



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

| | |
|---|--|
| Temat projektu: Aplikacja webowa: spoty-na-drony.pl Temat projektu po angielsku: Web application: spoty-na-drony.pl | Akronim: Merkury Data ustalenia tematu 2023-10-10 |
| Promotor: mgr Adam Urbanowicz | Konsultanci: 1. — brak — |
| Cele projektu: Stworzenie w pełni funkcjonalnej aplikacji internetowej do rozwijania hobby (latania dronem). | |
| Rezultaty projektu: 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. | |
| Miary sukcesu: Gotowa do wdrożenia aplikacja. Realizacja w terminie zgodnym z wymaganiami. | |
| Ograniczenia: 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 |

| | |
|---|--|
| Data ukończenia projektu: 23 listopada 2025 | Recenzent: dr Elżbieta Puźniakowska-Gałuch |
|---|--|

Spis treści

| | | |
|----------|--|-----------|
| 1 | Wstęp | 6 |
| 1.1 | O projekcie | 6 |
| 1.2 | Cel i zakres prac | 6 |
| 1.3 | Geneza pomysłu | 6 |
| 2 | Opis problemu | 7 |
| 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 |
| 3 | Planowanie | 10 |
| 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 |
| 4 | Analiza wymagań | 21 |
| 4.1 | Przypadki użycia | 21 |
| 4.1.1 | Aktorzy | 21 |
| 4.1.2 | Diagram przypadków użycia | 22 |
| 4.1.3 | Scenariusze przypadków użycia | 22 |
| 4.2 | Wymagania ogólne i dziedzinowe | 57 |
| 4.3 | Wymagania funkcjonalne | 57 |
| 4.3.1 | Funkcjonalności dla mapy | 57 |
| 4.3.2 | Funkcjonalności dla chatu | 57 |
| 4.3.3 | Funkcjonalności dla forum | 57 |
| 4.3.4 | Funkcjonalności dla konta użytkownika | 57 |
| 4.3.5 | Funkcjonalności dla logowania i rejestracji | 67 |
| 4.3.6 | Funkcjonalności dla wyszukiwarki spotów | 68 |
| 4.3.7 | Funkcjonalności dla motywu | 70 |
| 4.4 | Wymagania pozafunkcjonalne | 72 |
| 4.5 | Wymagania interfejs z otoczeniem | 72 |
| 4.6 | Wymagania na środowisko docelowe | 72 |
| 5 | Projekt | 73 |
| 5.1 | Wzorce projektowe | 73 |
| 5.2 | Architektura systemu | 73 |
| 5.2.1 | Diagram architektury | 73 |
| 5.2.2 | Komponenty systemu | 73 |
| 5.3 | Projekt bazy danych | 73 |
| 5.3.1 | Model danych | 73 |

| | | |
|----------|--|-----------|
| 5.3.2 | Diagram ERD | 73 |
| 5.4 | Architektura interfejsu użytkownika | 73 |
| 5.4.1 | Projekt strony głównej | 73 |
| 5.4.2 | Projekt panelu logowania | 73 |
| 5.4.3 | Projekt mapy | 73 |
| 5.4.4 | Projekt chatu | 73 |
| 5.4.5 | Projekt forum | 73 |
| 5.4.6 | Projekt konta użytkownika | 73 |
| 6 | Przebieg realizacji projektu | 74 |
| 6.1 | Sprint 1 | 74 |
| 6.2 | Sprint 2 | 74 |
| 7 | Realizacja Projektu | 75 |
| 7.1 | Implementacja backendu | 75 |
| 7.1.1 | Struktura projektu | 75 |
| 7.1.2 | Endpointy systemu | 75 |
| 7.1.3 | Integracja z bazą danych | 78 |
| 7.1.4 | Obsługa uwierzytelnienia | 78 |
| 7.1.5 | Konteneryzacja | 78 |
| 7.2 | Implementacja frontendu | 78 |
| 7.2.1 | Struktura aplikacji | 78 |
| 7.2.2 | Zarządzanie stanem i przepływ danych | 83 |
| 7.2.3 | Integracja i komunikacja z backendem | 85 |
| 7.2.4 | Style | 88 |
| 7.2.5 | Strona główna | 92 |
| 7.2.6 | Mapa | 92 |
| 7.2.7 | Chat | 92 |
| 7.2.8 | Forum | 92 |
| 7.2.9 | Konto użytkownika | 92 |
| 7.2.10 | Panel logowania | 92 |
| 7.3 | Implementacja CI/CD | 92 |

| | |
|---|------------|
| 8 Testy | 93 |
| 8.1 Testy jednostkowe | 93 |
| 8.2 Testy integracyjne | 93 |
| 8.3 Testy E2E | 93 |
| 8.4 Wyniki testów i wnioski | 93 |
| 9 Prezentacja systemu | 94 |
| 9.1 Strona główna | 94 |
| 9.2 Strona mapy | 94 |
| 9.3 Strona chatu | 94 |
| 9.4 Strona forum | 94 |
| 9.5 Panel logowania | 94 |
| 9.6 Panel konta użytkownika | 94 |
| 10 Nakład pracy | 95 |
| 10.1 Ogólny nakład pracy | 95 |
| 10.2 Indywidualne nakłady pracy | 95 |
| 10.2.1 Adam Langmesser | 95 |
| 10.2.2 Mateusz Redosz | 95 |
| 10.2.3 Stanisław Oziemczuk | 98 |
| 10.2.4 Kacper Badek | 98 |
| 11 Podsumowanie | 99 |
| 11.1 Osiągnięte rezultaty | 99 |
| 11.2 Napotkane wyzwania | 99 |
| 11.3 Plany na przyszłość | 99 |
| 12 Słownik pojęć i skrótów | 100 |
| Spis tabel | 101 |
| Załączniki | 104 |

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 | |
|------------------|---|
| Identyfikator: | UO1 |
| Nazwa: | Zespół projektowy |
| Opis: | Zespół czterech studentów odpowiedzialnych za analizę, projekt, implementację, testy oraz dokumentację systemu. |
| Typ udziałowca: | ożywiony, bezpośredni |
| Punkt widzenia: | techniczna, wykonawcza |
| Ograniczenia: | Ograniczone zasoby czasowe i doświadczenie komercyjne. |
| Wymagania: | Wymagania funkcjonalne i techniczne systemu, możliwość realizacji w ramach projektu dyplomowego. |

Tabela 2.1: Zespół projektowy

| KARTA UDZIAŁOWCA | |
|-------------------------|---|
| Identyfikator: | UO2 |
| Nazwa: | Promotor |
| Opis: | Osoba nadzorująca przebieg projektu, weryfikująca poprawność merytoryczną i zgodność z wymaganiami uczelni. |
| Typ udziałowca: | ożywiony, pośredni |
| Punkt widzenia: | merytoryczna, formalna, jakościowa |
| Ograniczenia: | Nie odpowiada za implementację; rekomenduje, opiniuje i zatwierdza. |
| Wymagania: | Czytelna dokumentacja, zgodność z wytycznymi kierunku i dobry poziom techniczny rozwiązania. |

Tabela 2.2: Promotor

| KARTA UDZIAŁOWCA | |
|-------------------------|---|
| Identyfikator: | UO3 |
| Nazwa: | Droniarze |
| Opis: | Główna grupa docelowa systemu – osoby latające dronami rekreacyjnie lub półprofesjonalnie, szukające miejsc do lotów i wymiany doświadczeń. |
| Typ udziałowca: | ożywiony, bezpośredni |
| Punkt widzenia: | użytkownik końcowy: prostota obsługi, rzetelne informacje o spotach, wygodne dzielenie się treściami. |
| Ograniczenia: | Brak wpływu na architekturę techniczną systemu; oczekują intuicyjnego interfejsu. |
| Wymagania: | Lista spotów, informacje o ograniczeniach prawnych, oceny/komentarze, dodawanie treści, podstawowe funkcje społecznościowe. |

Tabela 2.3: 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 [DAD]** w wariantcie **Lean Life Cycle [DAD-LLF]**, 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([uml-def]) czy BPMN([bpmn-def]). 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

| | |
|--------|--|
| Usługa | GitHub Actions (CI) [github-actions-billing] |
|--------|--|

| | |
|--------------|---|
| Opis | Uruchomienia pipeline'ów CI/CD dla repozytorium GitHub. |
| Limit | 3000 min/mies. |

Tabela 3.1: Usługa zewnętrzna: GitHub Actions (CI)

| | |
|---------------|--|
| Usługa | Azure Blob Storage [azure-blob-scalability] |
| Opis | Magazyn plików (m.in. zdjęcia spotów, załączniki z czatu). |
| Limit | 1 GB/mies. |

Tabela 3.2: Usługa zewnętrzna: Azure Blob Storage

| | |
|---------------|--|
| Usługa | Mailtrap [mailtrap-limits] |
| Opis | Środowisko testowe SMTP oraz Email API do wysyłki maili. |
| Limit | 150 maili/dzień |

Tabela 3.3: Usługa zewnętrzna: Mailtrap

| | |
|---------------|---|
| Usługa | LocationIQ [locationiq-pricing] |
| Opis | Geokodowanie adresu przy dodawaniu nowych spotów. |
| Limit | 5 000 zapytań/dzień |

Tabela 3.4: Usługa zewnętrzna: LocationIQ

| | |
|---------------|--|
| Usługa | Google Maps (Maps URLs) [google-maps-urls] |
| Opis | Otwieranie nawigacji w aplikacji Map Google (deep link/URL). |
| Limit | Brak limitu w ramach dokumentowanego sposobu użycia. |

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

| | |
|---------------|---|
| Usługa | OpenFreeMap [openfreemap-docs , openfreemap-quickstart] |
| Opis | Publiczny serwer kafelków do renderu mapy na froncie. |
| Limit | 30 000 zapytań/mies. |

Tabela 3.6: Usługa zewnętrzna: OpenFreeMap

| | |
|---------------|---|
| Usługa | Open-Meteo [open-meteo-usage] |
| Opis | Prognozy pogody wyświetlane dla spotów. |
| Limit | 10 000 zapytań/dzień |

Tabela 3.7: Usługa zewnętrzna: Open-Meteo

| | |
|---------------|---|
| Usługa | Tenor GIF API [tenor-docs] |
| Opis | Wyszukiwanie GIF-ów w czacie. |
| Limit | 1 zapytanie na sekundę; brak ogólnego limitu dziennego. |

Tabela 3.8: Usługa zewnętrzna: Tenor GIF API

| | |
|---------------|--|
| Usługa | Where the ISS at? [wheretheiss-docs] |
| Opis | HTTP API z bieżącą pozycją satelity, używane pomocniczo. |
| Limit | 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

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

Niniejszy rozdział zawiera scenariusze przypadków użycia. Zostały one wykonane dla wybranych przypadków użycia.

Każdy ze scenariuszy posiada jeden z trzech możliwych priorytetów implementacji: wysoki, średni albo niski.

Scenariusze przypadków użycia – funkcje ogólne

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|------------------------------------|---|
| Identyfikator: | PU1 |
| Nazwa: | Rejestracja użytkownika |
| Priorytet: | Wysoki |
| Aktorzy: | Użytkownik niezalogowany |
| Opis: | Użytkownik zakłada konto poprzez formularz rejestracji. |
| Warunki wstępne: | Użytkownik znajduje się na stronie z formularzem rejestracji. |

| | |
|--|---|
| Warunki końcowe: | Użytkownik posiada konto w systemie. |
| Główny przepływ zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik wypełnia formularz rejestracyjny. 2. Użytkownik naciska przycisk rejestracji. 3. System tworzy konto użytkownika. 4. System loguje użytkownika i przenosi go na stronę główną aplikacji. |
| Alternatywne przepływy zdarzeń: | <ol style="list-style-type: none"> 1a. Podane dane są niepoprawne – system wyświetla komunikat o błędzie oraz podświetla pola wymagające poprawy. 2a. Nazwa użytkownika jest już zajęta – system wyświetla komunikat o błędzie. |

Tabela 4.1: Scenariusz przypadku użycia: Rejestracja użytkownika

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|---|---|
| Identyfikator: | PU2 |
| Nazwa: | Logowanie użytkownika |
| Priorytet: | Wysoki |
| Aktorzy: | Użytkownik niezalogowany |
| Opis: | Użytkownik loguje się do systemu, podając login i hasło. |
| Warunki wstępne: | Użytkownik znajduje się na stronie logowania. |
| Warunki końcowe: | Użytkownik jest zalogowany i przeniesiony na stronę główną aplikacji. |

| | |
|--|---|
| Główny przepływ zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik wypełnia formularz logowania. 2. Użytkownik naciska przycisk logowania. 3. System loguje użytkownika i przenosi go na ostatnią odwiedzoną przez niego stronę aplikacji. |
| Alternatywne przepływy zdarzeń: | <ol style="list-style-type: none"> 2a. Podane dane są niepoprawne – system wyświetla komunikat o błędzie. |

Tabela 4.2: Scenariusz przypadku użycia: Logowanie użytkownika

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|---|--|
| Identyfikator: | PU3 |
| Nazwa: | Resetowanie hasła |
| Priorytet: | Wysoki |
| Aktorzy: | Użytkownik niezalogowany, Usługa SMTP |
| Opis: | Użytkownik inicjuje reset hasła, aby odzyskać dostęp do konta. |
| Warunki wstępne: | Użytkownik znajduje się na ekranie resetu hasła. |
| Warunki końcowe: | Użytkownik otrzymuje wiadomość e-mail z linkiem do ustawienia nowego hasła. |
| Główny przepływ zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik wpisuje adres e-mail powiązany z kontem. 2. Użytkownik zatwierdza żądanie resetu hasła. 3. System generuje token resetu hasła. 4. System wysyła e-mail z linkiem do zmiany hasła. |

| | |
|--|--|
| Alternatywne przebiegi zdarzeń: | <p>2a. Nie istnieje konto dla podanego adresu – system wyświetla komunikat o błędzie.</p> <p>4a. Występuje błąd połączenia z usługą SMTP – system informuje użytkownika o problemie technicznym.</p> |
|--|--|

Tabela 4.3: Scenariusz przypadku użycia: Resetowanie hasła

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|---|--|
| Identyfikator: | PU4 |
| Nazwa: | Zmiana hasła w ustawieniach konta |
| Priorytet: | Wysoki |
| Aktorzy: | Użytkownik zalogowany |
| Opis: | Użytkownik zmienia hasło do konta z poziomu ustawień profilu. |
| Warunki wstępne: | Użytkownik jest zalogowany i znajduje się na ekranie zmiany danych konta. |
| Warunki końcowe: | Hasło do konta użytkownika zostało zaktualizowane. |
| Główny przebieg zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik wpisuje aktualne hasło. 2. Użytkownik wpisuje nowe hasło i powtarza je. 3. Użytkownik zatwierdza formularz zmiany hasła. 4. System zapisuje nowe hasło i informuje o powodzeniu operacji. |

| | |
|--|---|
| Alternatywne przebiegi zdarzeń: | <p>3a. Aktualne hasło jest nieprawidłowe – system wyświetla komunikat i nie zapisuje zmian.</p> <p>3b. Nowe hasło nie spełnia wymagań bezpieczeństwa – system informuje o błędzie i podświetla pola do poprawy.</p> |
|--|---|

Tabela 4.4: Scenariusz przypadku użycia: Zmiana hasła w ustawieniach konta

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|---|--|
| Identyfikator: | PU5 |
| Nazwa: | Wylogowanie użytkownika |
| Priorytet: | Wysoki |
| Aktorzy: | Użytkownik zalogowany |
| Opis: | Użytkownik wylogowuje się z aplikacji. |
| Warunki wstępne: | Użytkownik jest zalogowany. |
| Warunki końcowe: | Sesja użytkownika została zakończona, użytkownik widzi stronę główną dla niezalogowanych. |
| Główny przebieg zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik wybiera opcję wylogowania z menu. 2. System unieważnia token dostępu użytkownika. 3. System przenosi użytkownika na stronę główną aplikacji. |
| Alternatywne przebiegi zdarzeń: | Brak istotnych alternatywnych przebiegów. |

Tabela 4.5: Scenariusz przypadku użycia: Wylogowanie użytkownika

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|--|--|
| Identyfikator: | PU6 |
| Nazwa: | Przeglądanie powiadomień |
| Priorytet: | Niski |
| Aktorzy: | Użytkownik zalogowany |
| Opis: | Użytkownik przegląda listę powiadomień. |
| Warunki wstępne: | Użytkownik jest na ekranie centra powiadomień. |
| Warunki końcowe: | Powiadomienia zostały wyświetlone, a wybrane oznaczone jako przeczytane. |
| Główny przepływ zdarzeń: | <ol style="list-style-type: none"> 1. System wyświetla powiadomienia w odwróconym porządku chronologicznym. 2. Użytkownik otwiera wybrane powiadomienie. 3. System oznacza powiadomienie jako przeczytane i ewentualnie przenosi użytkownika do powiązanego widoku. |
| Alternatywne przepływy zdarzeń: | <ol style="list-style-type: none"> 1a. System nie może pobrać powiadomień (błąd serwera) – użytkownik otrzymuje komunikat o błędzie i może spróbować ponownie. |

Tabela 4.6: Scenariusz przypadku użycia: Przeglądanie powiadomień

Scenariusze przypadków użycia dla funkcji premium

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|------------------------------------|--------------------------------|
| Identyfikator: | PU7 |
| Nazwa: | Wykupienie subskrypcji premium |
| Priorytet: | Niski |

| | |
|--|--|
| Aktorzy: | Użytkownik zalogowany, Bramka płatnicza, System finansowo-księgowy |
| Opis: | Użytkownik opłaca subskrypcję premium w celu uzyskania dodatkowych funkcji. |
| Warunki wstępne: | Użytkownik jest zalogowany i znajduje się w module subskrypcji. |
| Warunki końcowe: | Subskrypcja premium jest aktywna, a użytkownik ma dostęp do funkcji premium. |
| Główny przepływ zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik wybiera plan subskrypcji. 2. Użytkownik przechodzi do bramki płatniczej. 3. Użytkownik podaje dane płatnicze i zatwierdza transakcję. 4. Bramka płatnicza przetwarza płatność i zwraca wynik do systemu. 5. System zapisuje informację o opłaconej subskrypcji i aktualizuje uprawnienia. 6. System generuje wpis w systemie finansowo-księgowym. |
| Alternatywne przepływy zdarzeń: | <ol style="list-style-type: none"> 4a. Płatność nie powiodła się – system informuje użytkownika i umożliwia ponowną próbę. 5a. W czasie aktualizacji subskrypcji wystąpił błąd – system cofa zmiany i wyświetla komunikat o problemie. |

Tabela 4.7: Scenariusz przypadku użycia: Wykupienie subskrypcji premium

Scenariusze przypadków użycia dla wyszukiwarki

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|------------------------------------|-----|
| Identyfikator: | PU8 |

| | |
|--|---|
| Nazwa: | Wyszukiwanie spota na mapie |
| Priorytet: | Wysoki |
| Aktorzy: | Użytkownik niezalogowany |
| Opis: | Użytkownik wyszukuje spota po nazwie korzystając z pola wyszukiwania na mapie. |
| Warunki wstępne: | Użytkownik widzi mapę spotów. |
| Warunki końcowe: | Mapa zostaje ustawiona na wybranego spota lub listę dopasowań. |
| Główny przepływ zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik wpisuje frazę w polu wyszukiwania na mapie. 2. System podpowiada listę pasujących spotów. 3. Użytkownik wybiera spota z listy. 4. System przybliża mapę do wybranego spota i podświetla jego znacznik. |
| Alternatywne przepływy zdarzeń: | 2a. Brak wyników dla podanej frazy – system informuje użytkownika o braku dopasowań. |

Tabela 4.8: Scenariusz przypadku użycia: Wyszukiwanie spota na mapie

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|------------------------------------|---|
| Identyfikator: | PU9 |
| Nazwa: | Wyszukiwanie spota w globalnej wyszukiwarce |
| Priorytet: | Wysoki |
| Aktorzy: | Użytkownik niezalogowany, Usługa do wyświetlania mapy, Usługa do pogody |

| | |
|--|--|
| Opis: | Użytkownik wyszukuje spoty za pomocą globalnej wyszukiwarki w aplikacji. |
| Warunki wstępne: | Użytkownik znajduje się na stronie głównej z wyszukiwarką. |
| Warunki końcowe: | Użytkownik otrzymuje listę znalezionych spotów. |
| Główny przepływ zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik wpisuje frazę w globalnej wyszukiwarce. 2. System wyszukuje spoty spełniające kryteria. 3. System wyświetla listę wyników. |
| Alternatywne przepływy zdarzeń: | 3a. Brak wyników – system wyświetla komunikat i proponuje zmianę kryteriów wyszukiwania. |

Tabela 4.9: Scenariusz przypadku użycia: Wyszukiwanie spotu w globalnej wyszukiwarce

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|---|--|
| Identyfikator: | PU10 |
| Nazwa: | Przejdźcie do spotu na mapie z wyszukiwarki |
| Priorytet: | Wysoki |
| Aktorzy: | Użytkownik niezalogowany |
| Opis: | Użytkownik przechodzi z wyników wyszukiwarki do widoku mapy ustawionego na konkretny spot. |
| Warunki wstępne: | Wyświetlona jest lista wyników wyszukiwania spotów. |
| Warunki końcowe: | Mapa jest przybliżona do wybranego spotu, a jego szczegóły są dostępne. |

| | |
|--|---|
| Główny przepływ zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik wybiera spota z listy wyników. 2. System przełącza widok na moduł mapy. 3. System ustawia mapę na lokalizację spota i otwiera jego szczegóły. |
| Alternatywne przepływy zdarzeń: | Brak istotnych alternatywnych przepływów. |

Tabela 4.10: Scenariusz przypadku użycia: Przejście do spota na mapie z wyszukiwarki

Scenariusze przypadków użycia dla mapy

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|---|---|
| Identyfikator: | PU11 |
| Nazwa: | Przeglądanie mapy spotów |
| Priorytet: | Wysoki |
| Aktorzy: | Użytkownik niezalogowany, Usługa do wyświetlania mapy |
| Opis: | Użytkownik przegląda mapę spotów. |
| Warunki wstępne: | Użytkownik znajduje się w module mapy. |
| Warunki końcowe: | Mapa ze spotami została wyświetlona, a użytkownik może przybliżać, oddalać i przesuwać widok. |

| | |
|--|--|
| Główny przepływ zdarzeń: | <ol style="list-style-type: none"> 1. System inicjuje widok mapy z domyślnym obszarem. 2. System pobiera listę spotów w aktualnym zakresie mapy. 3. System rysuje znaczniki spotów na mapie. 4. Użytkownik przesuwa lub skaluje mapę. 5. System pobiera spoty dla nowego zakresu. |
| Alternatywne przepływy zdarzeń: | <ol style="list-style-type: none"> 2a. Usługa mapy jest niedostępna – system wyświetla komunikat o błędzie. |

Tabela 4.11: Scenariusz przypadku użycia: Przeglądanie mapy spotów

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|---|---|
| Identyfikator: | PU12 |
| Nazwa: | Przeglądanie komentarzy do spota |
| Priorytet: | Średni |
| Aktorzy: | Użytkownik niezalogowany |
| Opis: | Użytkownik czyta komentarze pod wybranym spotem. |
| Warunki wstępne: | Wyświetlany jest widok szczegółów spota. |
| Warunki końcowe: | Lista komentarzy do spota została wyświetlona. |
| Główny przepływ zdarzeń: | <ol style="list-style-type: none"> 1. System pobiera komentarze powiązane ze spotem. 2. System wyświetla komentarze w kolejności chronologicznej lub według popularności. 3. Użytkownik przewija listę komentarzy. |

| | |
|--|---|
| Alternatywne przebiegi zdarzeń: | 1a. Spot nie ma jeszcze komentarzy – system wyświetla odpowiednią informację. |
|--|---|

Tabela 4.12: Scenariusz przypadku użycia: Przeglądanie komentarzy do spota

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|---|--|
| Identyfikator: | PU13 |
| Nazwa: | Przeglądanie pogody na spocie |
| Priorytet: | Średni |
| Aktorzy: | Użytkownik zalogowany, Usługa danych pogodowych |
| Opis: | Użytkownik sprawdza prognozę pogody dla lokalizacji spota. |
| Warunki wstępne: | Wyświetlany jest widok szczegółów spota. |
| Warunki końcowe: | Prognoza pogody dla spota została wyświetlona. |
| Główny przebieg zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik otwiera zakładkę pogody. 2. System wysyła zapytanie do usługi pogodowej z lokalizacją spota. 3. System odbiera prognozę i prezentuje ją (temperatura, prędkość wiatru, opady). |
| Alternatywne przebiegi zdarzeń: | 2a. Usługa pogodowa jest niedostępna – system wyświetla komunikat o braku danych pogodowych. |

Tabela 4.13: Scenariusz przypadku użycia: Przeglądanie pogody na spocie

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|--|--|
| Identyfikator: | PU14 |
| Nazwa: | Zmiana typu mapy |
| Priorytet: | Niski |
| Aktorzy: | Użytkownik premium, Usługa do wyświetlania mapy |
| Opis: | Użytkownik zmienia typ mapy (np. standardowa, satelitarna, hybrydowa). |
| Warunki wstępne: | Użytkownik premium jest na ekranie mapy. |
| Warunki końcowe: | Mapa jest wyświetlana w wybranym typie. |
| Główny przepływ zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik otwiera ustawienia widoku mapy. 2. Użytkownik wybiera typ mapy z dostępnej listy. 3. System przełącza widok mapy na wybrany typ. |
| Alternatywne przepływy zdarzeń: | 3a. Wybrany typ mapy nie jest dostępny (błąd usługi mapowej) – system przywraca poprzedni typ i informuje o błędzie. |

Tabela 4.14: Scenariusz przypadku użycia: Zmiana typu mapy

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|------------------------------------|---|
| Identyfikator: | PU15 |
| Nazwa: | Przeglądanie stref PANSA |
| Priorytet: | Niski |
| Aktorzy: | Użytkownik premium, Usługa do wyświetlania mapy |

| | |
|--|--|
| Opis: | Użytkownik wyświetla na mapie strefy przestrzeni powietrznej PANSA. |
| Warunki wstępne: | Użytkownik premium ma otwarty moduł mapy. |
| Warunki końcowe: | Strefy PANSA zostały zwizualizowane na mapie. |
| Główny przepływ zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik włącza warstwę „Strefy PANSA”. 2. System pobiera dane o strefach. 3. System nakłada kontury stref na mapę. |
| Alternatywne przepływy zdarzeń: | 2a. Dane o strefach są chwilowo niedostępne – system komunikuje problem i nie włącza warstwy. |

Tabela 4.15: Scenariusz przypadku użycia: Przeglądanie stref PANSA

Scenariusze przypadków użycia dla czatu

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|------------------------------------|--|
| Identyfikator: | PU16 |
| Nazwa: | Utworzenie prywatnego czatu |
| Priorytet: | Wysoki |
| Aktorzy: | Użytkownik zalogowany |
| Opis: | Użytkownik tworzy prywatną konwersację z innym użytkownikiem. |
| Warunki wstępne: | Użytkownik jest zalogowany i znajduje się w zakładce społeczności. |

| | |
|--|---|
| Warunki końcowe: | Nowy czat prywatny został utworzony i wyświetlony użytkownikowi. |
| Główny przepływ zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik wybiera opcję utworzenia nowego czatu. 2. System tworzy nowy czat (jeśli nie istnieje). 3. System otwiera widok nowego czatu. |
| Alternatywne przepływy zdarzeń: | 1a. Taki czat już istnieje – system zamiast tworzyć nowy, otwiera istniejącą konwersację. |

Tabela 4.16: Scenariusz przypadku użycia: Utworzenie prywatnego czatu

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|---|---|
| Identyfikator: | PU17 |
| Nazwa: | Utworzenie czatu grupowego |
| Priorytet: | Średni |
| Aktorzy: | Użytkownik zalogowany |
| Opis: | Użytkownik tworzy nowy czat grupowy z kilkoma uczestnikami. |
| Warunki wstępne: | Użytkownik jest zalogowany i znajduje się na dowolnym czacie prywatnym. |
| Warunki końcowe: | Czat grupowy został utworzony i otwarty. |

| | |
|--|--|
| Główny przebieg zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik wybiera opcję utworzenia czatu grupowego. 2. Użytkownik wybiera uczestników grupy. 3. Użytkownik zatwierdza utworzenie czatu. 4. System tworzy czat grupowy i dodaje do niego wskazanych użytkowników. 5. System otwiera widok nowego czatu grupowego. |
| Alternatywne przebiegi zdarzeń: | <ol style="list-style-type: none"> 3a. System nie może utworzyć czatu – aplikacja informuje o błędzie. |

Tabela 4.17: Scenariusz przypadku użycia: Utworzenie czatu grupowego

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|---|--|
| Identyfikator: | PU18 |
| Nazwa: | Przeglądanie listy czatów |
| Priorytet: | Wysoki |
| Aktorzy: | Użytkownik zalogowany |
| Opis: | Użytkownik przegląda listę swoich czatów prywatnych i grupowych. |
| Warunki wstępne: | Użytkownik jest zalogowany i otwiera moduł czatu. |
| Warunki końcowe: | Lista czatów użytkownika została wyświetlona. |

| | |
|--|---|
| Główny przepływ zdarzeń: | <ol style="list-style-type: none"> 1. System pobiera listę czatów użytkownika. 2. System wyświetla listę czatów z podstawowymi informacjami. 3. Użytkownik wybiera czat z listy. 4. System otwiera widok wybranego czatu. |
| Alternatywne przepływy zdarzeń: | Brak istotnych alternatywnych przepływów. |

Tabela 4.18: Scenariusz przypadku użycia: Przeglądanie listy czatów

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|---|--|
| Identyfikator: | PU19 |
| Nazwa: | Wysyłanie wiadomości na czacie |
| Priorytet: | Wysoki |
| Aktorzy: | Użytkownik zalogowany |
| Opis: | Użytkownik wysyła wiadomość tekstową na czacie. |
| Warunki wstępne: | Użytkownik jest zalogowany i znajduje się w widoku konkretnego czatu. |
| Warunki końcowe: | Nowa wiadomość jest zapisana i widoczna w historii czatu. |
| Główny przepływ zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik wpisuje treść wiadomości. 2. Użytkownik wysyła wiadomość. 3. System zapisuje wiadomość i dostarcza ją do uczestników czatu. 4. System wyświetla wiadomość na liście wiadomości. |

| | |
|--|---|
| Alternatywne przebiegi zdarzeń: | 2a. Treść wiadomości jest pusta – system blokuje wysłanie i pozostaje w tym samym widoku. |
|--|---|

Tabela 4.19: Scenariusz przypadku użycia: Wysyłanie wiadomości na czacie

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|---|---|
| Identyfikator: | PU20 |
| Nazwa: | Wysyłanie GIF-a na czacie |
| Priorytet: | Średni |
| Aktorzy: | Użytkownik zalogowany, Usługa GIF-ów |
| Opis: | Użytkownik wysyła animację GIF w konwersacji czatowej. |
| Warunki wstępne: | Użytkownik jest zalogowany i znajduje się w widoku czatu. |
| Warunki końcowe: | Wybrany GIF został dodany jako wiadomość w czacie. |
| Główny przebieg zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik wybiera opcję dodania GIF-a. 2. System otwiera okno wyszukiwarki GIF-ów. 3. Użytkownik wybiera lub wyszukuje GIF-a. 4. Użytkownik zatwierdza wysłanie GIF-a. 5. System dodaje GIF-a jako wiadomość na czacie. |
| Alternatywne przebiegi zdarzeń: | 2a. Usługa GIF-ów jest niedostępna – system informuje o braku możliwości wysłania GIF-a. |

Tabela 4.20: Scenariusz przypadku użycia: Wysyłanie GIF-a na czacie

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|--|---|
| Identyfikator: | PU21 |
| Nazwa: | Wysyłanie pliku na czacie |
| Priorytet: | Średni |
| Aktorzy: | Użytkownik zalogowany, Usługa do przechowywania plików w chmurze |
| Opis: | Użytkownik wysyła plik (np. zdjęcie, film) w czacie. |
| Warunki wstępne: | Użytkownik jest zalogowany i znajduje się w widoku czatu. |
| Warunki końcowe: | Plik został zapisany w chmurze i powiązany z wiadomością na czacie. |
| Główny przepływ zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik wybiera opcję dodania pliku. 2. Użytkownik wybiera plik z urządzenia. 3. System przesyła plik do usługi przechowywania w chmurze. 4. System tworzy wiadomość z odnośnikiem do pliku. 5. System wyświetla wiadomość na liście czatu. |
| Alternatywne przepływy zdarzeń: | <ol style="list-style-type: none"> 3a. Przesyłanie pliku nie powiodło się – system informuje użytkownika i umożliwia ponowną próbę. |

Tabela 4.21: Scenariusz przypadku użycia: Wysyłanie pliku na czacie

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|------------------------------------|-----------------------|
| Identyfikator: | PU22 |
| Nazwa: | Edycja ustawień czatu |
| Priorytet: | Niski |

| | |
|--|---|
| Aktorzy: | Użytkownik zalogowany |
| Opis: | Użytkownik modyfikuje ustawienia czatu (np. nazwę, avatar, tryb powiadomień). |
| Warunki wstępne: | Użytkownik jest zalogowany i ma uprawnienia do edycji danego czatu. |
| Warunki końcowe: | Zaktualizowane ustawienia czatu są zapisane i widoczne dla uczestników. |
| Główny przepływ zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik otwiera panel ustawień czatu. 2. Użytkownik wprowadza zmiany (np. nazwę, opis, avatar). 3. Użytkownik zapisuje zmiany. 4. System waliduje dane i aktualizuje konfigurację czatu. |
| Alternatywne przepływy zdarzeń: | Brak istotnych alternatywnych przepływów poza walidacją pól. |

Tabela 4.22: Scenariusz przypadku użycia: Edycja ustawień czatu

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|------------------------------------|--|
| Identyfikator: | PU23 |
| Nazwa: | Dodanie członka do czatu grupowego |
| Priorytet: | Średni |
| Aktorzy: | Użytkownik zalogowany |
| Opis: | Użytkownik z uprawnieniami administratora dodaje nowego uczestnika do czatu grupowego. |
| Warunki wstępne: | Użytkownik jest zalogowany, znajduje się w czacie grupowym i ma prawo zarządzać członkami. |

| | |
|--|---|
| Warunki końcowe: | Nowy uczestnik został dodany do czatu grupowego. |
| Główny przepływ zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik otwiera listę uczestników czatu grupowego. 2. Użytkownik wybiera opcję dodania nowego członka. 3. Użytkownik wskazuje użytkownika do dodania i zatwierdza wybór. 4. System dodaje wskazanego użytkownika do czatu grupowego. |
| Alternatywne przepływy zdarzeń: | 3a. Operacja nie powiodła się – system informuje o błędzie. |

Tabela 4.23: Scenariusz przypadku użycia: Dodanie członka do czatu grupowego

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|---|---|
| Identyfikator: | PU24 |
| Nazwa: | Przeszukiwanie historii czatu |
| Priorytet: | Niski |
| Aktorzy: | Użytkownik premium |
| Opis: | Użytkownik wyszukuje konkretne wiadomości w historii czatu. |
| Warunki wstępne: | Użytkownik jest zalogowany jako użytkownik premium i znajduje się w widoku czatu. |
| Warunki końcowe: | Wiadomości spełniające kryteria wyszukiwania zostały wyświetlone. |

| | |
|--|--|
| Główny przepływ zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik otwiera pole wyszukiwania historii w czacie. 2. Użytkownik wpisuje frazę lub filtr (np. zakres dat, autor). 3. System filtruje wiadomości zgodnie z kryteriami. 4. System prezentuje listę dopasowanych fragmentów rozmowy. |
| Alternatywne przepływy zdarzeń: | Brak istotnych alternatywnych przepływów. |

Tabela 4.24: Scenariusz przypadku użycia: Przeszukiwanie historii czatu

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|---|---|
| Identyfikator: | PU25 |
| Nazwa: | Przeglądanie wysłanych plików na czacie |
| Priorytet: | Niski |
| Aktorzy: | Użytkownik premium, Usługa do przechowywania plików w chmurze |
| Opis: | Użytkownik przegląda listę plików wysłanych w ramach czatów. |
| Warunki wstępne: | Użytkownik jest zalogowany jako użytkownik premium. |
| Warunki końcowe: | Użytkownik widzi listę wysłanych plików i może przechodzić do powiązanych czatów. |

| | |
|--|--|
| Główny przepływ zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik otwiera sekcję „Wysłane pliki”. 2. System pobiera metadane plików z usługi przechowywania. 3. System wyświetla listę plików z podstawowymi informacjami (nazwa, typ, data). 4. Użytkownik wybiera plik, aby otworzyć go lub przejść do powiązanego czatu. |
| Alternatywne przepływy zdarzeń: | Brak istotnych alternatywnych przepływów. |

Tabela 4.25: Scenariusz przypadku użycia: Przeglądanie wysłanych plików na czacie

Scenariusze przypadków użycia dla forum

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|------------------------------------|---|
| Identyfikator: | PU26 |
| Nazwa: | Przeglądanie postów na forum |
| Priorytet: | Wysoki |
| Aktorzy: | Użytkownik niezalogowany |
| Opis: | Użytkownik przegląda listę postów na forum. |
| Warunki wstępne: | Użytkownik znajduje się w module forum. |
| Warunki końcowe: | Lista postów forum jest wyświetlona, a użytkownik może przechodzić do szczegółów. |

| | |
|--|---|
| Główny przebieg zdarzeń: | <ol style="list-style-type: none"> 1. System pobiera listę postów. 2. System wyświetla posty z podstawowymi informacjami. 3. Użytkownik wybiera post, który chce przeczytać. 4. System otwiera szczegółowy widok posta. |
| Alternatywne przebiegi zdarzeń: | <ol style="list-style-type: none"> 3a. System nie może pobrać szczegółów posta – system wyświetla komunikat o błędzie. |

Tabela 4.26: Scenariusz przypadku użycia: Przeglądanie postów na forum

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|---|--|
| Identyfikator: | PU27 |
| Nazwa: | Dodanie posta na forum |
| Priorytet: | Wysoki |
| Aktorzy: | Użytkownik zalogowany, Usługa do przechowywania plików w chmurze |
| Opis: | Użytkownik publikuje nowy post na forum. |
| Warunki wstępne: | Użytkownik znajduje się w module forum. |
| Warunki końcowe: | Nowy post jest widoczny na forum. |

| | |
|--|---|
| Główny przepływ zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik wybiera opcję dodania nowego posta. 2. Użytkownik wpisuje tytuł i treść posta. 3. (Opcjonalnie) Użytkownik dodaje załączniki (zdjęcia/filmy) do posta. 4. Użytkownik publikuje posta. 5. System zapisuje posta (oraz załączniki w chmurze) i wyświetla go na liście postów. |
| Alternatywne przepływy zdarzeń: | <ol style="list-style-type: none"> 3a. Załącznik nie może zostać zapisany – system informuje o błędzie i pozwala opublikować posta bez pliku. 4a. Formularz zawiera błędne lub niekompletne dane – system wyświetla komunikat i prosi o poprawę. |

Tabela 4.27: Scenariusz przypadku użycia: Dodanie posta na forum

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|------------------------------------|---|
| Identyfikator: | PU28 |
| Nazwa: | Dodanie komentarza na forum |
| Priorytet: | Wysoki |
| Aktorzy: | Użytkownik zalogowany |
| Opis: | Użytkownik dodaje komentarz pod postem na forum. |
| Warunki wstępne: | Użytkownik jest zalogowany i widzi szczegóły posta. |
| Warunki końcowe: | Nowy komentarz został zapisany i widoczny pod postem. |

| | |
|--|--|
| Główny przepływ zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik wpisuje treść komentarza w formularzu pod postem. 2. Użytkownik publikuje komentarz. 3. System zapisuje komentarz i odświeża listę komentarzy. |
| Alternatywne przepływy zdarzeń: | <ol style="list-style-type: none"> 2a. Treść komentarza jest niepoprawa – system wyświetla komunikat o błędzie. |

Tabela 4.28: Scenariusz przypadku użycia: Dodanie komentarza na forum

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|---|--|
| Identyfikator: | PU29 |
| Nazwa: | Przeglądanie historii interakcji z postami |
| Priorytet: | Średni |
| Aktorzy: | Użytkownik zalogowany |
| Opis: | Użytkownik przegląda historię swoich aktywności na forum (dodane posty, komentarze, reakcje). |
| Warunki wstępne: | Użytkownik jest zalogowany. |
| Warunki końcowe: | Lista interakcji użytkownika z postami jest wyświetlona. |
| Główny przepływ zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik przechodzi do sekcji historii aktywności. 2. System pobiera historię interakcji użytkownika. 3. System wyświetla listę interakcji z możliwością filtrowania. |

| | |
|--|---|
| Alternatywne przepływy zdarzeń: | Brak istotnych alternatywnych przepływów. |
|--|---|

Tabela 4.29: Scenariusz przypadku użycia: Przeglądanie historii interakcji z postami

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|---|---|
| Identyfikator: | PU30 |
| Nazwa: | Zarządzanie komentarzami do spotów |
| Priorytet: | Niski |
| Aktorzy: | Użytkownik zalogowany (właściciel spota lub moderator) |
| Opis: | Użytkownik zarządza komentarzami dodanymi do spota (edycja, usuwanie, ukrywanie). |
| Warunki wstępne: | Użytkownik jest zalogowany i wyświetla szczegóły spota. |
| Warunki końcowe: | Wybrane komentarze zostały zaktualizowane lub ukryte zgodnie z działaniem użytkownika. |
| Główny przepływ zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik otwiera panel zarządzania komentarzami dla danego spota. 2. System pobiera listę komentarzy wraz z możliwymi akcjami. 3. Użytkownik wybiera komentarz i akcję (np. edytuj, usuń, ukryj). 4. System wykonuje wybraną akcję na komentarzu. 5. System odświeża listę komentarzy. |

| | |
|--|--|
| Alternatywne przebiegi zdarzeń: | 3a. Użytkownik nie ma uprawnień do zarządzania komentarzem – system wyświetla komunikat o braku uprawnień. |
|--|--|

Tabela 4.30: Scenariusz przypadku użycia: Zarządzanie komentarzami do spotów

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|---|--|
| Identyfikator: | PU31 |
| Nazwa: | Zarządzanie komentarzami na forum |
| Priorytet: | Niski |
| Aktorzy: | Użytkownik zalogowany (autor posta lub moderator) |
| Opis: | Użytkownik zarządza komentarzami pod postami forum (edycja, usuwanie, przypinanie). |
| Warunki wstępne: | Użytkownik jest zalogowany i ma dostęp do danego wątku forum. |
| Warunki końcowe: | Komentarze zostały zaktualizowane zgodnie z działaniami użytkownika. |
| Główny przebieg zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik otwiera widok komentarzy pod postem. 2. Użytkownik wybiera komentarz i odpowiednią akcję. 3. System weryfikuje uprawnienia użytkownika. 4. System wykonuje wybraną akcję i aktualizuje widok. |
| Alternatywne przebiegi zdarzeń: | 3a. Użytkownik nie ma wymaganych uprawnień – system blokuje operację i informuje o tym. |

Tabela 4.31: Scenariusz przypadku użycia: Zarządzanie komentarzami na forum

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|--|--|
| Identyfikator: | PU32 |
| Nazwa: | Zarządzanie postami na forum |
| Priorytet: | Niski |
| Aktorzy: | Użytkownik zalogowany (autor posta lub moderator) |
| Opis: | Użytkownik edytuje, archiwizuje lub usuwa własne posty na forum. |
| Warunki wstępne: | Użytkownik jest zalogowany i otwiera listę swoich postów lub moderowany dział forum. |
| Warunki końcowe: | Status wybranych postów został zaktualizowany. |
| Główny przepływ zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik przechodzi do sekcji zarządzania postami. 2. System pobiera listę postów użytkownika (lub działu). 3. Użytkownik wybiera post i żadaną akcję (edycja, archiwizacja, usunięcie). 4. System zapisuje zmiany i aktualizuje listę postów. |
| Alternatywne przepływy zdarzeń: | 3a. Użytkownik próbuje usunąć post z zablokowanego wątku – system odmawia wykonania operacji. |

Tabela 4.32: Scenariusz przypadku użycia: Zarządzanie postami na forum

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|------------------------------------|---|
| Identyfikator: | PU33 |
| Nazwa: | Zgłoszenie komentarza naruszającego regulamin |
| Priorytet: | Średni |
| Aktorzy: | Użytkownik zalogowany |

| | |
|--|---|
| Opis: | Użytkownik zgłasza komentarz na forum. |
| Warunki wstępne: | Użytkownik widzi komentarz w aplikacji. |
| Warunki końcowe: | Zgłoszenie komentarza zostało zapisane i trafiło do kolejki moderacyjnej. |
| Główny przepływ zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik wybiera opcję „Zgłoś komentarz”. 2. Użytkownik określa powód zgłoszenia. 3. System zapisuje zgłoszenie i wiąże je z komentarzem i zgłaszającym. |
| Alternatywne przepływy zdarzeń: | Brak istotnych alternatywnych przepływów. |

Tabela 4.33: Scenariusz przypadku użycia: Zgłoszenie komentarza naruszającego regulamin

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|---|--|
| Identyfikator: | PU34 |
| Nazwa: | Zgłoszenie posta na forum |
| Priorytet: | Średni |
| Aktorzy: | Użytkownik zalogowany |
| Opis: | Użytkownik zgłasza post forum jako naruszający regulamin lub tematykę. |
| Warunki wstępne: | Wyświetlony jest widok posta na forum. |
| Warunki końcowe: | Zgłoszenie posta zostało zapisane i przekazane moderatorom. |

| | |
|--|--|
| Główny przepływ zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik wybiera opcję „Zgłoś post”. 2. Użytkownik wybiera kategorię naruszenia i potwierdza zgłoszenie. 3. System zapisuje zgłoszenie i oznacza post jako zgłoszony. |
| Alternatywne przepływy zdarzeń: | Brak istotnych alternatywnych przepływów. |

Tabela 4.34: Scenariusz przypadku użycia: Zgłoszenie posta na forum

Scenariusze przypadków użycia dla profilu użytkownika

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|---|---|
| Identyfikator: | PU35 |
| Nazwa: | Dodanie spota w profilu użytkownika |
| Priorytet: | Wysoki |
| Aktorzy: | Użytkownik zalogowany, Usługa do wyświetlania mapy, Usługa do przechowywania plików w chmurze |
| Opis: | Użytkownik dodaje nowy spot poprzez swój profil. |
| Warunki wstępne: | Użytkownik jest zalogowany i znajduje się w widoku swojego profilu. |
| Warunki końcowe: | Nowy spot został zapisany i widoczny na mapie oraz w profilu użytkownika. |

| | |
|--|---|
| Główny przebieg zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik wybiera opcję „Dodaj spota”. 2. Użytkownik uzupełnia podstawowe informacje o spocie (nazwa, opis, typ). 3. Użytkownik wskazuje lokalizację spota na mapie. 4. (Opcjonalnie) Użytkownik dodaje zdjęcia/filmy do spota. 5. Użytkownik zapisuje spota. 6. System zapisuje dane spota (oraz pliki w chmurze) i aktualizuje mapę oraz profil użytkownika. |
| Alternatywne przebiegi zdarzeń: | <ol style="list-style-type: none"> 2a. Formularz zawiera błędy – system wyświetla komunikat i zaznacza wymagające poprawy pola. |

Tabela 4.35: Scenariusz przypadku użycia: Dodanie spota w profilu użytkownika

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|---|--|
| Identyfikator: | PU36 |
| Nazwa: | Przeglądanie profilu użytkownika |
| Priorytet: | Wysoki |
| Aktorzy: | Użytkownik zalogowany |
| Opis: | Użytkownik przegląda swój profil (lista spotów, media, podstawowe dane). |
| Warunki wstępne: | Użytkownik jest zalogowany. |
| Warunki końcowe: | Wyświetlony jest widok profilu użytkownika wraz z jego zawartością. |

| | |
|--|--|
| Główny przepływ zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik otwiera swój profil. 2. System pobiera dane profilu (informacje podstawowe, spoty, media). 3. System wyświetla dane w odpowiednich sekcjach (spoty, zdjęcia, filmy, komentarze). |
| Alternatywne przepływy zdarzeń: | Brak istotnych alternatywnych przepływów. |

Tabela 4.36: Scenariusz przypadku użycia: Przeglądanie profilu użytkownika

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|---|---|
| Identyfikator: | PU37 |
| Nazwa: | Przeglądanie profilu innego użytkownika |
| Priorytet: | Średni |
| Aktorzy: | Użytkownik zalogowany |
| Opis: | Użytkownik ogląda profil innego użytkownika (np. z mapy, forum lub społeczności). |
| Warunki wstępne: | Użytkownik jest zalogowany i ma dostęp do odnośnika do profilu innego użytkownika. |
| Warunki końcowe: | Profil innego użytkownika został wyświetlony. |
| Główny przepływ zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik wybiera odnośnik do profilu innego użytkownika. 2. System pobiera dane profilu docelowego użytkownika. 3. System wyświetla profil (media, podstawowe informacje). |

| | |
|--|---|
| Alternatywne przebiegi zdarzeń: | 2a. Wystąpił błąd podczas pobierania danych użytkownika – system wyświetla informację o błędzie. |
|--|---|

Tabela 4.37: Scenariusz przypadku użycia: Przeglądanie profilu innego użytkownika

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|---|---|
| Identyfikator: | PU38 |
| Nazwa: | Dodanie użytkownika do znajomych |
| Priorytet: | Średni |
| Aktorzy: | Użytkownik zalogowany |
| Opis: | Użytkownik wysyła lub akceptuje zaproszenie do znajomych. |
| Warunki wstępne: | Użytkownik jest zalogowany i przegląda profil innego użytkownika. |
| Warunki końcowe: | Relacja „znajomy” została utworzona lub zaproszenie czeka na akceptację. |
| Główny przebieg zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik klika przycisk „Dodaj do znajomych”. 2. System sprawdza, czy relacja już istnieje. 3. System tworzy nowe zaproszenie. 4. System informuje o statusie o wysłaniu zaproszenia. |
| Alternatywne przebiegi zdarzeń: | Brak istotnych alternatywnych przebiegów. |

Tabela 4.38: Scenariusz przypadku użycia: Dodanie użytkownika do znajomych

| KARTA SCENARIUSZA PRZYPADKU UŻYCIA | |
|--|---|
| Identyfikator: | PU39 |
| Nazwa: | Przeglądanie społeczności |
| Priorytet: | Średni |
| Aktorzy: | Użytkownik zalogowany |
| Opis: | Użytkownik przegląda społeczności, grupy lub listy znajomych powiązane z aplikacją. |
| Warunki wstępne: | Użytkownik jest zalogowany. |
| Warunki końcowe: | Lista społeczności lub znajomych została wyświetlona. |
| Główny przepływ zdarzeń: | <ol style="list-style-type: none"> 1. Użytkownik przechodzi do sekcji społeczności. 2. System pobiera listę społeczności i znajomych użytkownika. 3. System wyświetla listę z możliwością przechodzenia do profili i czatów. |
| Alternatywne przepływy zdarzeń: | Brak istotnych alternatywnych przepływów. |

Tabela 4.39: Scenariusz przypadku użycia: Przeglądanie społeczności

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 <code>@tanstack/react-query</code> i <code>axios</code> z <code>withCredentials</code> . Prezentacja w widoku profilu. | | |
| Udziałowiec: | Zespół projektowy 2.1; promotor 2.2; droniarze 2.3. | | |
| Wymagania powiązane: | | | |

Tabela 4.40: 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 4.41: 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 2.1; promotor 2.2; droniarze 2.3. | | |
| Wymagania powiązane: | | | |

Tabela 4.42: 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 4.43: 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 4.44: 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 react-query + axios ; standardowa prezentacja listy. | | |
| Udziałowiec: | Zespół projektowy 2.1; promotor 2.2; droniarze 2.3. | | |
| Wymagania powiązane: | | | |

Tabela 4.45: 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 react-query + axios ; standardowa prezentacja listy. | | |
| Udziałowiec: | Zespół projektowy 2.1; promotor 2.2; droniarze 2.3. | | |
| Wymagania powiązane: | | | |

Tabela 4.46: 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 4.47: 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 4.48: 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 2.1; promotor 2.2; droniarze 2.3. | | |
| Wymagania powiązane: | | | |

Tabela 4.49: 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 2.1; promotor 2.2; droniarze 2.3. | | |
| Wymagania powiązane: | | | |

Tabela 4.50: 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 axios ; obsługa komunikatów o powodzeniu/błędach. | | |
| Udziałowiec: | Zespół projektowy 2.1; promotor 2.2; droniarze 2.3. | | |
| Wymagania powiązane: | | | |

Tabela 4.51: 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 axios ; po akceptacji — aktualizacja listy (odświeżenie react-query). | | |
| Udziałowiec: | Zespół projektowy 2.1; promotor 2.2; droniarze 2.3. | | |
| Wymagania powiązane: | | | |

Tabela 4.52: 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 2.1, promotor 2.2, droniarze 2.3. | | |
| Wymagania powiązane: | | | |

Tabela 4.53: 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 @tanstack/react-query i axios (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 4.54: 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 4.55: 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><html></code> ; | | |
| Udziałowiec: | Zespół projektowy 2.1, promotor 2.2, droniarze 2.3. | | |
| Wymagania powiązane: | | | |

Tabela 4.56: 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><html></code> (eliminuje mignięcie stylów). | | |
| Udziałowiec: | Zespół projektowy 2.1, promotor 2.2, droniarze 2.3. | | |
| Wymagania powiązane: | | | |

Tabela 4.57: Zapamiętanie preferencji motywu

| KARTA WYMAGANIA | | | |
|--------------------------|---|------------|---|
| Identyfikator: | FOXX | Priorytet: | S |
| Nazwa: | Przełącznik motywu w Sidebar | | |
| Opis: | Jako użytkownik chcę szybko zmieniać motyw bez wchodzenia w ustawienia. | | |
| Kryteria akceptacji: | W Sidebar dostępny jest przełącznik <i>Jasny-/Ciemny</i> ; posiada odpowiednio ikony <i>słońca/księżyca</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 2.1, promotor 2.2, droniarze 2.3. | | |
| Wymagania powiązane: | | | |

Tabela 4.58: 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 Endpointy systemu

GET /user-dashboard/profile

Opis: Zwraca profil aktualnie zalogowanego użytkownika.

Metoda: GET /user-dashboard/profile

Zwraca (200 OK): application/json — obiekt UserProfileDto.

Błąd (404 Not Found): text/plain —

komunikat z wyjątku UserNotFoundByUsernameException.

Przykładowa odpowiedź (200 OK):

```
1  {
2    "username": "john_doe",
3    "profilePhoto": "https://cdn.example.com/profiles/john_doe.
4      jpg",
5    "followersCount": 125,
6    "followedCount": 87,
7    "friendsCount": 32,
8    "photosCount": 58,
9    "mostPopularPhotos": [
10     {
11       "src": "https://cdn.example.com/photos/123.jpg",
12       "heartsCount": 240,
```

```
12     "viewsCount": 3400,
13     "title": "Sunset at the beach",
14     "id": 123
15   }
16 ]
17 }
```

Example response (404 Not Found):

Panel użytkownika

- GET /user-dashboard/profile
- GET /public/user-dashboard/profile/"targetUsername"
- PATCH /user-dashboard/profile
- GET /user-dashboard/friends
- GET /public/user-dashboard/friends/"targetUsername"
- PATCH /user-dashboard/friends
- PATCH /user-dashboard/friends/change-status
- GET /user-dashboard/followers
- GET /public/user-dashboard/followers/"targetUsername"
- GET /user-dashboard/followed
- GET /public/user-dashboard/followed/"targetUsername"
- GET /user-dashboard/friends/find
- GET /user-dashboard/friends/invites
- PATCH /user-dashboard/followed
- GET /user-dashboard/favorite-spots
- PATCH /user-dashboard/favorite-spots

- GET /user-dashboard/photos
- GET /user-dashboard/comments
- PATCH /user-dashboard/settings
- GET /user-dashboard/settings
- GET /user-dashboard/movies
- GET /user-dashboard/photos/"targetUsername"
- GET /user-dashboard/add-spot
- POST /user-dashboard/add-spot
- GET /user-dashboard/add-spot/coordinates

Strona główna

- GET /public/spot/most-popular
- GET /public/spot/search/home-page
- GET /public/spot/search/home-page/locations
- GET /public/spot/search/home-page/advance

Konto użytkownika

- POST /public/account/register
- POST /public/account/login
- GET /account/login-success
- POST /public/account/forgot-password
- POST /public/account/set-new-password
- GET /account/check

7.1.3 Integracja z bazą danych

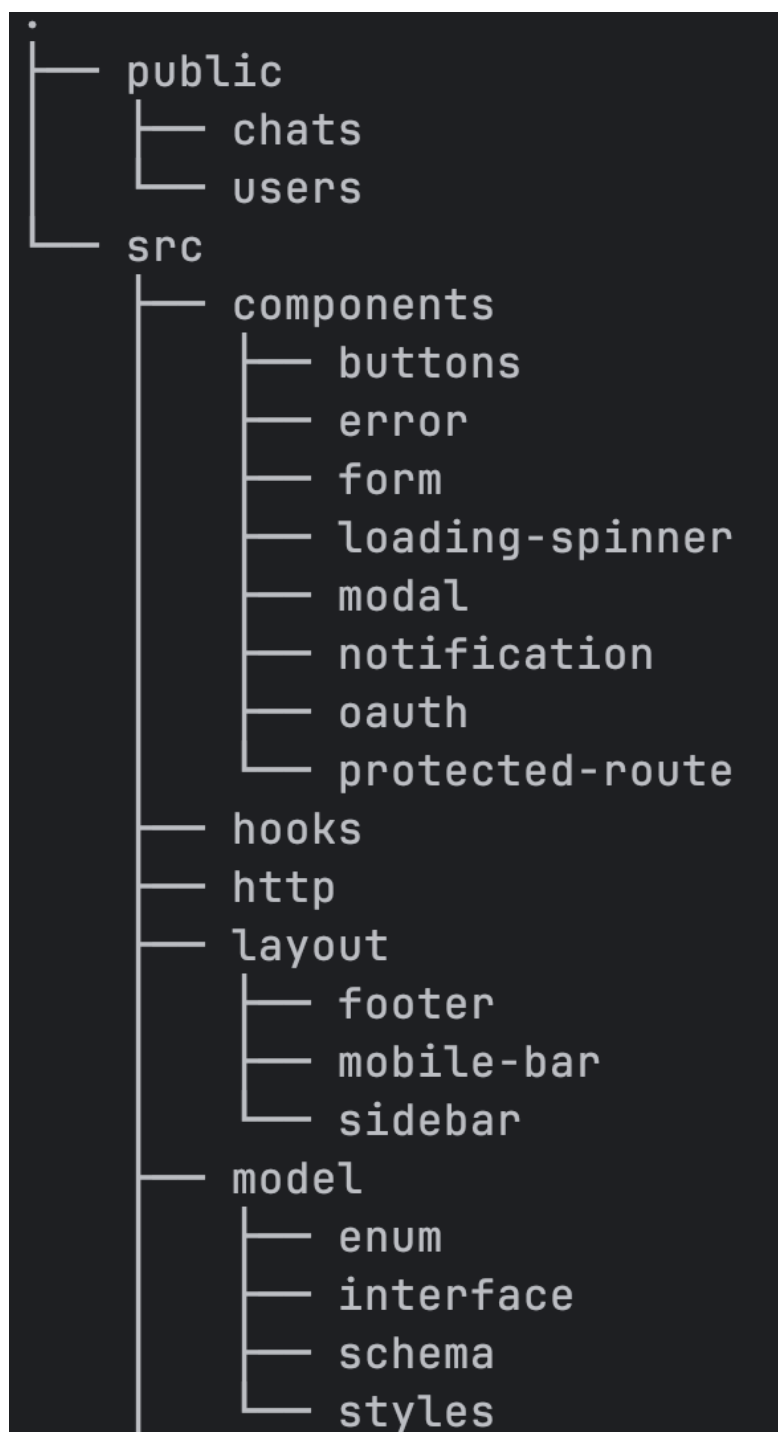
7.1.4 Obsługa uwierzytelnienia

7.1.5 Konteneryzacja

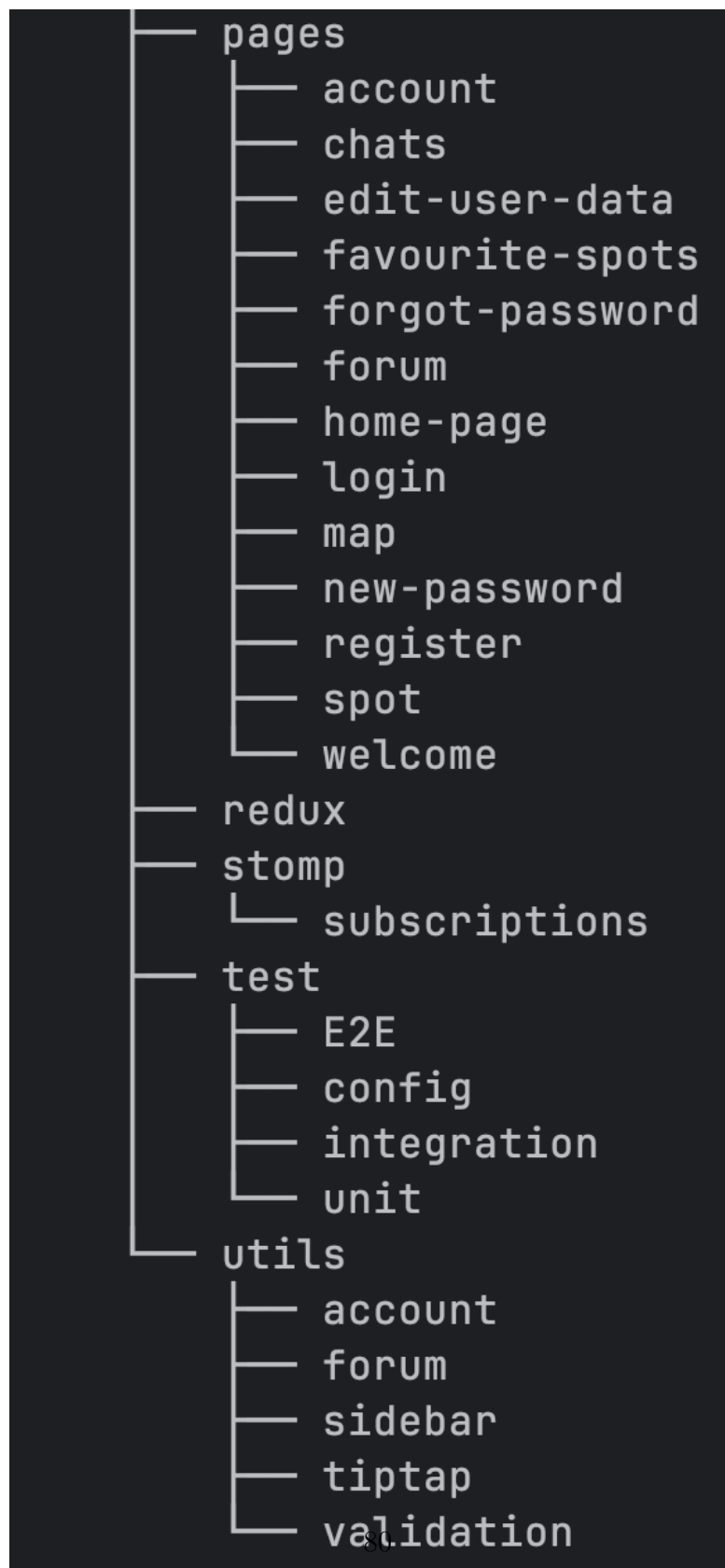
7.2 Implementacja frontendu

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`) [`react-use-state`], jak i ze Stanu globalnego, utrzymywanego przez Bibliotekę React Redux [`redux`]. 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` [axios] oraz Biblioteki `TanStack Query` [tanstack-query]. 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>) => Pr... = async (event: FormEvent<HTMLFormElement>) : Promise<void> => {
  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 [**tailwind**]. 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 [**motion**]. 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 Strona główna

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

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 | | 23 |
| Tabela 4.2: Scenariusz przypadku użycia: Logowanie użytkownika | | 24 |
| Tabela 4.3: Scenariusz przypadku użycia: Resetowanie hasła | | 25 |
| Tabela 4.4: Scenariusz przypadku użycia: Zmiana hasła w ustawieniach konta | | 26 |
| Tabela 4.5: Scenariusz przypadku użycia: Wylogowanie użytkownika . . . | | 26 |
| Tabela 4.6: Scenariusz przypadku użycia: Przeglądanie powiadomień . . . | | 27 |
| Tabela 4.7: Scenariusz przypadku użycia: Wykupienie subskrypcji premium | | 28 |
| Tabela 4.8: Scenariusz przypadku użycia: Wyszukiwanie spota na mapie | | 29 |
| Tabela 4.9: Scenariusz przypadku użycia: Wyszukiwanie spota w global- nej wyszukiwarce | | 30 |

| | |
|--|----|
| Tabela 4.10: Scenariusz przypadku użycia: Przejście do spota na mapie z wyszukiwarki | 31 |
| Tabela 4.11: Scenariusz przypadku użycia: Przeglądanie mapy spotów . . | 32 |
| Tabela 4.12: Scenariusz przypadku użycia: Przeglądanie komentarzy do spota | 33 |
| Tabela 4.13: Scenariusz przypadku użycia: Przeglądanie pogody na spocie | 33 |
| Tabela 4.14: Scenariusz przypadku użycia: Zmiana typu mapy | 34 |
| Tabela 4.15: Scenariusz przypadku użycia: Przeglądanie stref PANSA . . | 35 |
| Tabela 4.16: Scenariusz przypadku użycia: Utworzenie prywatnego czatu | 36 |
| Tabela 4.17: Scenariusz przypadku użycia: Utworzenie czatu grupowego . | 37 |
| Tabela 4.18: Scenariusz przypadku użycia: Przeglądanie listy czatów . . . | 38 |
| Tabela 4.19: Scenariusz przypadku użycia: Wysyłanie wiadomości na czacie | 39 |
| Tabela 4.20: Scenariusz przypadku użycia: Wysyłanie GIF-a na czacie . . | 39 |
| Tabela 4.21: Scenariusz przypadku użycia: Wysyłanie pliku na czacie . . | 40 |
| Tabela 4.22: Scenariusz przypadku użycia: Edycja ustawień czatu | 41 |
| Tabela 4.23: Scenariusz przypadku użycia: Dodanie członka do czatu grupowego | 42 |
| Tabela 4.24: Scenariusz przypadku użycia: Przeszukiwanie historii czatu . | 43 |
| Tabela 4.25: Scenariusz przypadku użycia: Przeglądanie wysłanych plików na czacie | 44 |
| Tabela 4.26: Scenariusz przypadku użycia: Przeglądanie postów na forum | 45 |
| Tabela 4.27: Scenariusz przypadku użycia: Dodanie posta na forum . . . | 46 |
| Tabela 4.28: Scenariusz przypadku użycia: Dodanie komentarza na forum | 47 |
| Tabela 4.29: Scenariusz przypadku użycia: Przeglądanie historii interakcji z postami | 48 |
| Tabela 4.30: Scenariusz przypadku użycia: Zarządzanie komentarzami do spotów | 49 |
| Tabela 4.31: Scenariusz przypadku użycia: Zarządzanie komentarzami na forum | 49 |
| Tabela 4.32: Scenariusz przypadku użycia: Zarządzanie postami na forum | 50 |

| | |
|---|----|
| Tabela 4.33: Scenariusz przypadku użycia: Zgłoszenie komentarza naruszającego regulamin | 51 |
| Tabela 4.34: Scenariusz przypadku użycia: Zgłoszenie posta na forum . . | 52 |
| Tabela 4.35: Scenariusz przypadku użycia: Dodanie spota w profilu użytkownika | 53 |
| Tabela 4.36: Scenariusz przypadku użycia: Przeglądanie profilu użytkownika | 54 |
| Tabela 4.37: Scenariusz przypadku użycia: Przeglądanie profilu innego użytkownika | 55 |
| Tabela 4.38: Scenariusz przypadku użycia: Dodanie użytkownika do znajomych | 55 |
| Tabela 4.39: Scenariusz przypadku użycia: Przeglądanie społeczności . . | 56 |
| 4.40 Profil użytkownika | 57 |
| 4.41 Lista dodanych spotów | 58 |
| 4.42 Dodanie spota | 59 |
| 4.43 Lista zdjęć | 60 |
| 4.44 Lista filmów | 60 |
| 4.45 Lista znajomych | 61 |
| 4.46 Lista obserwujących | 61 |
| 4.47 Lista obserwowanych | 62 |
| 4.48 Lista polubionych/odwiedzonych/planowanych spotów | 62 |
| 4.49 Lista komentarzy | 63 |
| 4.50 Ustawienia profilu | 64 |
| 4.51 Resetowanie hasła | 65 |
| 4.52 Dodawanie do znajomych | 66 |
| 4.53 Logowanie i rejestracja | 67 |
| 4.54 Strona główna — podstawowe filtry | 68 |
| 4.55 Strona główna — zaawansowane filtry | 69 |
| 4.56 Ustawienia motywu (ręczna zmiana) | 70 |
| 4.57 Zapamiętanie preferencji motywu | 71 |
| 4.58 Szybki przełącznik motywu w interfejsie | 72 |

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.