



POLSKO-JAPONSKA 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 stylów 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 albumu	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: 1 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	7
2.4 Wizja rozwiązania	7
2.5 Aspekty społeczne i biznesowe	7
2.5.1 Aspekty społeczne	7
2.5.2 Aspekty biznesowe	7
3 Planowanie	8
3.1 Metodologia pracy	8
3.1.1 Przegląd rozważanych podejść	8
3.1.2 Odrzucone podejścia	8
3.1.3 Wybrane podejście: Disciplined Agile Delivery (Lean Life Cycle)	9
3.1.4 Narzędzia i komunikacja	9
3.1.5 Podział ról w zespole	10
3.2 Harmonogram projektu	10
3.3 Technologie i narzędzia	12
3.3.1 Technologie	12

3.3.2	Narzędzia	12
3.4	Zasoby i ograniczenia	12
3.4.1	Zasoby	12
3.5	Zasoby projektu	12
3.5.1	Ograniczenia	13
3.6	Analiza ryzyka	13
4	Analiza wymagań	14
4.1	Przypadki użycia	14
4.1.1	Aktorzy	14
4.1.2	Diagram przypadków użycia	16
4.1.3	Scenariusz przypadków użycia	16
4.2	Wymagania ogólne i dziedzinowe	16
4.3	Wymagania funkcjonalne	16
4.3.1	Funkcjonalności dla mapy	16
4.3.2	Funkcjonalności dla chatu	16
4.3.3	Funkcjonalności dla forum	16
4.3.4	Funkcjonalności dla konta użytkownika	16
4.3.5	Funkcjonalności dla logowania i rejestracji	16
4.3.6	Funkcjonalności dla wyszukiwarki spotów	16
4.3.7	Funkcjonalności dla motywu	16
4.4	Wymagania pozafunkcjonalne	16
4.5	Wymagania interfejs z otoczeniem	16
4.6	Wymagania na środowisko docelowe	16
5	Projekt	17
5.1	Wzorce projektowe	17
5.2	Architektura systemu	17
5.2.1	Diagram architektury	17
5.2.2	Komponenty systemu	17
5.3	Projekt bazy danych	17
5.3.1	Model danych	17

5.3.2	Diagram ERD	17
5.4	Architektura interfejsu użytkownika	17
5.4.1	Projekt strony głównej	17
5.4.2	Projekt panelu logowania	17
5.4.3	Projekt mapy	17
5.4.4	Projekt chatu	17
5.4.5	Projekt forum	17
5.4.6	Projekt konta użytkownika	17
6	Przebieg realizacji projektu	18
6.1	Sprint 1	18
6.2	Sprint 2	18
7	Realizacja Projektu	19
7.1	Implementacja backendu	19
7.1.1	Struktura projektu	19
7.1.2	Endpointy systemu	19
7.1.3	Integracja z bazą danych	22
7.1.4	Obsługa uwierzytelnienia	22
7.1.5	Konteneryzacja	22
7.2	Implementacja frontendu	22
7.2.1	Struktura aplikacji	22
7.2.2	Zarządzanie stanem i przepływ danych	22
7.2.3	Integracja i komunikacja z backendem	22
7.2.4	Style	22
7.2.5	Strona główna	22
7.2.6	Mapa	22
7.2.7	Chat	22
7.2.8	Forum	22
7.2.9	Konto użytkownika	22
7.2.10	Panel logowania	22
7.3	Implementacja CI/CD	22

8 Testy	23
8.1 Testy jednostkowe	23
8.2 Testy integracyjne	23
8.3 Testy E2E	23
8.4 Wyniki testów i wnioski	23
9 Prezentacja systemu	24
9.1 Strona główna	24
9.2 Strona mapy	24
9.3 Strona chatu	24
9.4 Strona forum	24
9.5 Panel logowania	24
9.6 Panel konta użytkownika	24
10 Nakład pracy	25
10.1 Ogólny nakład pracy	25
10.2 Indywidualne nakłady pracy	25
10.2.1 Adam Langmesser	25
10.2.2 Mateusz Redosz	25
10.2.3 Stanisław Oziemczuk	28
10.2.4 Kacper Badek	28
11 Podsumowanie	29
11.1 Osiągnięte rezultaty	29
11.2 Napotkane wyzwania	29
11.3 Plany na przyszłość	29
12 Słownik pojęć i skrótów	30
Bibliografia	32
Załączniki	33

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

Nazwa udziałowca	Opis
Zespół projektowy	Odpowiedzialny za zaprojektowanie, implementację i testowanie systemu.
Promotor	Weryfikuje i nadzoruje pracę zespołu projektowego.
Droniarze	Główni użytkownicy systemu.

Tabela 2.2: Przedstawienie poszczególnych udziałowców wraz z ich znaczeniem

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, retrospekcja). Ze względu na nierównomierną dostępność zasobów w kolejnych miesiącach studiów nie zapewniono możliwości utrzymania stałej kadencji sprintów, dlatego z podejścia zrezygnowano.

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

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

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

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

Kluczowe argumenty wyboru:

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

3.1.4 Narzędzia i komunikacja

Do zarządzania zadaniami zastosowana zostanie **Jira** (monitorowanie postępu prac oraz ewidencja zadań członków zespołu). Komunikację w zespole zaplanowano w formie regularnych spotkań oraz asynchronicznie z wykorzystaniem **Discorda**.

3.1.5 Podział ról w zespole

- Adam
- Stanisław
- Kacper
- Mateusz

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

3.3.1 Technologie

3.3.2 Narzędzia

3.4 Zasoby i ograniczenia

3.4.1 Zasoby

3.5 Zasoby projektu

- **Czas zespołu** — realna dostępność godzin na tydzień (kolokwia, urlopy, inne zobowiązania).
- **Kompetencje** — umiejętności potrzebne do zadań (React, Spring, DevOps); braki tworzą wąskie gardła.
- **Minuty CI/CD** — limity uruchomień pipeline'ów (np. GitHub Actions).
- **Moc obliczeniowa** — CPU/RAM środowisk uruchomieniowych (klaster/K8s/Container Apps).
- **Pamięć masowa** — miejsce na obrazy, bazy, artefakty i logi (ACR/Blob/PostgreSQL).

- **Limity zewnętrznych API** — mapy, pogoda, SMTP (dzienne/miesięczne kwoty zapytań, throttling).
-

3.5.1 Ograniczenia

3.6 Analiza ryzyka

Rozdział 4

Analiza wymagań

4.1 Przypadki użycia

4.1.1 Aktorzy

Aktor	Krótki opis
Użytkownik niezalogowany	Gość przeglądający publiczne treści (mapa, spedy, forum), może się zarejestrować/zalogować.
Użytkownik (nie premium)	Zarejestrowany użytkownik podstawowy: zarządza kontem i ulubionymi, dodaje treści/komentarze, korzysta z czatu.
Użytkownik premium	Użytkownik z wykupioną subskrypcją; ma dostęp do funkcji premium (np. widok premium, rozszerzone wyszukiwanie).
Moderator	Nadzór treści: przegląda zgłoszenia, akceptuje/usuwa posty i komentarze, blokuje nadużycia.
Deweloper	Osoba utrzymująca system: inicjuje pipeline'y, monitoruje wdrożenia i infrastrukturę.

Kontynuacja na następnej stronie

Aktor	Krótki opis
Bramka płatnicza (Pytanie... bo nie bedzie u nas zaimplementowanej platnosci)	Zewnętrzny system obsługujący płatności/subskrypcje (autoryzacja, statusy).
Usługa mailowa (Mailtrap)	Serwis do wysyłki wiadomości e-mail (rejestracja, reset hasła, powiadomienia).
Dostawca API do map (OpenFreeMap)	Zewnętrzny dostawca map/tile/geokodowania wykorzystywany w widoku mapy.
Nawigacja (Google Maps)	Zewnętrzny dostawca map używany do nawigacji.
Dostawca API pogodowego (Open-Meteo)	Zewnętrzne API prognoz pogody używane do prezentacji warunków na spotach.
Dostawca API GIFów (Tenor)	Zewnętrzne API prognoz pogody używane do prezentacji warunków na spotach.
Dostawca API do określania strefy czasowej spota ("Where the ISS at?")	Zewnętrzne API prognoz pogody używane do prezentacji warunków na spotach.
Azure Blob Storage	Usługa chmurowa do przechowywania multimedialnych (np. zdjęcia spotów, pliki z czatu).
Baza danych (PostgreSQL)	System przechowywania danych aplikacji.
Cache (Redis)	System przechowywania danych aplikacji.
GitHub (repo + CI/CD)	Repozytorium kodu i pipeline'y automatyzujące build/test.
Dostawca OAuth (Google)	Zewnętrzny provider logowania/rejestracji.
Dostawca OAuth (Github)	Zewnętrzny provider logowania/rejestracji.

Tabela 4.1: Aktorzy systemu – przegląd

4.1.2 Diagram przypadków użycia

4.1.3 Scenariusz przypadków użycia

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

4.3.5 Funkcjonalności dla logowania i rejestracji

4.3.6 Funkcjonalności dla wyszukiwarki spotów

4.3.7 Funkcjonalności dla motywu

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 UserNotFoundException.

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

7.2.2 Zarządzanie stanem i przepływ danych

7.2.3 Integracja i komunikacja z backendem

7.2.4 Style

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 rozpoczęłem 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 Sidebaru
- Zmiana kropki na przyciemnienie tła na Sidebar
- Poprawa strony do logowania i rejestracji
- Usunięcie username z account Redux
- Dodanie zamknięcia Sidebaru na małych ekranach po kliknięciu nav linka
- Poprawić tooltipa na sidebar
- Zmiana sposobu pobierania username na backendzie z tokena jwt
- Paginacja z infinity scrolllem
- 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

Backend Część aplikacji odpowiedzialna za logikę biznesową, przetwarzanie danych i komunikację z bazą danych. Działa po stronie serwera i obsługuje żądania wysyłane przez frontend.. 2, 26

CI/CD Skrót od *Continuous Integration/Continuous Deployment*. Praktyka programistyczna polegająca na automatyzacji procesu budowania, testowania i wdrażania oprogramowania.. 25, 28

Design Etap lub proces projektowania wyglądu i funkcjonalności aplikacji, obejmujący zarówno aspekty wizualne, jak i użytkowe (UX/UI).. 25

Disciplined Agile Delivery - Lean Life Cycle Kanbanowy tryb pracy bez sprintów: zespół pobiera małe zadania z tablicy, pilnuje limitów pracy w toku i często wdraża małe zmiany. Planowanie jest lekkie i na bieżąco; priorytet mają rzeczy o najwyższej wartości. Skupiamy się na płynności—skracamy czas przejścia zadań, usuwamy blokady, usprawniamy proces małymi krokami. Jakość jest wbudowana: code review, testy automatyczne i prosta definicja „gotowe”.. 8

Framework Zestaw narzędzi, bibliotek i struktur wspomagających tworzenie aplikacji. Ułatwia programowanie poprzez dostarczenie gotowych komponentów oraz określenie zasad organizacji kodu.. 2

Frontend Warstwa aplikacji odpowiedzialna za interfejs użytkownika oraz interakcję z użytkownikiem. Zazwyczaj tworzona przy użyciu technologii takich jak HTML, CSS i JavaScript.. 2, 26

JWT Skrót od *JSON Web Token*. Standard służący do bezpiecznego przekazywania informacji między stronami w formacie JSON, często używany w procesach autoryzacji użytkowników.. 25

Review kodu Proces polegający na wzajemnym przeglądzie kodu źródłowego przez programistów w celu wykrycia błędów, poprawy jakości oraz zwiększenia spójności projektu.. 25

Sidebar Boczny panel w interfejsie użytkownika, zawierający menu nawigacyjne lub dodatkowe opcje funkcjonalne aplikacji.. 25–27

Bibliografia

- [1] *Disciplined Agile Delivery*. PMI. 1 stycznia 2025. URL: <https://www.pmi.org/disciplined-agile/process/introduction-to-dad/why> (dostęp 30.10.2025).
- [2] *Disciplined Agile Delivery — Lean Life Cycle*. PMI. 1 stycznia 2025. URL: <https://www.pmi.org/disciplined-agile/lifecycle/lean-lifecycle> (dostęp 30.10.2025).

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.