

Міністерство освіти і науки України

Національний технічний університет України «Київський політехнічний
інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи №3 з дисципліни

«Сучасні технології розробки WEB-застосунків на платформі
Microsoft.NET»

«Проектування REST веб-API»

Варіант 3

Виконав студент ІП-13 Макарчук Лідія Олександрівна

(шифр, прізвище, ім'я, по батькові)

Перевірив

Бардін В.

(прізвище, ім'я, по батькові)

Київ 2023

Лабораторна робота №3

Варіант 3

Тема: Проектування REST веб-API.

Постановка задачі

Теоретична частина:

1. Ознайомитися з основами створення REST веб-API та методологією С4 для відображення архітектури системи.
2. Ознайомитися з основами створення ER-діаграм для представлення структури бази даних.

Практична частина:

1. З дотриманням вимог REST-у спроектувати веб-API для обраної(згідно варіанту) доменної області, використовуючи методологію С4 для створення діаграми архітектури системи.
2. Створити ER-діаграму для DAL (Data Access Layer), яка відображатиме структуру бази даних веб-API.
3. Оформити спроектоване рішення у вигляді звіту до лабораторної роботи.

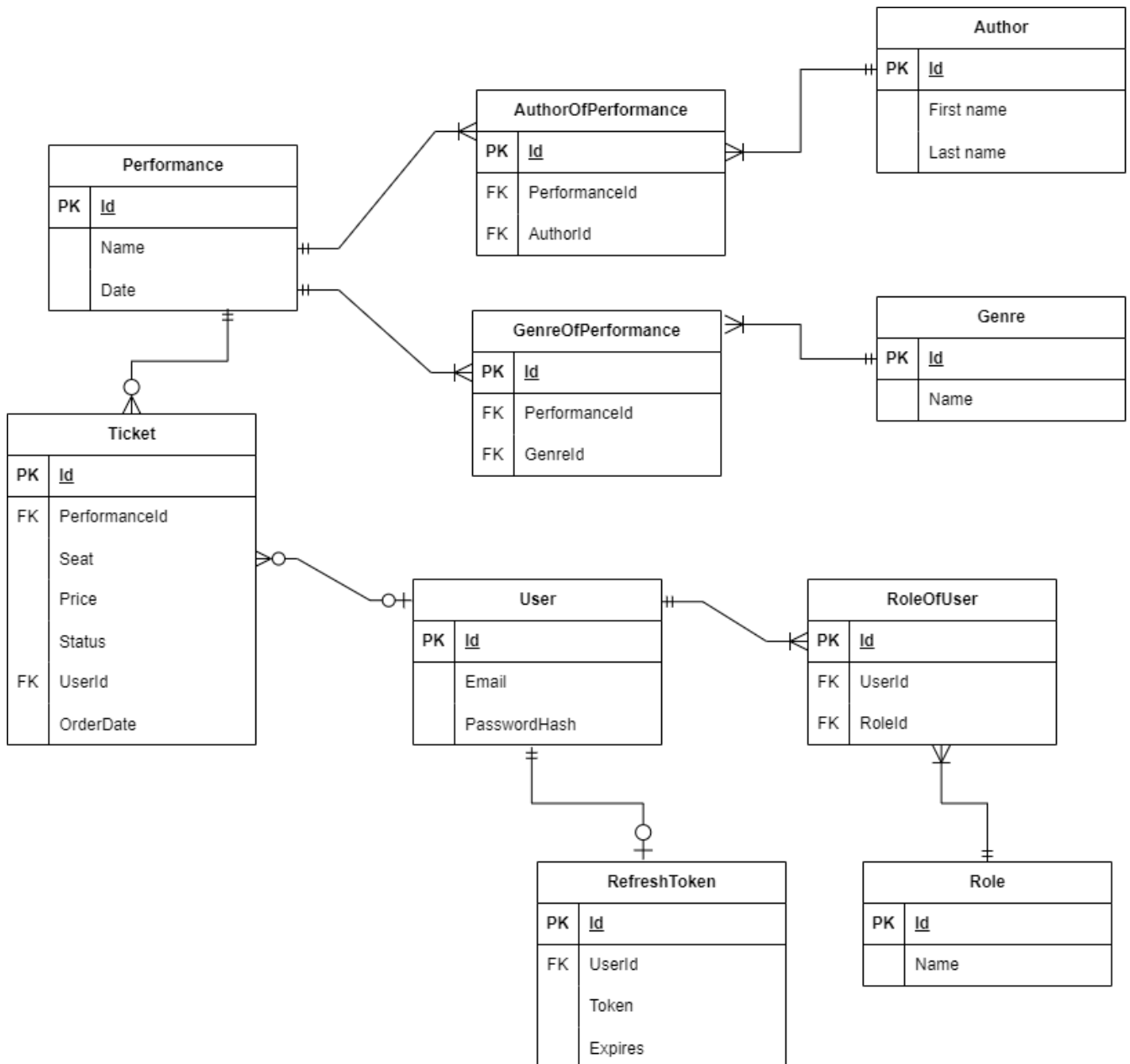
Документація:

1. Підготувати документацію(звіт до ЛР), яка включатиме опис веб-API, а також структуру бази даних з урахуванням ER-діаграми.

3	Театральна каса. Розповсюдження квитків	<div>1. Афіша вистав надає функції пошуку потрібної вистави за автором, назвою, жанром та датою.</div> <div>2. Кожній виставі відповідає кількість квитків різної вартості.</div> <div>3. Квитки можна безпосередньо продати чи попередньо забронювати та згодом перевести їх у стан проданих.</div> <div>Функціональні вимоги:</div> <div>1. Ведення афіши вистав;</div> <div>2. Продаж квитків</div>
---	--	---

Виконання завдань

ER-diagram:



Тип та опис сутностей:

Performance

- Id (PK) – bigint, Ідентифікатор
- Name – varchar(60), Назва вистави
- Date – datetime, Дата та час вистави

Author

- Id (PK) – bigint, Ідентифікатор
- FirstName – varchar(40), Ім'я автора
- LastName – varchar(40), Прізвище автора

Genre

- Id (PK) – smallint, Ідентифікатор
- Name – varchar(60), Назва жанру

AuthorOfPerformance

- Id (PK) – bigint, Ідентифікатор
- PerformanceId (FK) – bigint, Ідентифікатор вистави
- AuthorId (FK) – bigint, Ідентифікатор автора

GenreOfPerformance

- Id (PK) – bigint, Ідентифікатор
- PerformanceId (FK) – bigint, Ідентифікатор вистави
- GenreId (FK) – smallint, Ідентифікатор жанру

Ticket

- Id (PK) – bigint, Ідентифікатор
- PerformanceId (FK) – bigint, Ідентифікатор вистави
- Seat – smallint – номер місця
- Price – decimal(8, 2)
- Status – enum(“available”, “reserved”, “sold”)
- UserId (FK) – bigint, Ідентифікатор користувача
- OrderDate – datetime – час бронювання або купівлі квитка користувачем

User

- Id (PK) – bigint, Ідентифікатор
- Email – varchar(60), Електронна адреса
- PasswordHash - varchar(255), Хеш пароля

Role

- Id (PK) – tinyint, Ідентифікатор
- Name – varchar(60), Назва ролі

RoleOfUser

- Id (PK) – bigint, Ідентифікатор
- UserId (FK) – bigint, Ідентифікатор користувача
- RoleId (FK) – tinyint, Ідентифікатор ролі користувача

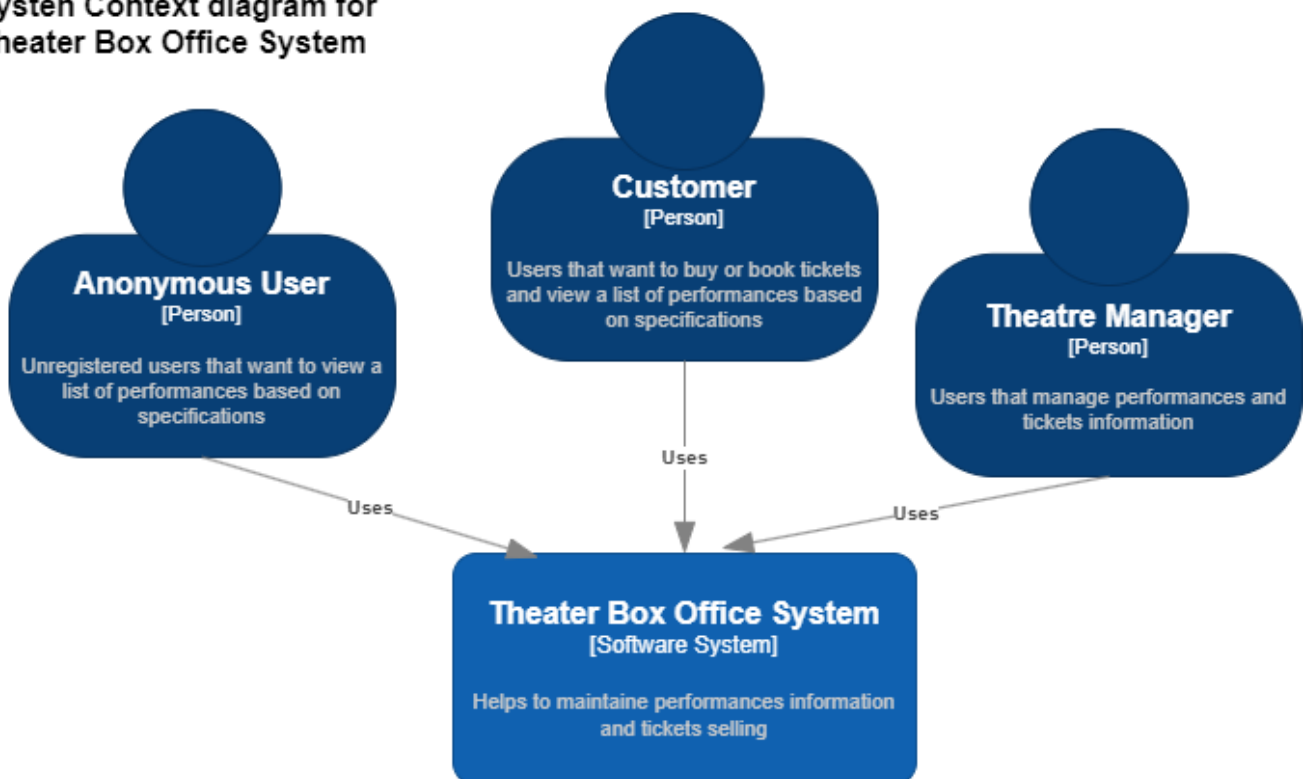
RefreshToken

- Id (PK) – bigint, Ідентифікатор
- UserId (FK) – bigint, Ідентифікатор користувача
- Token – varchar, значення токена
- Expires – datetime; час, після якого токен не є валідним

C4 diagram

1-й рівень

System Context diagram for
Theater Box Office System

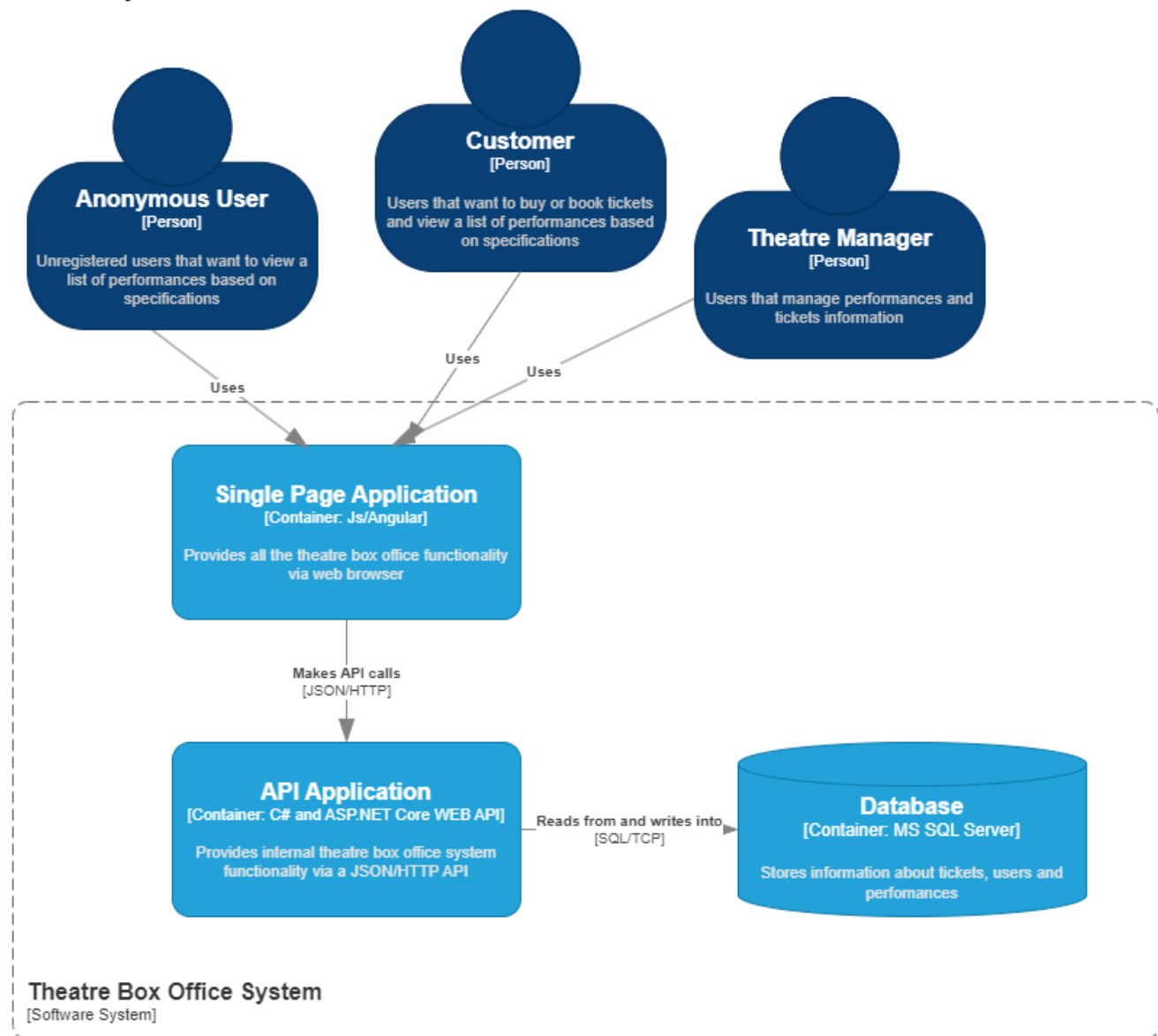


Системою можуть користуватися 3 види користувачів:

- Anonymous User – незареєстрований користувач, що може переглянути вистави за вказаними фільтрами;
- Customer – зареєстрований користувач, який може не лише переглядати список вистав, але і бронювати та купувати квитки;
- Theatre Manager – користувачі, які мають можливість редагувати, створювати, видаляти та переглядати дані вистав та квитків.

2-й рівень

Containers diagram for Theater Box Office System

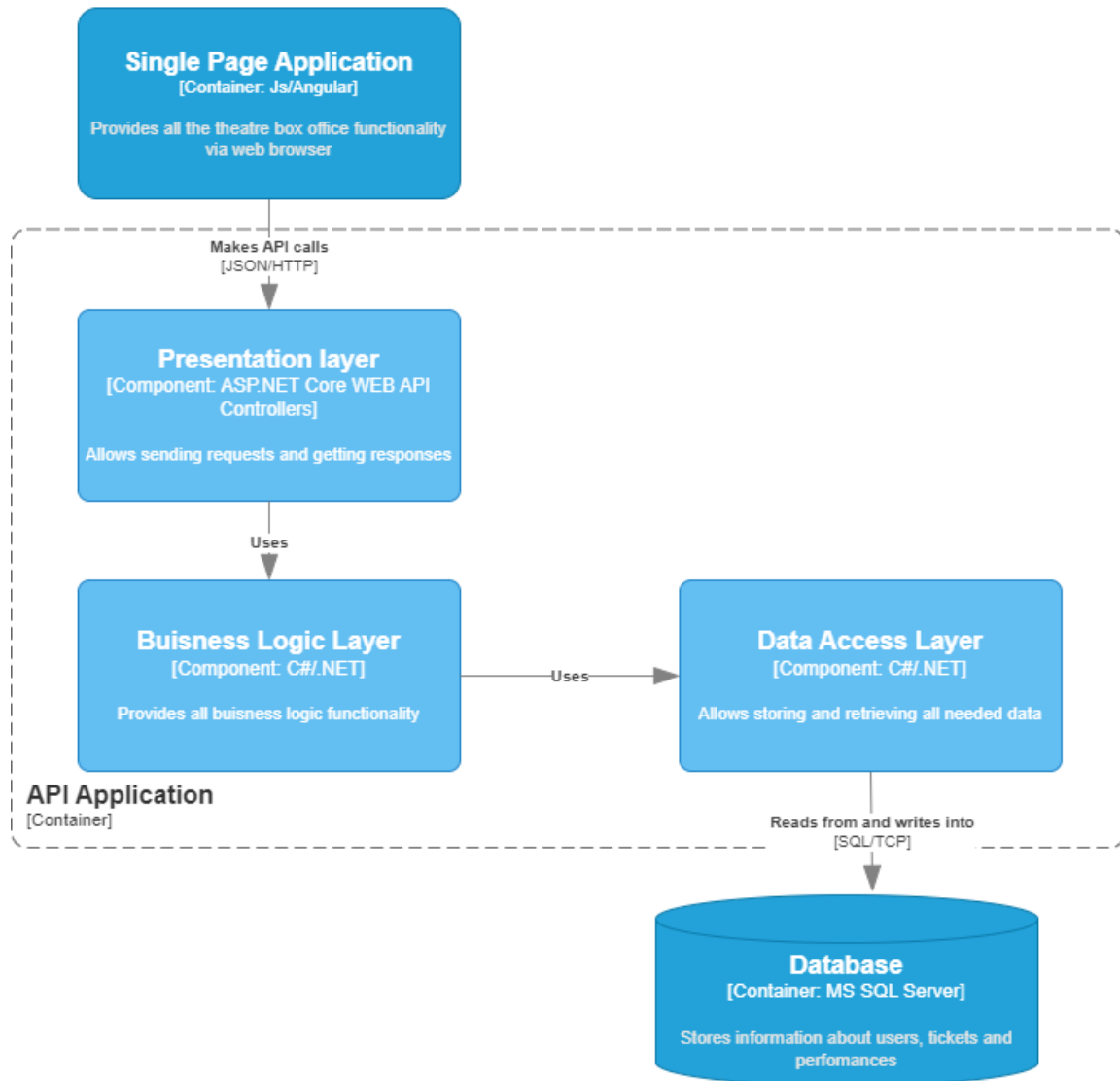


Система складається з 3-х контейнерів:

- Single Page Application – надає функціонал системи через браузер;
- API Application – надає функціонал системи, використовуючи сутності у форматі JSON та HTTP протокол;
- Database – дозволяє зберігати сутності та надає доступ до них.

3-й рівень

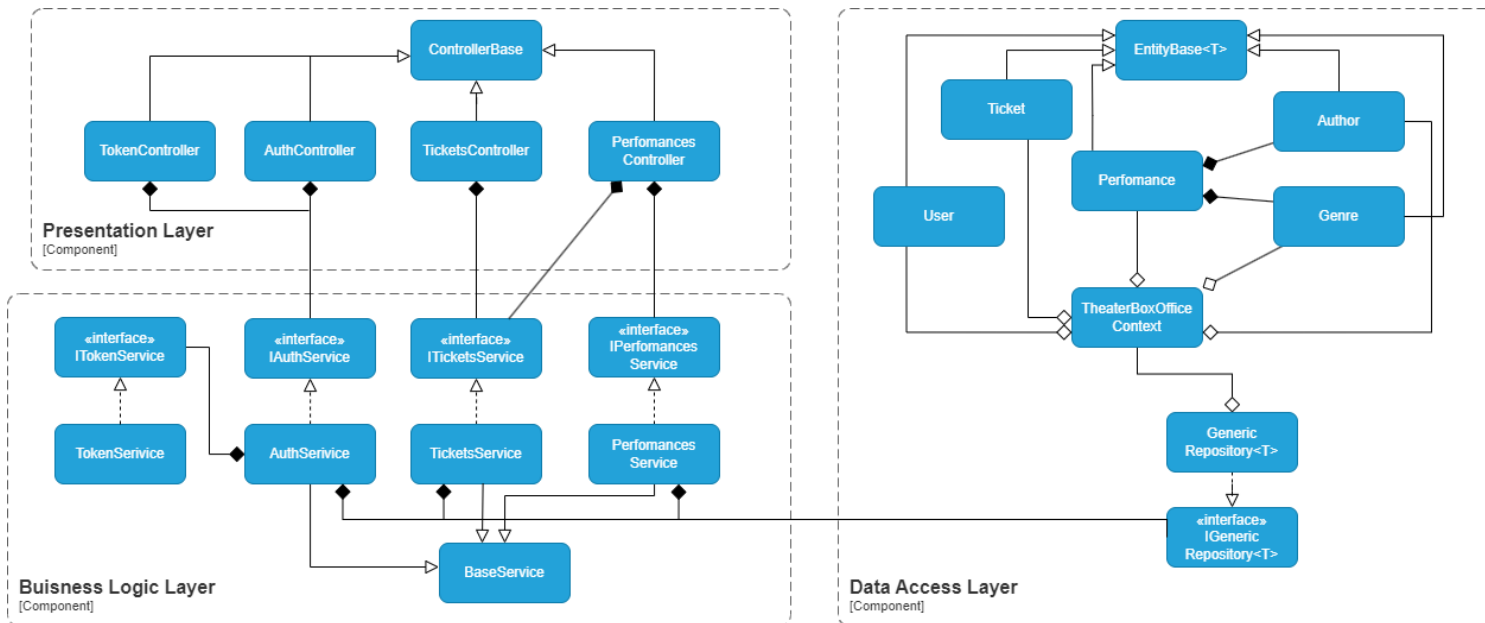
Components diagram for Theater Box Office System - API Application



API Application містить наступні 3 шари:

- Presentation Layer – вхідна точка взаємодії із API;
- Business Logic Layer – містить основний функціонал;
- Data Access Layer – забезпечує зберігання сутностей та надає до них доступ.

4-й рівень



PL: кожен контроллер успадковується від класу `ControllerBase` із ASP.NET Core. Контроллери пов'язані з одним чи більше сервісами.

BLL: тут розміщені інтерфейси сервісів та їх реалізації. Більшість реалізації успадковується від класу `BaseService`. Сервіси мають доступ до `GenericRepository` із наступного рівня.

DAL: `GenericRepository` клас, що реалізовує відповідний інтерфейс надає доступ до даних. Він використовує клас `TheatreBoxOfficeContext`, що наслідується від `IdentityDbContext`, який у свою чергу дає змогу використовувати EF Core та Identity Framework. Також тут розміщені сутності для роботи з даними.

WEB API:

Auth:

«Сучасні технології розробки WEB-застосунків на платформі Microsoft.NET»

POST

/api/Auth/register

^

Parameters

Try it out

No parameters

Request body

application/json

Example Value

Schema

```
{  "email": "string",  "password": "string"}
```

Responses

Code	Description	Links
200	Success	No links

Media type

text/plain

Controls Accept header.

Example Value

Schema

```
{  "accessToken": "string",  "refreshToken": "string"}
```

POST

/api/Auth/login

^

Parameters

Try it out

No parameters

Request body

application/json

Example Value

Schema

```
{  "email": "string",  "password": "string"}
```

Responses

Code	Description	Links
200	Success	No links

Media type

text/plain

Controls Accept header.

Example Value

Schema

```
{  "accessToken": "string",  "refreshToken": "string"}
```

«Сучасні технології розробки WEB-застосувань на платформі Microsoft.NET»

POST

/api/Auth/register-manager

Parameters

Try it out

No parameters

Request body

application/json

Example Value | Schema

```
{  "email": "string",  "password": "string"}
```

Responses

Code	Description	Links
200	Success	No links

Media type

text/plain

Controls Accept header.

Example Value | Schema

```
{  "accessToken": "string",  "refreshToken": "string"}
```

POST

/api/Auth/logout

Parameters

Try it out

No parameters

Request body

application/json

Example Value | Schema

```
"string"
```

Performances:

GET

/api/Perfomances/{id}

Parameters

Try it out

Name	Description
id required	
integer(\$int64)	id
(path)	

Responses

Code	Description	Links
200	Success	No links

Media type

text/plain

Controls Accept header.

Example Value | Schema

```
{  "id": 0,  "name": "string",  "date": "2023-11-21T13:34:49.200Z",  "author": [    {      "firstName": "string",      "lastName": "string"    }  ],  "genre": [    "string"  ]}
```

«Сучасні технології розробки WEB-застосунків на платформі Microsoft.NET»

PUT

/api/Perfomances/{id}

⌵

Parameters

Try it out

Name	Description
id * required	
integer(\$int64)	id
(path)	

Request body

application/json

Example Value

Schema

```
{
  "id": 0,
  "name": "string",
  "date": "2023-11-21T13:34:49.202Z",
  "authors": [
    {
      "firstName": "string",
      "lastName": "string"
    }
  ],
  "genres": [
    "string"
  ]
}
```

Responses

Code	Description	Links
200	Success	No links

Media type

text/plain

Controls Accept header.

Example Value

Schema

```
{
  "id": 0,
  "name": "string",
  "date": "2023-11-21T13:34:49.203Z",
  "author": [
    {
      "firstName": "string",
      "lastName": "string"
    }
  ],
  "genre": [
    "string"
  ]
}
```

DELETE

/api/Perfomances/{id}

⌵

Parameters

Try it out

Name	Description
id * required	
integer(\$int64)	id
(path)	

Responses

Code	Description	Links
200	Success	No links

«Сучасні технології розробки WEB-застосунків на платформі Microsoft.NET»

POST

/api/Perfomances

Try it out

Parameters

No parameters

Request body

application/json

Example Value | Schema

```
{  "name": "string",  "date": "2023-11-21T13:34:49.205Z",  "authors": [    {      "firstName": "string",      "lastName": "string"    }  ],  "genres": [    "string"  ]}
```

Responses

Code	Description	Links
200	Success	No links

Media type

text/plain

Controls Accept header.

Example Value | Schema

```
{  "id": 0,  "name": "string",  "date": "2023-11-21T13:34:49.206Z",  "author": {    "firstName": "string",    "lastName": "string"  },  "genre": [    "string"  ]}
```

GET

/api/Perfomances/search

Try it out

Parameters

Name	Description
Name string (query)	<input type="text" value="Name"/>
AuthorFirstName string (query)	<input type="text" value="AuthorFirstName"/>
AuthorLastName string (query)	<input type="text" value="AuthorLastName"/>
Genre string (query)	<input type="text" value="Genre"/>
Date.Year integer(\$int32) (query)	<input type="text" value="Date.Year"/>
Date.Month integer(\$int32) (query)	<input type="text" value="Date.Month"/>
Date.Day integer(\$int32) (query)	<input type="text" value="Date.Day"/>
Date.DayOfWeek integer(\$int32) (query)	Available values : 0, 1, 2, 3, 4, 5, 6 <div>...</div>
Date.DayOfYear integer(\$int32) (query)	<input type="text" value="Date.DayOfYear"/>
Date.DayNumber integer(\$int32) (query)	<input type="text" value="Date.DayNumber"/>

«Сучасні технології розробки WEB-застосунків на платформі Microsoft.NET»

Responses

Code	Description	Links
200	Success	No links

Media type

text/plain

Controls Accept header.

Example Value | Schema

```
[
  {
    "id": 0,
    "name": "string",
    "date": "2023-11-21T13:34:49.208Z",
    "author": [
      {
        "firstName": "string",
        "lastName": "string"
      }
    ],
    "genre": [
      "string"
    ]
  }
]
```

GET /api/Performances/{id}/tickets-info

Parameters

Try it out

Name	Description
id * required integer(\$int64) (path)	id

Responses

Code	Description	Links
200	Success	No links

Media type

text/plain

Controls Accept header.

Example Value | Schema

```
[
  {
    "status": 0,
    "price": 0,
    "ticketsNumber": 0,
    "tickets": [
      {
        "id": 0,
        "seat": 0
      }
    ]
  }
]
```

«Сучасні технології розробки WEB-застосунків на платформі Microsoft.NET»

POST

/api/Perfomances/{id}/add-tickets

Try it out

Parameters

Name	Description
id * required	
integer(\$int64)	id
(path)	

Request body

application/json

Example Value

Schema

```
{  "seats": [    0  ],  "price": 0,  "status": 0}
```

Responses

Code	Description	Links
200	Success	No links

Media type

text/plain

Controls Accept header

Example Value

Schema

```
{  "createdTicketsIds": [    0  ],  "failedTicketsSeats": [    0  ]}
```

Tickets:

GET

/api/Tickets/{id}

Try it out

Parameters

Name	Description
id * required	
integer(\$int64)	id
(path)	

Responses

Code	Description	Links
200	Success	No links

Media type

text/plain

Controls Accept header

Example Value

Schema

```
{  "id": 0,  "seat": 0,  "price": 0,  "status": 0,  "performance": {    "id": 0,    "name": "string",    "date": "2023-11-21T13:41:19.619Z",    "author": [      {        "firstName": "string",        "lastName": "string"      }    ],    "genre": [      "string"    ]  } }
```

«Сучасні технології розробки WEB-застосунків на платформі Microsoft.NET»

PUT

/api/Tickets/{id}

^

Parameters

Try it out

Name	Description
id * required	
integer(\$int64)	id
(path)	

Request body

application/json

Example Value | Schema

```
{  "id": 0,  "seat": 0,  "price": 0,  "status": 0,  "performanceId": 0}
```

Responses

Code	Description	Links
200	Success	No links

Media type

text/plain

Controls Accept header.

Example Value | Schema

```
{  "id": 0,  "seat": 0,  "price": 0,  "status": 0,  "performance": {    "id": 0,    "name": "string",    "date": "2023-11-21T13:41:19.621Z",    "author": [      {        "firstName": "string",        "lastName": "string"      }    ],    "genre": [      "string"    ]  } }
```

POST

/api/Tickets/{id}

^

Parameters

Try it out

Name	Description
id * required	
integer(\$int64)	id
(path)	

Request body

application/json

Example Value | Schema

```
{  "seat": 0,  "price": 0,  "performanceId": 0}
```

«Сучасні технології розробки WEB-застосунків на платформі Microsoft.NET»

Responses

Code	Description	Links
200	Success	No links

Media type

text/plain

Controls Accept header.

Example Value | Schema

```
{  "id": 0,  "seat": 0,  "price": 0,  "status": 0,  "performance": {    "id": 0,    "name": "string",    "date": "2023-11-21T13:41:19.624Z",    "author": [      {        "firstName": "string",        "lastName": "string"      }    ],    "genre": [      "string"    ]  } }
```

DELETE /api/Tickets/{id}

Parameters

Try it out

Name	Description
id * required integer(\$int64) (path)	id

Responses

Code	Description	Links
200	Success	No links

POST /api/Tickets/buy

Parameters

Try it out

No parameters

Request body

application/json

Example Value | Schema

```
[  0 ]
```


«Сучасні технології розробки WEB-застосунків на платформі Microsoft.NET»

Responses

Code	Description	Links
200	Success	No links

Media type

text/plain

Controls Accept header.

Example Value | Schema

```
{  "tickets": [    {      "id": 0,      "seat": 0,      "price": 0,      "status": 0,      "performance": {        "id": 0,        "name": "string",        "date": "2023-11-21T13:41:19.626Z",        "author": {          {            "firstName": "string",            "lastName": "string"          }        },        "genre": [          "string"        ]      }    },    {      "orderDate": "2023-11-21T13:41:19.626Z"    }  ],  "user": {    "id": 0,    "email": "string"  }}
```

POST /api/Tickets/book

Parameters

Try it out

No parameters

Request body

application/json

Example Value | Schema

```
[  0]
```

Responses

Code	Description	Links
200	Success	No links

Media type

text/plain

Controls Accept header.

Example Value | Schema

```
{  "tickets": [    {      "id": 0,      "seat": 0,      "price": 0,      "status": 0,      "performance": {        "id": 0,        "name": "string",        "date": "2023-11-21T13:41:19.627Z",        "author": {          {            "firstName": "string",            "lastName": "string"          }        },        "genre": [          "string"        ]      }    },    {      "orderDate": "2023-11-21T13:41:19.627Z"    }  ],  "user": {    "id": 0,    "email": "string"  }}
```

Tokens:

POST

/api/Tokens/refresh

Try it out

Parameters

No parameters

Request body

application/json

Example Value | Schema

"string"

Responses

Code	Description	Links
200	Success	No links

Media type

text/plain

Controls Accept header.

Example Value | Schema

```
{  "accessToken": "string",  "refreshToken": "string"}
```

Висновки:

Висновок: під час виконання лабораторної роботи я спроектувала систему для підтримки роботи театральної каси. При цьому були використані такі методи як С4 діаграми, ER діаграми та Swagger документація.