Міністерство освіти і науки України

Національний технічний університет України «КПІ ім. Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Лабораторна робота

з дисципліни «[Розподілені інформаційні системи](https://boldak.github.io/dis-edu/)»

Тема: «Квиткова каса. API сервіс і UI інтерфейс»

|  |  |
| --- | --- |
| Виконав:  студент групи ІП-з01  Максиміхін О.В.  Дата здачі \_\_\_\_\_\_\_\_\_\_\_\_  Захищено з балом \_\_\_\_\_\_\_\_\_\_\_ | Перевірив:  доц. Болдак А. О. |

Київ 2023

Зміст

[Вступ. 3](#_Toc135782624)

[Вимоги. 4](#_Toc135782625)

[API сервіс. 6](#_Toc135782626)

[Контролер Movies 6](#_Toc135782627)

[Контролер Tickets 9](#_Toc135782628)

[Контролер MovieTicket 12](#_Toc135782629)

[Контролер MoviesStats 13](#_Toc135782630)

[Schemas 14](#_Toc135782631)

[UI інтерфейс. 15](#_Toc135782632)

[Програма випробувань. 17](#_Toc135782633)

[Лістинг коду. 19](#_Toc135782634)

# Вступ.

Програмний модуль «Квиткова каса» призначений для опрацювання операцій із фільмами і квитками на фільми.

Програмний модуль складається із API сервісу розробленого на технології ASP.NET Core API та користувацького WEB інтерфейсу розробленого на технології REACT.

Призначення програмного модуля.

Для обробки операції із фільмами програмний модуль має виконувати операції:

додавання нових вистав фільмів,

коригування інформації у створених фільмах,

видалення вистав фільмів,

відображення наявних вистав фільмів.

Для обробки операції із квитками програмний модуль має виконувати операції:

додавання квитків до нових фільмів,

коригування інформації про квитки (придбання квитків),

видалення квитків,

відображення інформації про стан квитків (доступні для продажу, продані).

WEB інтерфейс надсилає запити/отримує результати запитів від API служби і відображає результати обробки запитів на екран пристрою.

# Вимоги.

Програмний модуль ASP.NET Core API служби є RESTful сервісом, що повинен задовольняти наступним вимогам.

Функціонування програмного модуля не має призводити до збою (фатального порушення роботи системи) та видавати діагностику стану системи та повідомлення про будь-які помилки, що виникли:

у відповідь на запит "Get all movies" повинен повертати код відповіді – 200 і колекцію Movie;

у відповідь на запит "Get movie by ID" повинен повертати код відповіді – 200 і повертати об’єкт Movie за обраним ID;

у відповідь на запит "Create movie" повинен повертати код відповіді – 200, створений об’єкт Movie і записувати створений Movie в базу даних ComosDB;

у відповідь на запит "Update movie" повинен повертати код відповіді – 200, змінений об’єкт Movie і записувати зміни в поля Movie в базі даних ComosDB;

у відповідь на запит "Delete movie by ID" повинен повертати код відповіді – 200, видалений об’єкт Movie і видаляти запис Movie в базі даних ComosDB;

у відповідь на запит "Get all tickets" повинен повертати код відповіді – 200 і колекцію Tickets;

у відповідь на запит "Get ticket by ID" повинен повертати код відповіді – 200 і повертати об’єкт Ticket за обраним ID;

у відповідь на запит "Create ticket" повинен повертати код відповіді – 200, створений об’єкт Ticket і записувати створений Ticket в базу даних MS SQL Server;

у відповідь на запит "Update ticket" повинен повертати код відповіді – 200, змінений об’єкт Ticket і записувати зміни в поля Ticket в базі даних MS SQL Server;

у відповідь на запит "Delete ticket by ID" повинен повертати код відповіді – 200, видалений об’єкт Ticket і видаляти запис Ticket в базі даних MS SQL Server;

у відповідь на запит "Create movieTicket chunk" повинен повертати код відповіді – 200, в Blob Storage у сервісі Azure створювати файл із назвою, що складається із Movie ID і Ticket ID в форматі "Movie ID"\_" Ticket ID" і повертати назву створеного файлу;

у відповідь на запит "Get movieTicket chunk by movieID" повинен повертати код відповіді – 200, і колекцію назв файлів із Blob Storage у сервісі Azure в яких присутній Movie ID;

у відповідь на запит "Get new added movies" повинен повертати код відповіді – 200, і колекцію Movie ID по всім створеним фільмам;

UI повинна бути побудована за допомогою framework REACT і відповідати вимогам:

1. responsive design;
2. період затримки відображенні відповіді від API сервісу не більше 2 секунд;
3. відсутність візуальних артефактів;
4. запуск в браузерах Chrome, Edge, Mozila;
5. наявність тем: темна, світла, синя, зелена.

# API сервіс.

API сервіс надає доступ до функціоналу по обробці операцій із виставами фільмів і квитків на фільми.

Опис API сервісу здійснено за допомогою специфікації OpenAPI (Swagger)

Base URLs:

<http://localhost:5127>

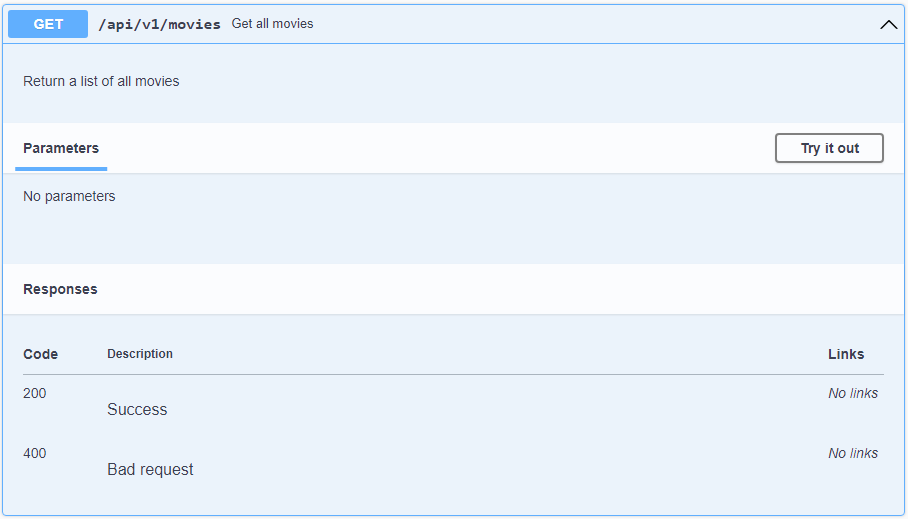
[Terms of service](https://example.com/terms)

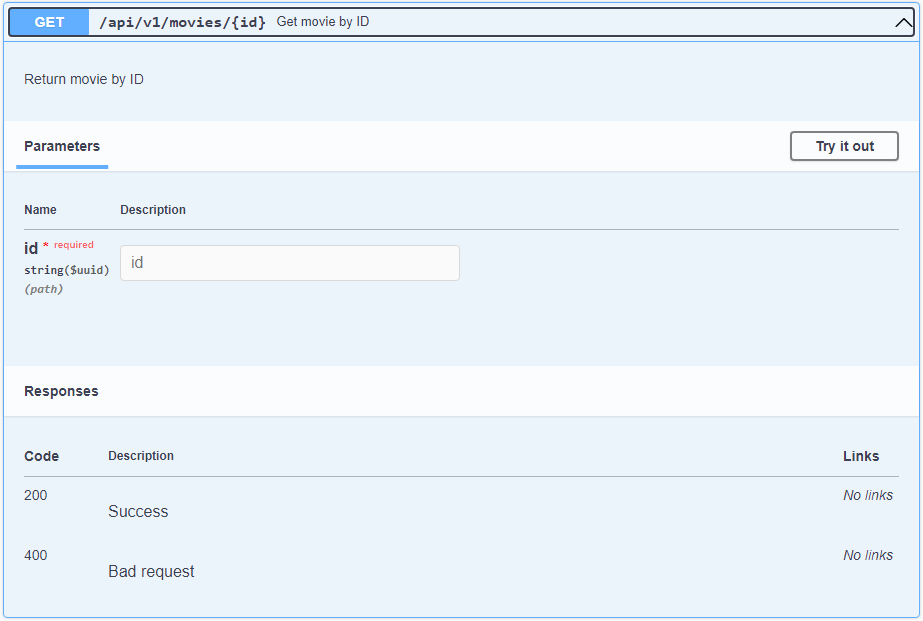
[Contact - Website](https://example.com/contact)

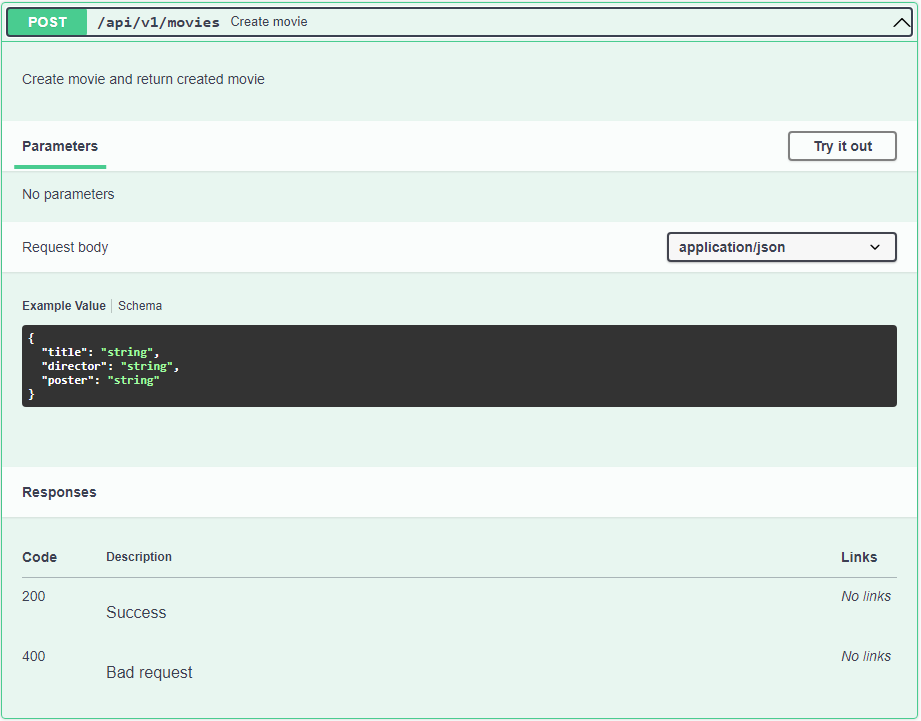
[License](https://example.com/license)

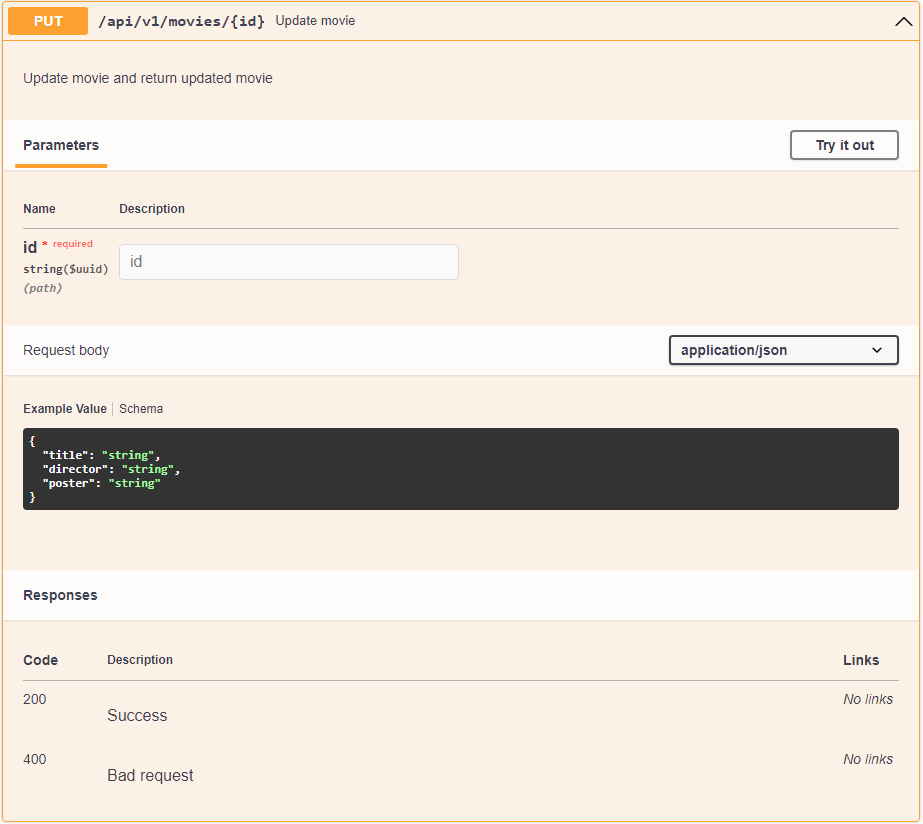
Загальна інформація.

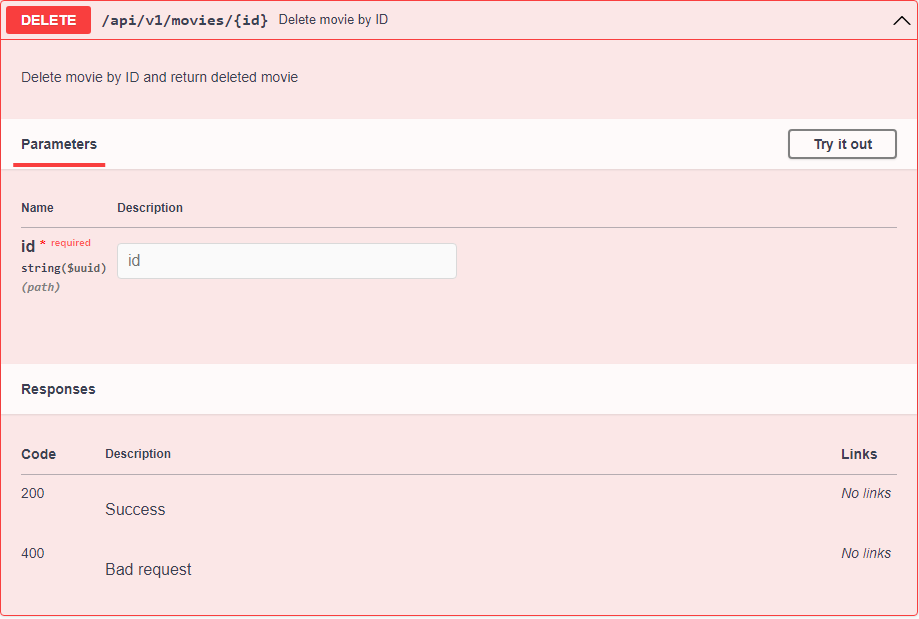
## Контролер Movies



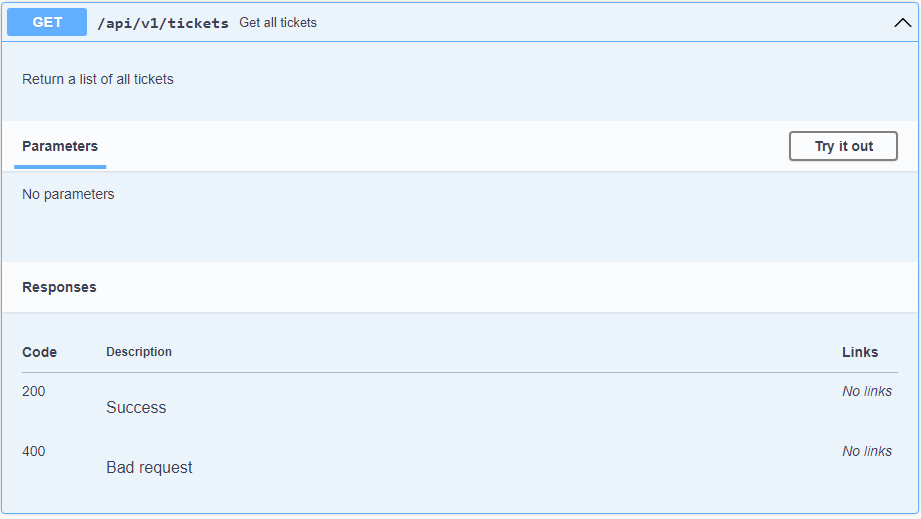


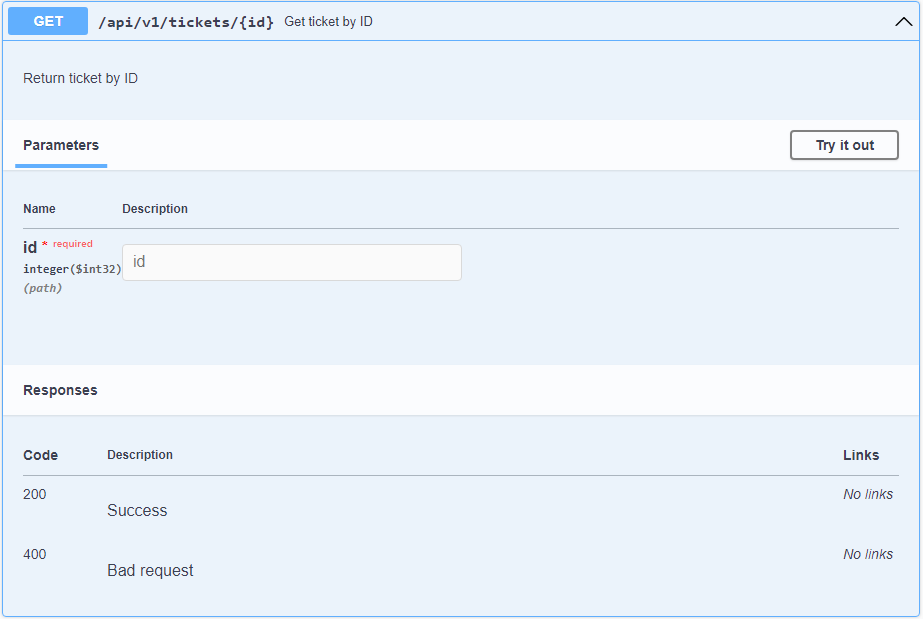


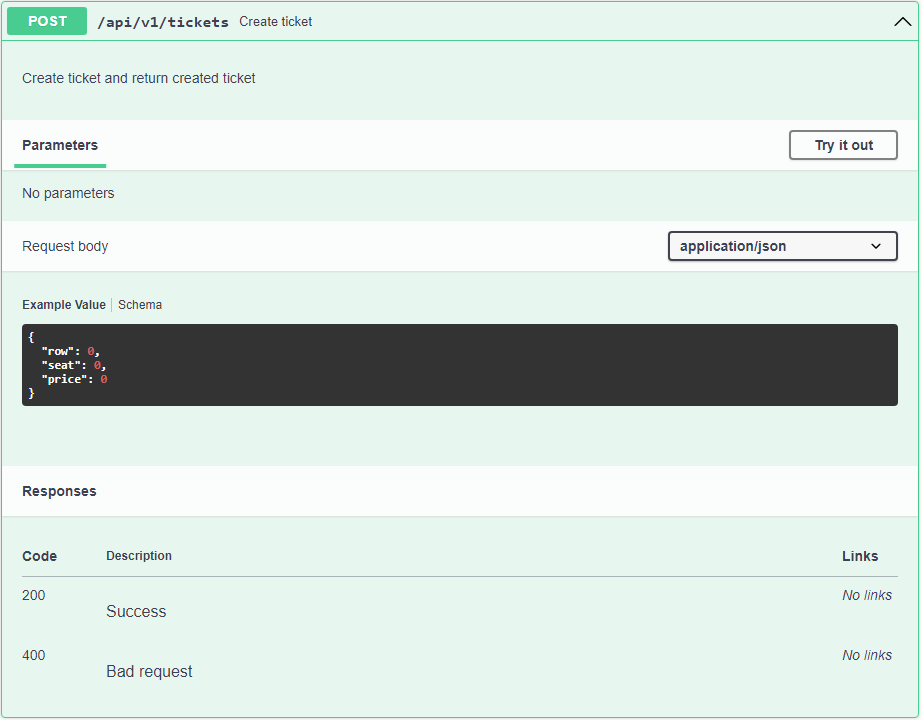


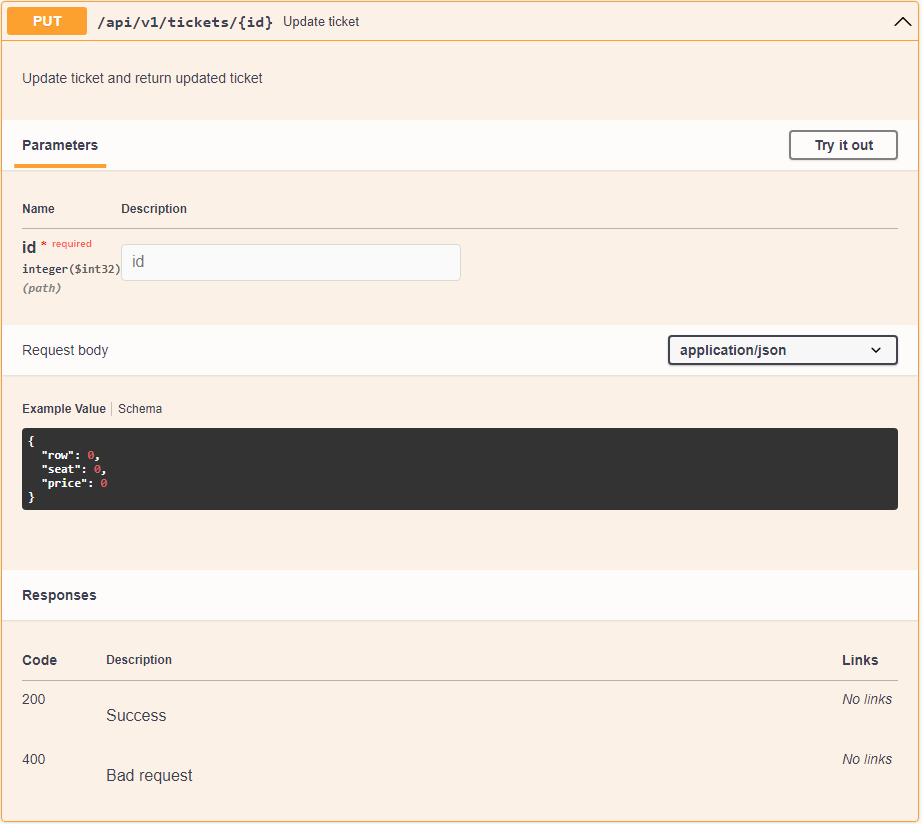


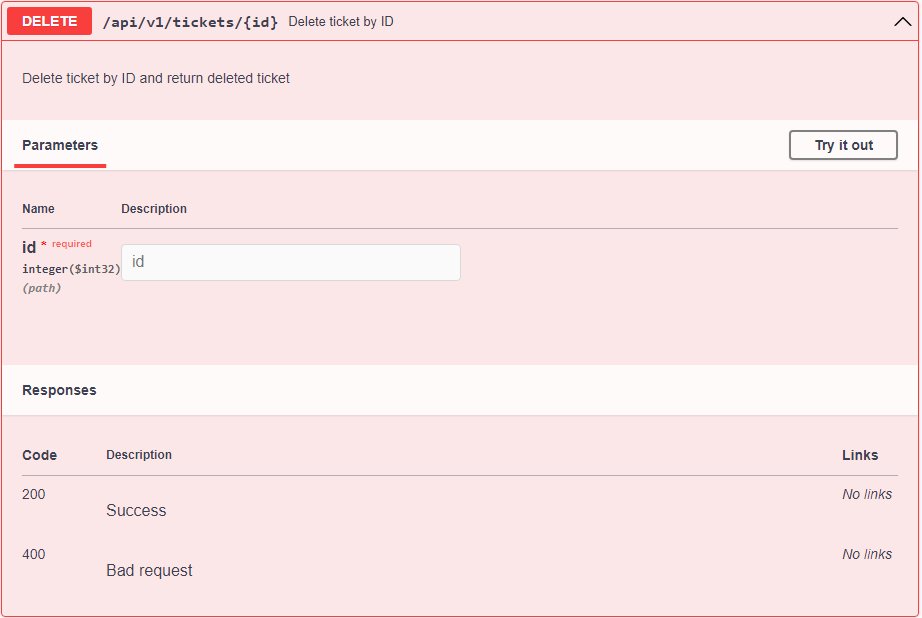
## Контролер Tickets



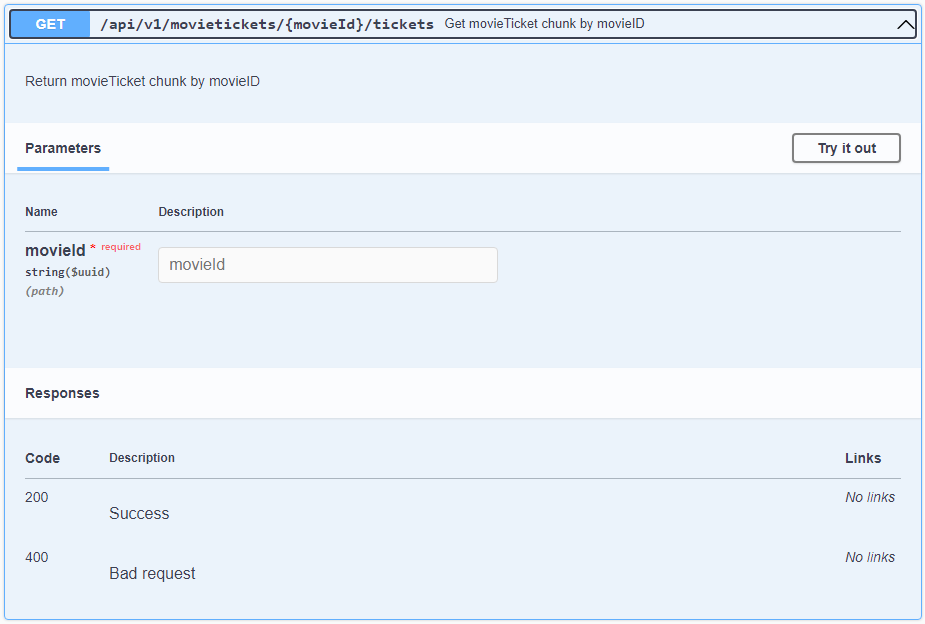


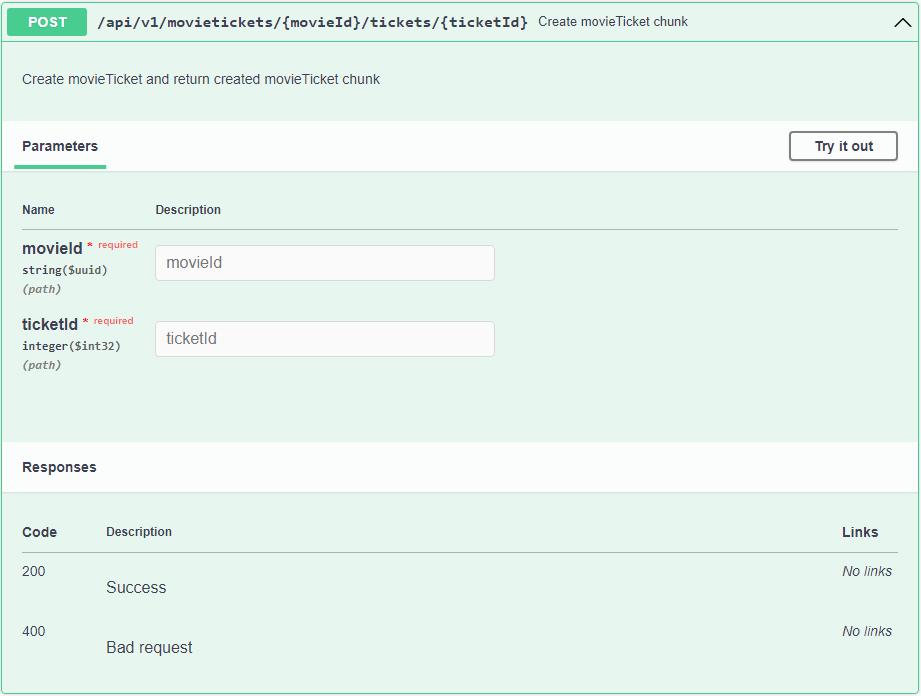




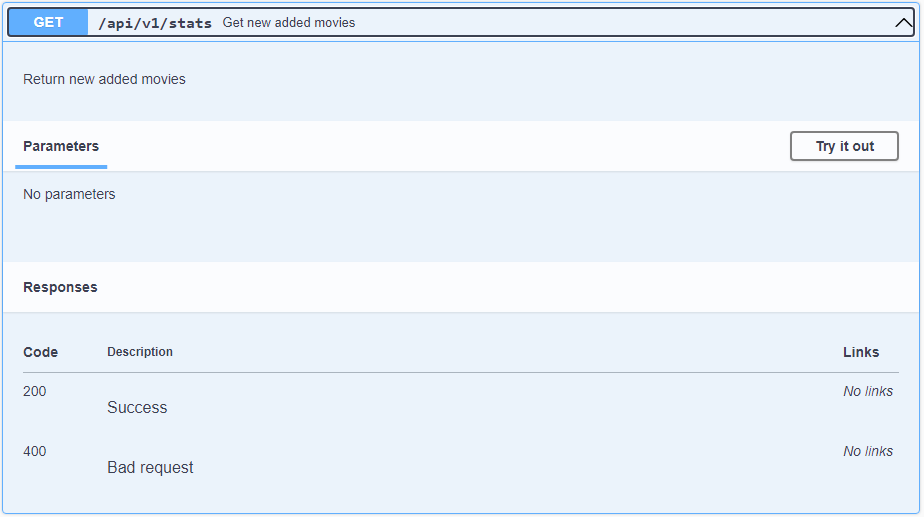


## Контролер MovieTicket

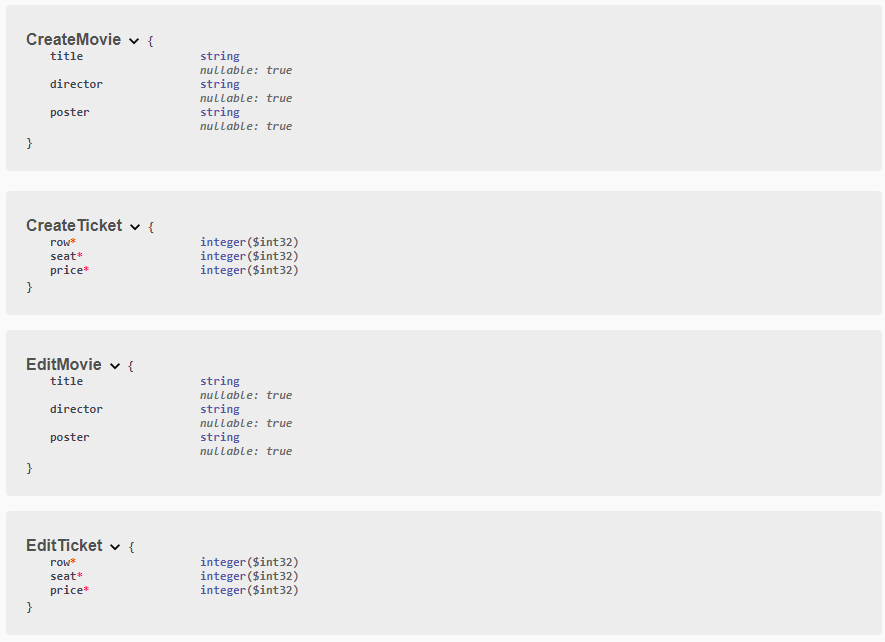




## Контролер MoviesStats



## Schemas

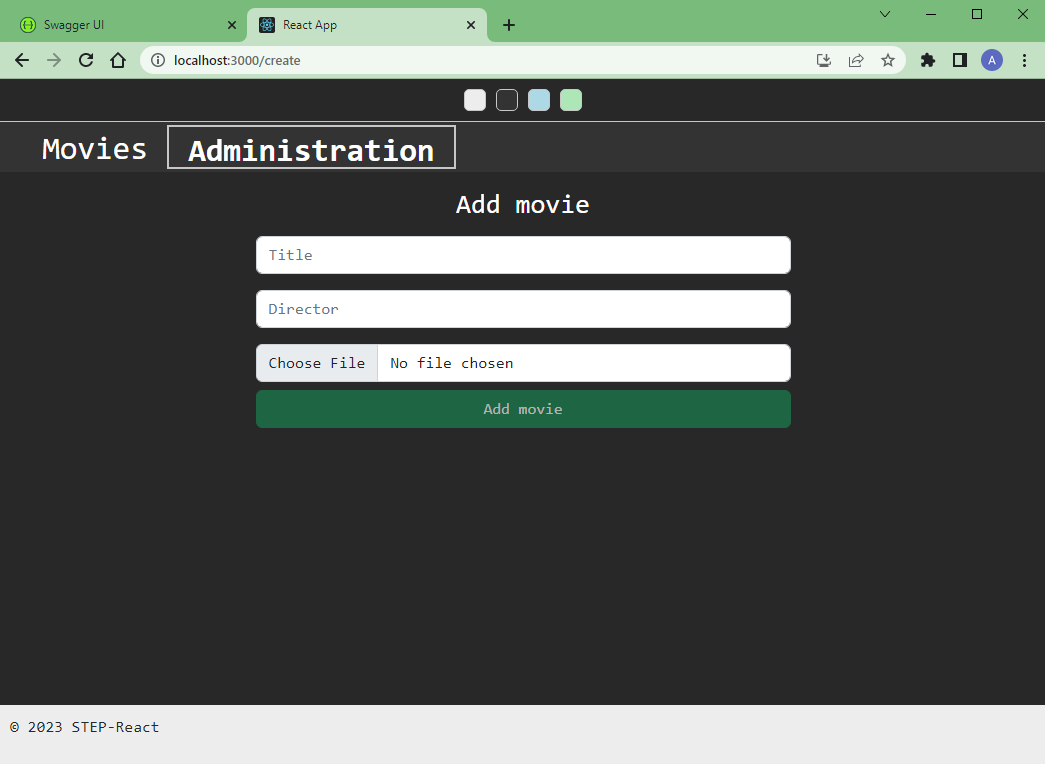


# UI інтерфейс.

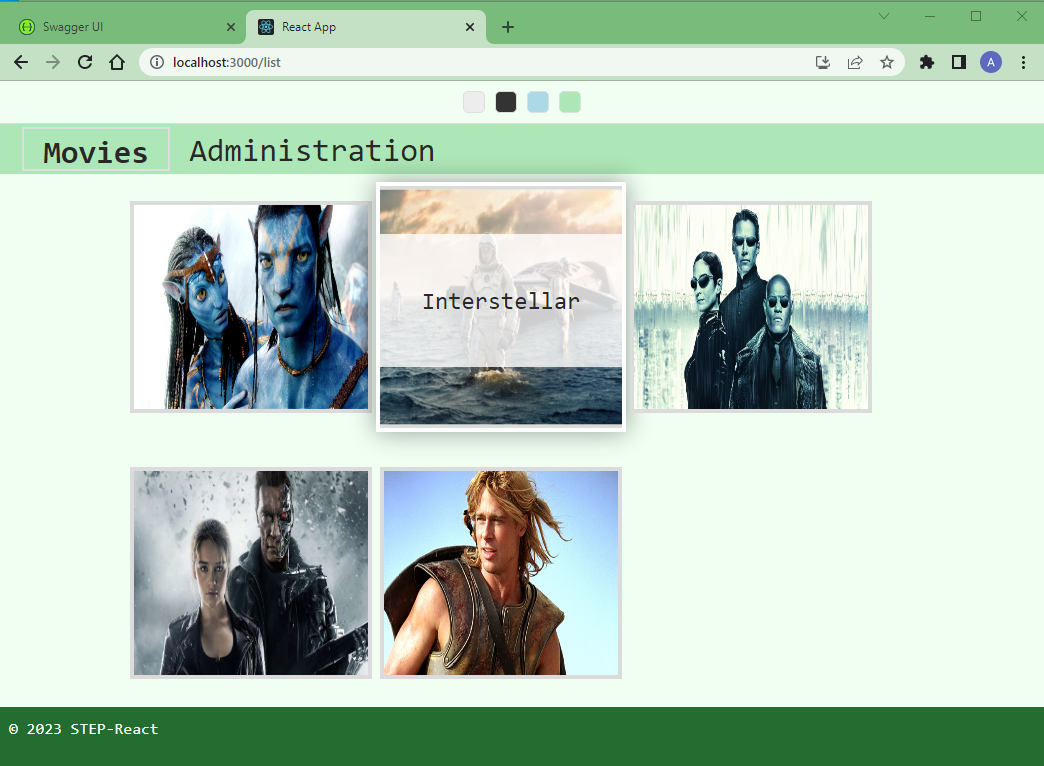
UI інтерфейс є WEB додатком.

Відображення роботи UI WEB інтерфейсу.

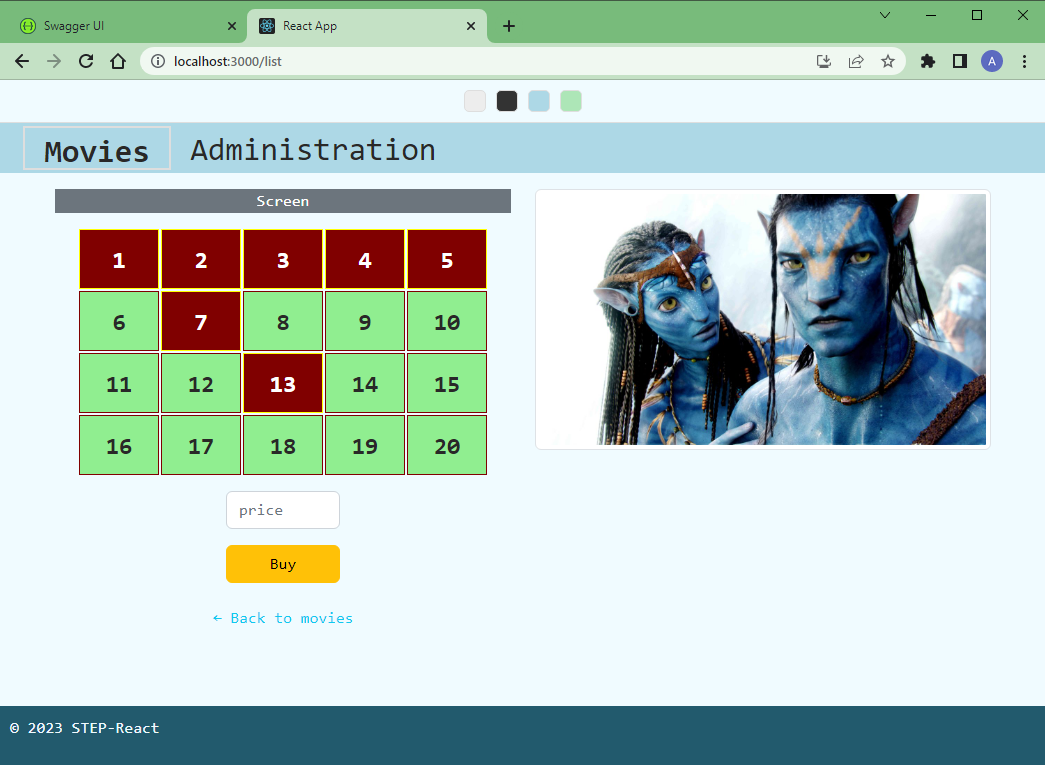
Панель роботи адміністратора (темна тема)



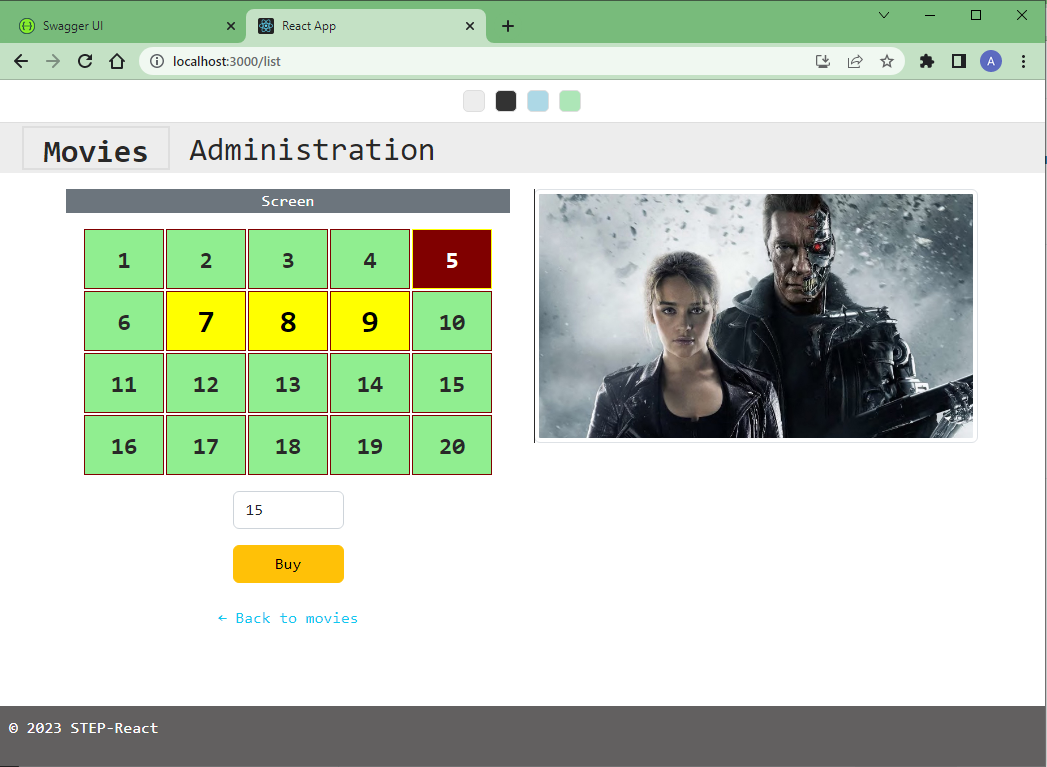
Відображення переліку фільмів (зелена тема)



Перегляд інформації про фільм і наявні/продані квитки (синя тема)



Придбання квитків (світла тема)

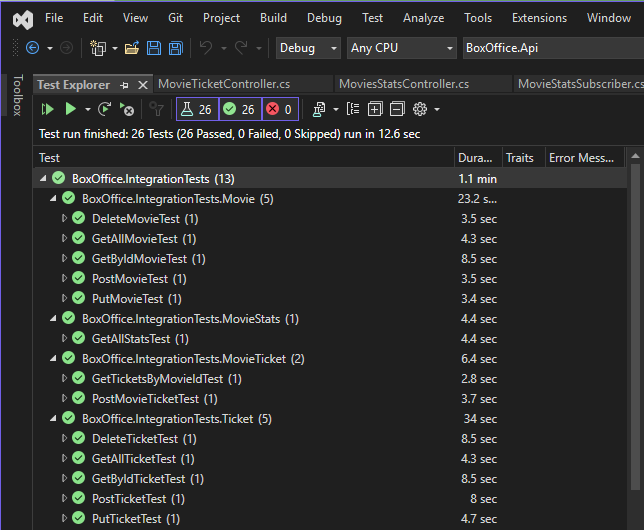


# Програма випробувань.

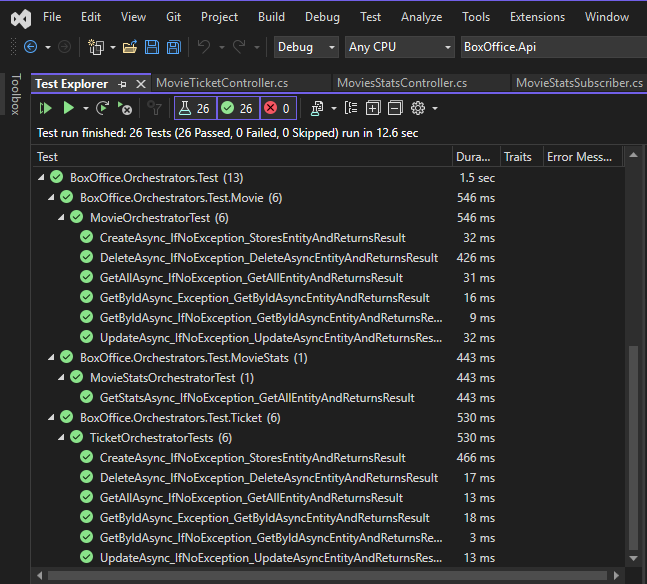
Програма випробувань передбачає проходження API сервісом інтеграційних тестів на кожний метод контролера і Unit тестів на кожний метод оркестраторів.

Результати проходження тестів.

Інтеграційні тести.



Unit тести



# Лістинг коду.

Програмний код застосунку викладений на GitHUB репзиторії за посиланням:

<https://github.com/oleksandr-maksymikhin/Distributed_Information_Systems>