . . . . .	80
<b>7</b>	<b>Realizacja Projektu</b>	<b>81</b>
7.1	Implementacja backendu . . . . .	81
7.1.1	Struktura projektu . . . . .	81
7.1.2	Integracja z bazą danych . . . . .	81
7.1.3	Obsługa uwierzytelnienia . . . . .	81
7.1.4	Konteneryzacja . . . . .	81
7.2	Implementacja frontendu . . . . .	81
7.2.1	Struktura aplikacji . . . . .	81
7.2.2	Zarządzanie stanem i przepływ danych . . . . .	86
7.2.3	Integracja i komunikacja z backendem . . . . .	88
7.2.4	Style . . . . .	91
7.2.5	Wyszukiwarka spotów . . . . .	95
7.2.6	Mapa . . . . .	102
7.2.7	Chat . . . . .	102
7.2.8	Forum . . . . .	102
7.2.9	Konto użytkownika . . . . .	102
7.2.10	Panel logowania . . . . .	102
7.3	Implementacja CI/CD . . . . .	102

<b>8 Testy</b>	<b>103</b>
8.1 Testy jednostkowe . . . . .	103
8.2 Testy integracyjne . . . . .	103
8.3 Testy E2E . . . . .	103
8.4 Wyniki testów i wnioski . . . . .	103
<b>9 Prezentacja systemu</b>	<b>104</b>
9.1 Strona główna . . . . .	104
9.2 Strona mapy . . . . .	104
9.3 Strona chatu . . . . .	104
9.4 Strona forum . . . . .	104
9.5 Panel logowania . . . . .	104
9.6 Panel konta użytkownika . . . . .	104
<b>10 Nakład pracy</b>	<b>105</b>
10.1 Ogólny nakład pracy . . . . .	105
10.2 Indywidualne nakłady pracy . . . . .	105
10.2.1 Adam Langmesser . . . . .	105
10.2.2 Mateusz Redosz . . . . .	105
10.2.3 Stanisław Oziemczuk . . . . .	108
10.2.4 Kacper Badek . . . . .	108
<b>11 Podsumowanie</b>	<b>109</b>
11.1 Osiągnięte rezultaty . . . . .	109
11.2 Napotkane wyzwania . . . . .	109
11.3 Plany na przyszłość . . . . .	109
<b>12 Słownik pojęć i skrótów</b>	<b>110</b>
<b>Spis tabel</b>	<b>117</b>
<b>Bibliografia</b>	<b>121</b>
<b>Załączniki</b>	<b>123</b>

# Rozdział 1

## Wstęp

1.1 O projekcie

1.2 Cel i zakres prac

1.3 Geneza pomysłu



# Rozdział 2

## Opis problemu

### 2.1 Rich picture

### 2.2 Udziałowcy

KARTA UDZIAŁOWCA	
<b>Identyfikator:</b>	UO1
<b>Nazwa udziałowca:</b>	Zespół projektowy
<b>Opis:</b>	Zespół czterech studentów odpowiedzialnych za analizę, projekt, implementację, testy oraz dokumentację systemu.
<b>Typ:</b>	ożywiony, bezpośredni
<b>Perspektywa:</b>	Techniczna, wykonawcza.
<b>Ograniczenia:</b>	Ograniczone zasoby czasowe i doświadczenie komercyjne.
<b>Powiązane wymagania:</b>	Wymagania funkcjonalne i techniczne systemu, możliwość realizacji w ramach projektu dyplomowego.

**Tabela 2.1:** Karta udziałowca: Zespół projektowy

KARTA UDZIAŁOWCA	
<b>Identyfikator:</b>	UO2
<b>Nazwa udziałowca:</b>	Promotor
<b>Opis:</b>	Osoba nadzorująca przebieg projektu, weryfikująca poprawność merytoryczną i zgodność z wymaganiami uczelni.
<b>Typ:</b>	ożywiony, pośredni
<b>Perspektywa:</b>	Merytoryczna, formalna, jakościowa.
<b>Ograniczenia:</b>	Nie odpowiada za implementację; rekomenduje, opiniuje i zatwierdza.
<b>Powiązane wymagania:</b>	Czytelna dokumentacja, zgodność z wytycznymi kierunku oraz odpowiedni poziom techniczny rozwiązania.

**Tabela 2.2:** Karta udziałowca: Promotor

KARTA UDZIAŁOWCA	
<b>Identyfikator:</b>	UO3
<b>Nazwa udziałowca:</b>	<a href="#">Droniarze</a>
<b>Opis:</b>	Główna grupa docelowa systemu – osoby latające dronami rekreacyjnie lub półprofesjonalnie, szukające miejsc do lotów i wymiany doświadczeń.
<b>Typ:</b>	ożywiony, bezpośredni
<b>Perspektywa:</b>	Użytkownik końcowy: prostota obsługi, rzetelne informacje o spotach, wygodne dzielenie się treściami.
<b>Ograniczenia:</b>	Brak wpływu na architekturę techniczną systemu; oczekują intuicyjnego interfejsu.

<b>Powiązane wymagania:</b>	Lista spotów, informacje o ograniczeniach prawnych, oceny i komentarze, dodawanie treści oraz podstawowe funkcje społecznościowe.
-----------------------------	---

**Tabela 2.3:** Karta udziałowca: [Droniarze](#)

## **2.3 Istniejące rozwiązania**

## **2.4 Wizja rozwiązania**

## **2.5 Aspekty społeczne i biznesowe**

### **2.5.1 Aspekty społeczne**

### **2.5.2 Aspekty biznesowe**

# Rozdział 3

## Planowanie

### 3.1 Metodologia pracy

#### 3.1.1 Przegląd rozważanych podejść

Przy wyborze metodologii pracy rozważono trzy podejścia do prowadzenia projektu informatycznego:

- klasyczny Agile (w praktyce: Scrum),
- model kaskadowy (Waterfall),
- [Disciplined Agile Delivery - Lean Life Cycle](#).

#### 3.1.2 Odrzucone podejścia

**„Klasyczny Agile” (Scrum).** Mimo elastyczności i popularności zakłada pracę w iteracjach 2–4 tygodni oraz stały zestaw ceremonii (planowanie, przegląd, retrospektywa). Ze względu na nierównomierną dostępność zasobów w kolejnych miesiącach studiów nie zapewniono możliwości utrzymania stałej kadencji sprintów, dlatego z podejścia zrezygnowano.

**Model kaskadowy (Waterfall).** Przewiduje sekwencyjne przechodzenie przez z góry określone etapy i ogranicza bieżącą weryfikację wymagań w trakcie prac deweloperskich. W projekcie wymagano możliwości częstych rewizji założeń oraz

wprowadzania istotnych zmian w docelowej wizji rozwiązania; dlatego z podejścia zrezygnowano.

### 3.1.3 Wybrane podejście: Disciplined Agile Delivery (Lean Life Cycle)

Podjęto decyzję o zastosowaniu **Disciplined Agile Delivery** [1] w wariacie **Lean Life Cycle** [2], ponieważ podejście to łączy pożądane cechy Agile i Waterfall, a jednocześnie eliminuje stałe sprinty na rzecz pracy w ciągłym przepływie.

#### Kluczowe argumenty wyboru:

- **Brak sprintów.** Zastosowano przepływ ciągły, co pozwala dopasować tempo do zmiennej dostępności zespołu i unikać sztucznego „domykania” iteracji.
- **Rozbudowana faza startowa.** Na początku przewidziano większy wysiłek planistyczny: doprecyzowanie zakresu, wstępna wizja architektury, identyfikacja ryzyk, plan publikacji oraz kryteria jakości – bez zamrażania szczegółów.
- **Ciągła weryfikacja wymagań.** W trakcie realizacji przewidziano bieżące doprecyzowywanie backlogu, regularny feedback promotora oraz możliwość korygowania kierunku bez kosztów „przeskakiwania” między fazami.
- **Praktyki Lean i koncentracja na wartości.** Priorytetyzacja wartości biznesowej, wizualizacja pracy, małe partie dostaw.
- **Lekka governance i kamienie milowe.** Zastosowano lekkie mechanizmy nadzoru (peer review, prezentacje postępów) zapewniające przejrzystość bez nadmiernej biurokracji.

### 3.1.4 Narzędzia i komunikacja

Do zarządzania zadaniami zastosowana została **Jira** (monitorowanie postępu prac oraz ewidencja zadań członków zespołu). Komunikację w zespole zaplanowano w

formie regularnych spotkań oraz asynchronicznie z wykorzystaniem **Discorda** oraz **Messengera**.

### 3.1.5 Podział ról w zespole

- Adam - fullstack developer, lider zespołu
- Stanisław - fullstack developer
- Kacper - fullstack developer
- Mateusz - fullstack developer

Każdy z członków zespołu uczestniczy również w przygotowaniu dokumentacji.

## 3.2 Harmonogram projektu

W poniższym harmonogramie przedstawiono plan prac nad poszczególnymi częściami projektu, rozłożony na miesiące.

### Rok 2024

**Czerwiec**     • Zebranie zespołu.

- Rozważenie potencjalnych pomysłów.

**Lipiec**     • Wybór technologii.

- Wstępne założenia architektoniczne.

**Sierpień**     • *(do uzupełnienia)*

- *(do uzupełnienia)*

**Wrzesień**     • *(do uzupełnienia)*

- *(do uzupełnienia)*

**Październik**     • *(do uzupełnienia)*

- *(do uzupełnienia)*

**Listopad** • *(do uzupełnienia)*

- *(do uzupełnienia)*

**Grudzień** • *(do uzupełnienia)*

- *(do uzupełnienia)*

## **Rok 2025**

**Styczeń** • *(do uzupełnienia)*

- *(do uzupełnienia)*

**Luty** • *(do uzupełnienia)*

- *(do uzupełnienia)*

**Marzec** • *(do uzupełnienia)*

- *(do uzupełnienia)*

**Kwiecień** • *(do uzupełnienia)*

- *(do uzupełnienia)*

**Maj** • *(do uzupełnienia)*

- *(do uzupełnienia)*

**Czerwiec** • *(do uzupełnienia)*

- *(do uzupełnienia)*

**Lipiec** • *(do uzupełnienia)*

- *(do uzupełnienia)*

**Sierpień** • *(do uzupełnienia)*

- *(do uzupełnienia)*

**Wrzesień** • *(do uzupełnienia)*

- *(do uzupełnienia)*

**Październik** • *(do uzupełnienia)*

- *(do uzupełnienia)*

**Listopad** • *(do uzupełnienia)*

- *(do uzupełnienia)*

**Grudzień** • *(do uzupełnienia)*

- *(do uzupełnienia)*

**Rok 2026**

**Styczeń** • *(do uzupełnienia)*

- *(do uzupełnienia)*

### 3.3 Technologie i narzędzia

Do realizacji projektu wykorzystano wiele technologii oraz narzędzi informatycznych. Przy wyborze technologii kierowaliśmy się ich popularnością, dostępnością dokumentacji oraz artykułów, a także łatwością użycia. Narzędzia zostały dopasowane do wybranych technologii i specyfikacji zadań. Poniżej przedstawiono opis wybranych opcji.

#### 3.3.1 Technologie

#### 3.3.2 Narzędzia

Do niektórych płatnych narzędzi mieliśmy bezpłatny dostęp za pośrednictwem uczelni, w innych mogliśmy założyć konta edukacyjne, które oferowały dostęp do wszystkich funkcji narzędzia. Gdy żadna z wymienionych opcji nie była udostępniona, wybieraliśmy rozwiązania darmowe.

- **IntelliJ IDEA Ultimate**

Jest to [IDE](#) od firmy JetBrains. Dzięki licznie dostępnym pluginom oferuje obsługę wielu języków programowania oraz innych składni. Pozwala również



na integrację z repozytorium. Używaliśmy go do programowania zarówno [frontendu](#), jak i [backendu](#) oraz tworzenia dokumentacji w LaTeX.

- **Docker Desktop**

To narzędzie do zarządzania obrazami, kontenerami oraz wolumenami Docker. Zawiera w sobie również silnik tej technologii. Wykorzystywaliśmy je do lokalnego uruchamiania bazy danych oraz serwisu do cachowania.

- **One Drive**

Usługa dysku chmurowego oferowana przez firmę Microsoft. Przechowywaliśmy tam dokumenty oraz obrazy diagramów.

- **Azure Blob Storage**

To rozwiązanie chmurowe Microsoft, służące do bezpiecznego przechowywania dużej ilości danych nieustrukturyzowanych, takich jak pliki multimedialne, dokumenty czy kopie zapasowe. Dane są dostępne poprzez interfejs [REST API](#) usługi Azure Storage. Wykorzystaliśmy go do przechowywania zdjęć profilowych użytkownika oraz multimedii (zdjęcia i filmy) ze spotów i forum.

- **Jira**

To narzędzie firmy Atlassian do zarządzania pracami nad projektem w metodykach zwinnych. Do [Backlogu](#) wpisywaliśmy zadania, a na [tablicy Kanbanowej](#) rejestrowaliśmy ich statusy oraz poświęcony czas.

- **GitHub**

Zdalne repozytorium służące do przechowywania i wersjonowania kodu aplikacji. Zamieściliśmy tam kod naszego projektu. Do każdego zadania tworzyliśmy osobną gałąź z właściwą nazwą, a po zakończeniu prac przeprowadzaliśmy [review kodu](#). Następnie łączyliśmy ją do głównej gałęzi deweloperskiej.

- **GitHub Actions**

To narzędzie do implementacji procesów [CI/CD](#) na platformie GitHub, które umożliwiają automatyczne testowanie lub wdrażanie kodu. Uruchamiają się w reakcji na różne operacje w repozytorium, na przykład przesłanie zmian na wybraną gałąź. Stosowaliśmy je do automatycznego testowania i budowania projektu po każdorazowym wprowadzeniu zmian.

- **GitHub Copilot**

To narzędzie sztucznej inteligencji będące asystentem programisty. W projekcie analizuje plik oraz pliki powiązane. Wykorzystywaliśmy go podczas [review kodu](#). Copilot skanował wszystkie pliki i w komentarzach opisywał sugerowane zmiany lub potencjalne błędy.

- **Discord**

Darmowa platforma komunikacyjna. Umożliwia udostępnienie obrazu z ekranu, komunikację głosową oraz tekstową, jak i również przesyłanie plików. Stosowaliśmy go do spotkań, na których omawialiśmy sprawy dotyczące projektu.

- **Messenger**

Komunikator będący usługą Facebooka. Daje możliwość tworzenia czatów grupowych lub prywatnych, a także udostępniania plików. Używaliśmy go do ustalania spotkań na Discordzie oraz szybkiej komunikacji.

- **Postman**

To narzędzie służące do testowania endpointów [API](#). Pozwala grupować zapytania w kolekcje, wysyłać ich różne typy oraz analizować odpowiedzi z serwera. Wykorzystywaliśmy go do testowania stworzonych endpointów oraz debugowania.

- **Figma**

Narzędzie chmurowe do projektowania interfejsów użytkownika ([UI](#)). Umożliwia zespołowe tworzenie w pełni interaktywnych prototypów. Wykonaliśmy w nim projekty ekranów naszej aplikacji.

- **Visual Paradigm**

To narzędzie do tworzenia różnych diagramów stosowanych w inżynierii oprogramowania, takich jak [UML](#)( [3]) czy [BPMN](#)( [4]). Zrobiliśmy w nim diagram przypadków użycia.

## 3.4 Zasoby i ograniczenia

### 3.4.1 Zasoby

- **Specjalizacja członków zespołu** — wszyscy członkowie zespołu projektowego specjalizują się w aplikacjach internetowych.
- **Dostęp do przedstawiciela grupy docelowej** — jeden z członków zespołu (Adam) jest [droniarzem foto/video](#).
- **Status studenta** — fakt bycia studentem zapewnia dostęp do wersji premium wielu usług (Figma Education, GitHub PRO).
- **Oprogramowanie zapewniane przez PJATK** - uczelnia zapewnia dostęp do pakietu JetBrains oraz usług firmy Microsoft (OneDrive).

### 3.4.2 Ograniczenia

- **Ograniczenia czasowe** — projekt jest ograniczony harmonogramem akademickim i terminem oddania pracy dyplomowej, co wymagało wysokiego tempa realizacji oraz sprawnej komunikacji w zespole.
- **Ograniczenia budżetowe** — projekt nie posiada finansowania i w związku z tym korzystano z rozwiązań darmowych oraz open source.

### 3.4.3 Usługi zewnętrzne

Niniejszy rozdział zawiera spis zewnętrznych [API](#) oraz usług użytych w projekcie.

KARTA USŁUGI ZEWNĘTRZNEJ	
<b>Identyfikator:</b>	UZ1
<b>Nazwa:</b>	GitHub Actions (CI) [5]
<b>Opis:</b>	Uruchomienia pipeline'ów CI/CD dla repozytorium GitHub.
<b>Limit:</b>	3000 min/mies.

**Tabela 3.1:** Usługa zewnętrzna: GitHub Actions (CI)

KARTA USŁUGI ZEWNĘTRZNEJ	
<b>Identyfikator:</b>	UZ2
<b>Nazwa:</b>	Azure Blob Storage [6]
<b>Opis:</b>	Magazyn plików (m.in. zdjęcia spotów, załączniki z czatu).
<b>Limit:</b>	1 GB/mies.

**Tabela 3.2:** Usługa zewnętrzna: Azure Blob Storage

KARTA USŁUGI ZEWNĘTRZNEJ	
<b>Identyfikator:</b>	UZ3
<b>Nazwa:</b>	Mailtrap [7]
<b>Opis:</b>	Środowisko testowe SMTP oraz Email API do wysyłki maili.
<b>Limit:</b>	150 maili/dzień

**Tabela 3.3:** Usługa zewnętrzna: Mailtrap

KARTA USŁUGI ZEWNĘTRZNEJ	
<b>Identyfikator:</b>	UZ4

<b>Nazwa:</b>	LocationIQ [8]
<b>Opis:</b>	Geokodowanie adresu przy dodawaniu nowych spotów.
<b>Limit:</b>	5 000 zapytań/dzień

**Tabela 3.4:** Usługa zewnętrzna: LocationIQ

KARTA USŁUGI ZEWNĘTRZNEJ	
<b>Identyfikator:</b>	UZ5
<b>Nazwa:</b>	Google Maps (Maps URLs) [9]
<b>Opis:</b>	Otwieranie nawigacji w aplikacji Map Google (deep link/URL).
<b>Limit:</b>	Brak limitu w ramach dokumentowanego sposobu użycia.

**Tabela 3.5:** Usługa zewnętrzna: Google Maps (Maps URLs)

KARTA USŁUGI ZEWNĘTRZNEJ	
<b>Identyfikator:</b>	UZ6
<b>Nazwa:</b>	OpenFreeMap [10]
<b>Opis:</b>	Publiczny serwer kafelków do renderu mapy na froncie.
<b>Limit:</b>	30 000 zapytań/mies.

**Tabela 3.6:** Usługa zewnętrzna: OpenFreeMap

KARTA USŁUGI ZEWNĘTRZNEJ	
<b>Identyfikator:</b>	UZ7

<b>Nazwa:</b>	Open-Meteo [12]
<b>Opis:</b>	Prognozy pogody wyświetlane dla spotów.
<b>Limit:</b>	10 000 zapytań/dzień

**Tabela 3.7:** Usługa zewnętrzna: Open-Meteo

KARTA USŁUGI ZEWNĘTRZNEJ	
<b>Identyfikator:</b>	UZ8
<b>Nazwa:</b>	Tenor GIF API [13]
<b>Opis:</b>	Wyszukiwanie GIF-ów w czacie.
<b>Limit:</b>	1 zapytanie na sekundę; brak ogólnego limitu dziennego.

**Tabela 3.8:** Usługa zewnętrzna: Tenor GIF API

KARTA USŁUGI ZEWNĘTRZNEJ	
<b>Identyfikator:</b>	UZ9
<b>Nazwa:</b>	Where the ISS at? [14]
<b>Opis:</b>	HTTP API z bieżącą pozycją satelity, używane pomocniczo.
<b>Limit:</b>	1 zapytanie na sekundę; brak ogólnego limitu dziennego.

**Tabela 3.9:** Usługa zewnętrzna: Where the ISS at?

## 3.5 Analiza ryzyka

# Rozdział 4

## Analiza wymagań

W niniejszym rozdziale przedstawiono analizę wymagań stawianych aplikacji. Punktem wyjścia była identyfikacja i opracowanie przypadków użycia systemu (zarówno w postaci diagramów, jak i szczegółowych scenariuszy), które stanowią podstawę do dalszej specyfikacji zachowania systemu.

Na tej podstawie zdefiniowano kolejno:

- wymagania ogólne oraz dziedzinowe,
- wymagania funkcjonalne,
- wymagania pozafunkcjonalne,
- wymagania dotyczące interfejsu z otoczeniem,
- wymagania na środowisko docelowe.

Do określania priorytetów realizacji poszczególnych wymagań wykorzystano technikę [MoSCoW](#). Metoda ta wyróżnia cztery kategorie:

**M (Must have)** wymagania kluczowe, które muszą zostać zrealizowane, aby system mógł zostać uznany za działający poprawnie;

**S (Should have)** wymagania bardzo istotne, jednak niewpływające bezpośrednio na minimalną zdolność operacyjną systemu;

**C (Could have)** wymagania opcjonalne, podnoszące wygodę użytkownika lub wartość biznesową rozwiązania;

**W (Won't have this time)** wymagania odłożone na przyszłość, które w bieżącej iteracji nie będą realizowane.

Tak przeprowadzona analiza pozwala w sposób uporządkowany opisać zarówno zakres funkcjonalny systemu, jak i ograniczenia oraz oczekiwania нефunkcjonalne, a także zewnętrzne uwarunkowania jego działania.

## 4.1 Przypadki użycia

### 4.1.1 Aktorzy

**Użytkownik niezalogowany** Gość przeglądający publiczne treści (mapa, spoty, forum): może się zarejestrować lub zalogować.

**Użytkownik (nie premium)** Zarejestrowany użytkownik: zarządza kontem i ulubionymi spotami, dodaje treści/komentarze, korzysta z czatu.

**Użytkownik premium** Użytkownik z wykupioną subskrypcją: ma dostęp do funkcji premium (np. oznaczenie stref [PANSA](#) na mapie, rozbudowana prognoza pogody).

**Moderator** Moderacja treści: posty na forum, komentarze spotów.

**Deweloper** Rozwija i utrzymuje system.

**Aktorzy będący zewnętrznymi usługami** Poniżej wymieniono aktorów, których opisy zamieszczono w rozdziale poświęconym integracji z zewnętrznymi usługami [3.4.3](#).

- Usługa mailowa (Mailtrap)
- Dostawca API do map (OpenFreeMap)
- Nawigacja (Google Maps)



- Dostawca API pogodowego (Open-Meteo)
- Dostawca API GIFów (Tenor)
- Dostawca API do określania strefy czasowej spota (“Where the ISS at?”)
- Dostawca API do geolokalizacji (LocationIQ)
- Azure Blob Storage
- Dostawca OAuth (Google)
- Dostawca OAuth (GitHub)

#### 4.1.2 Diagram przypadków użycia

#### 4.1.3 Scenariusze przypadków użycia

Scenariusze przypadków użycia – funkcje ogólne

KARTA SCENARIUSZA PRZYPADKU UŻYCIA	
<b>Identyfikator:</b>	PU1
<b>Nazwa:</b>	Rejestracja użytkownika
<b>Priorytet:</b>	Wysoki
<b>Aktorzy:</b>	Użytkownik niezalogowany
<b>Opis:</b>	Użytkownik zakłada konto poprzez formularz rejestracji.
<b>Warunki wstępne:</b>	Użytkownik znajduje się na stronie z formularzem rejestracji.
<b>Warunki końcowe:</b>	Użytkownik posiada konto w systemie.

<b>Główny przepływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik wypełnia formularz rejestracyjny.</li> <li>2. Użytkownik naciska przycisk rejestracji.</li> <li>3. System tworzy konto użytkownika.</li> <li>4. System loguje użytkownika i przenosi go na ostatnio odwiedzoną podstronę.</li> </ol>
<b>Alternatywne przepływy zdarzeń:</b>	<ol style="list-style-type: none"> <li>1a. Podane dane są niepoprawne – system wyświetla komunikat o błędzie oraz podświetla pola wymagające poprawy.</li> <li>2a. Nazwa użytkownika jest już zajęta – system wyświetla komunikat o błędzie.</li> <li>2b. Adres email jest już zajęty – system wyświetla komunikat o błędzie.</li> </ol>

**Tabela 4.1:** Scenariusz przypadku użycia: Rejestracja użytkownika

<b>KARTA SCENARIUSZA PRZYPADKU UŻYCIA</b>	
<b>Identyfikator:</b>	PU2
<b>Nazwa:</b>	Logowanie użytkownika
<b>Priorytet:</b>	Wysoki
<b>Aktorzy:</b>	Użytkownik niezalogowany
<b>Opis:</b>	Użytkownik loguje się do systemu, podając login i hasło.
<b>Warunki wstępne:</b>	Użytkownik znajduje się na stronie logowania.
<b>Warunki końcowe:</b>	Użytkownik jest zalogowany i przeniesiony na ostatnio odwiedzoną podstronę.

<b>Główny przebieg zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik wypełnia formularz logowania.</li> <li>2. Użytkownik naciska przycisk logowania.</li> <li>3. System loguje użytkownika i przenosi go na ostatnio odwiedzoną podstronę.</li> </ol>
<b>Alternatywne przebiegi zdarzeń:</b>	<ol style="list-style-type: none"> <li>2a. Podane dane są niepoprawne – system wyświetla komunikat o błędzie.</li> </ol>

**Tabela 4.2:** Scenariusz przypadku użycia: Logowanie użytkownika

<b>KARTA SCENARIUSZA PRZYPADKU UŻYCIA</b>	
<b>Identyfikator:</b>	PU3
<b>Nazwa:</b>	Wykupienie subskrypcji premium
<b>Priorytet:</b>	Niski
<b>Aktorzy:</b>	Użytkownik zalogowany, Bramka płatnicza, System finansowo-księgowy
<b>Opis:</b>	Użytkownik opłaca subskrypcję premium w celu uzyskania dodatkowych funkcji.
<b>Warunki wstępne:</b>	Użytkownik jest zalogowany i znajduje się w module subskrypcji.
<b>Warunki końcowe:</b>	Subskrypcja premium jest aktywna, a użytkownik ma dostęp do funkcji premium.

<b>Główny przepływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik wybiera plan subskrypcji.</li> <li>2. Użytkownik przechodzi do bramki płatniczej.</li> <li>3. Użytkownik podaje dane płatnicze i zatwierdza transakcję.</li> <li>4. Bramka płatnicza przetwarza płatność i zwraca wynik do systemu.</li> <li>5. System zapisuje informację o opłaconej subskrypcji i aktualizuje uprawnienia.</li> <li>6. System generuje wpis w systemie finansowo-księgowym.</li> </ol>
<b>Alternatywne przepływy zdarzeń:</b>	<ol style="list-style-type: none"> <li>4a. Płatność nie powiodła się – system informuje użytkownika i umożliwia ponowną próbę.</li> <li>5a. W czasie aktualizacji subskrypcji wystąpił błąd – system cofa zmiany i wyświetla komunikat o problemie.</li> </ol>

**Tabela 4.3:** Scenariusz przypadku użycia: Wykupienie subskrypcji premium

<b>KARTA SCENARIUSZA PRZYPADKU UŻYCIA</b>	
<b>Identyfikator:</b>	PU4
<b>Nazwa:</b>	Resetowanie hasła
<b>Priorytet:</b>	Wysoki
<b>Aktorzy:</b>	Użytkownik niezalogowany, Usługa SMTP
<b>Opis:</b>	Użytkownik inicjuje reset hasła, aby odzyskać dostęp do konta.
<b>Warunki wstępne:</b>	Użytkownik znajduje się na ekranie resetu hasła.
<b>Warunki końcowe:</b>	Użytkownik otrzymuje wiadomość e-mail z linkiem do ustawienia nowego hasła.

<b>Główny przebieg zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik wpisuje adres e-mail powiązany z kontem.</li> <li>2. Użytkownik zatwierdza żądanie resetu hasła.</li> <li>3. System generuje token resetu hasła.</li> <li>4. System wysyła e-mail z linkiem do zmiany hasła.</li> </ol>
<b>Alternatywne przebiegi zdarzeń:</b>	<ol style="list-style-type: none"> <li>2a. Nie istnieje konto dla podanego adresu – system wyświetla komunikat o błędzie.</li> <li>4a. Występuje błąd połączenia z usługą SMTP – system informuje użytkownika o problemie technicznym.</li> </ol>

**Tabela 4.4:** Scenariusz przypadku użycia: Resetowanie hasła

<b>KARTA SCENARIUSZA PRZYPADKU UŻYCIA</b>	
<b>Identyfikator:</b>	PU5
<b>Nazwa:</b>	Zmiana hasła w ustawieniach konta
<b>Priorytet:</b>	Wysoki
<b>Aktorzy:</b>	Użytkownik zalogowany
<b>Opis:</b>	Użytkownik zmienia hasło do konta z poziomu ustawień profilu.
<b>Warunki wstępne:</b>	Użytkownik jest zalogowany i znajduje się na ekranie zmiany danych konta.
<b>Warunki końcowe:</b>	Hasło do konta użytkownika zostało zaktualizowane.

<b>Główny przepływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik wpisuje aktualne hasło.</li> <li>2. Użytkownik wpisuje nowe hasło i powtarza je.</li> <li>3. Użytkownik zatwierdza formularz zmiany hasła.</li> <li>4. System zapisuje nowe hasło i informuje o powodzeniu operacji.</li> </ol>
<b>Alternatywne przepływy zdarzeń:</b>	<ol style="list-style-type: none"> <li>3a. Aktualne hasło jest nieprawidłowe – system wyświetla komunikat i nie zapisuje zmian.</li> <li>3b. Nowe hasło nie spełnia wymagań bezpieczeństwa – system informuje o błędzie i podświetla pola do poprawy.</li> </ol>

**Tabela 4.5:** Scenariusz przypadku użycia: Zmiana hasła w ustawieniach konta

<b>KARTA SCENARIUSZA PRZYPADKU UŻYCIA</b>	
<b>Identyfikator:</b>	PU6
<b>Nazwa:</b>	Wylogowanie użytkownika
<b>Priorytet:</b>	Wysoki
<b>Aktorzy:</b>	Użytkownik zalogowany
<b>Opis:</b>	Użytkownik wylogowuje się z aplikacji.
<b>Warunki wstępne:</b>	Użytkownik jest zalogowany.
<b>Warunki końcowe:</b>	Sesja użytkownika została zakończona, użytkownik widzi stronę główną dla niezalogowanych.
<b>Główny przepływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik wybiera opcję wylogowania z menu.</li> <li>2. System unieważnia token dostępu użytkownika.</li> <li>3. System przenosi użytkownika na stronę główną aplikacji.</li> </ol>

<b>Alternatywne przebiegi zdarzeń:</b>	Brak istotnych alternatywnych przebiegów.
--	---

**Tabela 4.6:** Scenariusz przypadku użycia: Wylogowanie użytkownika

<b>KARTA SCENARIUSZA PRZYPADKU UŻYCIA</b>	
<b>Identyfikator:</b>	PU7
<b>Nazwa:</b>	Przeglądanie powiadomień
<b>Priorytet:</b>	Niski
<b>Aktorzy:</b>	Użytkownik zalogowany
<b>Opis:</b>	Użytkownik przegląda listę powiadomień.
<b>Warunki wstępne:</b>	Użytkownik jest na ekranie centra powiadomień.
<b>Warunki końcowe:</b>	Powiadomienia zostały wyświetlone, a wybrane oznaczone jako przeczytane.
<b>Główny przebieg zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. System wyświetla powiadomienia w odwróconym porządku chronologicznym.</li> <li>2. Użytkownik otwiera wybrane powiadomienie.</li> <li>3. System oznacza powiadomienie jako przeczytane i ewentualnie przenosi użytkownika do powiązanego widoku.</li> </ol>
<b>Alternatywne przebiegi zdarzeń:</b>	<ol style="list-style-type: none"> <li>1a. System nie może pobrać powiadomień (błąd serwera) – użytkownik otrzymuje komunikat o błędzie i może spróbować ponownie.</li> </ol>

**Tabela 4.7:** Scenariusz przypadku użycia: Przeglądanie powiadomień

## Scenariusze przypadków użycia dla funkcji premium

KARTA SCENARIUSZA PRZYPADKU UŻYCIA	
<b>Identyfikator:</b>	PU8
<b>Nazwa:</b>	Przeszukiwanie historii czatu
<b>Priorytet:</b>	Niski
<b>Aktorzy:</b>	Użytkownik premium
<b>Opis:</b>	Użytkownik wyszukuje konkretne wiadomości w historii czatu.
<b>Warunki wstępne:</b>	Użytkownik jest zalogowany jako użytkownik premium i znajduje się w widoku czatu.
<b>Warunki końcowe:</b>	Wiadomości spełniające kryteria wyszukiwania zostały wyświetlone.
<b>Główny przepływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik otwiera pole wyszukiwania historii w czacie.</li> <li>2. Użytkownik wpisuje frazę lub filtr (np. zakres dat, autor).</li> <li>3. System filtruje wiadomości zgodnie z kryteriami.</li> <li>4. System prezentuje listę dopasowanych fragmentów rozmowy.</li> </ol>
<b>Alternatywne przepływy zdarzeń:</b>	<ol style="list-style-type: none"> <li>4a. Nie znaleziono wiadomości spełniających kryteria – system informuje o braku wyników wyszukiwania.</li> </ol>

Tabela 4.8: Scenariusz przypadku użycia: Przeszukiwanie historii czatu

KARTA SCENARIUSZA PRZYPADKU UŻYCIA	
<b>Identyfikator:</b>	PU9
<b>Nazwa:</b>	Przeglądanie wysłanych plików na czacie



<b>Priorytet:</b>	Niski
<b>Aktorzy:</b>	Użytkownik premium, Usługa do przechowywania plików w chmurze
<b>Opis:</b>	Użytkownik przegląda listę plików wysłanych w ramach czatów.
<b>Warunki wstępne:</b>	Użytkownik jest zalogowany jako użytkownik premium.
<b>Warunki końcowe:</b>	Użytkownik widzi listę wysłanych plików i może przechodzić do powiązanych czatów.
<b>Główny przepływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik otwiera sekcję „Wysłane pliki”.</li> <li>2. System pobiera metadane plików z usługi przechowywania.</li> <li>3. System wyświetla listę plików z podstawowymi informacjami (nazwa, typ, data).</li> <li>4. Użytkownik wybiera plik, aby otworzyć go lub przejść do powiązanego czatu.</li> </ol>
<b>Alternatywne przepływy zdarzeń:</b>	4a. Doszło do błędu podczas pobierania pliku - system wyświetla odpowiedni komunikat.

**Tabela 4.9:** Scenariusz przypadku użycia: Przeglądanie wysłanych plików na czacie

KARTA SCENARIUSZA PRZYPADKU UŻYCIA	
<b>Identyfikator:</b>	PU10
<b>Nazwa:</b>	Zmiana typu mapy
<b>Priorytet:</b>	Niski

<b>Aktorzy:</b>	Użytkownik premium, Usługa do wyświetlania mapy
<b>Opis:</b>	Użytkownik zmienia typ mapy (np. standardowa, satelitarna, hybrydowa).
<b>Warunki wstępne:</b>	Użytkownik premium jest na ekranie mapy.
<b>Warunki końcowe:</b>	Mapa jest wyświetlana w wybranym typie.
<b>Główny przebieg zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik otwiera ustawienia widoku mapy.</li> <li>2. Użytkownik wybiera typ mapy z dostępnej listy.</li> <li>3. System przełącza widok mapy na wybrany typ.</li> </ol>
<b>Alternatywne przebiegi zdarzeń:</b>	3a. Wybrany typ mapy nie jest dostępny (błąd usługi mapowej) – system przywraca poprzedni typ i informuje o błędzie.

**Tabela 4.10:** Scenariusz przypadku użycia: Zmiana typu mapy

KARTA SCENARIUSZA PRZYPADKU UŻYCIA	
<b>Identyfikator:</b>	PU11
<b>Nazwa:</b>	Przeglądanie stref PANSA
<b>Priorytet:</b>	Niski
<b>Aktorzy:</b>	Użytkownik premium, Usługa do wyświetlania mapy
<b>Opis:</b>	Użytkownik wyświetla na mapie strefy przestrzeni powietrznej <b>PANSA</b> .
<b>Warunki wstępne:</b>	Użytkownik premium ma otwarty moduł mapy.

<b>Warunki końcowe:</b>	Strefy PANSA zostały zwizualizowane na mapie.
<b>Główny przepływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik włącza warstwę „Strefy PANSA”.</li> <li>2. System pobiera dane o strefach.</li> <li>3. System nakłada kontury stref na mapę.</li> </ol>
<b>Alternatywne przepływy zdarzeń:</b>	2a. Dane o strefach są chwilowo niedostępne – system komunikuje problem i nie włącza warstwy.

**Tabela 4.11:** Scenariusz przypadku użycia: Przeglądanie stref PANSA

#### Scenariusze przypadków użycia dla wyszukiwarki

KARTA SCENARIUSZA PRZYPADKU UŻYCIA	
<b>Identyfikator:</b>	PU12
<b>Nazwa:</b>	Wyszukiwanie spota w globalnej wyszukiwarce
<b>Priorytet:</b>	Wysoki
<b>Aktorzy:</b>	Użytkownik niezalogowany, Usługa do wyświetlania mapy, Usługa do pogody
<b>Opis:</b>	Użytkownik wyszukuje spoty za pomocą globalnej wyszukiwarki w aplikacji.
<b>Warunki wstępne:</b>	Użytkownik znajduje się na stronie głównej z wyszukiwarką.
<b>Warunki końcowe:</b>	Użytkownik otrzymuje listę znalezionych spotów.

<b>Główny przepływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik wpisuje frazę w globalnej wyszukiwarce.</li> <li>2. System wyszukuje spoty spełniające kryteria.</li> <li>3. System wyświetla listę wyników.</li> </ol>
<b>Alternatywne przepływy zdarzeń:</b>	<ol style="list-style-type: none"> <li>3a. System informuje o braku wyników spełniających kryteria.</li> </ol>

**Tabela 4.12:** Scenariusz przypadku użycia: Wyszukiwanie spota w globalnej wyszukiwarce

<b>KARTA SCENARIUSZA PRZYPADKU UŻYCIA</b>	
<b>Identyfikator:</b>	PU13
<b>Nazwa:</b>	Przejsie do spota na mapie z wyszukiwarki
<b>Priorytet:</b>	Wysoki
<b>Aktorzy:</b>	Użytkownik niezalogowany
<b>Opis:</b>	Użytkownik przechodzi z wyników wyszukiwarki do widoku mapy ustawionego na konkretny spot.
<b>Warunki wstępne:</b>	Wyświetlona jest lista wyników wyszukiwania spotów.
<b>Warunki końcowe:</b>	Mapa jest przybliżona do wybranego spota, a jego szczegóły są dostępne.
<b>Główny przepływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik wybiera spota z listy wyników.</li> <li>2. System przełącza widok na moduł mapy.</li> <li>3. System ustawia mapę na lokalizację spota i otwiera jego szczegóły.</li> </ol>

<b>Alternatywne przebiegi zdarzeń:</b>	Brak istotnych alternatywnych przebiegów.
--	---

**Tabela 4.13:** Scenariusz przypadku użycia: Przejście do spota na mapie z wyszukiwarki

#### Scenariusze przypadków użycia dla mapy

KARTA SCENARIUSZA PRZYPADKU UŻYCIA	
<b>Identyfikator:</b>	PU14
<b>Nazwa:</b>	Przeglądanie mapy spotów
<b>Priorytet:</b>	Wysoki
<b>Aktorzy:</b>	Użytkownik niezalogowany, Usługa do wyświetlania mapy
<b>Opis:</b>	Użytkownik przegląda mapę spotów.
<b>Warunki wstępne:</b>	Użytkownik znajduje się w module mapy.
<b>Warunki końcowe:</b>	Mapa ze spotami została wyświetlona, a użytkownik może przybliżać, oddalać i przesuwać widok.
<b>Główny przebieg zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. System inicjuje widok mapy z domyślnym obszarem.</li> <li>2. System pobiera listę spotów.</li> <li>3. System rysuje znaczniki spotów na mapie.</li> <li>4. Użytkownik przesuwa lub skaluje mapę.</li> <li>5. System pobiera spoty dla nowego zakresu.</li> </ol>
<b>Alternatywne przebiegi zdarzeń:</b>	<ol style="list-style-type: none"> <li>2a. Usługa mapy jest niedostępna – system wyświetla komunikat o błędzie.</li> </ol>

**Tabela 4.14:** Scenariusz przypadku użycia: Przeglądanie mapy spotów

KARTA SCENARIUSZA PRZYPADKU UŻYCIA	
<b>Identyfikator:</b>	PU15
<b>Nazwa:</b>	Otwarcie szczegółów spota
<b>Priorytet:</b>	Wysoki
<b>Aktorzy:</b>	Użytkownik niezalogowany, Użytkownik zalogowany
<b>Opis:</b>	Użytkownik otwiera widok szczegółów wybranego spota.
<b>Warunki wstępne:</b>	Użytkownik widzi mapę spotów.
<b>Warunki końcowe:</b>	Wyświetlony został widok szczegółów spota z podstawowymi informacjami oraz jego lokalizacją na mapie.
<b>Główny przepływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik wybiera spota z mapy.</li> <li>2. System pobiera dane szczegółowe spota (informacje opisowe, lokalizacja).</li> <li>3. System otwiera widok szczegółów spota.</li> <li>4. System prezentuje informacje o spocie oraz mapę przybliżoną do jego lokalizacji.</li> </ol>
<b>Alternatywne przepływy zdarzeń:</b>	<ol style="list-style-type: none"> <li>2a. Spot nie istnieje (został usunięty lub ukryty) – system informuje użytkownika i powraca do poprzedniego widoku.</li> <li>2b. Wystąpił błąd podczas pobierania danych spota – system wyświetla komunikat o błędzie i umożliwia ponowną próbę.</li> </ol>

**Tabela 4.15:** Scenariusz przypadku użycia: Otwarcie szczegółów spota

KARTA SCENARIUSZA PRZYPADKU UŻYCIA	
<b>Identyfikator:</b>	PU16
<b>Nazwa:</b>	Przeglądanie komentarzy do spota

<b>Priorytet:</b>	Średni
<b>Aktorzy:</b>	Użytkownik niezalogowany
<b>Opis:</b>	Użytkownik czyta komentarze pod wybranym spotem.
<b>Warunki wstępne:</b>	Wyświetlany jest widok szczegółów spota.
<b>Warunki końcowe:</b>	Lista komentarzy do spota została wyświetlona.
<b>Główny przepływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. System pobiera komentarze powiązane ze spotem.</li> <li>2. System wyświetla komentarze w kolejności chronologicznej lub według popularności.</li> <li>3. Użytkownik przewija listę komentarzy.</li> </ol>
<b>Alternatywne przepływy zdarzeń:</b>	<ol style="list-style-type: none"> <li>1a. Spot nie ma jeszcze komentarzy – system wyświetla odpowiednią informację.</li> </ol>

**Tabela 4.16:** Scenariusz przypadku użycia: Przeglądanie komentarzy do spota

KARTA SCENARIUSZA PRZYPADKU UŻYCIA	
<b>Identyfikator:</b>	PU17
<b>Nazwa:</b>	Przeglądanie pogody na spocie
<b>Priorytet:</b>	Średni
<b>Aktorzy:</b>	Użytkownik zalogowany, Usługa danych pogodowych
<b>Opis:</b>	Użytkownik sprawdza prognozę pogody dla lokalizacji spota.
<b>Warunki wstępne:</b>	Wyświetlany jest widok szczegółów spota.

<b>Warunki końcowe:</b>	Prognoza pogody dla spota została wyświetlona.
<b>Główny przepływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik otwiera zakładkę pogody.</li> <li>2. System wysyła zapytanie do usługi pogodowej z lokalizacją spota.</li> <li>3. System odbiera prognozę i prezentuje ją (temperatura, prędkość wiatru, opady).</li> </ol>
<b>Alternatywne przepływy zdarzeń:</b>	2a. Usługa pogodowa jest niedostępna – system wyświetla komunikat o braku danych pogodowych.

**Tabela 4.17:** Scenariusz przypadku użycia: Przeglądanie pogody na spocie

KARTA SCENARIUSZA PRZYPADKU UŻYCIA	
<b>Identyfikator:</b>	PU18
<b>Nazwa:</b>	Wyszukiwanie spota na mapie
<b>Priorytet:</b>	Wysoki
<b>Aktorzy:</b>	Użytkownik niezalogowany
<b>Opis:</b>	Użytkownik wyszukuje spota po nazwie korzystając z pola wyszukiwania na mapie.
<b>Warunki wstępne:</b>	Użytkownik widzi mapę spotów.
<b>Warunki końcowe:</b>	Mapa zostaje ustawiona na wybranego spota lub listę dopasowań.



<b>Główny przepływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik wpisuje frazę w polu wyszukiwania na mapie.</li> <li>2. System podpowiada listę pasujących spotów.</li> <li>3. Użytkownik wybiera spota z listy.</li> <li>4. System przybliża mapę do wybranego spota i podświetla jego znacznik.</li> </ol>
<b>Alternatywne przepływy zdarzeń:</b>	<ol style="list-style-type: none"> <li>2a. Brak wyników dla podanej frazy – system informuje użytkownika o braku dopasowań.</li> </ol>

**Tabela 4.18:** Scenariusz przypadku użycia: Wyszukiwanie spota na mapie

#### Scenariusze przypadków użycia dla czatu

<b>KARTA SCENARIUSZA PRZYPADKU UŻYCIA</b>	
<b>Identyfikator:</b>	PU19
<b>Nazwa:</b>	Utworzenie prywatnego czatu
<b>Priorytet:</b>	Wysoki
<b>Aktorzy:</b>	Użytkownik zalogowany
<b>Opis:</b>	Użytkownik tworzy prywatną konwersację z innym użytkownikiem.
<b>Warunki wstępne:</b>	Użytkownik jest zalogowany i znajduje się w zakładce społeczności.
<b>Warunki końcowe:</b>	Nowy czat prywatny został utworzony i wyświetlony użytkownikowi.
<b>Główny przepływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik wybiera opcję utworzenia nowego czatu.</li> <li>2. System tworzy nowy czat (jeśli nie istnieje).</li> <li>3. System otwiera widok nowego czatu.</li> </ol>

<b>Alternatywne przebiegi zdarzeń:</b>	1a. Taki czat już istnieje – system zamiast tworzyć nowy, otwiera istniejącą konwersację.
--	---

**Tabela 4.19:** Scenariusz przypadku użycia: Utworzenie prywatnego czatu

<b>KARTA SCENARIUSZA PRZYPADKU UŻYCIA</b>	
<b>Identyfikator:</b>	PU20
<b>Nazwa:</b>	Otworzenie czatu
<b>Priorytet:</b>	Wysoki
<b>Aktorzy:</b>	Użytkownik zalogowany
<b>Opis:</b>	Użytkownik otwiera wybrany czat, aby wyświetlić historię rozmowy i móc wysyłać kolejne wiadomości.
<b>Warunki wstępne:</b>	Użytkownik jest zalogowany i widzi listę swoich czatów lub otrzymał powiadomienie prowadzące do czatu.
<b>Warunki końcowe:</b>	Wybrany czat został otworzony, a historia rozmowy jest widoczna dla użytkownika.
<b>Główny przebieg zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik wybiera czat z listy czatów lub z powiadomienia.</li> <li>2. System pobiera dane czatu (uczestników, ostatnie wiadomości).</li> <li>3. System oznacza nieprzeczytane wiadomości na czacie jako przeczytane.</li> <li>4. System wyświetla widok czatu wraz z historią rozmowy.</li> </ol>

<b>Alternatywne przebiegi zdarzeń:</b>	<p>2a. Czat nie jest już dostępny (np. został usunięty lub użytkownik utracił do niego dostęp) – system wyświetla komunikat o braku dostępu i powraca do listy czatów.</p> <p>2b. Wystąpił błąd podczas pobierania danych czatu – system wyświetla komunikat o błędzie i umożliwia ponowną próbę.</p>
--	---

**Tabela 4.20:** Scenariusz przypadku użycia: Otworzenie czatu

<b>KARTA SCENARIUSZA PRZYPADKU UŻYCIA</b>	
<b>Identyfikator:</b>	PU21
<b>Nazwa:</b>	Utworzenie czatu grupowego
<b>Priorytet:</b>	Średni
<b>Aktorzy:</b>	Użytkownik zalogowany
<b>Opis:</b>	Użytkownik tworzy nowy czat grupowy z kilkoma uczestnikami.
<b>Warunki wstępne:</b>	Użytkownik jest zalogowany i znajduje się na dowolnym czacie prywatnym.
<b>Warunki końcowe:</b>	Czat grupowy został utworzony i wyświetlony na ekranie.
<b>Główny przebieg zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik wybiera opcję utworzenia czatu grupowego.</li> <li>2. Użytkownik wybiera uczestników grupy.</li> <li>3. Użytkownik zatwierdza utworzenie czatu.</li> <li>4. System tworzy czat grupowy i dodaje do niego wskazanych użytkowników.</li> <li>5. System otwiera widok nowego czatu grupowego.</li> </ol>

<b>Alternatywne przebiegi zdarzeń:</b>	3a. System nie może utworzyć czatu – aplikacja informuje o błędzie.
--	---

**Tabela 4.21:** Scenariusz przypadku użycia: Utworzenie czatu grupowego

<b>KARTA SCENARIUSZA PRZYPADKU UŻYCIA</b>	
<b>Identyfikator:</b>	PU22
<b>Nazwa:</b>	Przeglądanie listy czatów
<b>Priorytet:</b>	Wysoki
<b>Aktorzy:</b>	Użytkownik zalogowany
<b>Opis:</b>	Użytkownik przegląda listę swoich czatów prywatnych i grupowych.
<b>Warunki wstępne:</b>	Użytkownik jest zalogowany i otwiera moduł czatu.
<b>Warunki końcowe:</b>	Lista czatów użytkownika została wyświetlona.
<b>Główny przebieg zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. System pobiera listę czatów użytkownika.</li> <li>2. System wyświetla listę czatów z podstawowymi informacjami.</li> <li>3. Użytkownik wybiera czat z listy.</li> <li>4. System otwiera widok wybranego czatu.</li> </ol>
<b>Alternatywne przebiegi zdarzeń:</b>	Brak istotnych alternatywnych przebiegów.

**Tabela 4.22:** Scenariusz przypadku użycia: Przeglądanie listy czatów

KARTA SCENARIUSZA PRZYPADKU UŻYCIA	
<b>Identyfikator:</b>	PU23
<b>Nazwa:</b>	Wysyłanie wiadomości na czacie
<b>Priorytet:</b>	Wysoki
<b>Aktorzy:</b>	Użytkownik zalogowany
<b>Opis:</b>	Użytkownik wysyła wiadomość tekstową na czacie.
<b>Warunki wstępne:</b>	Użytkownik jest zalogowany i znajduje się w widoku konkretnego czatu.
<b>Warunki końcowe:</b>	Nowa wiadomość jest zapisana i widoczna w historii czatu.
<b>Główny przepływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik wpisuje treść wiadomości.</li> <li>2. Użytkownik wysyła wiadomość.</li> <li>3. System zapisuje wiadomość i dostarcza ją do uczestników czatu.</li> <li>4. System wyświetla wiadomość na liście wiadomości.</li> </ol>
<b>Alternatywne przepływy zdarzeń:</b>	2a. Treść wiadomości jest pusta – system blokuje wysłanie i pozostaje w tym samym widoku.

**Tabela 4.23:** Scenariusz przypadku użycia: Wysyłanie wiadomości na czacie

KARTA SCENARIUSZA PRZYPADKU UŻYCIA	
<b>Identyfikator:</b>	PU24
<b>Nazwa:</b>	Wysyłanie GIF-a na czacie
<b>Priorytet:</b>	Średni
<b>Aktorzy:</b>	Użytkownik zalogowany, Usługa GIF-ów
<b>Opis:</b>	Użytkownik wysyła animację GIF w konwersacji czatowej.

<b>Warunki wstępne:</b>	Użytkownik jest zalogowany i znajduje się w widoku czatu.
<b>Warunki końcowe:</b>	Wybrany GIF został dodany jako wiadomość w czacie.
<b>Główny przepływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik wybiera opcję dodania GIF-a.</li> <li>2. System otwiera okno wyszukiwarki GIF-ów.</li> <li>3. Użytkownik wybiera lub wyszukuje GIF-a.</li> <li>4. Użytkownik zatwierdza wysłanie GIF-a.</li> <li>5. System dodaje GIF-a jako wiadomość na czacie.</li> </ol>
<b>Alternatywne przepływy zdarzeń:</b>	2a. Usługa GIF-ów jest niedostępna – system informuje o braku możliwości wysłania GIF-a.

**Tabela 4.24:** Scenariusz przypadku użycia: Wysyłanie GIF-a na czacie

<b>KARTA SCENARIUSZA PRZYPADKU UŻYCIA</b>	
<b>Identyfikator:</b>	PU25
<b>Nazwa:</b>	Wysyłanie pliku na czacie
<b>Priorytet:</b>	Średni
<b>Aktorzy:</b>	Użytkownik zalogowany, Usługa do przechowywania plików w chmurze
<b>Opis:</b>	Użytkownik wysyła plik (np. zdjęcie, film) na czacie.
<b>Warunki wstępne:</b>	Użytkownik jest zalogowany i znajduje się w widoku czatu.
<b>Warunki końcowe:</b>	Plik został zapisany w chmurze i powiązany z wiadomością na czacie.

<b>Główny przebieg zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik wybiera opcję dodania pliku.</li> <li>2. Użytkownik wybiera plik z urządzenia.</li> <li>3. System przesyła plik do usługi przechowywania w chmurze.</li> <li>4. System tworzy wiadomość z odnośnikiem do pliku.</li> <li>5. System wyświetla wiadomość na liście czatu.</li> </ol>
<b>Alternatywne przebiegi zdarzeń:</b>	<ol style="list-style-type: none"> <li>3a. Przesyłanie pliku nie powiodło się – system informuje użytkownika i umożliwia ponowną próbę.</li> </ol>

**Tabela 4.25:** Scenariusz przypadku użycia: Wysyłanie pliku na czacie

<b>KARTA SCENARIUSZA PRZYPADKU UŻYCIA</b>	
<b>Identyfikator:</b>	PU26
<b>Nazwa:</b>	Edycja ustawień czatu
<b>Priorytet:</b>	Niski
<b>Aktorzy:</b>	Użytkownik zalogowany
<b>Opis:</b>	Użytkownik modyfikuje ustawienia czatu (np. nazwę, avatar, tryb powiadomień).
<b>Warunki wstępne:</b>	Użytkownik jest zalogowany i ma uprawnienia do edycji danego czatu.
<b>Warunki końcowe:</b>	Zaktualizowane ustawienia czatu są zapisane i widoczne dla uczestników.
<b>Główny przebieg zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik otwiera panel ustawień czatu.</li> <li>2. Użytkownik wprowadza zmiany (np. nazwę, opis, avatar).</li> <li>3. Użytkownik zapisuje zmiany.</li> <li>4. System waliduje dane i aktualizuje konfigurację czatu.</li> </ol>

<b>Alternatywne przebiegi zdarzeń:</b>	Brak istotnych alternatywnych przebiegów poza walidacją pól.
--	--

**Tabela 4.26:** Scenariusz przypadku użycia: Edycja ustawień czatu

<b>KARTA SCENARIUSZA PRZYPADKU UŻYCIA</b>	
<b>Identyfikator:</b>	PU27
<b>Nazwa:</b>	Dodanie członka do czatu grupowego
<b>Priorytet:</b>	Średni
<b>Aktorzy:</b>	Użytkownik zalogowany
<b>Opis:</b>	Użytkownik z uprawnieniami administratora dodaje nowego uczestnika do czatu grupowego.
<b>Warunki wstępne:</b>	Użytkownik jest zalogowany, znajduje się w czacie grupowym i ma prawo zarządzać członkami.
<b>Warunki końcowe:</b>	Nowy uczestnik został dodany do czatu grupowego.
<b>Główny przebieg zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik otwiera listę uczestników czatu grupowego.</li> <li>2. Użytkownik wybiera opcję dodania nowego członka.</li> <li>3. Użytkownik wskazuje użytkownika do dodania i zatwierdza wybór.</li> <li>4. System dodaje wskazanego użytkownika do czatu grupowego.</li> </ol>
<b>Alternatywne przebiegi zdarzeń:</b>	3a. Operacja nie powiodła się – system informuje o błędzie.

**Tabela 4.27:** Scenariusz przypadku użycia: Dodanie członka do czatu grupowego



## Scenariusze przypadków użycia dla forum

KARTA SCENARIUSZA PRZYPADKU UŻYCIA	
<b>Identyfikator:</b>	PU28
<b>Nazwa:</b>	Przeglądanie postów na forum
<b>Priorytet:</b>	Wysoki
<b>Aktorzy:</b>	Użytkownik niezalogowany, Użytkownik zalogowany
<b>Opis:</b>	Użytkownik przegląda listę postów na forum z możliwością sortowania wyników.
<b>Warunki wstępne:</b>	Użytkownik znajduje się w module forum.
<b>Warunki końcowe:</b>	Lista postów forum jest wyświetlona, a użytkownik może przechodzić do szczegółów wybranego posta.
<b>Główny przepływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik przechodzi do widoku listy postów forum.</li> <li>2. System pobiera listę postów i domyślnie wyświetla je w kolejności od najnowszych.</li> <li>3. Użytkownik wybiera sposób sortowania listy.</li> <li>4. System aktualizuje listę postów zgodnie z wybranym kryterium sortowania.</li> <li>5. Użytkownik wybiera post, który chce przeczytać.</li> <li>6. System otwiera szczegółowy widok wybranego posta.</li> </ol>
<b>Alternatywne przepływy zdarzeń:</b>	<ol style="list-style-type: none"> <li>6a. System nie może pobrać szczegółów posta – system wyświetla komunikat o błędzie.</li> </ol>

**Tabela 4.28:** Scenariusz przypadku użycia: Przeglądanie postów na forum

## KARTA SCENARIUSZA PRZYPADKU UŻYCIA

<b>Identyfikator:</b>	PU29
<b>Nazwa:</b>	Wyszukiwanie postów na forum
<b>Priorytet:</b>	Średni
<b>Aktorzy:</b>	Użytkownik niezalogowany, Użytkownik zalogowany
<b>Opis:</b>	Użytkownik wyszukuje posty na forum na podstawie tytułu, kategorii, tagów oraz autora.
<b>Warunki wstępne:</b>	Użytkownik znajduje się w module forum lub na stronie wyszukiwarki postów.
<b>Warunki końcowe:</b>	Lista postów spełniających zadane kryteria wyszukiwania jest wyświetlona.
<b>Główny przebieg zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik otwiera panel wyszukiwania postów na forum.</li> <li>2. Użytkownik określa kryteria wyszukiwania (np. tytuł, kategoria, tagi, autor).</li> <li>3. Użytkownik uruchamia wyszukiwanie.</li> <li>4. System filtruje posty zgodnie z podanymi kryteriami i wyświetla listę wyników.</li> </ol>
<b>Alternatywne przebiegi zdarzeń:</b>	<ol style="list-style-type: none"> <li>4a. Brak postów spełniających zadane kryteria – system wyświetla informację o braku wyników.</li> <li>4b. Wystąpił błąd podczas wyszukiwania – system wyświetla komunikat o błędzie i umożliwia ponowną próbę.</li> </ol>

**Tabela 4.29:** Scenariusz przypadku użycia: Wyszukiwanie postów na forum

KARTA SCENARIUSZA PRZYPADKU UŻYCIA	
<b>Identyfikator:</b>	PU30
<b>Nazwa:</b>	Dodanie posta na forum

<b>Priorytet:</b>	Wysoki
<b>Aktorzy:</b>	Użytkownik zalogowany, Usługa do przechowywania plików w chmurze
<b>Opis:</b>	Użytkownik publikuje nowy post na forum, określając jego treść, kategorię, tagi oraz opcjonalne załączniki.
<b>Warunki wstępne:</b>	Użytkownik znajduje się w module forum.
<b>Warunki końcowe:</b>	Nowy post jest poprawnie zapisany i widoczny na forum.
<b>Główny przebieg zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik wybiera opcję dodania nowego posta.</li> <li>2. Użytkownik wpisuje tytuł i treść posta.</li> <li>3. Użytkownik wybiera kategorię posta.</li> <li>4. (Opcjonalnie) Użytkownik wybiera tagi przypisane do posta.</li> <li>5. (Opcjonalnie) Użytkownik dodaje załączniki (zdjęcia/filmy) do posta.</li> <li>6. Użytkownik publikuje posta.</li> <li>7. System zapisuje posta (oraz poprawne załączniki w chmurze) i wyświetla go na liście postów.</li> </ol>
<b>Alternatywne przebiegi zdarzeń:</b>	<ol style="list-style-type: none"> <li>6a. Załącznik nie może zostać zapisany lub nie spełnia wymagań (np. zbyt duży rozmiar, nieobsługiwany format) – system informuje użytkownika o błędzie, blokuje publikację posta i wymaga usunięcia lub podmiany problematycznego pliku.</li> <li>6b. Formularz zawiera błędne lub niekompletne dane (np. brak tytułu lub treści) – system wyświetla komunikat i prosi o poprawę danych przed publikacją.</li> </ol>

**Tabela 4.30:** Scenariusz przypadku użycia: Dodanie posta na forum

KARTA SCENARIUSZA PRZYPADKU UŻYCIA	
<b>Identyfikator:</b>	PU31
<b>Nazwa:</b>	Dodanie komentarza na forum
<b>Priorytet:</b>	Wysoki
<b>Aktorzy:</b>	Użytkownik zalogowany
<b>Opis:</b>	Użytkownik dodaje komentarz pod postem na forum, opcjonalnie z załącznikami (zdjęcia/filmy).
<b>Warunki wstępne:</b>	Użytkownik jest zalogowany i widzi szczegóły posta.
<b>Warunki końcowe:</b>	Nowy komentarz został zapisany i jest widoczny pod postem.
<b>Główny przepływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik wpisuje treść komentarza w formularzu pod postem.</li> <li>2. (Opcjonalnie) Użytkownik dodaje załączniki (zdjęcia/filmy) do komentarza.</li> <li>3. Użytkownik publikuje komentarz.</li> <li>4. System zapisuje komentarz (oraz poprawne załączniki) i odświeża listę komentarzy.</li> </ol>
<b>Alternatywne przepływy zdarzeń:</b>	<ol style="list-style-type: none"> <li>3a. Treść komentarza lub załączniki są niepoprawne (np. naruszają walidację) – system wyświetla komunikat o błędzie i blokuje publikację do czasu poprawy danych.</li> </ol>

**Tabela 4.31:** Scenariusz przypadku użycia: Dodanie komentarza na forum

KARTA SCENARIUSZA PRZYPADKU UŻYCIA	
<b>Identyfikator:</b>	PU32
<b>Nazwa:</b>	Przeglądanie historii interakcji z postami

<b>Priorytet:</b>	Średni
<b>Aktorzy:</b>	Użytkownik zalogowany
<b>Opis:</b>	Użytkownik przegląda historię swoich aktywności na forum (dodane posty, komentarze, reakcje).
<b>Warunki wstępne:</b>	Użytkownik jest zalogowany.
<b>Warunki końcowe:</b>	Lista interakcji użytkownika z postami jest wyświetlona.
<b>Główny przepływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik przechodzi do sekcji historii aktywności.</li> <li>2. System pobiera historię interakcji użytkownika.</li> <li>3. System wyświetla listę interakcji z możliwością filtrowania.</li> </ol>
<b>Alternatywne przepływy zdarzeń:</b>	Brak istotnych alternatywnych przepływów.

**Tabela 4.32:** Scenariusz przypadku użycia: Przeglądanie historii interakcji z postami

KARTA SCENARIUSZA PRZYPADKU UŻYCIA	
<b>Identyfikator:</b>	PU33
<b>Nazwa:</b>	Zarządzanie komentarzami na forum
<b>Priorytet:</b>	Niski
<b>Aktorzy:</b>	Użytkownik zalogowany (autor posta lub moderator)
<b>Opis:</b>	Użytkownik zarządza komentarzami pod postami forum (edycja, usuwanie, zgłaszanie komentarzy innych użytkowników).

<b>Warunki wstępne:</b>	Użytkownik jest zalogowany i ma dostęp do danego wątku forum.
<b>Warunki końcowe:</b>	Komentarze zostały zaktualizowane zgodnie z działaniami użytkownika.
<b>Główny przepływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik otwiera widok komentarzy pod postem.</li> <li>2. Użytkownik wybiera komentarz i odpowiednią akcję (edycja, usunięcie, zgłoszenie).</li> <li>3. System weryfikuje uprawnienia użytkownika oraz zgodność akcji z jego rolą.</li> <li>4. System wykonuje wybraną akcję (np. zapisuje zmiany, usuwa komentarz lub przygotowuje zgłoszenie) i aktualizuje widok.</li> </ol>
<b>Alternatywne przepływy zdarzeń:</b>	<ol style="list-style-type: none"> <li>2a. Użytkownik próbuje zgłosić własny komentarz – system blokuje operację i informuje, że nie można zgłaszać własnych treści.</li> <li>3a. Użytkownik nie ma wymaganych uprawnień do wykonania wybranej akcji – system blokuje operację i informuje o braku uprawnień.</li> </ol>

**Tabela 4.33:** Scenariusz przypadku użycia: Zarządzanie komentarzami na forum

KARTA SCENARIUSZA PRZYPADKU UŻYCIA	
<b>Identyfikator:</b>	PU34
<b>Nazwa:</b>	Zgłoszenie komentarza naruszającego regulamin
<b>Priorytet:</b>	Średni
<b>Aktorzy:</b>	Użytkownik zalogowany

<b>Opis:</b>	Użytkownik zgłasza komentarz na forum jako naruszający regulamin.
<b>Warunki wstępne:</b>	Użytkownik widzi komentarz w aplikacji.
<b>Warunki końcowe:</b>	Zgłoszenie komentarza zostało zapisane i trafiło do kolejki moderacyjnej.
<b>Główny przepływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik wybiera opcję „Zgłoś komentarz”.</li> <li>2. Użytkownik wybiera kategorię naruszenia, podaje szczegóły i potwierdza zgłoszenie.</li> <li>3. System zapisuje zgłoszenie i wiąże je z komentarzem oraz zgłaszającym użytkownikiem.</li> </ol>
<b>Alternatywne przepływy zdarzeń:</b>	<ol style="list-style-type: none"> <li>1a. Użytkownik próbuje zgłosić własny komentarz – system blokuje operację i wyświetla komunikat, że nie można zgłaszać własnych treści.</li> <li>3a. Komentarz został już wcześniej zgłoszony – system informuje użytkownika, że komentarz znajduje się już w kolejce moderacyjnej i nie zapisuje kolejnego zgłoszenia.</li> </ol>

**Tabela 4.34:** Scenariusz przypadku użycia: Zgłoszenie komentarza naruszającego regulamin

KARTA SCENARIUSZA PRZYPADKU UŻYCIA	
<b>Identyfikator:</b>	PU35
<b>Nazwa:</b>	Zgłoszenie posta na forum
<b>Priorytet:</b>	Średni
<b>Aktorzy:</b>	Użytkownik zalogowany

<b>Opis:</b>	Użytkownik zgłasza post forum jako naruszający regulamin lub tematykę.
<b>Warunki wstępne:</b>	Wyświetlony jest widok posta na forum.
<b>Warunki końcowe:</b>	Zgłoszenie posta zostało zapisane i przekazane moderatorom.
<b>Główny przeływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik wybiera opcję „Zgłoś post”.</li> <li>2. Użytkownik wybiera kategorię naruszenia, podaje szczegóły i potwierdza zgłoszenie.</li> <li>3. System zapisuje zgłoszenie i oznacza post jako zgłoszony.</li> </ol>
<b>Alternatywne przeływy zdarzeń:</b>	<ol style="list-style-type: none"> <li>1a. Użytkownik próbuje zgłosić własny post – system blokuje operację i wyświetla komunikat, że nie można zgłaszać własnych treści.</li> <li>3a. Post został już wcześniej zgłoszony – system informuje użytkownika, że post jest już zgłoszony i nie zapisuje kolejnego zgłoszenia.</li> </ol>

**Tabela 4.35:** Scenariusz przypadku użycia: Zgłoszenie posta na forum

<b>KARTA SCENARIUSZA PRZYPADKU UŻYCIA</b>	
<b>Identyfikator:</b>	PU36
<b>Nazwa:</b>	Przeglądanie komentarzy pod postem
<b>Priorytet:</b>	Średni
<b>Aktorzy:</b>	Użytkownik niezalogowany, Użytkownik zalogowany
<b>Opis:</b>	Użytkownik przegląda komentarze dodane pod wybranym postem na forum z możliwością zmiany kolejności ich wyświetlania.



<b>Warunki wstępne:</b>	Wyświetlany jest szczegółowy widok posta na forum.
<b>Warunki końcowe:</b>	Lista komentarzy powiązanych z postem została wyświetlona zgodnie z wybranym kryterium sortowania.
<b>Główny przepływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. System pobiera komentarze powiązane z wybranym postem i domyślnie wyświetla je w kolejności od najnowszych.</li> <li>2. Użytkownik wybiera sposób sortowania komentarzy.</li> <li>3. System aktualizuje listę komentarzy zgodnie z wybranym kryterium sortowania.</li> <li>4. Użytkownik przewija listę komentarzy i zapoznaje się z ich treścią.</li> </ol>
<b>Alternatywne przepływy zdarzeń:</b>	<ol style="list-style-type: none"> <li>1a. Post nie ma jeszcze komentarzy – system wyświetla informację o braku komentarzy.</li> <li>1b. Wystąpił błąd podczas pobierania komentarzy – system wyświetla komunikat o błędzie i umożliwia ponowną próbę.</li> </ol>

**Tabela 4.36:** Scenariusz przypadku użycia: Przeglądanie komentarzy pod postem

#### Scenariusze przypadków użycia dla panelu użytkownika

KARTA SCENARIUSZA PRZYPADKU UŻYCIA	
<b>Identyfikator:</b>	PU37
<b>Nazwa:</b>	Dodanie spota w panelu użytkownika
<b>Priorytet:</b>	Wysoki

<b>Aktorzy:</b>	Użytkownik zalogowany, Usługa do wyświetlania mapy, Usługa do przechowywania plików w chmurze
<b>Opis:</b>	Użytkownik dodaje nowy spot poprzez panel.
<b>Warunki wstępne:</b>	Użytkownik jest zalogowany i znajduje się w widoku panelu użytkownika.
<b>Warunki końcowe:</b>	Nowy spot został zapisany i jest widoczny na mapie oraz w panelu użytkownika.
<b>Główny przebieg zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik wybiera opcję „Dodaj spot”.</li> <li>2. Użytkownik uzupełnia podstawowe informacje o spocie (nazwa, opis, tagi).</li> <li>3. Użytkownik wskazuje lokalizację spota na mapie.</li> <li>4. Użytkownik dodaje zdjęcia/filmy do spota.</li> <li>5. Użytkownik zapisuje spota.</li> <li>6. System zapisuje dane spota (oraz pliki w chmurze) i aktualizuje mapę.</li> </ol>
<b>Alternatywne przebiegy zdarzeń:</b>	<ol style="list-style-type: none"> <li>3a. Podane dane wejściowe są niepoprawne – system wyświetla komunikat i zaznacza wymagające poprawy pola.</li> </ol>

**Tabela 4.37:** Scenariusz przypadku użycia: Dodanie spota w panelu użytkownika

KARTA SCENARIUSZA PRZYPADKU UŻYCIA	
<b>Identyfikator:</b>	PU38
<b>Nazwa:</b>	Przeglądanie profilu użytkownika
<b>Priorytet:</b>	Wysoki
<b>Aktorzy:</b>	Użytkownik zalogowany

<b>Opis:</b>	Użytkownik przegląda swój profil (lista spotów, media, podstawowe dane).
<b>Warunki wstępne:</b>	Użytkownik jest zalogowany.
<b>Warunki końcowe:</b>	Wyświetlony jest widok profilu użytkownika wraz z jego wartością.
<b>Główny przepływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik otwiera swój profil.</li> <li>2. System pobiera dane profilu (informacje podstawowe, spoty, media).</li> <li>3. System wyświetla dane w odpowiednich sekcjach (spoty, zdjęcia, filmy, komentarze).</li> </ol>
<b>Alternatywne przepływy zdarzeń:</b>	2a. Wystąpił błąd podczas pobierania danych użytkownika – system wyświetla informację o błędzie.

**Tabela 4.38:** Scenariusz przypadku użycia: Przeglądanie profilu użytkownika

KARTA SCENARIUSZA PRZYPADKU UŻYCIA	
<b>Identyfikator:</b>	PU39
<b>Nazwa:</b>	Przeglądanie profilu innego użytkownika
<b>Priorytet:</b>	Średni
<b>Aktorzy:</b>	Użytkownik niezalogowany
<b>Opis:</b>	Użytkownik ogląda profil innego użytkownika (np. z mapy, forum lub społeczności).
<b>Warunki wstępne:</b>	Użytkownik jest zalogowany i ma dostęp do odnośnika do profilu innego użytkownika.

<b>Warunki końcowe:</b>	Profil innego użytkownika został wyświetlony.
<b>Główny przepływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik wybiera odnośnik do profilu innego użytkownika.</li> <li>2. System pobiera dane profilu docelowego użytkownika.</li> <li>3. System wyświetla profil (media, podstawowe informacje).</li> </ol>
<b>Alternatywne przepływy zdarzeń:</b>	<ol style="list-style-type: none"> <li>2a. Wystąpił błąd podczas pobierania danych użytkownika – system wyświetla informację o błędzie.</li> </ol>

**Tabela 4.39:** Scenariusz przypadku użycia: Przeglądanie profilu innego użytkownika

<b>KARTA SCENARIUSZA PRZYPADKU UŻYCIA</b>	
<b>Identyfikator:</b>	PU40
<b>Nazwa:</b>	Dodanie użytkownika do znajomych
<b>Priorytet:</b>	Średni
<b>Aktorzy:</b>	Użytkownik zalogowany
<b>Opis:</b>	Użytkownik wysyła lub akceptuje zaproszenie do znajomych.
<b>Warunki wstępne:</b>	Użytkownik jest zalogowany i przegląda profil innego użytkownika.
<b>Warunki końcowe:</b>	Relacja „znajomy” została utworzona lub zaproszenie czeka na akceptację.
<b>Główny przepływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik klika przycisk „Dodaj do znajomych”.</li> <li>2. System sprawdza, czy relacja już istnieje.</li> <li>3. System tworzy nowe zaproszenie.</li> <li>4. System informuje o statusie o wysłaniu zaproszenia.</li> </ol>

<b>Alternatywne przeływy zdarzeń:</b>	Brak istotnych alternatywnych przeływów.
---------------------------------------	--

**Tabela 4.40:** Scenariusz przypadku użycia: Dodanie użytkownika do znajomych

<b>KARTA SCENARIUSZA PRZYPADKU UŻYCIA</b>	
<b>Identyfikator:</b>	PU41
<b>Nazwa:</b>	Przeglądanie społeczności
<b>Priorytet:</b>	Średni
<b>Aktorzy:</b>	Użytkownik zalogowany
<b>Opis:</b>	Użytkownik przegląda społeczności, grupy lub listy znajomych powiązane z aplikacją.
<b>Warunki wstępne:</b>	Użytkownik jest zalogowany.
<b>Warunki końcowe:</b>	Lista społeczności lub znajomych została wyświetlona.
<b>Główny przeływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik przechodzi do sekcji społeczności.</li> <li>2. System pobiera listę społeczności i znajomych użytkownika.</li> <li>3. System wyświetla listę z możliwością przechodzenia do profili i czatów.</li> </ol>
<b>Alternatywne przeływy zdarzeń:</b>	Brak istotnych alternatywnych przeływów.

**Tabela 4.41:** Scenariusz przypadku użycia: Przeglądanie społeczności

KARTA SCENARIUSZA PRZYPADKU UŻYCIA	
<b>Identyfikator:</b>	PU42
<b>Nazwa:</b>	Przeglądanie dodanych spotów
<b>Priorytet:</b>	Wysoki
<b>Aktorzy:</b>	Użytkownik zalogowany, Usługa do wyświetlania mapy
<b>Opis:</b>	Użytkownik przegląda listę/siatkę spotów, które sam dodał.
<b>Warunki wstępne:</b>	Użytkownik jest zalogowany i znajduje się w widoku panelu użytkownika lub sekcji „Moje spoty”.
<b>Warunki końcowe:</b>	Lista dodanych spotów użytkownika została wyświetlona (np. na mapie i/lub w formie listy).
<b>Główny przepływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik przechodzi do sekcji „Moje spoty”.</li> <li>2. System pobiera listę spotów dodanych przez użytkownika.</li> <li>3. System wyświetla listę spotów oraz znaczniki na mapie.</li> <li>4. Użytkownik wybiera spota, aby przejść do jego szczegółów.</li> </ol>
<b>Alternatywne przepływy zdarzeń:</b>	2a. Użytkownik nie dodał jeszcze żadnego spota – system wyświetla komunikat i proponuje dodanie pierwszego spota.

**Tabela 4.42:** Scenariusz przypadku użycia: Przeglądanie dodanych spotów

KARTA SCENARIUSZA PRZYPADKU UŻYCIA	
<b>Identyfikator:</b>	PU43
<b>Nazwa:</b>	Edycja danych użytkownika
<b>Priorytet:</b>	Wysoki
<b>Aktorzy:</b>	Użytkownik zalogowany

<b>Opis:</b>	Użytkownik modyfikuje swoje dane profilu (np. nazwę, opis, avatar).
<b>Warunki wstępne:</b>	Użytkownik jest zalogowany i znajduje się w widoku edycji profilu.
<b>Warunki końcowe:</b>	Zaktualizowane dane profilu są zapisane i widoczne w aplikacji.
<b>Główny przepływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik otwiera widok edycji profilu.</li> <li>2. Użytkownik wprowadza zmiany w danych profilu.</li> <li>3. Użytkownik zapisuje zmiany.</li> <li>4. System waliduje dane i zapisuje zaktualizowany profil.</li> </ol>
<b>Alternatywne przepływy zdarzeń:</b>	4a. Dane są niepoprawne lub niekompletne – system wyświetla komunikat o błędzie i zaznacza pola do poprawy.

**Tabela 4.43:** Scenariusz przypadku użycia: Edycja danych użytkownika

<b>KARTA SCENARIUSZA PRZYPADKU UŻYCIA</b>	
<b>Identyfikator:</b>	PU44
<b>Nazwa:</b>	Przeglądanie dodanych zdjęć do spotów
<b>Priorytet:</b>	Średni
<b>Aktorzy:</b>	Użytkownik zalogowany, Usługa do przechowywania plików w chmurze
<b>Opis:</b>	Użytkownik przegląda wszystkie zdjęcia powiązane ze spotami, które dodał.
<b>Warunki wstępne:</b>	Użytkownik jest zalogowany i znajduje się w sekcji mediów (np. „Moje zdjęcia”).

<b>Warunki końcowe:</b>	Lista lub galeria zdjęć powiązanych ze spotami użytkownika została wyświetlona.
<b>Główny przepływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik otwiera sekcję przeglądania zdjęć ze spotów.</li> <li>2. System pobiera metadane zdjęć z usługi przechowywania plików.</li> <li>3. System wyświetla galerię zdjęć z podstawowymi informacjami (np. nazwa spotu, data dodania).</li> <li>4. Użytkownik wybiera zdjęcie, aby zobaczyć je w powiększeniu lub przejść do spotu.</li> </ol>
<b>Alternatywne przepływy zdarzeń:</b>	2a. Użytkownik nie dodał jeszcze zdjęć – system wyświetla informację o braku zdjęć.

**Tabela 4.44:** Scenariusz przypadku użycia: Przeglądanie dodanych zdjęć do spotów

<b>KARTA SCENARIUSZA PRZYPADKU UŻYCIA</b>	
<b>Identyfikator:</b>	PU45
<b>Nazwa:</b>	Przeglądanie dodanych filmów do spotów
<b>Priorytet:</b>	Średni
<b>Aktorzy:</b>	Użytkownik zalogowany, Usługa do przechowywania plików w chmurze
<b>Opis:</b>	Użytkownik przegląda filmy powiązane ze spotami, które dodał.
<b>Warunki wstępne:</b>	Użytkownik jest zalogowany i znajduje się w sekcji mediów (np. „Moje filmy”).



<b>Warunki końcowe:</b>	Lista lub galeria filmów powiązanych ze spotami użytkownika została wyświetlona.
<b>Główny przepływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik otwiera sekcję przeglądania filmów ze spotów.</li> <li>2. System pobiera metadane filmów z usługi przechowywania plików.</li> <li>3. System wyświetla listę/galerię filmów z podstawowymi informacjami.</li> <li>4. Użytkownik wybiera film, aby go odtworzyć lub przejść do spotu.</li> </ol>
<b>Alternatywne przepływy zdarzeń:</b>	<ol style="list-style-type: none"> <li>2a. Użytkownik nie dodał jeszcze filmów – system wyświetla informację o braku filmów.</li> <li>3a. Nie udało się pobrać filmów – system wyświetla komunikat o błędzie.</li> </ol>

**Tabela 4.45:** Scenariusz przypadku użycia: Przeglądanie dodanych filmów do spotów

KARTA SCENARIUSZA PRZYPADKU UŻYCIA	
<b>Identyfikator:</b>	PU46
<b>Nazwa:</b>	Przeglądanie dodanych komentarzy do spotów
<b>Priorytet:</b>	Średni
<b>Aktorzy:</b>	Użytkownik zalogowany
<b>Opis:</b>	Użytkownik przegląda komentarze dodane do spotów, które sam utworzył.
<b>Warunki wstępne:</b>	Użytkownik jest zalogowany i otwiera sekcję komentarzy do swoich spotów.

<b>Warunki końcowe:</b>	Lista komentarzy do spotów użytkownika została wyświetlona.
<b>Główny przepływ zdarzeń:</b>	<ol style="list-style-type: none"> <li>1. Użytkownik przechodzi do sekcji komentarzy do własnych spotów.</li> <li>2. System pobiera komentarze powiązane ze spotami użytkownika.</li> <li>3. System wyświetla komentarze (np. w kolejności chronologicznej).</li> </ol>
<b>Alternatywne przepływy zdarzeń:</b>	<ol style="list-style-type: none"> <li>2a. Żaden z spotów użytkownika nie ma komentarzy – system wyświetla odpowiednią informację.</li> <li>3a. Nie udało się pobrać komentarzy – system wyświetla komunikat o błędzie.</li> </ol>

**Tabela 4.46:** Scenariusz przypadku użycia: Przeglądanie dodanych komentarzy do spotów

## 4.2 Wymagania ogólne i dziedzinowe

## 4.3 Wymagania funkcjonalne

### 4.3.1 Funkcjonalności dla mapy

### 4.3.2 Funkcjonalności dla chatu

### 4.3.3 Funkcjonalności dla forum

### 4.3.4 Funkcjonalności dla konta użytkownika

KARTA WYMAGANIA			
Identyfikator:	jednoznaczny symbol np. FO1, FO2 ..	Priorytet:	M
Nazwa:	Profil użytkownika		
Opis:	Jako użytkownik chcę mieć dostęp do strony profilu, aby sprawdzić informacje o swoim koncie.		
Kryteria akceptacji:	Użytkownik widzi liczby: znajomych, obserwowanych i obserwujących, a także najpopularniejsze zdjęcia.		
Dane wejściowe:	Lista zdjęć oraz liczby: znajomych, obserwujących i obserwowanych.		
Warunki początkowe:	Użytkownik jest zalogowany.		
Warunki końcowe:	Wyświetlone informacje o profilu.		
Sytuacje wyjątkowe:	Błąd połączenia z API; brak danych profilu; brak uprawnień (401/403).		
Szczegóły implementacji:	Frontend: React + Tailwind; pobieranie danych profilu przez @tanstack/react-query i axios z withCredentials. Prezentacja w widoku profilu.		
Udziałowiec:	Zespół projektowy <a href="#">2.1</a> ; promotor <a href="#">2.2</a> ; droniarze <a href="#">2.3</a> .		
Wymagania powiązane:			

Tabela 4.47: Profil użytkownika

KARTA WYMAGANIA			
Identyfikator:	jednoznaczny symbol np. FO1, FO2 ..	Priorytet:	M
Nazwa:	Lista dodanych spotów		
Opis:	Jako użytkownik chcę sprawdzić listę spotów, które dodałem.		
Kryteria akceptacji:	Użytkownik widzi listę własnych dodanych spotów.		
Dane wejściowe:	Lista dodanych spotów.		
Warunki początkowe:	Użytkownik jest zalogowany.		
Warunki końcowe:	Wyświetlona lista dodanych spotów.		
Sytuacje wyjątkowe:	Brak wyników; błąd połączenia z API.		
Szczegóły implementacji:	Pobranie listy z backendu (endpoint listy własnych spotów) przez <code>react-query</code> + <code>axios</code> ; prezentacja listy z podstawowymi danymi.		
Udziałowiec:	Zespół projektowy 2.1; promotor 2.2; droniarze 2.3.		
Wymagania powiązane:			

Tabela 4.48: Lista dodanych spotów

KARTA WYMAGANIA			
Identyfikator:	jednoznaczny symbol np. FO1, FO2 ..	Priorytet:	M
Nazwa:	Dodanie spota		
Opis:	Jako użytkownik chcę mieć dostęp do formularza dodania spota.		
Kryteria akceptacji:	Użytkownik ma dostęp do formularza dodania spota i może go wysłać.		
Dane wejściowe:	Formularz dodania spota.		
Warunki początkowe:	Użytkownik jest zalogowany.		
Warunki końcowe:	Wyświetlony formularz dodania spota (po wysłaniu: zapis na backendzie).		
Sytuacje wyjątkowe:	Nieprawidłowe dane formularza; błąd połączenia z API.		
Szczegóły implementacji:	Formularz w React; walidacja przeglądarkowa; wysyłka przez <code>axios</code> (POST) z <code>withCredentials</code> .		
Udziałowiec:	Zespół projektowy <a href="#">2.1</a> ; promotor <a href="#">2.2</a> ; droniarze <a href="#">2.3</a> .		
Wymagania powiązane:			

Tabela 4.49: Dodanie spota

KARTA WYMAGANIA			
Identyfikator:	jednoznaczny symbol np. FO1, FO2 ..	Priorytet:	M
Nazwa:	Lista zdjęć		
Opis:	Jako użytkownik chcę mieć dostęp do listy zdjęć, które dodałem na forum, do komentarzy pod spotem oraz do spota.		
Kryteria akceptacji:	Użytkownik widzi listę swoich zdjęć.		
Dane wejściowe:	Lista zdjęć.		
Warunki początkowe:	Użytkownik jest zalogowany.		
Warunki końcowe:	Wyświetlona lista zdjęć.		
Sytuacje wyjątkowe:	Brak wyników; błąd połączenia z API.		
Szczegóły implementacji:	Pobranie listy zdjęć użytkownika przez <code>react-query</code> + <code>axios</code> ; prezentacja z miniaturowymi.		
Udziałowiec:	Zespół projektowy 2.1; promotor 2.2; droniarze 2.3.		
Wymagania powiązane:			

Tabela 4.50: Lista zdjęć

KARTA WYMAGANIA			
Identyfikator:	jednoznaczny symbol np. FO1, FO2 ..	Priorytet:	M
Nazwa:	Lista filmów		
Opis:	Jako użytkownik chcę mieć dostęp do listy filmów, które dodałem na forum, do komentarzy pod spotem oraz do spota.		
Kryteria akceptacji:	Użytkownik widzi listę swoich filmów.		
Dane wejściowe:	Lista filmów.		
Warunki początkowe:	Użytkownik jest zalogowany.		
Warunki końcowe:	Wyświetlona lista filmów.		
Sytuacje wyjątkowe:	Brak wyników; błąd połączenia z API.		
Szczegóły implementacji:	Pobranie listy filmów użytkownika przez <code>react-query</code> + <code>axios</code> ; prezentacja z miniaturowymi.		
Udziałowiec:	Zespół projektowy 2.1; promotor 2.2; droniarze 2.3.		
Wymagania powiązane:			

Tabela 4.51: Lista filmów

KARTA WYMAGANIA			
Identyfikator:	jednoznaczny symbol np. FO1, FO2 ..	Priorytet:	M
Nazwa:	Lista znajomych		
Opis:	Jako użytkownik chcę mieć dostęp do listy znajomych.		
Kryteria akceptacji:	Użytkownik ma dostęp do listy znajomych.		
Dane wejściowe:	Lista znajomych.		
Warunki początkowe:	Użytkownik jest zalogowany.		
Warunki końcowe:	Wyświetlona lista znajomych.		
Sytuacje wyjątkowe:	Brak wyników; błąd połączenia z API.		
Szczegóły implementacji:	Pobranie listy znajomych przez <b>react-query</b> + <b>axios</b> ; standardowa prezentacja listy.		
Udziałowiec:	Zespół projektowy <a href="#">2.1</a> ; promotor <a href="#">2.2</a> ; droniarze <a href="#">2.3</a> .		
Wymagania powiązane:			

Tabela 4.52: Lista znajomych

KARTA WYMAGANIA			
Identyfikator:	jednoznaczny symbol np. FO1, FO2 ..	Priorytet:	M
Nazwa:	Lista obserwujących		
Opis:	Jako użytkownik chcę mieć dostęp do listy obserwujących.		
Kryteria akceptacji:	Użytkownik ma dostęp do listy obserwujących.		
Dane wejściowe:	Lista obserwujących.		
Warunki początkowe:	Użytkownik jest zalogowany.		
Warunki końcowe:	Wyświetlona lista obserwujących.		
Sytuacje wyjątkowe:	Brak wyników; błąd połączenia z API.		
Szczegóły implementacji:	Pobranie listy obserwujących przez <b>react-query</b> + <b>axios</b> ; standardowa prezentacja listy.		
Udziałowiec:	Zespół projektowy <a href="#">2.1</a> ; promotor <a href="#">2.2</a> ; droniarze <a href="#">2.3</a> .		
Wymagania powiązane:			

Tabela 4.53: Lista obserwujących

KARTA WYMAGANIA			
Identyfikator:	jednoznaczny symbol np. FO1, FO2 ..	Priorytet:	M
Nazwa:	Lista obserwowanych		
Opis:	Jako użytkownik chcę mieć dostęp do listy obserwowanych.		
Kryteria akceptacji:	Użytkownik ma dostęp do listy obserwowanych.		
Dane wejściowe:	Lista obserwowanych.		
Warunki początkowe:	Użytkownik jest zalogowany.		
Warunki końcowe:	Wyświetlona lista obserwowanych.		
Sytuacje wyjątkowe:	Brak wyników; błąd połączenia z API.		
Szczegóły implementacji:	Pobranie listy obserwowanych przez <code>react-query</code> + <code>axios</code> ; standardowa prezentacja listy.		
Udziałowiec:	Zespół projektowy 2.1; promotor 2.2; droniarze 2.3.		
Wymagania powiązane:			

Tabela 4.54: Lista obserwowanych

KARTA WYMAGANIA			
Identyfikator:	jednoznaczny symbol np. FO1, FO2 ..	Priorytet:	M
Nazwa:	Lista spotów		
Opis:	Jako użytkownik chcę mieć dostęp do listy spotów, które polubiłem, odwiedziłem i planuję odwiedzić.		
Kryteria akceptacji:	Użytkownik ma dostęp do listy spotów w wymienionych kategoriach.		
Dane wejściowe:	Listy spotów: polubione, odwiedzone, planowane.		
Warunki początkowe:	Użytkownik jest zalogowany.		
Warunki końcowe:	Wyświetlone listy spotów.		
Sytuacje wyjątkowe:	Brak wyników; błąd połączenia z API.		
Szczegóły implementacji:	Pobranie list przez <code>react-query</code> + <code>axios</code> ; prezentacja w zakładkach/kategoriach.		
Udziałowiec:	Zespół projektowy 2.1; promotor 2.2; droniarze 2.3.		
Wymagania powiązane:			

Tabela 4.55: Lista polubionych/odwiedzonych/planowanych spotów



KARTA WYMAGANIA			
Identyfikator:	jednoznaczny symbol np. FO1, FO2 ..	Priorytet:	M
Nazwa:	Lista komentarzy		
Opis:	Jako użytkownik chcę mieć dostęp do listy komentarzy.		
Kryteria akceptacji:	Użytkownik ma dostęp do listy swoich komentarzy.		
Dane wejściowe:	Lista komentarzy.		
Warunki początkowe:	Użytkownik jest zalogowany.		
Warunki końcowe:	Wyświetlona lista komentarzy.		
Sytuacje wyjątkowe:	Brak wyników; błąd połączenia z API.		
Szczegóły implementacji:	Pobranie listy komentarzy użytkownika przez <code>react-query</code> + <code>axios</code> ; standardowa prezentacja listy.		
Udziałowiec:	Zespół projektowy <a href="#">2.1</a> ; promotor <a href="#">2.2</a> ; droniarze <a href="#">2.3</a> .		
Wymagania powiązane:			

Tabela 4.56: Lista komentarzy

KARTA WYMAGANIA			
Identyfikator:	jednoznaczny symbol np. FO1, FO2 ..	Priorytet:	M
Nazwa:	Ustawienia		
Opis:	Jako użytkownik chcę mieć możliwość zmiany danych.		
Kryteria akceptacji:	Użytkownik może edytować wybrane dane profilu i zapisać zmiany.		
Dane wejściowe:	Formularz edycji danych.		
Warunki początkowe:	Użytkownik jest zalogowany.		
Warunki końcowe:	Wyświetlony formularz edycji; po zapisie — zaktualizowane dane.		
Sytuacje wyjątkowe:	Nieprawidłowe dane formularza; błąd połączenia z API.		
Szczegóły implementacji:	Formularz w React; walidacja pól; wysyłka przez <code>axios</code> (PUT/PATCH) z <code>withCredentials</code> . Po sukcesie — komunikat i odświeżenie danych przez <code>react-query</code> .		
Udziałowiec:	Zespół projektowy <a href="#">2.1</a> ; promotor <a href="#">2.2</a> ; droniarze <a href="#">2.3</a> .		
Wymagania powiązane:			

Tabela 4.57: Ustawienia profilu

KARTA WYMAGANIA			
Identyfikator:	jednoznaczny symbol np. FO1, FO2 ..	Priorytet:	M
Nazwa:	Resetowanie hasła		
Opis:	Jako użytkownik chcę mieć możliwość zresetowania hasła do swojego konta.		
Kryteria akceptacji:	Po kliknięciu w odpowiedni link użytkownik może zresetować hasło do konta.		
Dane wejściowe:	Adres e-mail użytkownika do wysłania linku resetującego.		
Warunki początkowe:	Użytkownik podał poprawny adres e-mail użyty przy rejestracji.		
Warunki końcowe:	Hasło zresetowane po przejściu całej procedury.		
Sytuacje wyjątkowe:	Niepoprawny adres e-mail; wygasły lub nieprawidłowy token resetu; błąd połączenia z API.		
Szczegóły implementacji:	Frontend: formularz „zapomniałem hasła” (POST do endpointu wysyłającego link resetu) oraz formularz ustawienia nowego hasła (POST/PATCH z tokenem). Wysyłka przez <b>axios</b> ; obsługa komunikatów o powodzeniu/błędach.		
Udziałowiec:	Zespół projektowy <a href="#">2.1</a> ; promotor <a href="#">2.2</a> ; droniarze <a href="#">2.3</a> .		
Wymagania powiązane:			

Tabela 4.58: Resetowanie hasła

KARTA WYMAGANIA			
Identyfikator:	jednoznaczny symbol np. FO1, FO2 ..	Priorytet:	M
Nazwa:	Dodawanie użytkowników do listy znajomych		
Opis:	Jako użytkownik chcę mieć możliwość dodawania innych użytkowników do listy znajomych.		
Kryteria akceptacji:	Użytkownik może dodać innego użytkownika do swojej listy znajomych.		
Dane wejściowe:	Dane użytkownika, którego chcemy dodać do znajomych.		
Warunki początkowe:	Użytkownik jest zalogowany.		
Warunki końcowe:	Znajomy dodany do listy i widoczny w profilu użytkownika.		
Sytuacje wyjątkowe:	Brak uprawnień; użytkownik już jest znajomym; błąd połączenia z API.		
Szczegóły implementacji:	Akcja wysłania zaproszenia do znajomych przez <code>axios</code> ; po akceptacji — aktualizacja listy (odświeżenie <code>react-query</code> ).		
Udziałowiec:	Zespół projektowy <a href="#">2.1</a> ; promotor <a href="#">2.2</a> ; droniarze <a href="#">2.3</a> .		
Wymagania powiązane:			

Tabela 4.59: Dodawanie do znajomych

### 4.3.5 Funkcjonalności dla logowania i rejestracji

KARTA WYMAGANIA			
Identyfikator:	FOXX	Priorytet:	M
Nazwa:	Logowanie i rejestracja		
Opis:	Jako użytkownik chcę mieć możliwość zalogowania się do aplikacji, korzystając z formularza lub poprzez konto Google lub GitHub.		
Kryteria akceptacji:	Użytkownik może zalogować się do aplikacji zarówno za pomocą standardowego formularza, jak i przy użyciu konta w serwisie Google lub GitHub.		
Dane wejściowe:	Dane użytkownika: adres e-mail, hasło; przy rejestracji dodatkowo nazwa użytkownika.		
Warunki początkowe:	Użytkownik niezalogowany.		
Warunki końcowe:	Działające formularze rejestracji i logowania oraz możliwość logowania za pomocą konta Google i GitHub.		
Sytuacje wyjątkowe:	Błędne dane logowania; przerwana lub nieudana autoryzacja u dostawcy (Google/GitHub).		
Szczegóły implementacji:	Frontend: formularze w React; wysyłka żądań przez <code>axios</code> z <code>withCredentials</code> . SSO: integracja z Google i GitHub (OAuth 2.0) z przekierowaniem i ustawieniem sesji po stronie backendu ( <code>httpOnly</code> cookie). Obsługa statusu 401 zgodnie z mechanizmem wylogowania.		
Udziałowiec:	Zespół projektowy <a href="#">2.1</a> , promotor <a href="#">2.2</a> , droniarze <a href="#">2.3</a> .		
Wymagania powiązane:			

Tabela 4.60: Logowanie i rejestracja

### 4.3.6 Funkcjonalności dla wyszukiwarki spotów

KARTA WYMAGANIA			
Identyfikator:	FOXX	Priorytet:	M
Nazwa:	Strona główna z podstawowymi filtrami		
Opis:	Jako użytkownik chcę mieć dostęp do strony głównej, która wyświetla karuzelę z najpopularniejszymi spotami oraz listę spotów, które można filtrować.		
Kryteria akceptacji:	Użytkownik widzi karuzelę najpopularniejszych miejsc. Karuzela zawiera zdjęcia, nazwę miejsca i miasto. Użytkownik może filtrować miejsca według lokalizacji (kraj, region, miasto).		
Dane wejściowe:	Lokalizacja użytkownika (kraj, region, miasto); dane z bazy spotów.		
Warunki początkowe:	Użytkownik nie musi być zalogowany.		
Warunki końcowe:	Użytkownik widzi popularne miejsca z wybranego miasta (np. Gdańsk) i może przejść do szczegółów danego miejsca.		
Sytuacje wyjątkowe:	Brak wyników dla wybranych filtrów; błąd połączenia z API.		
Szczegóły implementacji:	Frontend: React + Tailwind. Pobieranie danych przez <code>@tanstack/react-query</code> i <code>axios</code> (GET do backendu z parametrami lokalizacji). Filtry lokalizacji mapowane na parametry zapytania.		
Udziałowiec:	Zespół projektowy 2.1, promotor 2.2, droniarze 2.3.		
Wymagania powiązane:			

Tabela 4.61: Strona główna — podstawowe filtry

KARTA WYMAGANIA			
Identyfikator:	FOXX	Priorytet:	M
Nazwa:	Strona główna z zaawansowanymi filtrami		
Opis:	Jako użytkownik chcę mieć dostęp do strony głównej, która wyświetla listę spotów, które można filtrować i sortować.		
Kryteria akceptacji:	Użytkownik widzi listę, którą może filtrować według miasta, tagów i oceny spotu, a także sortować po ocenie i popularności.		
Dane wejściowe:	Lokalizacja użytkownika (miasto), wartości filtrów i sortowania; dane z bazy spotów.		
Warunki początkowe:	Użytkownik nie musi być zalogowany.		
Warunki końcowe:	Użytkownik widzi wyniki zgodne z zastosowanymi filtrami i sortowaniem oraz może przejść do szczegółów danego miejsca.		
Sytuacje wyjątkowe:	Brak wyników po zastosowaniu filtrów; błąd połączenia z API.		
Szczegóły implementacji:	Frontend: React + Tailwind. Pobieranie danych przez <code>@tanstack/react-query</code> i <code>axios</code> z parametrami: lokalizacja, tagi, minimalna ocena oraz kryterium sortowania.		
Udziałowiec:	Zespół projektowy 2.1, promotor 2.2, droniarze 2.3.		
Wymagania powiązane:	SPXX		

Tabela 4.62: Strona główna — zaawansowane filtry

### 4.3.7 Funkcjonalności dla motywu

KARTA WYMAGANIA			
Identyfikator:	FOXX	Priorytet:	M
Nazwa:	Ustawienia motywu		
Opis:	Jako użytkownik chcę móc zmienić motyw aplikacji.		
Kryteria akceptacji:	Dostępna jest opcja przełączenia motywu na <i>jaśny</i> lub <i>ciemny</i> ; zmiana następuje bez przeładowania strony; ustawienie działa we wszystkich widokach.		
Dane wejściowe:	Preferencje użytkownika dotyczące motywu.		
Warunki początkowe:	Brak.		
Warunki końcowe:	Zmiana motywu widoczna jest natychmiast po kliknięciu przycisku.		
Sytuacje wyjątkowe:	Brak.		
Szczegóły implementacji:	Tailwind CSS z <code>darkMode: 'class'</code> ; motyw przełączany przez dodanie/usunięcie klasy <code>dark</code> na elemencie <code>&lt;html&gt;</code> ;		
Udziałowiec:	Zespół projektowy 2.1, promotor 2.2, droniarze 2.3.		
Wymagania powiązane:			

Tabela 4.63: Ustawienia motywu (ręczna zmiana)



KARTA WYMAGANIA			
Identyfikator:	FOXX	Priorytet:	M
Nazwa:	Zapamiętywanie preferencji motywu		
Opis:	Jako użytkownik chcę, aby moja preferencja motywu była zapamiętana i przywracana przy kolejnym użyciu aplikacji.		
Kryteria akceptacji:	Wybrany motyw jest przywracany po ponownym włączeniu i odświeżeniu strony; preferencja jest zapamiętywana lokalnie w przeglądarce.		
Dane wejściowe:	Preferencje użytkownika zapisane lokalnie.		
Warunki początkowe:	FOXX dostępne.		
Warunki końcowe:	Motyw po uruchomieniu odpowiada ostatniej decyzji użytkownika.		
Sytuacje wyjątkowe:	Brak dostępu do magazynu trwałego — preferencja przechowywana w local storage.		
Szczegóły implementacji:	Zapis w <code>localStorage</code> pod kluczem <code>theme</code> ( <code>dark</code> lub <code>light</code> ); krótki skrypt umieszczony w <code>App.jsx</code> przed startem odczytuje <code>localStorage</code> i odpowiednio dodaje lub usuwa klasę <code>dark</code> na <code>&lt;html&gt;</code> (eliminuje mignięcie stylów).		
Udziałowiec:	Zespół projektowy 2.1, promotor 2.2, droniarze 2.3.		
Wymagania powiązane:			

Tabela 4.64: Zapamiętanie preferencji motywu

KARTA WYMAGANIA			
Identyfikator:	FOXX	Priorytet:	S
Nazwa:	Przełącznik motywu w <a href="#">Sidebar</a>		
Opis:	Jako użytkownik chcę szybko zmieniać motyw bez wchodzenia w ustawienia.		
Kryteria akceptacji:	W <a href="#">Sidebar</a> dostępny jest przełącznik <i>Jasny-/Ciemny</i> ; posiada odpowiednio ikony <i>słońca/księżycy</i> ; zmiana następuje natychmiast.		
Dane wejściowe:	Bieżąca preferencja motywu.		
Warunki początkowe:	FOXX, FOXX dostępne.		
Warunki końcowe:	Motyw zmieniony; preferencja zaktualizowana.		
Sytuacje wyjątkowe:	Brak.		
Szczegóły implementacji:	Przycisk typu <i>toggle</i> wywołuje funkcję, która przełącza klasę <code>dark</code> na <code>document.documentElement</code> oraz aktualizuje <code>localStorage</code> ( <code>theme = 'dark' 'light'</code> ); brak przeładowania strony.		
Udziałowiec:	Zespół projektowy <a href="#">2.1</a> , promotor <a href="#">2.2</a> , droniarze <a href="#">2.3</a> .		
Wymagania powiązane:			

Tabela 4.65: Szybki przełącznik motywu w interfejsie

## 4.4 Wymagania pozafunkcjonalne

## 4.5 Wymagania interfejs z otoczeniem

## 4.6 Wymagania na środowisko docelowe

# Rozdział 5

## Projekt

### 5.1 Wzorce projektowe

### 5.2 Architektura systemu

#### 5.2.1 Diagram architektury

#### 5.2.2 Komponenty systemu

### 5.3 Projekt bazy danych

#### 5.3.1 Model danych

#### 5.3.2 Diagram ERD

### 5.4 Architektura interfejsu użytkownika

#### 5.4.1 Projekt strony głównej

#### 5.4.2 Projekt panelu logowania

#### 5.4.3 Projekt mapy

#### 5.4.4 Projekt chatu

#### 5.4.5 Projekt forum

#### 5.4.6 Projekt konta użytkownika

## Rozdział 6

# Przebieg realizacji projektu

### 6.1 Sprint 1

### 6.2 Sprint 2

# Rozdział 7

## Realizacja Projektu

### 7.1 Implementacja backendu

#### 7.1.1 Struktura projektu

#### 7.1.2 Integracja z bazą danych

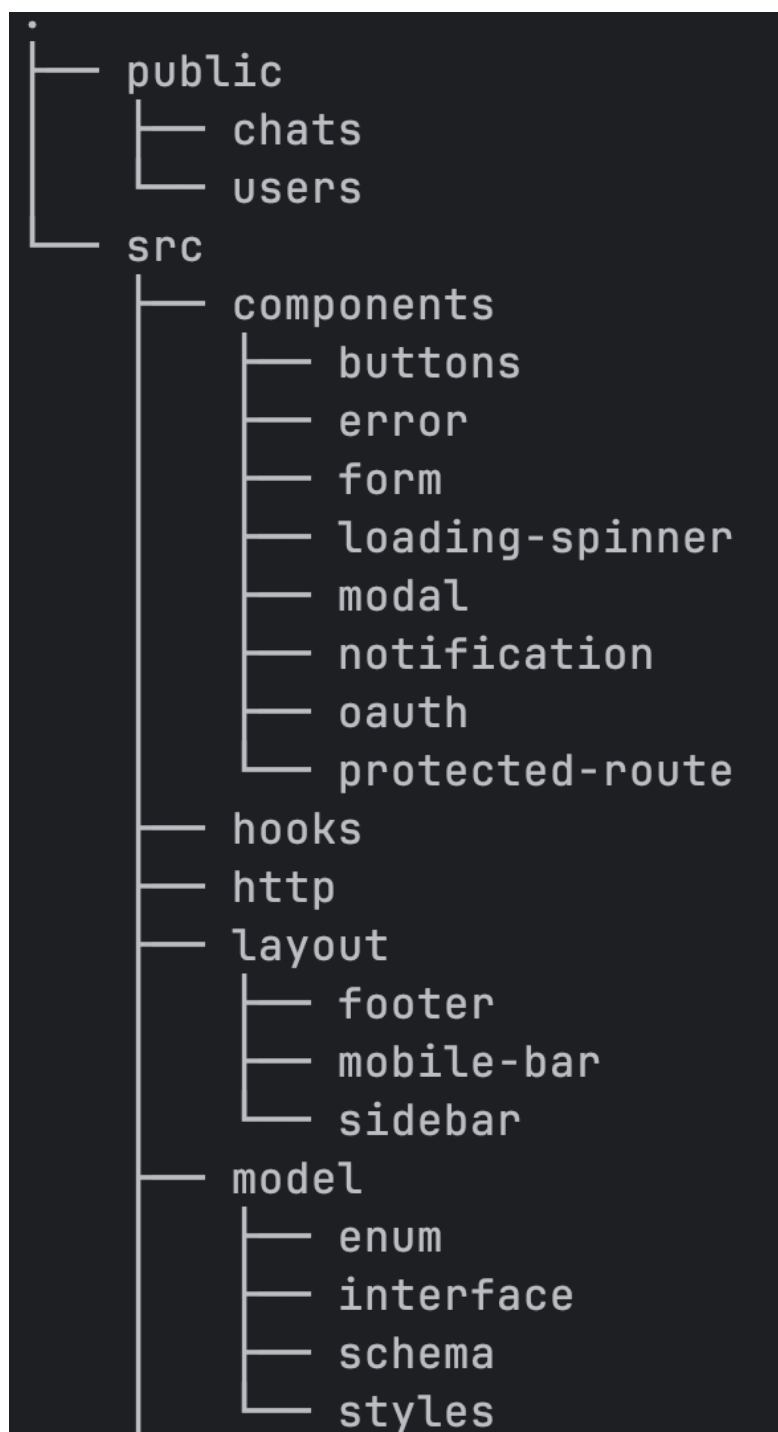
#### 7.1.3 Obsługa uwierzytelnienia

#### 7.1.4 Konteneryzacja

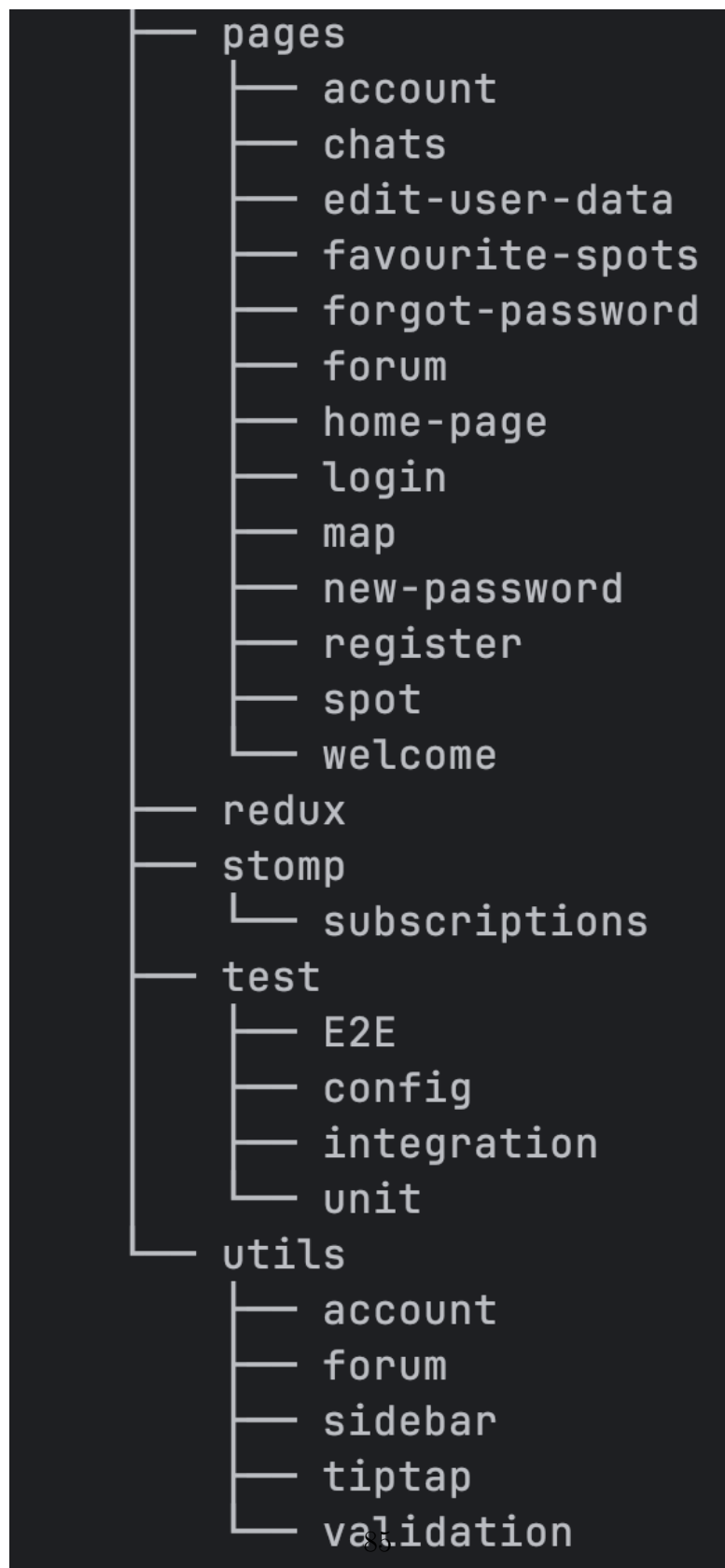
### 7.2 Implementacja frontendu

#### 7.2.1 Struktura aplikacji

Architektura aplikacji frontendowej została zaprojektowana w strukturze [Folder by type](#), która polega na podziale kodu według typu zasobu (komponenty, strony, modele itd.). Każdy plik znajduje się w katalogu odpowiadającym jego przeznaczeniu, co jest przedstawione na rysunkach [7.1](#) oraz [7.2](#).



Rysunek 7.1: Struktura katalogów (1)



Rysunek 7.2: Struktura katalogów (2)

Głównym elementem aplikacji jest mechanizm routingu oparty na [Bibliotece React Router](#). Definiuje on ścieżki do poszczególnych funkcjonalności aplikacji. Dzięki temu możliwa jest płynna nawigacja między różnymi widokami bez konieczności przeładowywania strony.

```
const router : Router = createBrowserRouter([
  {
    path: "/",
    element: <Layout />,
    errorElement: <Error error={undefined} />,
    children: [
      {
        index: true,
        element: <HomePage />,
      },
      {
        path: "advanced",
        element: <AdvanceHomePage />,
      },
      {
        path: "account",
        children: [ 11 elements... ],
      },
      {
        path: "register",
        element: <Register />,
      },
      {
        path: "login",
        element: <Login />,
      },
      {
        path: "forgot-password",
        element: <ForgotPassword />,
      },
    ],
  },
]);
```

Rysunek 7.3: Implementacja routera (1)



```

    {
      path: "new-password",
      element: <NewPassword />,
    },
    {
      path: "forum",
      element: <Forum />,
    },
    {
      path: "forum/:postId/:slugTitle?",
      element: <ForumThread />,
    },
    {
      path: "map",
      element: <MapPage />,
    },
    {
      path: "chat",
      element: (
        <ProtectedRoute>
        |   <ChatsPage />
        | </ProtectedRoute>
      ),
    },
  ],
);

export default router;

```

Rysunek 7.4: Implementacja routera (2)

W projekcie zastosowano również wzorzec [Protected route](#), który służy do zabezpieczania wybranych tras przed dostępem użytkowników niezalogowanych. W pliku `router.tsx`, znajdującym się w głównym katalogu projektu, w konfiguracji przekazywanej do funkcji `createBrowserRouter` (rysunki [7.3](#) oraz [7.4](#)), wybrane

ścieżki zostały opakowane w komponent `ProtectedRoute`. Komponent ten pełni rolę bramki (rysunek 7.5).

Przykładem takiej chronionej ścieżki jest trasa `/chat`, prowadząca do modułu czatu dostępnego wyłącznie dla zalogowanych użytkowników. Jeśli niezalogowany użytkownik spróbuje uzyskać dostęp do tej ścieżki, zostanie automatycznie przekierowany na stronę główną.

```
export default function ProtectedRoute({ children }) {
  const isLoggedIn = useSelector((state) => state.account.isLoggedIn);

  return isLoggedIn ? children : <Navigate to="/" />;
}
```

Rysunek 7.5: Implementacja komponentu bramki (`ProtectedRoute`)

### 7.2.2 Zarządzanie stanem i przepływ danych

W projekcie postawiliśmy na zrównoważone podejście do zarządzania [Stanem](#). Korzystamy zarówno z lokalnego [Stanu](#) komponentów (za pomocą [Hook \(React\)](#) `useState`) [15], jak i ze [Stanu](#) globalnego, utrzymywanego przez [Bibliotekę React Redux](#) [16]. Globalny [Stan](#) został wprowadzony po to, aby możliwie najbardziej ograniczyć przekazywanie [Propsów](#) w głąb drzewa komponentów oraz uniknąć niepotrzebnych ponownych renderów.

Do przechowywania [Stanu](#) lokalnego, ograniczonego tylko do danego komponentu (lub jego najbliższych elementów podrzędnych), wykorzystujemy [Hook \(React\)](#) `useState`. Natomiast efekty uboczne i synchronizację realizujemy za pomocą `useEffect`. W przypadku bardziej złożonej logiki lub potrzeby ponownego wykorzystania kodu powstały [Hook \(React\)](#)i niestandardowe, takie jak `useScreenSize`, `useDarkMode` czy `useClickOutside`. Dzięki temu większość logiki prezentacji została wydzielona z warstwy [UI](#), co poprawia czytelność i ułatwia utrzymanie kodu.

Z racji tego, że korzystamy z [Reacta](#) w połączeniu z [TypeScriptem](#), przygotowaliśmy również własne [Hook \(React\)](#)i wspomagające typowanie, takie jak `useDispatchTyped` oraz `useSelectorTyped`. Pozwalają one na bezpieczne typowanie

wanie akcji oraz selektorów [Reduxa](#) bez konieczności powtarzania adnotacji typów w każdym komponencie. Fragmenty tej implementacji przedstawiono na rysunkach [7.6](#) oraz [7.7](#).

```
const store : EnhancedStore<{ account: AccountSliceProp... = configureStore({
  reducer: {
    account: accountSlice.reducer,
    notification: notificationSlice.reducer,
    spotDetails: spotDetailsModalSlice.reducer,
    searchedSpotsListModal: searchedSpotListModalSlice.reducer,
    expandedSpotMediaGallery: expandedSpotMediaGallerySlice.reducer,
    spotFilters: spotFiltersSlice.reducer,
    chats: chatsSlice.reducer,
    map: mapSlice.reducer,
    sidebar: sidebarSlice.reducer,
    searchedSpots: searchedSpotsSlice.reducer,
    social: socialSlice.reducer,
    spotComments: spotCommentSlice.reducer,
    currentViewSpots: currentViewSpotsSlice.reducer,
    currentViewSpotsListModal: currentViewSpotsListModalSlice.reducer,
    currentViewSpotsParams: currentViewSpotParamsSlice.reducer,
    spotWeather: spotWeatherSlice.reducer,
    expandedSpotGalleryMediaList: expandedSpotGalleryMediaListSlice.reducer,
    expandedSpotMediaGalleryModals:
      expandedSpotMediaGalleryModalsSlice.reducer,
    expandedSpotMediaGalleryFullscreenSizeModal:
      expandedSpotMediaGalleryFullscreenSizeSlice.reducer,
    expandedSpotGalleryCurrentMedia:
      expandedSpotGalleryCurrentMediaSlice.reducer,
  },
});

export default store; Show usages  Mredosz
export type RootState = ReturnType<typeof store.getState>;
export type AppDispatch = typeof store.dispatch;
```

Rysunek 7.6: Konfiguracja sklepu (Redux store)

```

interface AccountSliceProps { Show usages  Mredosz +1
  isLoggedIn: boolean;
  username: string;
}

const initialState: AccountSliceProps = {
  isLoggedIn: localStorage.getItem("is_logged_in") === "true",
  username: localStorage.getItem("username") || "",
};

export const accountSlice : Slice<AccountSliceProps, { setIsLoggedIn(st... = createSlice({ Show usages  Mredosz +1
  name: "account",
  initialState,
  reducers: {
    setIsLoggedIn(state : WritableDraft<AccountSliceProps> ) : void {
      localStorage.setItem("is_logged_in", "true");
      state.isLoggedIn = true;
    },
    signOut(state : WritableDraft<AccountSliceProps> ) : void {
      localStorage.removeItem("is_logged_in");
      localStorage.removeItem("username");
      state.isLoggedIn = false;
      state.username = "";
    },
    setUsername(state : WritableDraft<AccountSliceProps> , action: PayloadAction<string>) : void {
      localStorage.setItem("username", action.payload);
      state.username = action.payload;
    },
  },
});

export const accountAction : CaseReducerActions<{ setIsLoggedIn(state: W... = accountSlice.actions; Show usages  Mredosz

```

Rysunek 7.7: Przykładowy slice odpowiedzialny za sprawdzenie czy użytkownik jest zalogowany

### 7.2.3 Integracja i komunikacja z backendem

Jest to kluczowy element aplikacji, ponieważ wymaga bezpiecznego przesyłania danych użytkownika. W celu uproszczenia komunikacji z serwerem skorzystaliśmy z biblioteki [axios](#) [17] oraz [Biblioteki TanStack Query](#) [18]. We wszystkich ścieżkach, które wymagają aby użytkownik był zalogowany, do zapytania dołączany jest token [JWT](#). Token jest przekazywany w ciasteczku dzięki ustawieniu parametru `withCredentials` na wartość `true`. Przykładem pliku odpowiedzialnego za taką komunikację jest `account.js` (rys. 7.8 i 7.9), który obsługuje operacje związane z

logowaniem, rejestracją, zmianą hasła oraz wylogowaniem.

```
export async function loginUser(user) { Show usages new *
  return await axios.post(`${BASE_URL}/public/account/login`, user, {
    withCredentials: true,
  });
}

export async function registerUser(user) { Show usages Adam Langmesser +2
  return await axios.post(`${BASE_URL}/public/account/register`, user, {
    withCredentials: true,
  });
}

export async function setEmailWithNewPasswordLink(email) { Show usages Adam Langmesser +1
  console.log("sending email...");
  return await axios.post(
    `${BASE_URL}/public/account/forgot-password`,
    email,
    {
      headers: {
        "Content-Type": "text/plain",
      },
    },
  );
}
```

Rysunek 7.8: Implementacja modułu account (1)

```

export async function changePassword(userData) { Show usages  ⓘ stanoz +1
  return await axios.post(
    `${BASE_URL}/public/account/set-new-password`,
    userData,
  );
}

export async function logout() { Show usages  ⓘ stanoz +1
  await axios.post(
    `${BASE_URL}/account/oauth2/logout`,
    {},
    {
      withCredentials: true,
    },
  );
}

export const googleLoginUrl = `${BASE_URL}/oauth2/authorization/google`; Show usages  ⓘ stanoz
export const githubLoginUrl = `${BASE_URL}/oauth2/authorization/github`; Show usages  ⓘ stanoz

```

Rysunek 7.9: Implementacja modułu `account` (2)

Funkcje odpowiedzialne za komunikację z backendem zostały umieszczone w katalogu `/http`. Dzięki temu są one scentralizowane i mogą być w prosty sposób wykorzystywane w różnych częściach aplikacji. Zastosowaliśmy TanStack Query, ponieważ znacząco ogranicza on powtarzalny kod oraz upraszcza obsługę błędów i stanów zapytania (takich jak ładowanie danych, błąd, sukces). Udostępnia m.in. wartość `isLoading`, dzięki czemu komponent może łatwo wyświetlić ekran ładowania bez ręcznego zarządzania własnym stanem. Dodatkowo [Hook \(React\) `useQuery`](#) z tej [Biblioteki](#) umożliwia automatyczne pobieranie danych po wejściu na daną podstronę. Oznacza to, że komponent deklaruje jedynie „jakie dane są mu potrzebne”, a TanStack Query zajmuje się ich pobraniem, cache’owaniem oraz odświeżaniem. Do operacji, które wymagają wywołania akcji po stronie użytkownika (np. wysłania formularza logowania), wykorzystujemy [Hook \(React\) `useMutation`](#) z TanStack Query. Przykład użycia tego rozwiązania w procesie logowania został przedstawiony na rys. 7.10.
















```
const { mutateAsync, isSuccess, error } = useMutation({
  mutationFn: loginUser,
});

const handleSubmit: (event: FormEvent<HTMLFormElement>) => Promise<void> = async (event: FormEvent<HTMLFormElement>) => {
  event.preventDefault();
  await mutateAsync({
    username: enteredValue.username,
    password: enteredValue.password,
  });
  navigate(-1);
};
```

Rysunek 7.10: Wykorzystanie TanStack Query przy logowaniu użytkownika

## 7.2.4 Style

Do stylowania interfejsu wykorzystaliśmy [Framework](#) Tailwind CSS [19]. Dzięki gotowym klasom udostępnianym przez Tailwind mogliśmy definiować wygląd elementów bezpośrednio w kodzie komponentu, bez konieczności przechodzenia do osobnych plików ze stylami. Ułatwia to zarówno tworzenie widoków, jak i późniejsze modyfikacje — w przypadku zmiany stylu dokładnie wiadomo, gdzie należy jej dokonać. Korzystanie ze zdefiniowanych klas pozwoliło nam również zachować spójność wizualną w całej aplikacji. W pliku `index.css` zdefiniowaliśmy zmienne kolorystyczne (rys. 7.11 i 7.12). Dzięki temu zmiana motywu kolorystycznego w przyszłości sprowadza się do edycji wartości w jednym miejscu.

	<code>--color-violetDark: #363041;</code>
	<code>--color-violetLight: #6d6183;</code>
	<code>--color-violetLightDarker: #4f4660;</code>
	<code>--color-violetLightDark: #554a69;</code>
	<code>--color-violetLighter: #9b8cbd;</code>
	<code>--color-violetDarker: #2c2734;</code>
	<code>--color-violetHeavyDark: #1e1b23;</code>
	<code>--color-violetBtnBorderDark: #625b6e;</code>
	<code>--color-violetBright: #835ace;</code>
	<code>--color-darbVioletBtnOutline: #816ba6;</code>
	<code>--color-mediumDarkBlue: #424b77;</code>
	<code>--color-first: #2c3e50;</code>
	<code>--color-second: #34495e;</code>
	<code>--color-third: #1abc9c;</code>
	<code>--color-fourth: #16a085;</code>
	<code>--color-fifth: #ecf0f1;</code>
	<code>--color-sixth: #e94560;</code>
	<code>--color-magenta: #a01bc1;</code>
	<code>--color-darkYellow: #c5a03c;</code>
	<code>--color-ratingStarColor: #fadb14;</code>
	<code>--color-locationMarkerDarkerBlue: #a3dcff;</code>
	<code>--color-locationMarkerLightBlue: #52bafb;</code>
	<code>--color-userLocationDot: #4285f4;</code>
	<code>--color-spotLocationMarker: #a8071a;</code>

Rysunek 7.11: Implementacja zmiennych kolorystycznych (1)





Rysunek 7.12: Implementacja zmiennych kolorystycznych (2)

W niektórych miejscach konieczne było zapisanie stylów w czystym [CSS](#), ponieważ część użytych [Bibliotek](#) tego wymagała. W innych przypadkach wystarczyło skorzystać z klas zdefiniowanych w `index.css` oraz klas Tailwinda. Cała aplikacja

jest [Responsywna](#). Tailwind udostępnia predefiniowane prefiksy [Responsywne](#) (np. `md:`, `lg:`) (rys. 7.13), stworzyliśmy również własny (`3xl:`) na ekrany o rozdzielczości 2560px. Pozwalają one przypisywać style zależnie od szerokości ekranu bez pisania własnych reguł `@media`. Dzięki temu implementacja widoków mobilnych i desktopowych była znacząco szybsza.

```
<div className="mt-17 flex flex-col items-center gap-7 lg:mt-0 lg:-ml-40 lg:flex-row xl:-ml-42 xl:gap-10 2xl:-ml-80">
  <div className="relative">
    <img
      alt="profileImage"
      src={userData?.profilePhoto}
      className="dark:drop-shadow-darkBgMuted aspect-square h-64 rounded-full
        shadow-md sm:h-80 lg:h-85 xl:h-96 dark:drop-shadow-md"
    />
  </div>
</div>
```

Rysunek 7.13: Przykładowe użycie klas Tailwind (w tym prefiksów responsywności)

Tailwind został też wykorzystany do obsługi trybu jasnego i ciemnego. Wystarczy dodać klasę z prefiksem `dark:` (np. `dark:bg-black`), aby zmienić kolorystykę elementu, gdy aplikacja jest w trybie ciemnym (rys. 7.14).

```
<input
  id={id}
  value={value}
  type={type}
  onChange={onChange}
  onFocus={setFocusedToTrue}
  onBlur={handleOnBlur}
  className="dark:bg-darkBgMuted bg-lightBgMuted dark:text-darkText text-lightText w-full
    rounded-md p-2 shadow-md focus:outline-none dark:shadow-black/50"
/>
```

Rysunek 7.14: Przykładowe użycie klas Tailwind (w tym wariantu `dark:`)

Aby uzyskać płynniejsze i przyjemniejsze animacje, wykorzystaliśmy [Bibliotekę Motion](#) [20]. Pozwala ona w prosty sposób tworzyć animacje elementów interfejsu, bez potrzeby ręcznego pisania złożonych reguł [CSS](#). W naszej aplikacji użyliśmy jej `m.in.` w polach formularza logowania i rejestracji (rys. 7.15). Na początku etykieta pola (np. „username”) jest wyświetlana wewnątrz pola tekstowego, natomiast po

kliknięciu w pole jest płynnie przesuwana nad to pole, co poprawia czytelność i ergonomię formularza.

```
<div className="relative">
  <motion.label
    htmlFor={id}
    initial={false}
    animate={{
      top: shouldFloat ? "-0.7rem" : "0.5rem",
      left: "0.75rem",
      fontSize: shouldFloat ? "0.75rem" : "1rem",
      opacity: shouldFloat ? 1 : 0.6,
    }}
    transition={{ type: "spring", stiffness: 300, damping: 25 }}
    className="dark:text-darkText text-lightText pointer-events-none absolute z-10 px-1 capitalize"
  >
    {label}
  </motion.label>
  <input
    id={id}
    value={value}
    type={type}
    onChange={onChange}
    onFocus={setFocusedToTrue}
    onBlur={handleOnBlur}
    className="dark:bg-darkBgMuted bg-lightBgMuted dark:text-darkText text-lightText w-full rounded-md p-2 shadow-md focus:outline-none dark:shadow-black/50"
  />
```

Rysunek 7.15: Implementacja animacji z wykorzystaniem Motion

## 7.2.5 Wyszukiwarka spotów

Niniejszy rozdział opisuje sposób implementacji wyszukiwarki spotów.

Jednym z głównych modułów aplikacji jest wyszukiwarka spotów, która umożliwia użytkownikowi szybkie odnalezienie interesujących lokalizacji. Funkcjonuje ona w dwóch wariantach: prostym i zaawansowanym (rys. 7.16 oraz 7.17).

```

<div className={`${dark:bg-darkBg} ${dark:text-darkText} ${bg-lightBg} ${text-lightText}
flex min-h-screen w-full flex-col items-center space-y-4 overflow-hidden p-8 pt-18">
  <Switch />
  <SearchBar
    onSetSpots={handleSetSearchedSpots}
    loadMoreRef={loadMoreRef}
    onSetFetchingNextPage={setIsFetchingNextPage}
  />
  <div className="flex w-full flex-col items-center space-y-4">
    <h1 className="text-center text-3xl">The Most Popular Spots</h1>
    <div className="flex w-full flex-col items-center space-y-5">
      <Carousel spots={data!} spotsPerPage={spotsPerPage} />
      <SearchSpotList
        spots={searchedSpots}
        isFetchingNextPage={isFetchingNextPage}
        loadMoreRef={loadMoreRef}
      />
    </div>
  </div>
</div>

```

Rysunek 7.16: Implementacja prostej wersji wyszukiwarki

```

<div className={`${dark:bg-darkBg} ${dark:text-darkText} ${bg-lightBg} ${text-lightText}
flex min-h-screen w-full flex-col items-center space-y-4 overflow-hidden p-8 pt-18">
  <Switch />
  <AdvanceSearchBar
    onSetSpots={handleSetSearchedSpots}
    loadMoreRef={loadMoreRef}
    onSetFetchingNextPage={setIsFetchingNextPage}
  />
  <div className="flex w-full flex-col items-center space-y-10">
    <SearchSpotList
      spots={searchedSpots}
      loadMoreRef={loadMoreRef}
      isFetchingNextPage={isFetchingNextPage}
    />
  </div>
</div>

```

Rysunek 7.17: Implementacja zaawansowanej wersji wyszukiwarki

Przełączanie pomiędzy tymi widokami odbywa się za pomocą przycisku umieszczonego w górnej części strony (rys. 7.18).

```
<div className="dark:shadow-darkBgSoft flex rounded-full shadow-lg shadow-black/20">
  <NavLink
    to="/"
    className={({ isActive } : NavLinkRenderProps ) : string =>
      `dark:shadow-darkBgSoft flex rounded-full shadow-lg shadow-black/20
        hover:dark:bg-violetDark hover:bg-violetLight rounded-l-full px-2.5 py-1.5
        transition-all duration-300 ${isActive ? "dark:bg-violetDark bg-violetLight" : ""}`
  >
    Simple filters
  </NavLink>
  <NavLink
    to="/advanced"
    className={({ isActive } : NavLinkRenderProps ) : string =>
      `dark:shadow-darkBgSoft flex rounded-full shadow-lg shadow-black/20
        hover:dark:bg-violetDark hover:bg-violetLight rounded-r-full px-2.5 py-1.5
        transition-all duration-300 ${isActive ? "dark:bg-violetDark bg-violetLight" : ""}`
  >
    Advanced filters
  </NavLink>
</div>
```

Rysunek 7.18: Implementacja komponentu do przełączania trybów

W trybie prostym prezentowana jest karuzela (rys. 7.19) z dwunastoma najpopularniejszymi **spotami** w całej aplikacji. Użytkownik może w tym miejscu wyszukiwać **spoty** po lokalizacji (kraj, region, miasto).

```

<div className="relative flex w-full items-center justify-center">
  <button
    onClick={() : void => paginate(-1)}
    className="hover:text-darkBorder z-10 cursor-pointer transition-all duration-300"
  >
    <RiArrowLeftWideFill className="text-5xl sm:text-6xl" />
  </button>

  <div className="relative h-[440px] w-full max-w-[1200px] overflow-hidden">
    <AnimatePresence custom={direction} initial={false} mode="sync">
      <motion.div
        key={page}
        custom={direction}
        variants={sliderVariants}
        initial="incoming"
        animate="active"
        exit="exit"
        transition={[ 3 elements... ]}
        className="grid w-full grid-cols-1 grid-rows-1 justify-items-center gap-4
          lg:grid-cols-2 lg:grid-rows-2 2xl:grid-cols-3 2xl:grid-rows-2"
      >
        {currentSpots.map((spot : TopRatedSpot ) : Element => (
          <MostPopularSpot
            spot={spot}
            key={` ${spot.id}-${page}`}
          />
        ))}
      </motion.div>
    </AnimatePresence>
  </div>

  <button
    onClick={() : void => paginate(1)}
    className="hover:text-darkBorder z-10 cursor-pointer transition-all duration-300"
  >
    <RiArrowRightWideFill className="text-5xl sm:text-6xl" />
  </button>
</div>

```

Rysunek 7.19: Implementacja karuzeli z najpopularniejszymi [spotami](#)

Widok zaawansowany udostępnia rozszerzoną wyszukiwarkę, która umożliwia filtrowanie wyników po mieście, tagach oraz ocenie, a także ich sortowanie według popularności i średniej oceny (rys. 7.17).

Wyszukiwarka spotów została zbudowana z dwóch głównych komponentów: `HomePage` oraz `AdvanceHomePage`. W skład prostej wersji wchodzi następujące komponenty:

- `Switch` – służy do przełączania widoku między trybem podstawowym a zaawansowanym,
- `SearchBar` – podstawowa wyszukiwarka [spotów](#),
- `Carousel` – wyświetla najpopularniejsze [spoty](#),
- `SearchSpotList` – wyświetla wyszukiwane [spoty](#).

W skład zaawansowanej wersji wchodzi następujące komponenty:

- `Switch` – służy do przełączania widoku między trybem podstawowym a zaawansowanym,
- `AdvanceSearchBar` – zaawansowana wyszukiwarka [spotów](#),
- `SearchSpotList` – wyświetla wyszukiwane [spoty](#).

Komponent `Switch` (rys. 7.18) zawiera dwa elementy `NavLink` z [biblioteki](#) `React Router`, co pozwala na przełączanie widoków bez konieczności przeładowywania całej strony.

W komponencie `SearchBar` (rys. 7.20) po wpisaniu co najmniej dwóch znaków wyświetlana jest lista podpowiedzi dla kraju, regionu oraz miasta, w zależności od aktualnie uzupełnianego pola. Po pojawieniu się listy użytkownik może wybrać interesującą go lokalizację, co ułatwia określenie, w jakich miejscach znajdują się dostępne [spoty](#).

```

<div className="dark:bg-darkBgSoft light:bg-lightBgSoft flex w-full flex-col items-center justify-between
space-y-3 rounded-md px-3 py-2 shadow-md md:flex-row md:space-y-0 lg:w-3/4 lg:space-x-3 xl:w-1/2
dark:shadow-black">
  <div className="flex w-full flex-col space-y-2">
    <h1>Location</h1>
    <div className="flex w-full flex-col space-y-3 md:flex-row md:space-y-0 md:space-x-2">
      {inputList.map(({ id, label } : { readonly label: "Your Country"; readonl... } : Element ) => (
        <div key={id} className="relative w-full">
          <SearchInput
            label={label}
            id={id}
            value={searchLocation[id] ?? ""}
            onChange={(e : ChangeEvent<HTMLInputElement> ) : void =>
              handleSetLocation(id, e.target.value)
            }
            onFocus={() : void => setActiveInput(id)}
          />
          {activeInput === id && suggestions.length > 0 && (
            <SearchSuggestions
              suggestions={suggestions}
              onClick={handleSuggestionClick}
              id={id}
              onClose={() : void => setActiveInput(null)}
            />
          )}
        </div>
      )}
    </div>
  </div>
  <button
    className="dark:bg-darkBgMuted dark:hover:bg-darkBgMuted/80 light:bg-lightBgMuted
    light:hover:bg-lightBgMuted/80 flex w-full cursor-pointer justify-center rounded-md p-2 md:w-fit"
    onClick={handleSearchSpots}
  >
    <FaSearch />
  </button>
</div>

```

Rysunek 7.20: Implementacja prostej wyszukiwarki

Komponent `SearchSpotList` (rys. 7.21) odpowiada za prezentację wyników wyszukiwania. Został w nim zaimplementowany mechanizm przewijania nieskończonego (*infinite scroll*), który automatycznie pobiera kolejne strony wyników w momencie, gdy użytkownik zbliża się do końca listy. Wykorzystuje on listę komponentów `SpotTile`, a także komponent `LoadingSpinner` oraz komunikat informujący o braku wyników, jeżeli nie zostanie odnaleziony żaden *spot*.



```

</>
<ul className="grid w-full grid-cols-1 place-items-center gap-8 xl:grid-cols-2 2xl:grid-cols-3">
  {spots.map((spot : HomePageSpotDto ) : Element => (
    |
    <SpotTile key={spot.id} spot={spot} />
    |
  ))}
</ul>
<div ref={loadMoreRef} className="h-10" />
{isFetchingNextPage && <LoadingSpinner />}
{spots.length === 0 && (
  <p className="text-center text-2xl">
    Search for spots to see results.
  </p>
)}
</>

```

Rysunek 7.21: Implementacja listy do wyświetlania [spotów](#)

Komponent `SpotTile` zawiera następujące informacje:

- zdjęcie [spota](#),
- miasto, w którym się znajduje,
- nazwę [spota](#),
- ocenę oraz liczbę ocen,
- tagi,
- podstawowe informacje pogodowe (temperatura i typ pogody),
- dwa przyciski: jeden prowadzący do widoku szczegółów [spota](#) oraz drugi informujący, jak daleko znajduje się dany [spot](#); po kliknięciu przycisku lokalizacja [spota](#) jest prezentowana na mapie.

Komponent `AdvanceSearchBar` jest zbliżony wyglądem i strukturą do wersji podstawowej, jednak w polu lokalizacji można podać wyłącznie miasto. Dodatkowo dostępna jest możliwość dodawania tagów z przygotowanej listy. Wyszukiwarka umożliwia także filtrowanie po ocenie oraz sortowanie wyników według oceny i popularności z wykorzystaniem komponentów typu `Dropdown`.

Oba widoki (`HomePage` i `AdvanceHomePage`) współdzielą część komponentów, między innymi `Switch` oraz `SearchSpotList`. Dzięki temu kod odpowiedzialny za wyświetlanie listy wyników jest zdefiniowany w jednym miejscu, a zmiany w sposobie prezentacji [spotów](#) wymagają modyfikacji tylko w komponentach współdzielonych.

#### **7.2.6 Mapa**

#### **7.2.7 Chat**

#### **7.2.8 Forum**

#### **7.2.9 Konto użytkownika**

#### **7.2.10 Panel logowania**

### **7.3 Implementacja CI/CD**

# Rozdział 8

## Testy

8.1 Testy jednostkowe

8.2 Testy integracyjne

8.3 Testy E2E

8.4 Wyniki testów i wnioski

## Rozdział 9

### Prezentacja systemu

9.1 Strona główna

9.2 Strona mapy

9.3 Strona chatu

9.4 Strona forum

9.5 Panel logowania

9.6 Panel konta użytkownika

# Rozdział 10

## Nakład pracy

### 10.1 Ogólny nakład pracy

### 10.2 Indywidualne nakłady pracy

#### 10.2.1 Adam Langmesser

#### 10.2.2 Mateusz Redosz

Na projekt poświęciłem łącznie 324 godziny, z czego 237 przeznaczyłem na prace deweloperskie, 111 na pisanie dokumentacji, 19 godzin na [Review kodu](#), 19 na spotkania dotyczące omówienia dalszych prac projektowych oraz przy pomocy innym członkom zespołu oraz 49 godzin poświęciłem nad stworzeniem widoków na figmie. Prace nad częścią deweloperską rozpocząłem 04.08.2024 a zakończyłem 08.09.2025. W projekcie pracowałem nad Rejestracją użytkownika, tokenem [JWT](#), częściową implementacją [CI/CD](#), stroną główną, zaimplementowaniem [Sidebara](#) oraz podstroną dla użytkownika. Moje wylistowane zadania z Jira:

#### 1. Dokumentacja

- TODO

#### 2. [Design](#)

- Ustalić paletę kolorystyczną

- Propozycja wyglądu

### 3. [Backend](#) i [Frontend](#)

- Formularz rejestracji
- Routing
- Formatowanie w React (prettier)
- Obsługa JWT na frontend
- OAuth Frontend
- Update JWT
- Refactor JWT
- Stworzenie komponentu Notification i poprawa błędów
- Implementacja pierwszych testów
- Zaimplementowanie kolejki w komponencie notification
- Dodanie reduxa do rejestracji
- Zmiana sposobu pobierania danych o spotach
- Obsługa customowych błędów z jakarta.validation
- Obsługa auto wylogowania przy starcie
- Domyślna wiadomość w notification
- Poprawa headera
- Ciemny motyw
- Refactor pogody
- Propozycja wyglądu
- Przeniesienie zdjęć z google drive
- Dodać Type script do Reacta
- Aktualizacja tailwinda i dodanie kolorów
- Podstawowy [Sidebar](#)

- Strona główna z prostymi filtrami
- Strona główna z zaawansowanymi filtrami
- [Sidebar](#)
- Strona profilu
- Ustawienia
- Listy spotów
- Lista zdjęć
- Lista filmów
- Lista znajomych
- Dodanie spotów
- Lista komentarzy
- Strona główna profilu
- Listy
- Poprawa [Sidebara](#)
- Zmiana kropki na przyciemnienie tła na [Sidebar](#)
- Poprawa strony do logowania i rejestracji
- Usunięcie username z account Redux
- Dodanie zamknięcia [Sidebara](#) na małych ekranach po kliknięciu nav linka
- Poprawić tooltipa na sidebar
- Zmiana sposobu pobierania username na backendzie z tokena jwt
- Paginacja z infinity scrollem
- Lista zdjęć innego usera
- Walidacja i responsywność w dodaniu spotów
- Dodanie sortowania i filtrów na zaawansowanej stronie
- Zmiana na infinity scrola

- Zmiana zdjęcia profilowego użytkownika
- Czyszczenie formularza w dodawaniu spota
- Dodanie wyszukiwarki znajomych w Social
- Zatwierdzenie przez drugiego użytkownika dodania do znajomych
- Sprawdzenie czy wszystko działa i poprawki Mateusz

#### 4. [CI/CD](#)

- Dodanie testów z frontendu do github actions
- Poprawa github actions
- Poprawa pipeline od Javy i Reacta

#### 5. Praca dyplomowa

- Uzupełnienie informacji o zespole i podział na rozdziały

### **10.2.3 Stanisław Oziemczuk**

### **10.2.4 Kacper Badek**



# Rozdział 11

## Podsumowanie

- 11.1 Osiągnięte rezultaty
- 11.2 Napotkane wyzwania
- 11.3 Plany na przyszłość

# Rozdział 12

## Słownik pojęć i skrótów

### API

(ang. *application programming interface*); zbiór reguł i operacji do komunikacji z oprogramowaniem.. [16](#)

### Backend

Część aplikacji odpowiedzialna za logikę biznesową, przetwarzanie danych i komunikację z bazą danych. Działa po stronie serwera i obsługuje żądania wysyłane przez frontend. [2](#), [15](#), [106](#)

### Backlog

Lista zadań, które należy wykonać w ramach projektu, używane w metodykach zwinnych.. [15](#)

### Biblioteka

Zewnętrzny lub wewnętrzny zestaw gotowych funkcji, klas, komponentów lub modułów, który można wielokrotnie wykorzystywać w projekcie zamiast pisać wszystko od zera. [84](#), [86](#), [88](#), [90](#), [93](#), [94](#), [99](#)

### BPMN

(ang. *Business Process Model and Notation*); standardowa notacja graficzna, która umożliwia szczegółowe przedstawienie i dokumentowanie procesów biznesowych.. [17](#)

## CI/CD

Skrót od *Continuous Integration/Continuous Deployment*. Praktyka programistyczna polegająca na automatyzacji procesu budowania, testowania i wdrażania oprogramowania. [16](#), [105](#), [108](#)

## CSS

Kaskadowe arkusze stylów (Cascading Style Sheets) — język opisu prezentacji dokumentów (np. HTML). Definiuje wygląd interfejsu: układ, kolory, typografię, odstępy, animacje i zachowania responsywne, oddzielając warstwę treści od warstwy prezentacji.. [93](#), [94](#)

## Design

Etap lub proces projektowania wyglądu i funkcjonalności aplikacji, obejmujący zarówno aspekty wizualne, jak i użytkowe (UX/UI). [105](#)

## Disciplined Agile Delivery - Lean Life Cycle

Disciplined Agile Delivery w wariacie Lean Life Cycle to sposób prowadzenia projektu, który łączy elastyczność Agile z przewidywalnością Waterfalla, ale bez stałych sprintów — praca toczy się w ciągłym przepływie. Na starcie zakłada mocniejszą fazę przygotowawczą: doprecyzowanie zakresu, szkic architektury, identyfikację ryzyk i kryteria jakości. W realizacji następuje ciągłe doprecyzowywanie wymagań i backlogu, oparte na regularnym feedbacku udziałowców. Całość opiera się na praktykach Lean oraz lekkim governance: code review i regularnych przeglądach postępów. . [10](#)

## Droniarz

Potoczne określenie osoby, która jest jednocześnie pilotem oraz operatorem drona. Zwykle entuzjasta dronów.. [8](#)

## Droniarz foto/video

Pilot wykorzystujący drony fotograficzne/filmowe do rejestracji materiałów wizualnych (zdjęcia, wideo), zwykle z naciskiem na stabilizację i jakość obrazu.. [17](#)

## Folder by type

Sposób organizowania struktury katalogów w projekcie, w którym pliki są grupowane według rodzaju (typu) zasobu, a nie według funkcjonalności. Na przykład wszystkie komponenty trafiają do jednego folderu, wszystkie style do innego itd. [81](#)

## Framework

Zestaw narzędzi, bibliotek i struktur wspomagających tworzenie aplikacji. Ułatwia programowanie poprzez dostarczenie gotowych komponentów oraz określenie zasad organizacji kodu. [2](#), [91](#)

## Frontend

Warstwa aplikacji odpowiedzialna za interfejs użytkownika oraz interakcję z użytkownikiem. Zazwyczaj tworzona przy użyciu technologii takich jak HTML, CSS i JavaScript. [2](#), [15](#), [106](#)

## Hook (React)

Prosta funkcja w React, która „dodaje” możliwości do elementu interfejsu — np. pozwala mu coś zapamiętać (stan) albo zrobić coś po zmianie/załadowaniu. Wszystkie hooki zaczynają się od `use...` (np. `useState`, `useEffect`).. [86](#), [90](#)

## IDE

(ang. *integrated development environment*); to zintegrowane środowisko programistyczne, służące do tworzenia, modyfikowania, testowania i konserwacji oprogramowania. [14](#)

## Infinite scroll

Wzorzec interfejsu użytkownika, w którym kolejne porcje treści są automatycznie doładowywane podczas przewijania strony w dół, zamiast być podzielone na odrębne, ręcznie przełączane strony. [100](#)

## JWT

Skrót od *JSON Web Token*. Standard służący do bezpiecznego przekazywania informacji między stronami w formacie JSON, często używany w procesach autoryzacji użytkowników. [88](#), [105](#)

## Media queries

Konstrukcja CSS pozwalająca stosować reguły stylów w zależności od cech urządzenia/okna (np. szerokości ekranu, orientacji, preferencji użytkownika). Podstawa responsywnego projektowania (*responsive design*).. [114](#)

## MoSCoW

Metoda nadawania priorytetów wymaganiom, wyróżniająca kategorie: *Must have*, *Should have*, *Could have* oraz *Won't have this time*. [21](#)

## PANSA

Polish Air Navigation Services Agency, pol. Polska Agencja Żeglugi Powietrznej. Instytucja ta zapewnia m.in. mapę z zaznaczonymi strefami lotów. Każda strefa ma swoje właściwości prawne. . [22](#), [32](#)

## Props

Właściwości przekazywane do komponentu React przez komponent nadrzędny; służą do konfiguracji i przekazywania danych. Powinny być traktowane jako tylko do odczytu (read-only) wewnątrz komponentu potomnego.. [86](#)

## Protected route

Trasa w aplikacji, do której dostęp jest ograniczony, zwykle tylko dla zalogowanych użytkowników lub użytkowników z odpowiednimi uprawnieniami. Jeżeli użytkownik nie spełnia warunków, jest przekierowywany (np. na stronę główną). [85](#)

## React

Biblioteka JavaScript do budowy interfejsów użytkownika w oparciu o komponenty deklaratywne i wirtualny DOM. Zapewnia jednokierunkowy przepływ danych oraz zarządzanie stanem komponentów.. [86](#)

## Redux

Biblioteka do przewidywalnego zarządzania stanem aplikacji. Opiera się na jednym *store*, akcjach i czystych *reducerach*, promuje niemutowalność i jednokierunkowy przepływ danych. Często używana z Reactem, ale niezależna od niego.. [86](#), [87](#)

## Responsywność

Określenie związane z projektowaniem responsywnym (Responsive Web Design, RWD), czyli dostosowywaniem interfejsu do różnych rozmiarów i parametrów ekranów. Obejmuje m.in. elastyczne siatki, grafiki i [Media queries](#), tak aby układ i czytelność były zachowane na telefonach, tabletach i desktopach.. [94](#)

## REST API

Architektura budowania usług sieciowych komunikujących się poprzez metody protokołu HTTP (GET, PUT, POST, DELETE, PATCH). Wymiana danych występuje często w formacie JSON lub XML.

REST API musi spełniać następujące reguły:

1. **Rozdzielenie klient-serwer** — klient i serwer są od siebie niezależne, komunikują się poprzez interfejs.
2. **Bezstanowość** — każde żądanie przez klienta zawiera wszystkie informacje niezbędne do jego obsłużenia. Po otrzymaniu żądania serwer nie przechowuje o nim żadnych informacji.
3. **Buforowalność (cache)** — odpowiedzi z API powinny informować, czy dane można cache'ować. Jeśli tak, to przy kolejnym żądaniu mogą być zwrócone z cache'a.
4. **Jednolity interfejs:**
  - **Identyfikacja zasobów** — każdy zasób musi być jednoznacznie zidentyfikowany w interakcji klient-serwer.
  - **Manipulacja zasobów poprzez reprezentację** — po otrzymaniu reprezentacji klient może zmienić stan zasobu przesyłając zmodyfikowaną reprezentację.
  - **Samoopisujące się wiadomości** — każde żądanie i odpowiedź powinny zawierać informacje do jego poprawnego przetworzenia.

- **Hypermedia jako silnik stanu aplikacji (HATEOAS)** — po otrzymaniu odpowiedzi klient powinien móc dynamicznie poznać inne interakcje przez linki.
5. **Warstwowość** — klient nie wie czy komunikuje się bezpośrednio z serwerem, czy poprzez pośrednika (np. proxy) oraz nie wie z czym komunikuje się obsługująca go warstwa.
  6. **Kod na żądanie (opcjonalnie)** — serwer może przesłać fragment kodu, który zostanie wykonany przez klienta.

[15](#)

## Review kodu

Proces polegający na wzajemnym przeglądzie kodu źródłowego przez programistów w celu wykrycia błędów, poprawy jakości oraz zwiększenia spójności projektu. [15](#), [16](#), [105](#)

## Sidebar

Boczny panel w interfejsie użytkownika, zawierający menu nawigacyjne lub dodatkowe opcje funkcjonalne aplikacji. [78](#), [105–107](#)

## Spot

Spotkanie zespołu projektowego, zazwyczaj krótkie i regularne, służące omówieniu postępów prac, problemów oraz planów na najbliższy okres. [97–102](#)

## Stan

Aktualny zestaw danych przechowywanych przez aplikację lub komponent, na podstawie którego renderowany jest interfejs użytkownika. Stan może być lokalny (utrzymywany w pojedynczym komponencie) albo globalny (wspólny dla wielu komponentów).. [86](#)

## Tablica Kanban

Narzędzie do zarządzania przepływem pracy, które pomaga zespołom śledzić zadania oraz ich postępy. Składa się z kolumn reprezentujących stan etapu prac, na przykład „Do zrobienia” lub „W trakcie”.. [15](#)

## TypeScript

Rozszerzenie do języka JavaScript dodający statyczne typowanie, interfejsy i narzędzia do większych projektów. Kompiluje się do czystego JavaScript, ułatwiając wykrywanie błędów w czasie kompilacji i refaktoryzację.. [86](#)

## UI

Interfejs użytkownika (ang. *User Interface*); warstwa prezentacji odpowiedzialna za sposób wyświetlania danych oraz interakcji użytkownika z aplikacją.. [16](#), [86](#)

## UML

(ang. *Unified Modeling Language*); graficzny język wizualizacji, specyfikowania oraz dokumentowania składników systemów informatycznych. . [17](#)



# Spis tabel

2.1	Zespół projektowy . . . . .	7
2.2	Promotor . . . . .	8
2.3	Droniarze . . . . .	8
Tabela 3.1: Usługa zewnętrzna: GitHub Actions (CI) . . . . .		18
Tabela 3.2: Usługa zewnętrzna: Azure Blob Storage . . . . .		18
Tabela 3.3: Usługa zewnętrzna: Mailtrap . . . . .		18
Tabela 3.4: Usługa zewnętrzna: LocationIQ . . . . .		18
Tabela 3.5: Usługa zewnętrzna: Google Maps (Maps URLs) . . . . .		19
Tabela 3.6: Usługa zewnętrzna: OpenFreeMap . . . . .		19
Tabela 3.7: Usługa zewnętrzna: Open-Meteo . . . . .		19
Tabela 3.8: Usługa zewnętrzna: Tenor GIF API . . . . .		19
Tabela 3.9: Usługa zewnętrzna: Where the ISS at? . . . . .		20
Tabela 4.1: Scenariusz przypadku użycia: Rejestracja użytkownika . . . . .		24
Tabela 4.2: Scenariusz przypadku użycia: Logowanie użytkownika . . . . .		25
Tabela 4.3: Scenariusz przypadku użycia: Wykupienie subskrypcji premium . . . . .		26
Tabela 4.4: Scenariusz przypadku użycia: Resetowanie hasła . . . . .		27
Tabela 4.5: Scenariusz przypadku użycia: Zmiana hasła w ustawieniach konta . . . . .		28
Tabela 4.6: Scenariusz przypadku użycia: Wylogowanie użytkownika . . . . .		29
Tabela 4.7: Scenariusz przypadku użycia: Przeglądanie powiadomień . . . . .		29
Tabela 4.8: Scenariusz przypadku użycia: Przeszukiwanie historii czatu . . . . .		30
Tabela 4.9: Scenariusz przypadku użycia: Przeglądanie wysłanych plików na czacie . . . . .		31

Tabela 4.10: Scenariusz przypadku użycia: Zmiana typu mapy . . . . .	32
Tabela 4.11: Scenariusz przypadku użycia: Przeglądanie stref PANSA . .	33
Tabela 4.12: Scenariusz przypadku użycia: Wyszukiwanie spota w glo- balnej wyszukiwarce . . . . .	34
Tabela 4.13: Scenariusz przypadku użycia: Przejście do spota na mapie z wyszukiwarki . . . . .	35
Tabela 4.14: Scenariusz przypadku użycia: Przeglądanie mapy spotów . .	36
Tabela 4.15: Scenariusz przypadku użycia: Otwarcie szczegółów spota . .	36
Tabela 4.16: Scenariusz przypadku użycia: Przeglądanie komentarzy do spota . . . . .	37
Tabela 4.17: Scenariusz przypadku użycia: Przeglądanie pogody na spocie	38
Tabela 4.18: Scenariusz przypadku użycia: Wyszukiwanie spota na mapie	39
Tabela 4.19: Scenariusz przypadku użycia: Utworzenie prywatnego czatu	40
Tabela 4.20: Scenariusz przypadku użycia: Otworzenie czatu . . . . .	41
Tabela 4.21: Scenariusz przypadku użycia: Utworzenie czatu grupowego .	42
Tabela 4.22: Scenariusz przypadku użycia: Przeglądanie listy czatów . . .	42
Tabela 4.23: Scenariusz przypadku użycia: Wysyłanie wiadomości na czacie	43
Tabela 4.24: Scenariusz przypadku użycia: Wysyłanie GIF-a na czacie . .	44
Tabela 4.25: Scenariusz przypadku użycia: Wysyłanie pliku na czacie . .	45
Tabela 4.26: Scenariusz przypadku użycia: Edycja ustawień czatu . . . .	46
Tabela 4.27: Scenariusz przypadku użycia: Dodanie członka do czatu gru- powego . . . . .	46
Tabela 4.28: Scenariusz przypadku użycia: Przeglądanie postów na forum	47
Tabela 4.29: Scenariusz przypadku użycia: Dodanie posta na forum . . .	48
Tabela 4.30: Scenariusz przypadku użycia: Dodanie komentarza na forum	49
Tabela 4.31: Scenariusz przypadku użycia: Przeglądanie historii interakcji z postami . . . . .	50
Tabela 4.32: Scenariusz przypadku użycia: Zarządzanie komentarzami na forum . . . . .	51
Tabela 4.33: Scenariusz przypadku użycia: Zgłoszenie komentarza naru- szającego regulamin . . . . .	51

Tabela 4.34: Scenariusz przypadku użycia: Zgłoszenie posta na forum . .	52
Tabela 4.35: Scenariusz przypadku użycia: Przeglądanie komentarzy pod postem . . . . .	53
Tabela 4.36: Scenariusz przypadku użycia: Dodanie spota w panelu użyt- kownika . . . . .	54
Tabela 4.37: Scenariusz przypadku użycia: Przeglądanie profilu użytkow- nika . . . . .	55
Tabela 4.38: Scenariusz przypadku użycia: Przeglądanie profilu innego użytkownika . . . . .	56
Tabela 4.39: Scenariusz przypadku użycia: Dodanie użytkownika do zna- jomych . . . . .	57
Tabela 4.40: Scenariusz przypadku użycia: Przeglądanie społeczności . .	57
Tabela 4.41: Scenariusz przypadku użycia: Przeglądanie dodanych spotów	58
Tabela 4.42: Scenariusz przypadku użycia: Edycja danych użytkownika .	59
Tabela 4.43: Scenariusz przypadku użycia: Przeglądanie dodanych zdjęć do spotów . . . . .	60
Tabela 4.44: Scenariusz przypadku użycia: Przeglądanie dodanych filmów do spotów . . . . .	61
Tabela 4.45: Scenariusz przypadku użycia: Przeglądanie dodanych ko- mentarzy do spotów . . . . .	62
4.46 Profil użytkownika . . . . .	63
4.47 Lista dodanych spotów . . . . .	64
4.48 Dodanie spota . . . . .	65
4.49 Lista zdjęć . . . . .	66
4.50 Lista filmów . . . . .	66
4.51 Lista znajomych . . . . .	67
4.52 Lista obserwujących . . . . .	67
4.53 Lista obserwowanych . . . . .	68
4.54 Lista polubionych/odwiedzonych/planowanych spotów . . . . .	68
4.55 Lista komentarzy . . . . .	69
4.56 Ustawienia profilu . . . . .	70

4.57	Resetowanie hasła . . . . .	71
4.58	Dodawanie do znajomych . . . . .	72
4.59	Logowanie i rejestracja . . . . .	73
4.60	Strona główna — podstawowe filtry . . . . .	74
4.61	Strona główna — zaawansowane filtry . . . . .	75
4.62	Ustawienia motywu (ręczna zmiana) . . . . .	76
4.63	Zapamiętanie preferencji motywu . . . . .	77
4.64	Szybki przełącznik motywu w interfejsie . . . . .	78

# Bibliografia

- [1] *Disciplined Agile Delivery*. PMI. 1 stycznia 2025. URL: <https://www.pmi.org/disciplined-agile/process/introduction-to-dad/why> (dostęp 30.10.2025).
- [2] *Disciplined Agile Delivery — Lean Life Cycle*. PMI. 1 stycznia 2025. URL: <https://www.pmi.org/disciplined-agile/lifecycle/lean-lifecycle> (dostęp 30.10.2025).
- [3] Stanisław Wrycza, Bartosz Marcinkowski i Krzysztof Wyrzykowski. „Język UML 2.0 w modelowaniu systemów informatycznych”. Warszawa: Helion, 2006. ISBN: 83-736-1892-9, 8373618929.
- [4] Michał Wolski. *10 wskazówek poprawiających modelowanie procesów biznesowych w notacji BPMN*. 14 maja 2024. URL: <https://wolski.pro/2024/05/10-wskazowek-poprawiajacych-modelowanie-procesow-biznesowych-w-notacji-bpmn/> (dostęp 19.11.2025).
- [5] *About billing for GitHub Actions*. GitHub Docs. 1 stycznia 2024. URL: <https://docs.github.com/en/billing/managing-billing-for-github-actions/about-billing-for-github-actions> (dostęp 2.11.2025).
- [6] *Scalability and performance targets for Blob storage*. Microsoft Learn. 1 stycznia 2024. URL: <https://learn.microsoft.com/azure/storage/blobs/scalability-targets> (dostęp 2.11.2025).
- [7] *What are the limitations in Mailtrap?* Mailtrap Docs. 1 stycznia 2024. URL: <https://help.mailtrap.io/article/111-what-are-the-limitations-in-mailtrap/> (dostęp 2.11.2025).
- [8] *LocationIQ Pricing*. LocationIQ. 1 stycznia 2024. URL: <https://locationiq.com/pricing> (dostęp 2.11.2025).
- [9] *Google Maps (Maps URLs)*. Google Maps. 1 stycznia 2024. URL: <https://developers.google.com/maps/documentation/urls/get-started?hl=pl> (dostęp 2.11.2025).
- [10] *OpenFreeMap Documentation*. OpenFreeMap. 1 stycznia 2024. URL: <https://openfreemap.org/docs> (dostęp 2.11.2025).

- [11] *OpenFreeMap Quick Start*. OpenFreeMap. 1 stycznia 2024. URL: <https://openfreemap.org/docs/quick-start> (dostęp 2.11.2025).
- [12] *Open-Meteo API Usage & Pricing*. Open-Meteo. 1 stycznia 2024. URL: <https://open-meteo.com/en/docs/usage-and-pricing> (dostęp 2.11.2025).
- [13] *Tenor API — Documentation*. Tenor. 1 stycznia 2024. URL: <https://tenor.com/gifapi/documentation> (dostęp 2.11.2025).
- [14] *Where the ISS at? API*. wheretheiss.at. 1 stycznia 2024. URL: <https://wheretheiss.at/> (dostęp 2.11.2025).
- [15] *React useState*. 1 stycznia 2025. URL: <https://react.dev/reference/react/useState> (dostęp 3.11.2025).
- [16] *Redux*. 1 stycznia 2025. URL: <https://redux.js.org/> (dostęp 3.11.2025).
- [17] *Axios*. 1 stycznia 2025. URL: <https://axios-http.com/> (dostęp 3.11.2025).
- [18] *Tanstack Query*. 1 stycznia 2025. URL: <https://tanstack.com/query/latest> (dostęp 3.11.2025).
- [19] *Tailwind*. 1 stycznia 2025. URL: <https://tailwindcss.com/> (dostęp 3.11.2025).
- [20] *Motion*. 1 stycznia 2025. URL: <https://motion.dev/> (dostęp 3.11.2025).

# Załączniki

Płyta CD z następującą zawartością:

- *pliki projektowe* – pliki składające się na całość projektu
  - repozytorium kodu źródłowego wraz z instrukcją zbudowania i uruchomienia projektu
  - źródło pracy inżynierskiej.
- *Langmesser Adam\_Redosz Mateusz\_Oziemczuk Stanisław\_Badek Kacper\_praca pisemna* – katalog zawierający plik PDF z pracą inżynierską.