

Sprawozdanie z projektu – Program *G4B Store*

Amadeusz Żyła
Nr indeksu 43907

Sebastian Wróblewski
Nr indeksu 43902

Adrian Jurak
Nr indeksu 43847

PWSZ w Legnicy – Wydział Nauk Technicznych i Ekonomicznych
Informatyka, semestr IV – Legnica 2025

Przedmiot: Projektowanie i Programowanie Obiektowe (PIPO)

Rok akademicki 2024/2025

3 czerwca 2025

Spis treści

1	Opis funkcjonalny systemu	2
2	Podstawowe funkcje programu w diagramach UML	3
2.1	Diagram czynności (Activity Diagram)	4
2.2	Diagram przypadków użycia (Use Case)	5
2.3	Diagram klas (Class Diagram)	6
2.4	Diagram schematu bazy danych	7
3	Wdrożone zagadnienia kwalifikacyjne	8
4	Wbudowane funkcje i klasy	9
5	Dalsze możliwości rozwoju programu	10
6	Opis użytych narzędzi i bibliotek	11
7	Uproszczona instrukcja obsługi i FAQ	12
7.1	Krótką instrukcja obsługi	12
7.1.1	Instalacja i konfiguracja	12
7.2	Uruchomienie zdalne (Docker)	12
7.3	Najczęstsze pytania (FAQ)	13
8	Wnioski projektowe	14

Rozdział 1

Opis funkcjonalny systemu

"G4B Store" to pełnowartościowa aplikacja webowa oparta o architekturę Model–View–Controller (MVC), przeznaczona do sprzedaży cyfrowych gier komputerowych. System pozwala użytkownikom na przeglądanie oferty, zakup gier, zarządzanie koszykiem, subskrypcjami oraz historią zamówień. Kluczowe cechy:

- **Wielojęzyczność** – dynamiczne przełączanie języka interfejsu użytkownika (i18n) bez odświeżania sesji.
- **Paginacja** – dzielenie wyników na strony z możliwością przechodzenia między nimi bez przeładowywania całej strony.
- **Filtrowanie wyników** – interaktywne zawężanie listy gier na podstawie wybranych kryteriów, z natychmiastową aktualizacją widoku.
- **Przeglądanie gier** – dynamiczne wyświetlanie listy produktów z możliwością filtrowania po gatunku, platformie i przedziale cenowym.

Repozytorium projektu: <https://github.com/AdrianJurak/GameKeySite-Project-PSBD>

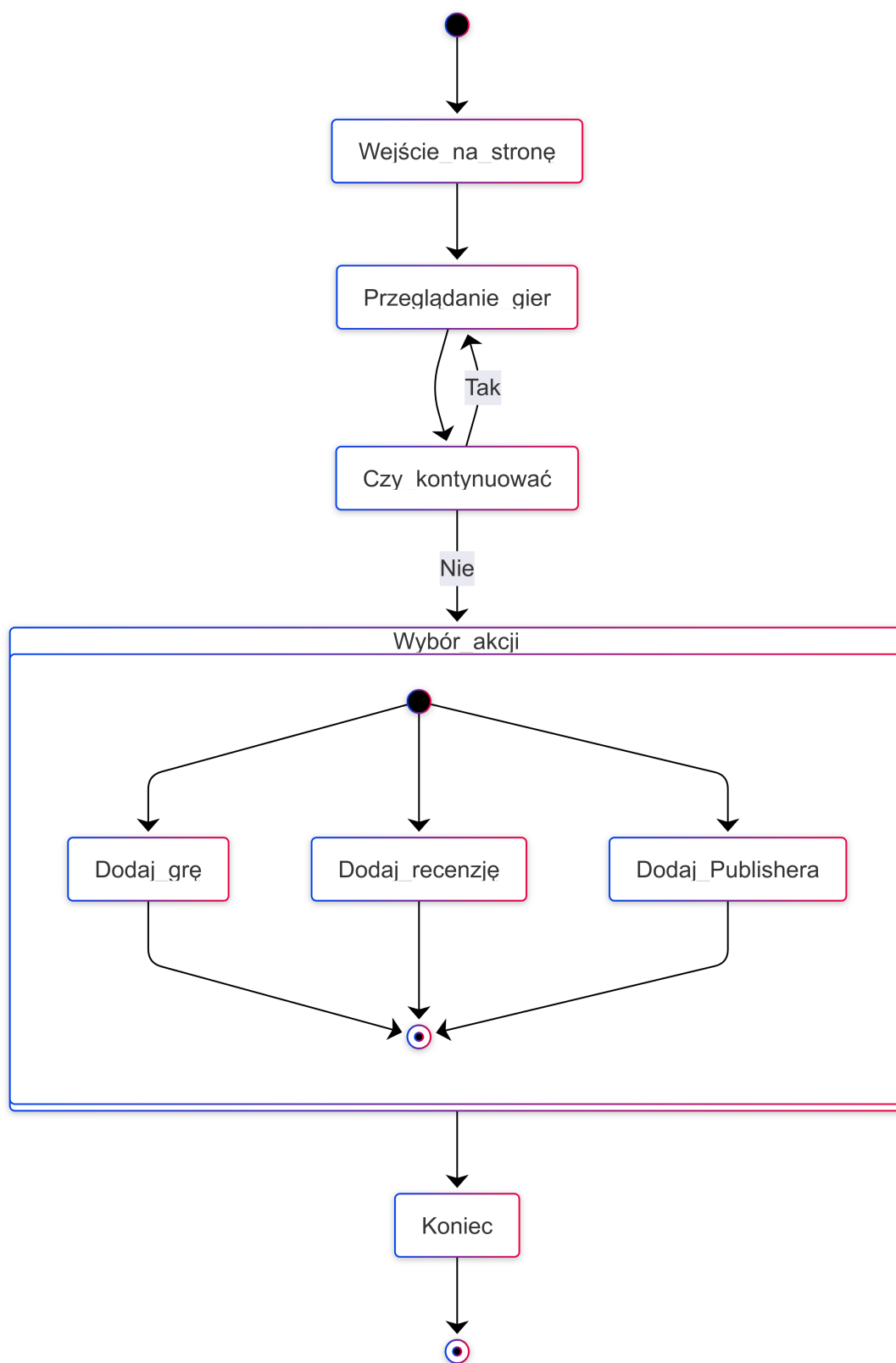
Baza danych MySQL przechowuje informacje o grach, wydawcach i recenzjach, a aplikacja wykorzystuje JPA/Hibernate do odwzorowania relacyjno–obiektowego.

Rozdział 2

Podstawowe funkcje programu w diagramach UML

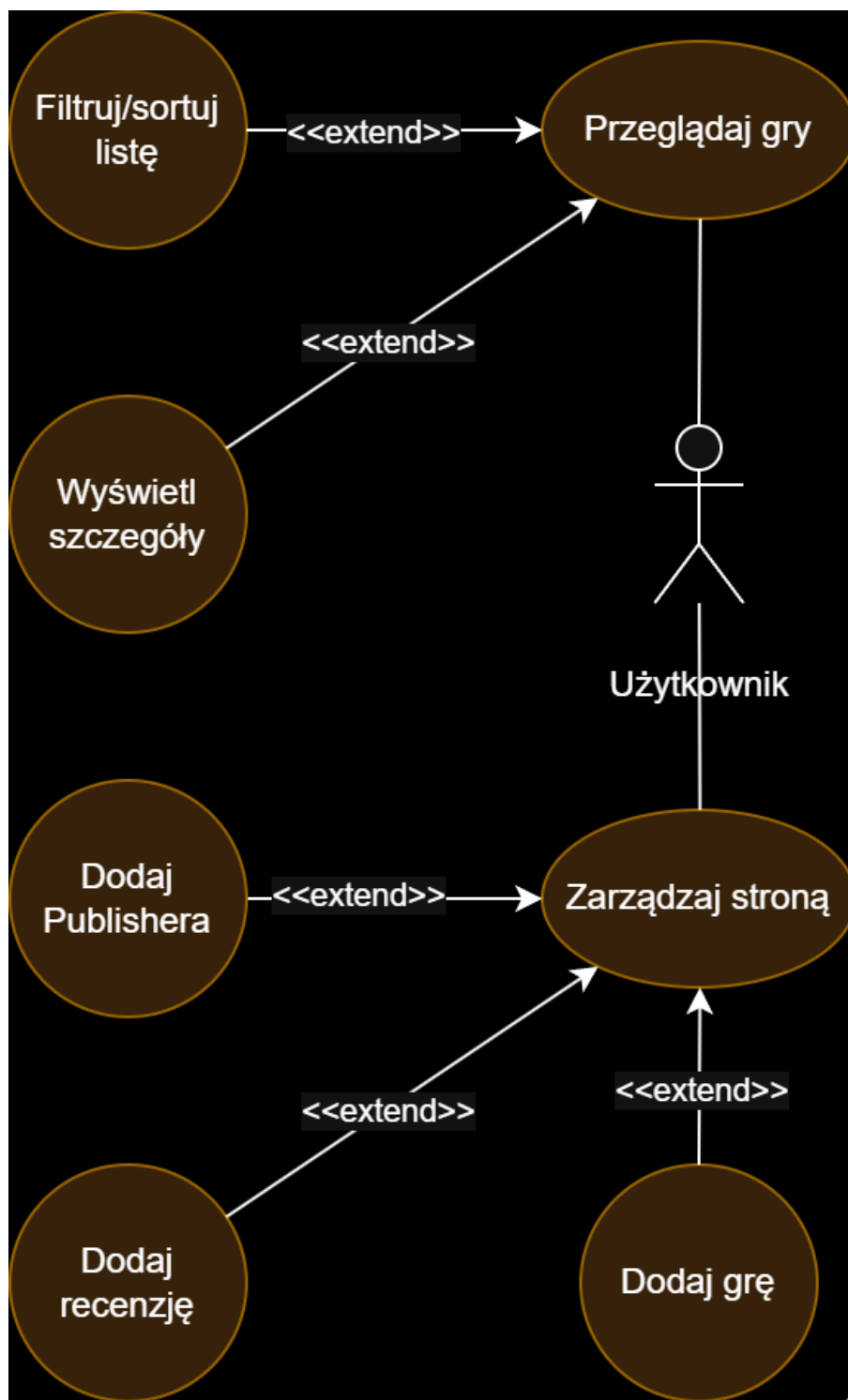
W tym rozdziale zaprezentowano cztery główne diagramy UML przygotowane w narzędziu `diagrams.net` oraz `Mermaid.js`.

2.1 Diagram czynności (Activity Diagram)



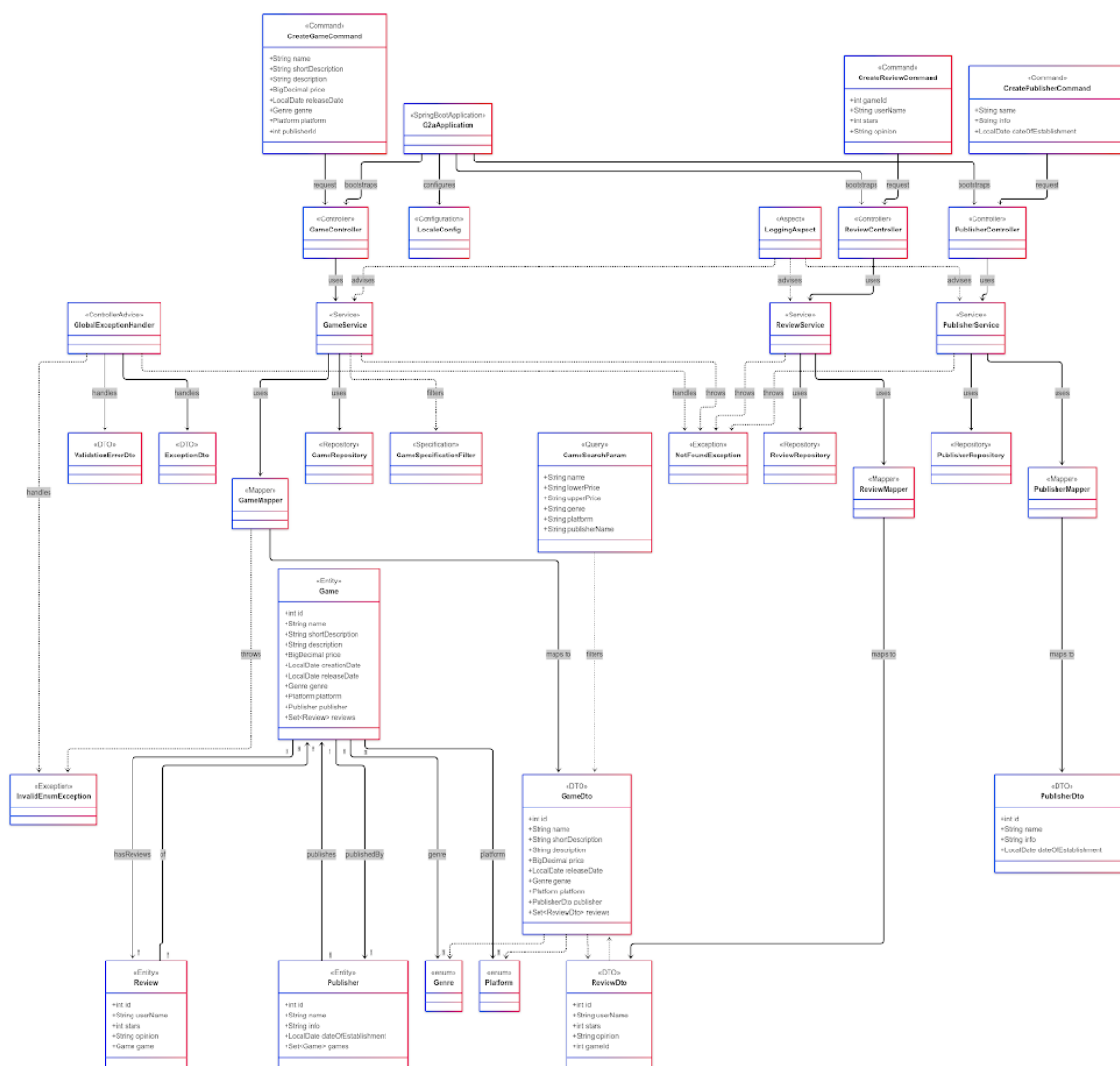
Rysunek 2.1: Przeływ czynności użytkownika w aplikacji *G4B Store*.

2.2 Diagram przypadków użycia (Use Case)



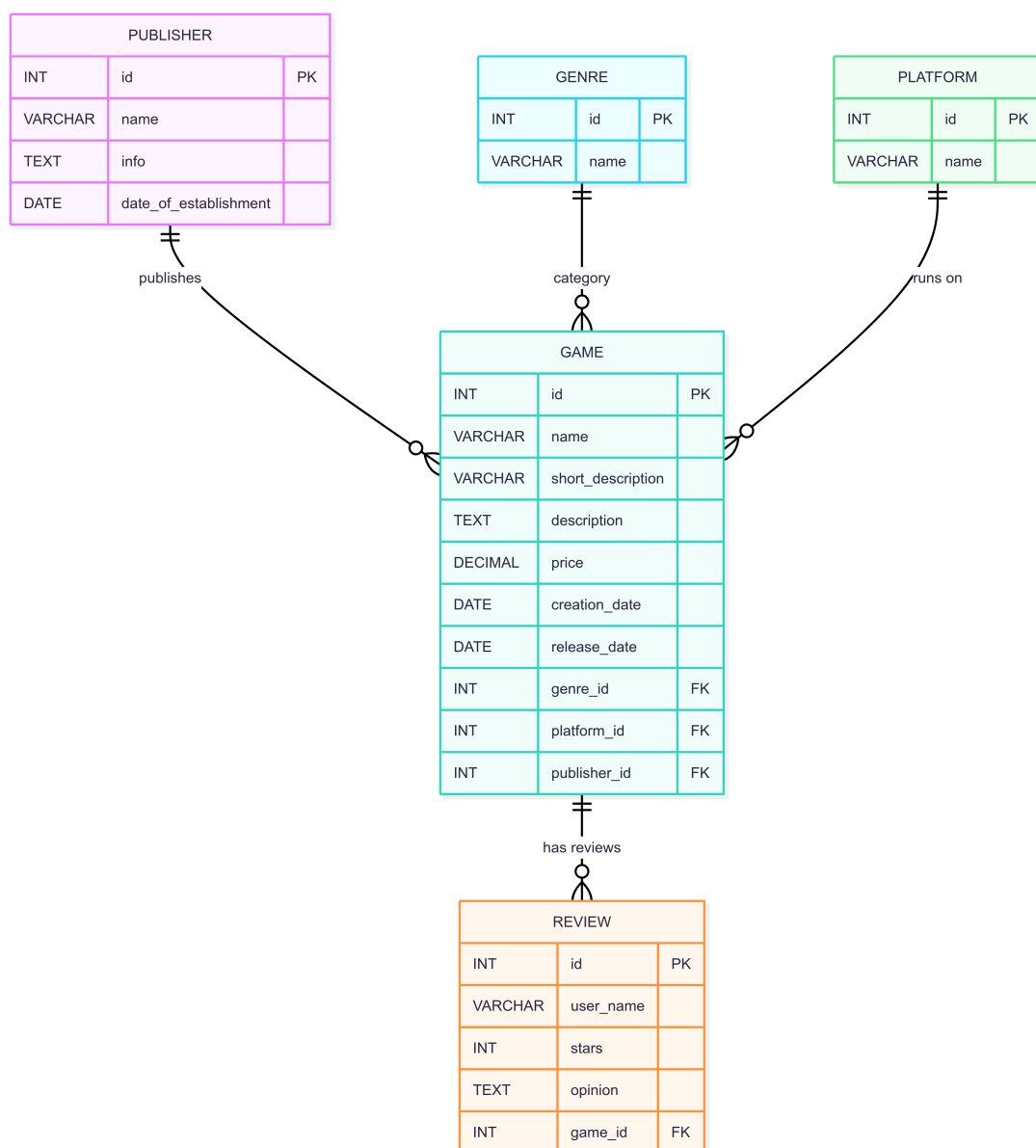
Rysunek 2.2: Przypadki użycia oraz aktorzy systemu.

2.3 Diagram klas (Class Diagram)



Rysunek 2.3: Struktura klas domenowych i serwisów.

2.4 Diagram schematu bazy danych



Rysunek 2.4: Model relacyjny bazy danych MySQL.

Rozdział 3

Wdrożone zagadnienia kwalifikacyjne

Poniższa tabela przedstawia listę zaimplementowanych zagadnień wraz z odpowiednimi powiązaniami z wykładami przedmiotu.

#	Zagadnienie	Opis	Wykład
1	framework MVC	wykorzystanie frameworka na backendzie	VI
3	baza danych	dołączenie do projektu bazy danych	VIII
5	dependency manager	dołączenie do projektu systemu zarządzania zależnościami	III
6	HTML	szkielet aplikacji internetowej	II
7	CSS	ostylowanie aplikacji internetowej	II
8	JavaScript	uinteraktywnienie aplikacji internetowej	XI
9	routing	wykorzystany routing i tzw. pretty URLs	V
10	ORM	wykorzystane mapowanie obiektowo-relacyjne	IX
12	lokalizacja	możliwość przełączania języka aplikacji	XV
14	formularze	przesyłanie danych do aplikacji przez formularze	II
19	logger	logowanie akcji w systemie	XV
Łącznie punktów			11

Tabela 3.1: Podsumowanie zrealizowanych zagadnień i powiązanych wykładów.

Rozdział 4

Wbudowane funkcje i klasy

Klasa / komponent	Zadanie
GameService	Logika biznesowa związana z wyszukiwaniem, filtrowaniem i dodawaniem gier.
GameSpecificationFilter	Dynamiczne budowanie kryteriów wyszukiwania (<i>Spring Data JPA Specification</i>).
GlobalExceptionHandler	Przechwytywanie i serializacja wyjątków do formatu JSON.

Tabela 4.1: Wybrane klasy i ich przeznaczenie.

Klasy encji oznaczone są adnotacją `@Entity` oraz mapowane na tabele MySQL, a relacje odzwierciedlają zależności *many to one* / *one to many* między grami, wydawcami i recenzjami.

Rozdział 5

Dalsze możliwości rozwoju programu

- **Konteneryzacja** – przygotowanie obrazów Docker oraz orkiestracja Kubernetes.
- **Użytkownicy** – rejestracja, logowanie, zarządzanie profilem, uwierzytelnianie JWT.
- **Koszyk** – dynamiczne dodawanie i aktualizacja produktów, lokalne przechowywanie (`localStorage`), synchronizacja z bazą danych.
- **Płatności** – integracja z bramkami płatniczymi (Stripe, PayPal), obsługa transakcji i walidacja danych.
- **Historia zakupów** – filtrowanie wg daty i statusu, generowanie faktur w formacie PDF.

Rozdział 6

Opis użytych narzędzi i bibliotek

- **IntelliJ IDEA 2024.1.2** – główne IDE z obsługą Spring Boot oraz Maven.
- **Spring Boot 3.2.1** – szkielet aplikacji, wbudowany kontener Tomcat, autokonfiguracja.
- **Hibernate 6.5** – ORM z mapowaniem JPA i migracjami Flyway.
- **Thymeleaf 3.1** – silnik szablonów HTML.
- **Lombok 1.18** – generowanie kodu boilerplate (getterzy, konstruktory).
- **Maven 3.9** – system budowania i zarządzania zależnościami.
- **MySQL 8.4** – relacyjna baza danych.
- **diagrams.net** – tworzenie diagramów UML (wersja 25.04.2025).
- **Mermaid.js** – generowanie diagramów w przeglądarce.

Rozdział 7

Uproszczona instrukcja obsługi i FAQ

7.1 Krótka instrukcja obsługi

7.1.1 Instalacja i konfiguracja

1. Sklonuj repozytorium:

```
git clone https://github.com/AdrianJurak/GameKeySite-Project-PSBD.git
cd GameKeySite-Project-PSBD
```

2. Skonfiguruj bazę danych w pliku `application.yml`.
3. Uruchom aplikację:

```
mvn spring-boot:run
```

4. Aplikacja będzie dostępna pod adresem `http://localhost:8080`.

7.2 Uruchomienie zdalne (Docker)

1. Zbuduj obraz kontenera:

```
docker build -t g4b-store .
```

2. Uruchom kontener na serwerze produkcyjnym:

```
docker run -d -p 80:8080 \
  --env MYSQL_URL="jdbc:mysql://<ip-serwera>:3306/g4b" \
  --env MYSQL_USER="g4b" \
  --env MYSQL_PASSWORD="*****" \
  g4b-store
```

3. Zweryfikuj logi:

```
docker logs -f $(docker ps -q --filter ancestor=g4b-store)
```

4. Aplikacja będzie dostępna pod `http://<IP_serwera>`.

7.3 Najczęstsze pytania (FAQ)

Aplikacja nie uruchamia się po wpisaniu `mvn spring-boot:run`. Sprawdź, czy plik `application.yml` został poprawnie skonfigurowany (dane do bazy danych) oraz czy port 3306 nie jest zajęty.

Jak zmienić język interfejsu? Kliknij ikonę języka w górnym menu aplikacji. UI przełączy się natychmiast, bez odświeżania strony.

Dlaczego nie widzę wszystkich gier na liście? Włączona jest paginacja. Użyj przycisków *Poprzednia/Następna* na dole strony, aby przejść do kolejnych wyników.

Jak usunąć zastosowane filtry? Kliknij przycisk *Wyczyść filtry* znajdujący się nad listą gier.

Czy mogę filtrować wiele kategorii jednocześnie (np. platforma i gatunek)? Tak. System pozwala na nakładanie wielu filtrów równocześnie – ich efekty są łączone i stosowane w czasie rzeczywistym.

Rozdział 8

Wnioski projektowe

- Projekt zrealizował wymagane minimum funkcjonalne i spełnia kryteria oceny **3** (dostateczny).
- Największe wyzwania dotyczyły paginacji z dynamicznym filtrowaniem wyników oraz wielojęzycznej prezentacji cen.
- W przyszłych projektach planujemy lepsze przygotowanie poprzez:
 - wprowadzenie pełnego zestawu testów jednostkowych i integracyjnych,
 - automatyzację procesu CI/CD (GitHub Actions),
 - dodanie kompleksowego systemu uwierzytelniania i autoryzacji użytkowników (np. JWT + OAuth2),
 - zastosowanie responsywnego frameworka CSS (np. Tailwind) oraz frameworka JS (np. Vue.js) do budowy front-endu,