Національний технічний університет України

«Київський політехнічний інститут ім. Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

**Звіт**

Тема: “Розробка сервісів обробки інформації про ігри та налаштування взаємодіїі мікросервісів”

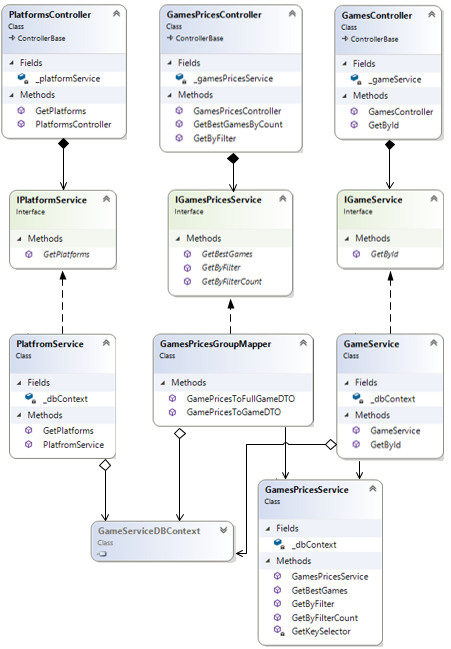
Виконав: Перевірено:

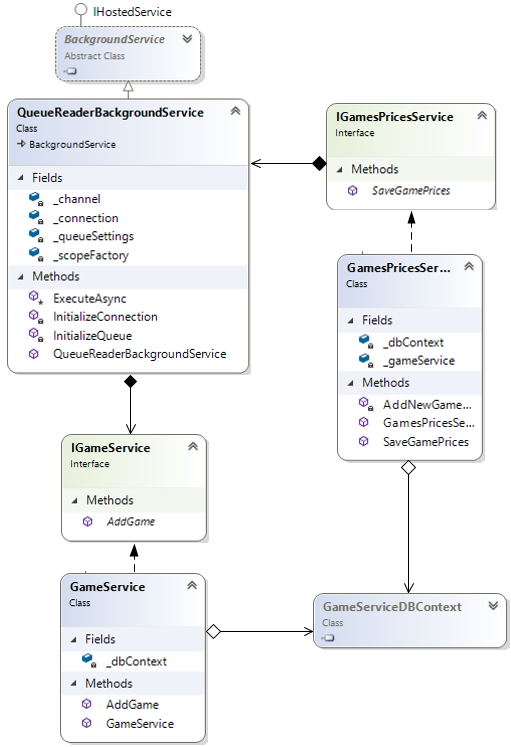
студент групи ІТ-71 асист. кафедри АУТС

Ковтко Юрій Галушко Дмитро

Андрійович Олександрович

Київ 2020

Одним з сервісів обробки інформації про ігри є сервіс доступу до інформації. Мікросервіс реалізований за стандартною багатошаровою архітектурою. Загальна структура мікросервісу зображена на рисунку.

Проектування сервісу збереження ігор полягало в розробці системи, яка має реалізований функціонал збереження ігор та має можливість постійно прослуховувати чергу повідомлень на наявність нової інформації про ігри. Загальна структура сервісу зображена на рисунку.

Для легкого розгортання мікросервісів використовувався docker-compose. Опис всіх сервісів вказано в файлі docker-compose:

version: "3.3"

services:

    frontend:

        image: frontend

        container\_name: frontend

        ports:

            - 2080:80

        build:

            context: ./game-sales-tracker

    rabbitmq:

        image: rabbitmq:3-management

        container\_name: rabbitmq

        ports:

            - 5672:5672

            - 15672:15672

    db:

        image: postgres:11

        container\_name: postgres

        ports:

            - 5433:5432

        environment:

            POSTGRES\_PASSWORD: salespassword

            POSTGRES\_USER: salesservice

            PGDATA : /var/lib/postgresql/data

        volumes:

            - data:/var/lib/postgresql/data

    ps-parser:

        image: ps-parser

        container\_name: ps-parser

        depends\_on:

            - rabbitmq

        build:

            context: ./Parsers/

            dockerfile: ./PSStoreParser/Dockerfile

        env\_file:

            - ./env/PSStoreParser.env

    steam-parser:

        image: steam-parser

        container\_name: steam-parser

        depends\_on:

            - rabbitmq

        build:

            context: ./Parsers/

            dockerfile: ./SteamParser/Dockerfile

        env\_file:

            - ./env/SteamParser.env

    uplay-parser:

        image: uplay-parser

        container\_name: uplay-parser

        depends\_on:

            - rabbitmq

        build:

            context: ./Parsers/

            dockerfile: ./UplayParser/Dockerfile

        env\_file:

            - ./env/UplayParser.env

    nintendo-parser:

        image: nintendo-parser

        container\_name: nintendo-parser

        depends\_on:

            - rabbitmq

        build:

            context: ./Parsers/

            dockerfile: ./NintendoParser/Dockerfile

        env\_file:

            - ./env/NintendoParser.env

    epic-parser:

        image: epic-parser

        container\_name: epic-parser

        depends\_on:

            - rabbitmq

        build:

            context: ./Parsers/EpicGamesParser

    gateway-api:

        image: gateway-api

        container\_name: gateway-api

        ports:

            - 5080:80

        depends\_on:

            - db

        build:

            context: ./GameSalesApi/

            dockerfile: ./GameSalesApi/Dockerfile

        env\_file:

            - ./env/GatewayApi.env

    games-saver:

        image: games-saver

        container\_name: games-saver

        depends\_on:

            - db

            - rabbitmq

        build:

            context: ./GamesServices/

            dockerfile: ./GamesSaver/Dockerfile

        env\_file:

            - ./env/GamesSaver.env

        restart: on-failure

    games-provider:

        image: games-provider

        container\_name: games-provider

        ports:

            - 6080:80

        depends\_on:

            - db

        build:

            context: ./GamesServices/

            dockerfile: ./GamesProvider/Dockerfile

        env\_file:

            - ./env/GamesProvider.env

volumes:

    data:

        driver: local