**КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ**

**ІМЕНІ ТАРАСА ШЕВЧЕНКА**

Факультет комп’ютерних наук та кібернетики

**Курсова робота**

За спеціальністю 122 Комп’ютерні науки

на тему:

**ЗВІТ ДО ЛАБОРАТОРНОЇ РОБОТИ №1**

Виконав студент 2 курсу

Бобиль Мирослав Володимирович

Науковий керівник:

асистент

Белова Анна Сергіївна

Засвідчую, що в цій курсовій роботі

немає запозичень з працю інших авторів

без відповідних посилань

Київ – 2024

**Зміст**

|  |  |
| --- | --- |
| **Реферат** | **3** |
| **Скорочення та умовні позначення** | **4** |
| **Вступ** | **5** |
| **Розділ 1. Огляд використаних технологій** | **7** |
| **Розділ 2. Призначення і цілі створення системи** | **8** |
| **Розділ 3. Опис організації інформаційної бази** | **11** |
| **Розділ 4. Реалізація системи** | **15** |
| **Розділ 5. Інструкція користувачу** | **18** |
| **Висновки** | **20** |
| **Перелік використаних джерел** | **21** |

**Реферат**

**Ключові слова** : ASP.NET CORE, ENTITY FRAMEWORK, MICROSOFT VISUAL STUDIO 2022, .NET, ВЕБ-ЗАСТОСУНОК, ПЗ, РОЗРОБКА ПЗ, ФОРМА, ЗАПИТ, ПРЕДСТАВЛЕННЯ, DATA FIRST, DB.

**Об’єктом дослідження** є процес реалізації сервісу для управління мережею кінотеатрів, а саме – процесом організації показу сеансів та продажу квитків.

**Предметом дослідження** є служба управління базою даних мережі кінотеатрів.

**Метою** даної роботи є розробка служби управлінням мережею кінотеатрів, а саме показом фільмів, продажем квитків, наповненням фільмотеки.

**Методи дослідження**: спостереження, порівняння та аналіз.

**Інструментальні розроблення**: платформа .NET Core, технологія роботи з даними Entity Framework Core, СКБД SQL Management Studio, база даних SQL Server, організація архітектури серверного застосунку на основі ASP.NET Core з використанням багаторівневої архітектури, середовище розробки Microsoft Visual Studio 2022.

**Скорочення та умовні позначення**

**API –** application programming interface, прикладний програмний інтерфейс;

**ASP –** Active Server Pages, технологія створення веб-застосунків і веб-сервісів від компанії Microsoft;

**EF Core –** Entity Framework Core, об'єктно-орієнтована, легковажна технологія;

**HTML –** HyperText Markup Language, мова гипертекстової розмітки;

**IDE –** Integrated Design Environment, інтегроване середовище розробки;

**LINQ –** Language Integrated Query, мова запитів для платформи програмування .NET;

**.NET –** платформа програмування компанії Microsoft;

**NuGet –** система керування пакетами для платформ Microsoft;

**ORM –** Object-Relational Mapping, об’єктно- реляційне відображення;

**SQL –** Structured query language, мова структурованих запитів;

**WEB –** World Wide Web, всесвітня мережа;

**БД –** база даних;

**Вступ**

**Актуальність**. Розробка інтернет-сервісу керування мережею кінотеатрів та його використання є актуальним питанням на сьогоднішній день. В світі і особливо в Україні, величезними темпами росте кількість користувачів Інтернет і, як наслідок, кількість «електронних» замовників.

**Мета дослідження** - розробка інтернет-сервісу для керування базою даних та створення продажу квитків.

**Завдання дослідження** – розробка алгоритму функціонування інтернет-сервісу програмних засобів, проектування структури інтернет-сервісу, керування базою даних мережі кінотеатрів, внесених в базу їх персоналом записів; поглиблення та закріплення знань з технологій ASP.NET MVC та Entity Framework Core – Data First

**Предметом дослідження** є веб-сервіс, що реалізує ідею сервісу для надання можливості керувати базою даних мережі кінотеатрів.

**Методи дослідження**, використані у роботі:

* аналіз;
* моделювання;
* спостереження.

**Використані інструменти** для досягнення остаточного результату:

* платформа .NET;
* інтегроване середовище розробки Visual Studio 2022;
* мова програмування C#;
* додаток для створення веб-застосунків ASP.NET Core;
* WPS Office для створення даного звіту.

**Практична цінність** отриманих результатів полягає у використанні інтернет – сервісу для комерційних цілей. Взаємозв’язок з іншими роботами. В процесі роботи були використані та поглиблені знання з створення та зміни баз даних за допомогою Data First Workflow, а також набуті знання в роботи з MVC мовою програмування C#

**Розділ 1. Огляд використаних технологій**

1.1 Entity Framework Core Data First - ORM Framework

Об'єктно-орієнтована, розширювана технологія від компанії Microsoft для доступу до даних. EF Core є ORM-інструментом (object-relational mapping – відображення даних на реальні об'єкти). Тобто EF Core дозволяє працювати базами даних, але є більш високий рівень абстракції: EF Core дозволяє абстрагуватися від самої бази даних та її таблиць та працювати з даними незалежно від типу сховища. Якщо фізично ми оперуємо таблицями, індексами, первинними та зовнішніми ключами, але на концептуальному рівні, який нам пропонує Entity Framework, ми вже працюємо з об'єктами.

1.2 ASP.NET MVC

ASP.NET MVC (Model-View-Controller) – це фреймворк для створення веб-додатків, розроблений компанією Microsoft. Він реалізує патерн MVC, який розділяє додаток на три основні компоненти: модель (Model), уявлення (View) та контролер (Controller). Цей підхід дозволяє розробникам створювати веб-додатки, які легко підтримувати та розширювати.

**Основні компоненти ASP.NET MVC**

1. **Model (Модель)**:
   * Відповідає за бізнес-логіку додатку та роботу з даними.
   * Містить дані, що надходять з бази даних, і методи для їхньої обробки.
   * Зазвичай моделі є простими об'єктами (Plain Old CLR Objects, POCOs), що представляють структуру даних.
2. **View (Уявлення)**:
   * Відповідає за відображення інтерфейсу користувача.
   * Уявлення отримує дані з моделі та формує HTML для відправлення на клієнт.
   * Часто використовуються Razor шаблони для створення уявлень.
3. **Controller (Контролер)**:
   * Відповідає за обробку вхідних запитів від користувача.
   * Викликає методи моделі для обробки даних і передає результати уявленню.
   * Контролери містять дії (action methods), які відповідають на різні запити (HTTP GET, POST тощо).

1.3 Bootstrap

Bootstrap — це популярний фреймворк для розробки адаптивних (responsive) і мобільно-орієнтованих веб-сайтів та веб-додатків. Він містить набір інструментів для створення інтерфейсів користувача, таких як шаблони HTML та CSS для типографії, форм, кнопок, навігаційних елементів та інших компонентів інтерфейсу. Bootstrap також включає JavaScript-плагіни для покращення функціональності.

1.4 JavaScript

Потужна мова програмування, яка може додати веб-сайту інтерактивності. Його винайшов Брендан Айх (співзасновник проекту Mozilla, Mozilla Foundation і Mozilla Corporation). JavaScript є універсальним і зручним для початківців.

**Розділ 2. Призначення і цілі створення системи**

2.1 Призначення системи

A diagram of a website

Description automatically generated

Use-case діаграма

Призначення веб-системи «CMS Web App» є автоматизація процесу управління базою даних мережі кінотеатрів, що дає змогу ефективно керувати усіма задачами, від створення сеансу, продажу квитків, до аналізу попиту на ту чи іншу кінопродукцію в той чи інший час.

Робота передбачає:

* Аналіз методів, методик і моделей, що застосовуються для розв’язання задач створення комплексних вебсистем
* Проектування та програмну реалізацію системи «CMS Web App»

2.2 Цілі створення системи

«CMS Web App» створюється з метою надання користувачам системи управління базою даних, організаціїї її в єдину систему, задля поліпшення та полегшення робочого процесу.

2.3 Вимоги до системи

Система повинна будуватись з використанням підходів ООП, з уніфікацією програмно-технічних засобів розробки прикладної функціональності з використанням сучасних технологій. А також система має бути реалізована з використанням технології Data First.

2.3.1 Вимоги до системи в цілому

Система «CMS Web App» повинна реалізовувати функції створення, редагування та видалення сеансів, фільмів, кінотеатрів, залів та квитків.

В системі передбачається виділити наступні функціональні підсистеми:

* адміністративна, призначена для керування базою даних, зміною записів;
* підсистема користувача, призначена для перегляду вмісту бази даних;

2.3.2 Вимоги до функції, які виконуються системою

**Підсистема адміністратора**

|  |  |
| --- | --- |
| Функція | Задача |
| Робота з базою даних | Перегляд, створення, оновлення та видалення даних про кінотеатри, сеанси, фільми та квитки |
| Робота з замовленнями | Перегляд та аналіз попиту |

**Підсистема користувача**

|  |  |
| --- | --- |
| Функція | Задача |
| Робота з базою даних | Перегляд, даних про фільми, сеанси, багатокритеріальний пошук |
| Робота з замовленнями | Покупка квитків |

**Розділ 3. Опис організації інформаційної бази**

3.1 Логічна структура бази даних

**A screenshot of a computer screen

Description automatically generated**

Діаграма бази даних.

Перелік таблиць системи «Restaurant API»

|  |  |  |
| --- | --- | --- |
| Номер | Таблиця | Опис |
| 1 | Cinemas | Таблиця для збереження інформації кінотеатри |
| 2 | Halls | Таблиця для збереження інформації про зали кінотеатрів |
| 3 | Sessions | Таблиця для збереження інформації про сеанси |
| 4 | Movies | Таблиця для збереження інформації про фільми |
| 5 | Persons | Таблиця для збереження інформації про важливих персон, які брали участь в зйомці фільму |
| 6 | PersonsToMovies | Таблиця для збереження інформації про зв’язок між учасниками зйомки та фільмами |
| 7 | Tickets | Таблиця для збереження інформації про куплені квитки |

3.2 Опис таблиць

Таблиця Cinemas

|  |  |  |
| --- | --- | --- |
| Атрибут | Тип | Опис |
| Id | int | Ідентифікатор кінотеатру |
| CinemaName | Nvarchar(255) | Назва кінотеатру |
| Adress | Nvarchar(255) | Адреса кінотеатру |

Таблиця Halls

|  |  |  |
| --- | --- | --- |
| Атрибут | Тип | Опис |
| Id | int | Ідентифікатор зали |
| CinemaId | int | Зовнішній ключ |
| HallName | Nvarchar(255) | Назва зали |
| MaxPlaces | int | Кількість доступних місць у залі |

Таблиця Sessions

|  |  |  |
| --- | --- | --- |
| Атрибут | Тип | Опис |
| Id | int | Ідентифікатор сеансу |
| HallId | int | Зовнішній ключ |
| MovieId | int | Зовнішній ключ |
| StartTime | DateTime | Початок сеансу |
| Price | Currency | Ціна квитка |

Таблиця Tickets

|  |  |  |
| --- | --- | --- |
| Атрибут | Тип | Опис |
| Id | int | Ідентифікатор квитка |
| SessionId | int | Зовнішній ключ |
| SoldTime | DateTime | Час продажу |
| SeatNumber | int | Номер місця |

Таблиця Movies

|  |  |  |
| --- | --- | --- |
| Атрибут | Тип | Опис |
| Id | int | Ідентифікатор фільма |
| Title | Nvarchar(50) | Назва |
| Genre | Nvarchar(50) | Жанр |
| ReleaseYear | int | Рік випуску |
| Duration | int | Тривалість у хвилинах |

Таблиця Persons

|  |  |  |
| --- | --- | --- |
| Атрибут | Тип | Опис |
| Id | int | Ідентифікатор персони |
| PersonName | Nvarchar(50) | Ім’я |
| TookPartAs | Nvarchar(50) | В якій ролі брав участь під час зйомки |

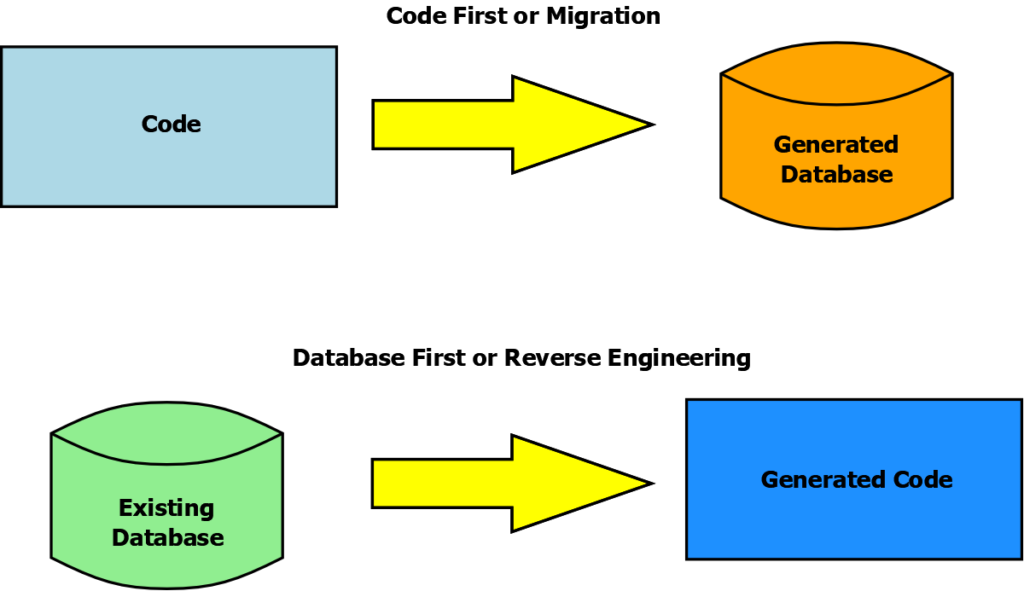
Таблиця PersonsToMovies

|  |  |  |
| --- | --- | --- |
| Атрибут | Тип | Опис |
| Id | int | Ідентифікатор зв’язку |
| PersonId | int | Ідентифікатор персони |
| MovieId | int | Ідентифікатор фільму |

**Розділ 4. Реалізація системи**

4.1.Бази даних

Весь процес кодування виконувався в середовищі програмування Visual Studio 2022 на мові програмування C# із застосуванням технології Entity Framework Core. Був застосований підхід Data First.



Для того щоб використовувати БД у програмі, необхадно під’єднатися до SQL-сервера, вкзавши ConnectionString, і створити необхадні моделі та контекст з цими моделями.

4.2 Створення моделей

За замовчуванням всі типи сутностей, для яких визначені в контексті даних набори DbSet, включаються в модель і надалі зіставляються з таблицями в базі даних. Але крім того, в модель також включаються типи, на які є посилання в сутності, які вже включені в модель, наприклад, через властивості DbSet.

За замовчуванням кожна сутність зіставляється з таблицею, яка називається за іменем властивості DbSet <T> в контексті даних, що представляє дану сутність. Якщо в контексті даних подібного властивості не визначено, то для назви таблиці використовується ім'я класу сутності.

A screen shot of a computer program

Description automatically generated

Модель для зберігання сезону

A screen shot of a computer

Description automatically generated

Контекст

4.3 Створення контролера

Контролери можна створити автоматично при наявності коректної моделі даних та працюючого контексту з’єднання з БД – цього достатньо для автогенерації не тільки контролера, а і вигляду для перегляду і редагування даних – так званих CRUD-операцій.

A screen shot of a computer program

Description automatically generated

Контролер для обробки запитів для пошуку сеансу по назві фільму, імені актора або режисера з реалізацією авторизації та розділення ролей

**Розділ 5. Інструкція користувачу**

Користувач має доступ до всіх функцій з головної сторінки. Додатово, імплементовано функцію реєстрації та авторизації.

A screenshot of a computer

Description automatically generated

Головна сторінка

A screenshot of a login form

Description automatically generated

Форма реєстрації

A screenshot of a login screen

Description automatically generated

Форма входу

У сервісі, кожен користувач має 1 з двох ролей – Admin, який має право на будь які операції, та User – який може виконувати лише операції читання. Єдине виключення – це операція покупки квитків, яка міняє значення в БД.

Для кожної із 7 таблиць реалізовані CRUD-операції: Create, Read, Update, Delete.

Додатково, імплементовано генерацію звітів у форматі .xlsx та .docx по таблиці Cinemas: отримати список всіх сеансів із відповідним даними по кожному кінотеатру.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Також, імплементовано можливість багатокритеріального пошуку сеансу і покупки квитка:

A screenshot of a computer

Description automatically generated

A screenshot of a web page

Description automatically generated

**Висновки**

Було розроблено систему, що надає користувачу можливість керування базою даних мережі кінотеатрів, зокрема переглядати, видаляти, створювати та редагувати кожну таблицю даних.

У процесі роботи над розробкою веб-додатку навчився новій технології з розробки веб-додатків, поглибив знання з HTML та CSS, а також ознайомився з технологією Data First.

В результаті роботи над проектом було окремо розроблено backend та frontend частину, що в майбутньому дозволить реалізувати повноцінний програмний продукт «CMS Web App», що буде відповідати усім технічним вимогам. Цей досвід дуже важливий для підвищення технічних навичок, покращення знань у сфері інформаційних технологій та оформленні необхідної документації до проекту.

**Перелік використаних джерел**

1. Bootswatch [Електронний ресурс] – Режим доступу до ресурсу:

https://bootswatch.com/

2. ASP.NET MVC documentation [Електронний ресурс] – Режим

доступу до ресурсу: https://docs.microsoft.com/en

us/aspnet/mvc/overview/getting-started/introduction/getting-started/

3. Entity Framework documentation [Електронний ресурс] – Режим

доступу до ресурсу: https://docs.microsoft.com/en-us/ef/

4. JavaScript documentation [Електронний ресурс] – Режим доступу до ресурсу: https://devdocs.io/javascript/