**Документация на групов проект**

Video Game Platform

**1. Цели:**

- Проектът ни трябва да съдържа класове – модели за Видео игри, Потребители, Администратори, Разпродажби, Подаръчни карти, Държави, Студия за игри, Жанрове, Отзиви и библиотека съдържаща кои игри кои потребители притежават.

- Да се добави подходящ интерфейс за потребители и администратори, със функционалност за влизане и излизане от акаунти и хеширане на пароли.

- Поддръжка за различни държави и валути при купуване на игри и добавяне на баланс

- Поддръжка за запазване на Библиотеки от видео игри

Потребителския интерфейса трябва да съдържа методи за:

- Добавяне на баланс във валутата на потребителя, използвайки подаръчни карти

- Разглеждане на възможните игри за закупуване със изписани – жанра, студио и цена във валутата на потребителя

- Купуване на игри от потребителя

- Писане на отзиви за игри, които потребителя притежава

- Разглеждане на отзиви за игра

- Разглеждане на профили на други потребители

- Виждане на собствената библиотека от игри

-Излизане от профила

Администраторския интерфейс трябва да съдържа:

- Възможност за промяна на: Видео игри, студия за игри, потребители, разпродажби

- Добавяне на: игри, разпродажби, студия за игри, потребители?

- Масово генериране на случайно избрани подаръчни карти

- Излизане от профила

Трябва да има и общо меню с функционалности:

- регистриране на потребителски профил

- влизане като потребител

- влизане като администратор

- за спиране на програмата

**2. Разпределение на ролите**

Слой за данни – Ата Темур

Бизнес слой – Ата Темур (Със съдействие на двамата други участника)

Презентационен слой – Светослав Мирославов Балтаджиев

Тестове – Кристиян Костадинов Джургов

**3. Основни етапи в развитието на проекта**

**Етап 1:** Създаване и правилно описване на моделите, с правилни анотации, за да се подсигури коректно мигриране към база данни

**Етап 2:** Описване на контекст класа и подготвяне на проекта за миграция

**Етап 3:** Миграция

**Етап 4:** Разработка на основните методи в контролер класа

**Етап 5:** Разработка на презентационния слой, и разширяване функционалността на бизнес слоя, за да поддържа методите в новите класове

**4. Реализация**

1. Създаване и правилно описване на моделите, с правилни анотации, за да се подсигури коректно мигриране към база данни

Започнахме със създаването на проект на Net 5.0, където направихме папки за слоя за данни, бизнес слоя и презентационния слой. Изтеглихме пакетите:

- Microsoft.EntityFrameworkCore.SqlServer – версия 5.0.17

- Microsoft.EntityFrameworkCore.Design – версия 5.0.17

- Microsoft.EntityFrameworkCore.Tools – версия 5.0.17

- System.Security.Cryptography – версия 4.3.1 – за хеширане на пароли

* 1. Таблица Countries

За да дефинираме таблицата Countries, създадохме клас Country, който е модел.

Моделът трябва да съдържа основна информация за държавата, като име и валута. За да може да се имплементира функционалност за преобразуване на валути ни трябва и колона/поле за валутния курс на държавата спрямо еврото.

За Primary key на таблицата сме избрали CountryCode (трибуквен код отговарящ на държавата), защото е достатъчен идентификатор:

Text

Description automatically generated

Името на държавата е string/varchar с дължина до 50 символа и не може да бъде null.

Валутата е string/varchar с дължина до 5 символа и не може да бъде null.

Валутния курс е decimal с дължина 7 символа, 5 от тях след запетаята и не може да бъде null.

* 1. Таблица Admins

За да дефинираме таблицата Admins, създадохме клас-модел Admin.

В него се съдържат полета за username, първо име, фамилия, и две хеширани пароли.

Администраторите имат Id, което е primary key и е с Auto increment

Username, първо име и фамилия са полета, които са varchar/string, имат максимална дължина от 50 символа, и не могат да бъдат null.

Username полето също така трябва да е unique.

1.3 Таблица GameStudios

За да дефинираме таблицата GameStudios, създадохме клас-модел GameStudio.

Студията за игри имат Id, което е primary key и е с Auto increment.

Те имат име, което е varchar/string с максимална дължина от 50 символа и не може да бъде null.

Имат и поле isUnderContract, което е Boolean със стойност под разбиране false.

1.4 Таблица Genres

За да дефинираме таблицата Genres, създадохме клас-модел Genre.

Жанровете имат Id, което е primary key и е с Auto increment.

Те имат име, което е varchar/string с максимална дължина от 50 символа и не може да бъде null.

1.5 Таблица GiftCards

За да дефинираме таблицата GiftCards, създадохме клас-модел GiftCard.

Подаръчните карти имат Id, което е primary key и е с Auto increment.

Те имат код, който е varchar/string с максимална дължина от 50 символа и не може да бъде null.

Кодът е Unique property, т. е. не може да се повтаря.

Имат и поле стойност, което е стойността на картата в евро, и е поле/колона от вид decimal.

1.6 Таблица Users

За да дефинираме таблицата Users, създадохме клас-модел User.

Той съдържа полета за username, първо име, фамилия, държавен код , email и баланс.

Потребители имат Id, което е primary key и е с Auto increment.

Username, първо име и фамилия са от тип string/varchar с максимална дължина от 50 думи и не могат да бъдат null.

Emailът е от тип string/varchar и не може да бъде null.

Хашираната парола държи паролата след хеширане и е от тип stirng/varchar, не може да бъде null.

Баланса е от тип decimal и има стойност по под разбиране 0.

Държавния код е foreign key, който е reference към таблицата Countries.

За да се осъществи този reference има и Navigational property от класа Country.

1.7 Таблица VideoGames

За да дефинираме таблицата VideoGames, създадохме клас-модел VideoGame.

Видео игрите имат Id, което е primary key и е с Auto increment.

Те имат и заглавие, което е от вид string/varchar и не може да бъде null.

Те имат и цена, която е в евро, и е от вид decimal, не може да бъде null.

Те имат и StudioId, което е foreign key с reference към таблицата GameStudios.

За да се осъществи този reference има и Navigational Property от класа GameStudio.

Те имат и GenreId, което е foreign key с reference към таблицата Genres.

За да се осъществи този reference има и Navigational Property от класа Genre.

Има и поле isAvaiable, което определя дали играта може да се закупи, и има стойност по подразбиране true.

1.8 Таблица Reviews

За да дефинираме таблицата Reviews, създадохме клас-модел Review.

Тази таблица е устроена с композитен Primary Key, образуван от UserId и GameId.

UserId на своя страна е Foreign key към таблицата Users, a GameId към таблицата VideoGames.

И за двете има Navigational Property от съответния клас.

Таблицата има и поле за текста на отзива, което е от тип string/Text.

Таблицата има и поле за дали отзива е позитивен, което е от вид boolean и не може да бъде null.

1.9 Таблица Sales

За да дефинираме таблицата Sales, създадохме клас-модел Sale.

Те имат за Primary key GameId, което е foreign key с reference към таблицата VideoGames.

За да се осъществи този reference имат Navigational property.

Те имат поле discount, което е int с максимална дължина до 3 елемента и е в %. Например ако играта трябва да се намали с 30%, то полето ще е със стойност 30, не може да бъде null.

Те имат полета за начална и крайна дата от тип DateTime(във C#)/Date(в database) , които не могат да бъдат null.

1.10 Таблица UserGameCollection

Таблицата е UserGameCollection е mapping таблица, затова тя се състои само от UserId, GameId, които са композитен Primary Key.

Двете полета са и foreign key към съответната им таблица.

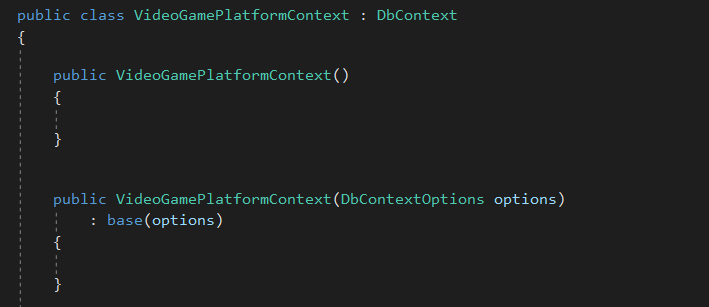
Има и Navigational property за всяко от тях.

Целта на играта е да бази игрите, които определен потребител е закупил.

**Етап 2:**

Етап 2 се състои от дефиниране на context класа.

За да се извърши това нашия context клас наследява класа от EnityFramework DbContext.



След това за всяка таблица създаваме DbSet с класа на модела.



След това override-ваме OnConfiguring метода където си слагаме connection string-а.

След него override-ваме OnModelCreating метода, където определяме спецификациите на колоните, които не могат да се определят от анотации.

Най- важни от тях са преобразуването на началната и крайната дата от DateTime на Date.

Text

Description automatically generated

Както и определянето на дължината на цената.

Text

Description automatically generated

Определянето на кои колона си уникални.

A picture containing text

Description automatically generated

**Етап 3:**

По време на миграция се сблъскахме с няколко проблеми.



Първоначално трябваше да се добавят полета от тип ICollection, където имаме обект, който е под reference, от други обекти.

Graphical user interface, text

Description automatically generated with medium confidence

Пример от класа VideoGame, защото таблиците Reviews, Sales и UserGameCollections всички имат reference към таблицата VideoGames, то класа трябва да държи колекции от тези класове.

Друг проблем, с който се сблъскахме е, че след преместване на някои от моделите имахме проблеми с namespace-овете.

**Етап 4:**

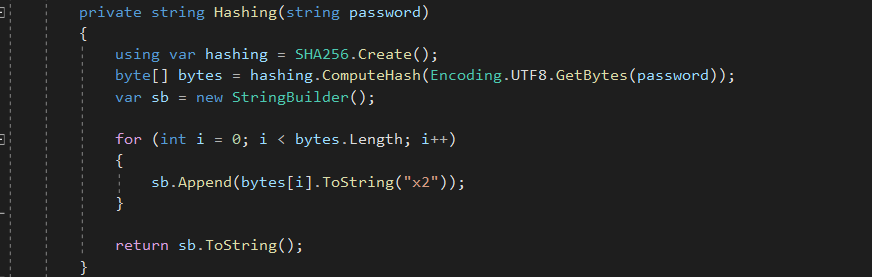
Създадохме контролер клас за всеки от моделите за да имаме възможността да извличаме и променяме информацията в базата данни.

Основни методи на класовете.

CountryController:

* GetCountryByCountryName - взима име на държава и връща държавата с това име.
* GetCountryByCountryCode - взима код на държава и връща държавата с този код.
* GetCountryCurrencyExchangeRate – взима код на държава и връща валутния курс на държавата към евро

AdminController:

* LoginAttempt – получава потребителско име и две пароли и проверява дали съвпадат с базата данни, ако съвпадат връща true, иначе false.
* GetAdminByUsername – взима потребителско име и връща администратор с това потребителско име
* Hashing – взима string и го връща хеширан.
* 

Text

Description automatically generated

GameStudioController:

* AddStudio – получава име и bool за дали е в договор и създава ново студио в базата данни.
* EditStudio – получава всички параметри на Studio със старо име, за да го намери и ги сменя на новите.
* GetStudioById – получава id и връша студиото с това id.
* GetStudioByName – получава име и връща студио с това име.

GenreController:

* AddGenre – получава име и добавя нов жанр.
* GetGenreById – получава id и връща жанр с това id
* GetGenreByName – получава име и връща жанр, на който съвпада.

GiftCardController:

* GenerateGiftCards – премахва всички подаръчни карти от системата и генерира 1000 нови, със случайно избрани кодове и стойност.
* RedeemGistCard – взима код на карта и връща стойността и, като в процеса маха тая подаръчна карта от базата данни.
* GetCardByCode – взима код на подаръчна карта и я връща.

ReviewController:

* AddReview – взима параметрите и добавя ново Review към базата данни.
* DeleteReview – взима потребител и заглавие на игра и трие отзива, който съвпада с тях.
* GetUserReviews – взима отзивите на определен потребител.
* GetReviewInfoByGameTitle – взима всички отзиви на игра определена от заглавието и връща списък от string-ове .
* GetUserReviewInfoOfSpecificGame – взима потребител и заглавие на игра и връща отзива на този потребител за тази игра.
* GetUserReviews – Връща всички отзиви на определен потребител.
* UserHasReviewFor – проверя дали потребител е оставил отзив за определена игра и връща bool.
* GetReviewsByGameTitle – връща всички отзиви на игра определена от заглавие и връща списък от игри.

SalesController:

* AddSale – Добавя ново намаление към базата данни.
* GetSaleByGameTitle – Взима намаление на игра спрямо заглавие на играта, на която трябва да бъде.
* RefreshSales – проверява кои намаления са си преминали през крайната дата и ги маха от базата данни.
* MakeSaleDiscountEffective – взима намаление и променя цената на играта което то афектира, така че да е намалена до новата и стойност.
* RevertGamePriceToBeforeSale – Връща цената на игра до стойността и преди намалението.
* RemoveSale – премахва разпродажба спрямо заглавието на играта, което афектира.
* GetSaleByGameId – взима id на игра и връща намалението спрямо това id
* GetInfoOfAllSales – взима информация за всички намаления.

Text

Description automatically generated

Text

Description automatically generatedText

Description automatically generated

UserController:

* Register – регистрира нов потребител към системата, като му хешира паролата
* Hashing – получава парола и я хешира за пазене в базата данни.
* LoginAttempt – проверява дали потребителско име и парола съвпадат с такива от базата данни и връща съответно true или false.
* DeleteUser – трие потребител от базата данни
* GetUserByUsername – взима потребител по неговото потребителско име и го връща
* GetUserById – взима потребител по неговото id и го връща.
* AddBalanceInUserCurrency – добавя определена сума към баланс на потребителя като взима под внимания валутния курс на потребителя.
* GetUserInfoByUsername – взима потребителското име на потребител и връща основната информация за него.
* UserTriesToBuyGame – получава потребител и заглавие на игра и опитва да закупи играта
* GetInfoOfAllUsers – връща информация за всички потребители.
* GetUserByUsernameOrEmail – връща потребител спрямо потребителско име или email.
* GetUserInfoByUsernameOrEmail – връща информация за потребител спрямо потребителско име или email.

Text

Description automatically generated

Text

Description automatically generated

UserGameCollectionController:

* GetGameCollectionByUser – връща колекция от игри които притежава определен потребител
* GetGameInfoCollectionByUser -връща колекция от string-ове държаща информация за всяка от игрите в библиотеката на потребителя
* UserOwnsGame – връща true ако потребителя притежава играта

Text

Description automatically generated

Text

Description automatically generated

VideoGameController:

* AddGame – Добавя игра към базата данни
* МакеGameUnavailable – сваля играта от пазара за закупуване
* ChangeGameTitle, ChangeGamePrice, ChangeGameStudio и ChangeGameGenre променят съответното поле на игра
* EditGame – взима нови параметри и ги слага на стара игра спрямо заглавието и
* GetGameByTitle – взима игра по заглавие
* GetGamesStartingWithLetter – взима всички игри започващи с определена буква
* GetGameInfoByTitle – взима информация за игра спрямо заглавието и

Text

Description automatically generated

Text

Description automatically generated

**Етап 5:**

За презентационния слой, сме разделили класовете на менюта. За всяко ново меню се вика нов клас, който му отговаря.

A screenshot of a computer

Description automatically generated with medium confidence

Като Display.cs вика главното меню, откъдето потребителите и администраторите влизат.

Класовете се състоят от ShowMenu методи и MenuInput методи, като първия принтира менюто, а втория чете число, което е една от опциите и вика метод, който изпълнява съответното действие.

Ето главните менюта:

Display.cs:

Text

Description automatically generated with medium confidence

UserDisplay.cs:

Text

Description automatically generated

ReviewsDisplay.cs:

Text

Description automatically generated

AdminDisplay.cs:

Text

Description automatically generated

GamesManagerDisplay.cs:

Text

Description automatically generated

UsersManagerDisplay.cs:

Text

Description automatically generated

StudiosManagerDisplay.cs:

Text

Description automatically generated with low confidence

SalesManagerDisplay.cs:

Text

Description automatically generated

Всяка от опциите е имплементирана с подходящо четене от конзолата и викане на подходящи методи.

**5. Развитие и нововъведения**

През процеса на писане на кода и на презентационния слой, много от методите в бизнес слоя бяха променени, за да приемат правилните параметри.

Също така бяха добавени валидации за повечето функции и параметрите, които приемат.

**6.Заключение**

Постигнахме функционалността, която си бяхме наумили да имаме.

Научихме много неща за разработката и писането на код по време на процеса.

**7.Изпозлзвани технологии и литература**

- <https://www.youtube.com/watch?v=IO4xdQVuRZQ> - за хеширане на пароли

- Stack Overflow – хилядите отговори там, когато ни се наложи да търсим нещо, което ни трябва да използваме свързано с Entity Framework.

- модела за езикова обработка базиран на изкуствен интелект, ChatGPT на OpenAI – за помощ със нещата и логиката, които не разбирахме и генериране на dataset с държави, студия за игри, жанрове и видеоигри