Sprawozdanie z projektu UML – Program „MVC Game 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

Data: 3.06.2025

Przedmiot: *Projektowanie i Programowanie Obiektowe (PIPO)*

Rok akademicki: 2024/2025

Spis treści

1. Ogólny zarys programu

„MVC Game Store” to pełnowartościowa aplikacja webowa oparta o architekturę **ModelViewController (MVC)**, przeznaczona do sprzedaży cyfrowych gier komputerowych. System pozwala użytkownikom na przeglądanie oferty, – na zakup gier, zarządzanie koszykiem, subskrypcjami oraz historią zamówień. Kluczowe cechy:

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

Repozytorium GitHub: https://github.com/AdrianJurak/GameKeySite-Project-PSBD

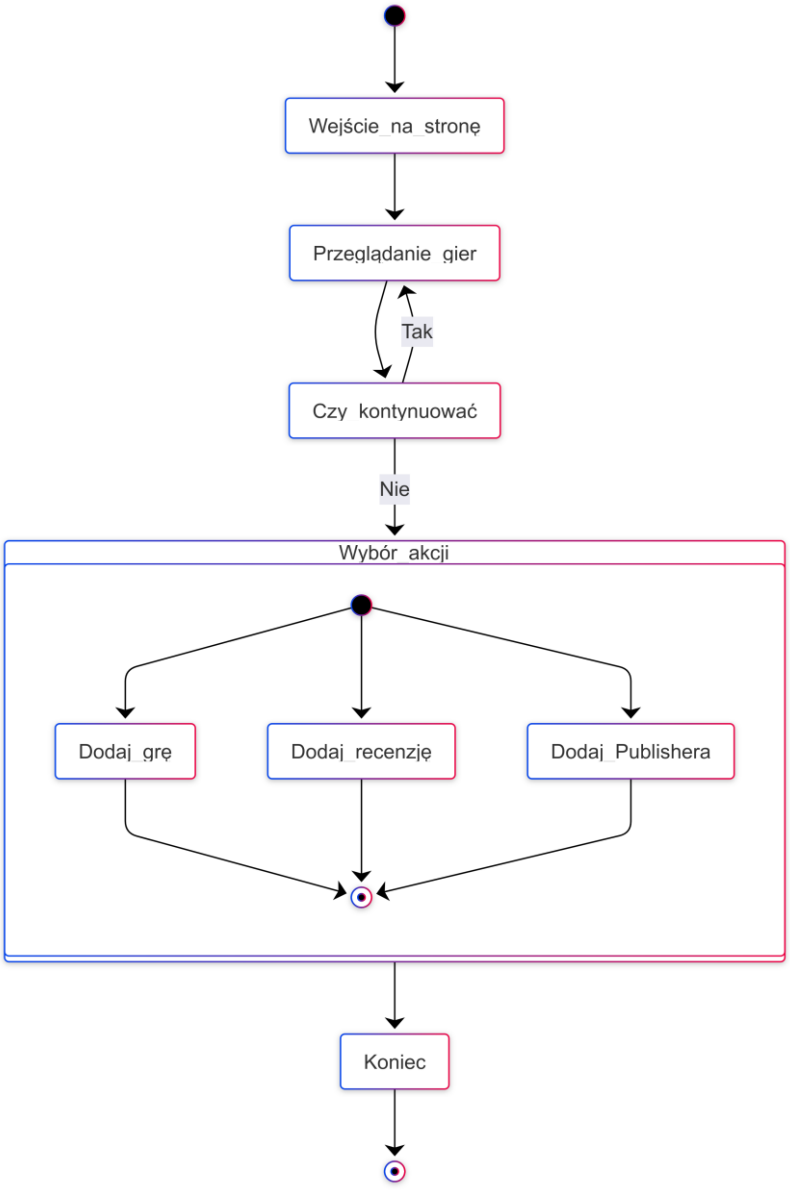
Baza danych MySQL przechowuje informacje o grach, wydawcach, recenzjach. Aplikacja wykorzystuje JPA/Hibernate do odwzorowania relacyjno-obiektowego.

2. Podstawowe funkcje programu w diagramach UML

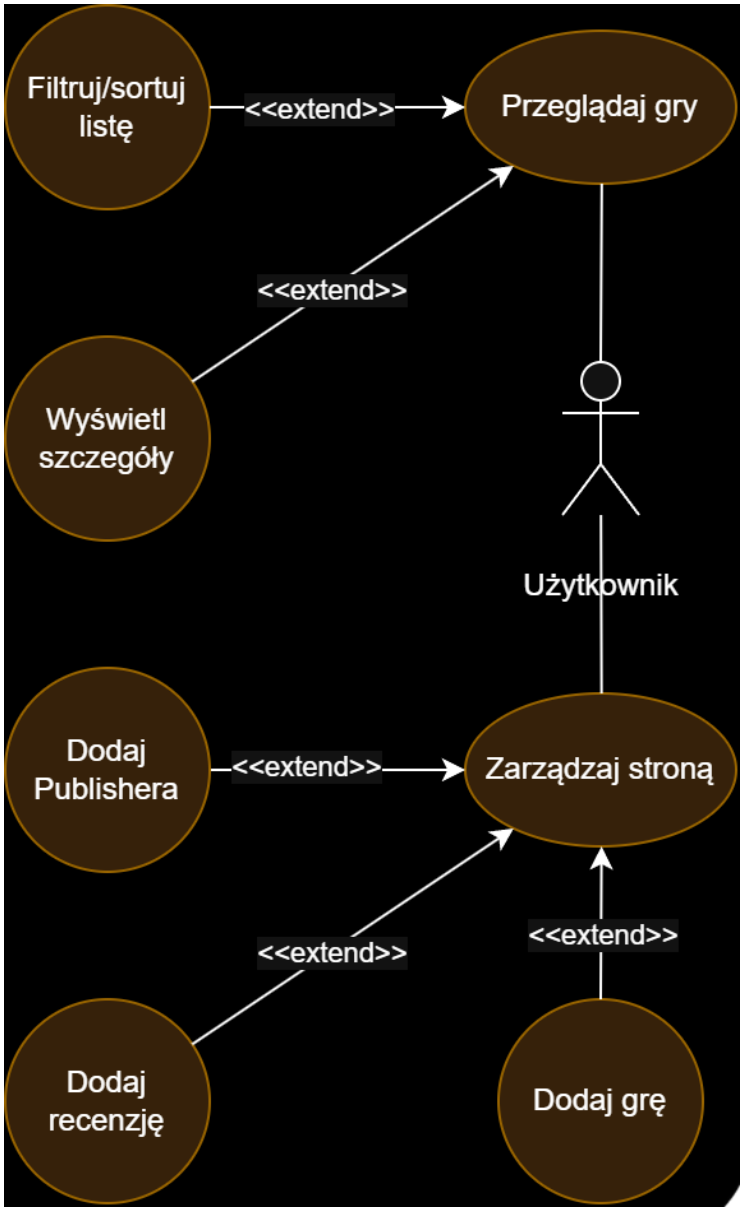
Poniższy opis odnosi się do czterech głównych diagramów UML oraz Mermaid przygotowanych w narzędziu **diagrams.net oraz mermaid.js.org**

2.1 Diagram czynności (Activity Diagram)

Przedstawia przepływ użytkownika: od wejścia na stronę.



2.2 Diagram przypadków użycia (Use Case)



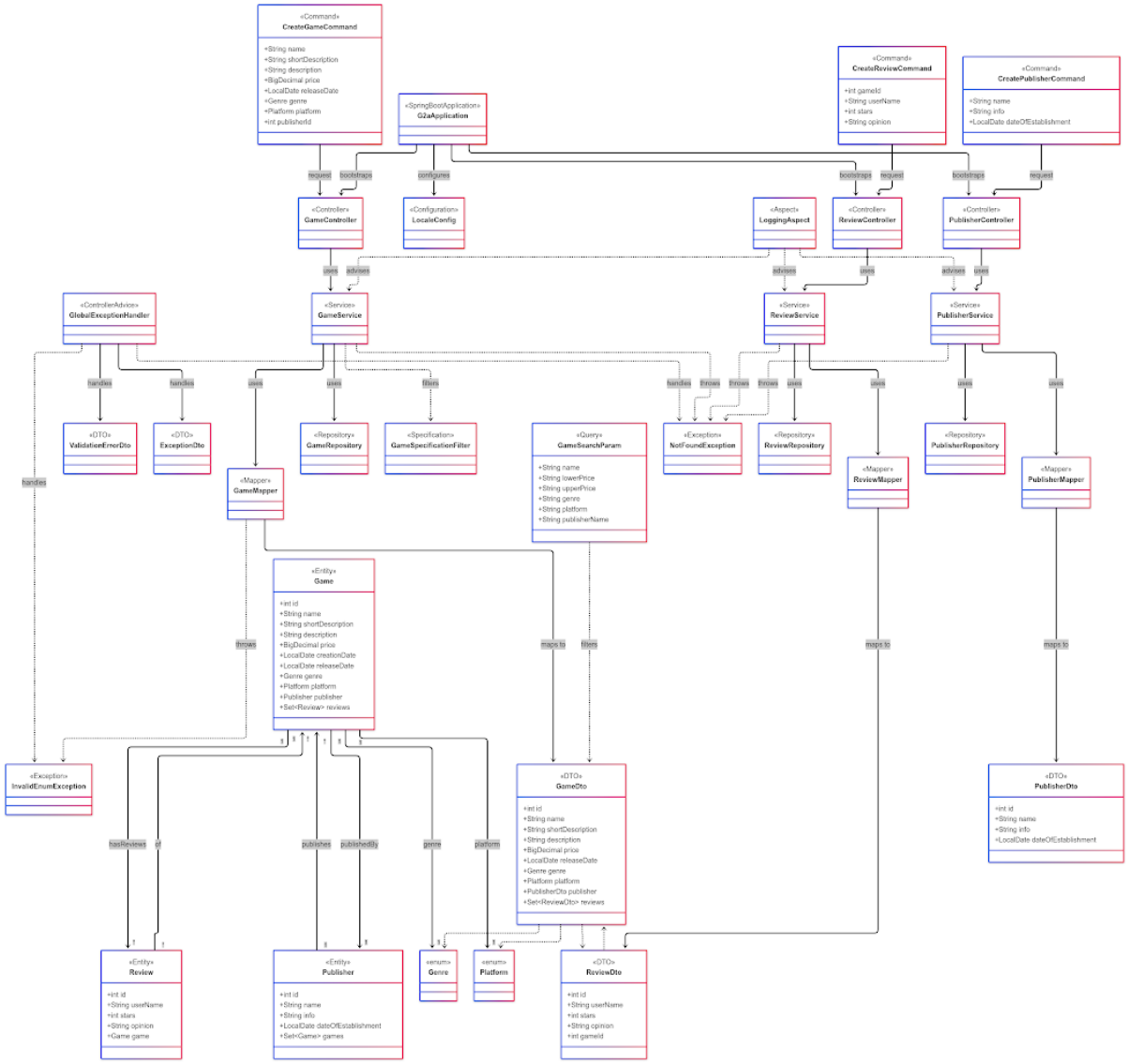
Aktorzy:

Użytkownik

Główne przypadki użycia:

Przeglądaj gry, Filtruj/sortuj listę, Wyświetl szczegóły, Dodaj.

2.3 Diagram klas (Class Diagram)

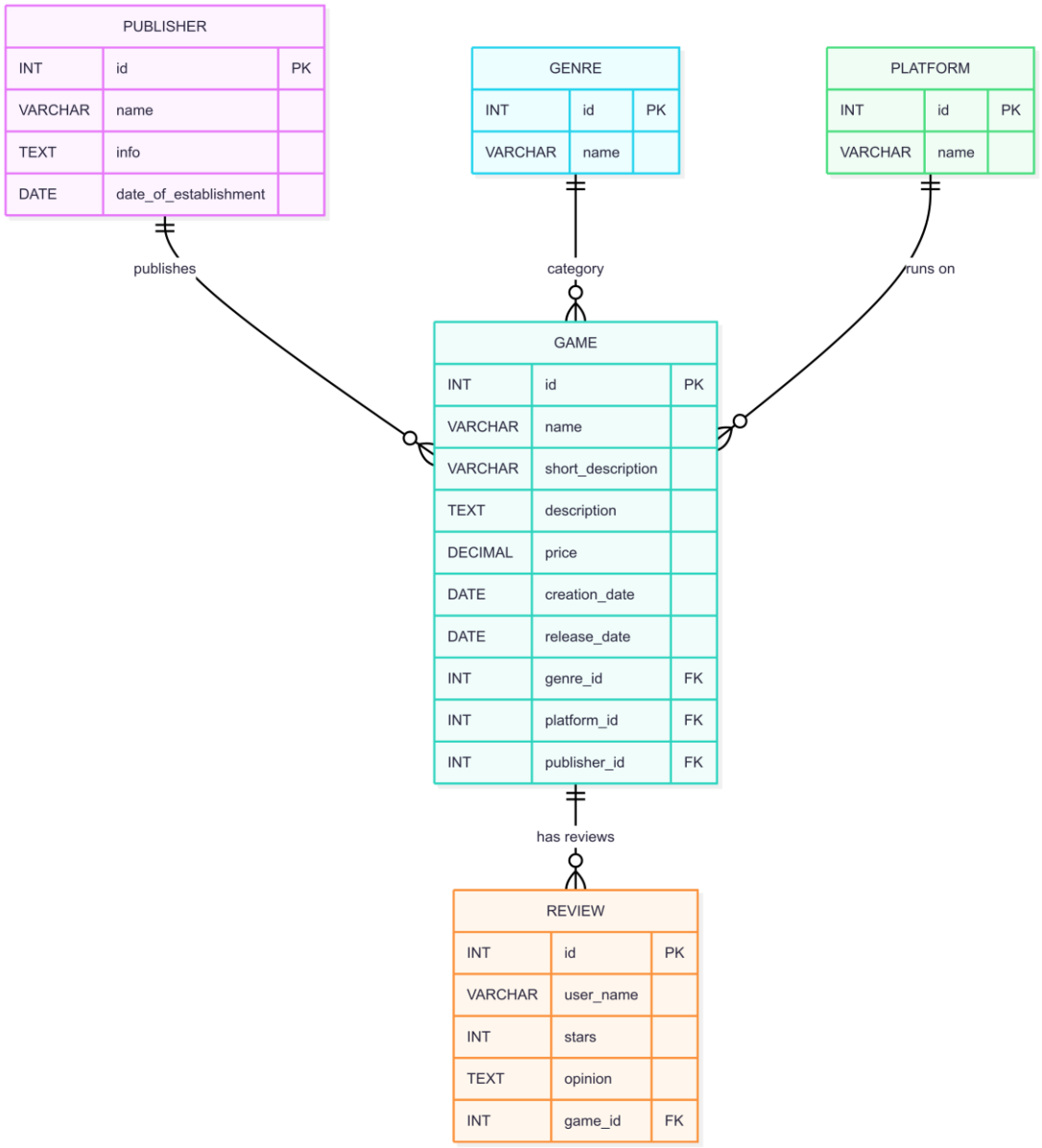


W warstwie modelu wyróżniono klasy:

- **Game** (id, name, short description, description, creation date, release date, genre, platform, publisher)
- **Publisher** (id, name, info, date of establishment)
- **Review** (id, user name ,stars , opinion, game)

W warstwie serwisu znajdują się klasy GameService, PublisherService, ReviewService, a dostęp do danych realizują repozytoria Spring Data (GameRepository itd.).

2.4 Diagram Schematu bazy danych



3. Wbudowane funkcje i klasy

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

Klasy encji są oznaczone adnotacją `@Entity` i mapowane na tabele MySQL, przy czym relacje odzwierciedlają zależności manytoone / onetomany między grami, wydawcami i recenzjami.

4. Dalsze możliwości rozwoju programu

- **Konteneryzacja** – obrazy Docker oraz orkiestracja Kubernetes.
- **Użytkownicy** – rejestracja, logowanie, zarządzanie profilem, uwierzytelnianie JWT.
- **Koszyk** – dynamiczne dodawanie/aktualizacja produktów, lokalne przechowywanie (localStorage), synchronizacja z bazą danych.
- **Płatności** – integracja z bramkami płatniczymi (np. Stripe, PayPal), obsługa transakcji, walidacja danych.
- **Historia zakupów** – przechowywanie danych w bazie, filtrowanie według daty/statusu, możliwość generowania faktur.

5. Opis użytych narzędzi i bibliotek

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

6. Uproszczona instrukcja obsługi + FAQ

6.1 Krótka instrukcja obsługi

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.

6.2 Najczęstsze pytania (FAQ)

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

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

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

4. Jak usunąć zastosowane filtry?
Kliknij przycisk „Wyczyść filtry” znajdujący się nad listą gier. Lista zostanie natychmiast zaktualizowana.

5. 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.