**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ**

**РОССИЙСКОЙФЕДЕРАЦИИ**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

федеральное государственное автономное образовательное учреждение высшего образования

«Санкт-Петербургский политехнический университет Петра Великого»  
(ФГАОУ ВО «СПБПУ»)

**Институт среднего профессионального образования**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Отделение информационных технологий

Специальность: 09.02.07 Информационные системы и программирование

Дисциплина: МДК 11.01 Технологии разработки и защиты баз данных

Группа: 32919/6

**КУРСОВОЙ ПРОЕКТ**

**на тему:**

**«Информационная система игровой компании»**

|  |  |  |  |
| --- | --- | --- | --- |
| Студент | \_\_Орлов И.А.\_\_\_\_ | | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  | (ФИО) | | (подпись) |
| Руководитель | \_\_Резанова А.О.\_\_\_\_ | | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  | (ФИО) | | (подпись) |
|  | | | |
| Оценка | \_\_\_\_\_\_\_\_\_\_\_\_ | \_\_\_\_\_\_\_\_\_\_\_\_ | \_\_\_\_Резанова А.О.\_\_\_\_\_\_ |
|  | (оценка) | (подпись) | (ФИО) |

Санкт - Петербург

2021

СОДЕРЖАНИЕ

[ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ 4](#_Toc90717623)

[ВВЕДЕНИЕ 5](#_Toc90717624)

[ОБЩИЙ РАЗДЕЛ 6](#_Toc90717625)

[СПЕЦИАЛЬНЫЙ РАЗДЕЛ 7](#_Toc90717626)

[1 Аналитический разбор 7](#_Toc90717627)

[2 Анализ конкурентов разрабатываемой ИС 7](#_Toc90717628)

[3 Требования к функционалу 7](#_Toc90717629)

[3.1 Требования к функционалу клиента 7](#_Toc90717630)

[3.2 Требования к функционалу сотрудника 8](#_Toc90717631)

[4 Технические средства разработки 9](#_Toc90717632)

[5 Диаграмма прецендентов использования 10](#_Toc90717633)

[6 Сущности и атрибуты сущностей 11](#_Toc90717634)

[7 Инфологическая модель 13](#_Toc90717635)

[8 Даталогическая модель 14](#_Toc90717636)

[9 ER-диаграмма 15](#_Toc90717637)

[10 Проект основных дисплейных фрагментов 16](#_Toc90717638)

[10.1 Форма авторизации 16](#_Toc90717639)

[10.2 Форма регистрации 16](#_Toc90717640)

[10.3 Основная форма приложения 16](#_Toc90717641)

[10.3.1 Вкладка «Магазин» основной формы 16](#_Toc90717642)

[10.3.2 Вкладка «Библиотека» основной формы 17](#_Toc90717643)

[10.3.3 Вкладка «Избранное» основной формы 18](#_Toc90717644)

[10.4 Форма авторизированного пользователя 19](#_Toc90717645)

[10.5 Форма добавления игры 20](#_Toc90717646)

[11 Исключительные ситуации 21](#_Toc90717647)

[12 Реализация добавления, удаления, обновления 22](#_Toc90717648)

[13 Разработанные запросы 22](#_Toc90717649)

[14 Обеспечение целостности БД 22](#_Toc90717650)

[14.1 Уникальность атрибутов 22](#_Toc90717651)

[14.2 Проверка данных 23](#_Toc90717652)

[15 Разграничение прав доступа и защита данных 23](#_Toc90717653)

[16 Разработка алгоритмов программных модулей 23](#_Toc90717654)

[16.1 Авторизация 23](#_Toc90717655)

[16.2 Регистрация 24](#_Toc90717656)

[17 Справочная система 25](#_Toc90717657)

[18 Тестирование приложения 25](#_Toc90717658)

[ЗАКЛЮЧЕНИЕ 31](#_Toc90717659)

[СПИСОК ИСПОЛЬЗВОАННЫХ ИСТОЧНИКОВ 32](#_Toc90717660)

[ПРИЛОЖЕНИЕ А Характеристики программного и аппаратного обеспечения 33](#_Toc90717661)

[ПРИЛОЖЕНИЕ Б Руководство системного программиста 34](#_Toc90717662)

[ПРИЛОЖЕНИЕ В Листинг SQL-скрипта 43](#_Toc90717663)

[ПРИЛОЖЕНИЕ Г Листинг приложения 45](#_Toc90717664)

[1 LibraryComponent.cs 45](#_Toc90717665)

[2 ShopAssortiment.cs 46](#_Toc90717666)

[3 Settings.cs using System; 50](#_Toc90717667)

[4 Registration.cs 53](#_Toc90717668)

[5 GlobalFields.cs 54](#_Toc90717669)

[6 Form1.cs 55](#_Toc90717670)

[7 FavouriteGames.cs 60](#_Toc90717671)

[8 AddGame.cs 63](#_Toc90717672)

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ

1. ЕСПД – единая система программной документации.
2. СУБД – система управления базами данных.
3. БД – база данных.
4. ИС – информационная система.
5. ПК – первичный ключ.
6. ВК – внешний ключ.
7. ПО – программное обеспечение.
8. Инди (от англ. «independent») - независимый

ВВЕДЕНИЕ

Разрабатывается интерфейс для базы данных «Игровая компания». В которой есть пользователи (игроки или разработчики) и игры. Пользователи приобретают игры или добавляют их в избранные, игры образуют группу основного контента приложения. Администраторы добавляют игры.

Основной целью курсового проектирования является:

1. закрепление практических умений, полученных студентами на лабораторных занятиях, и получение дополнительных навыков обследования предметной области, концептуального, логического и физического проектирования базы данных, освоение средств поддержания целостности БД, формирования разнообразных запросов и т. д.
2. знакомство и использование единой системы программной документации (ЕСПД): комплекса государственных стандартов, устанавливающих взаимоувязанные правила разработки, оформления и обращения программ и программной документации и обеспечивающих снижение трудоемкости и повышение эффективности разработки, сопровождения и эксплуатации программных изделий.

Задачами курсового проекта являются проектирование и создание прикладной программы (веб-приложения), обеспечивающей создание и ведение базы данных по выбранной предметной области. На примере этой программы должны быть показаны примеры реализации вопросов, перечисленных выше, и связанных с конкретной СУБД и операционной системой.

При формулировании заданий на курсовое проектирование учитывалась тесная связь дисциплины МДК 11.01 «Технологии разработки и защиты баз данных» с другими дисциплинами.

ОБЩИЙ РАЗДЕЛ

Предметная область – «Игровая компания». ИС предоставляет возможность получать информацию об играх компании, приобретать их, помещать в избранное. Так же добавлять игры.

Разрабатываемая ИС актуальна в связи с ростом популярности компьютерных игр, а, в следствие, и игровых компаний. Игровые компании получают преимущество перед инди проектами, так как компания вкладывается в различные ресурсы игровой разработки приводят к более крупным, проработанным и желанным продуктам деятельности(играм).

Основной цель данного курсового проекта является разработка интерфейса и базы данных для ИС «Игровая компания»

Основные задачи курсового проекта:

* Изучить предметную область
* Провести анализ конкурентов разрабатываемой ИС
* Выявить требования к функционалу разрабатываемой ИС
* Выбрать технологии для разработки
* Провести инфологическое моделирование
* Описать даталогическую модель
* Создать ER-диаграмму
* Выявить и описать исключительные ситуации
* Реализовать запросы на добавление, удаление, обновление данных в БД;
* Обеспечить целостность БД
* Обеспечить защиту данных
* Разработать алгоритмы программных модулей
* Создать справочную систему
* Провести тестирование приложения

СПЕЦИАЛЬНЫЙ РАЗДЕЛ

1. Аналитический разбор

Выбранная предметная область – «Игровая компания».

Приложение фитнес-клуба дает возможность игрокам приобретать игры, добавлять их в избранное и просматривать игры по отдельным жанрам, а разработчикам добавлять игры различных жанров.

1. Анализ конкурентов разрабатываемой ИС

В таблице 1 представлена сравнительная характеристика конкурентов разрабатываемой ИС.

1. Анализ конкурентов

|  |  |  |
| --- | --- | --- |
| Конкурент | riotgames.com | rockstargames.com |
| Авторизация и регистрация | да | да |
| Возможность приобретать игру | да | да |
| Возможность добавлять игру в избранное | нет | да |
| Возможность просмотра игр по отдельному жанру | нет | нет |
| Возможность разработчикам добавлять игру различных жанров | нет | нет |

1. Требования к функционалу
   1. Требования к функционалу клиента

* Войти в аккаунт;
* Выйти из аккаунта;
* Зарегистрироваться;
* Настроить аккаунт;
* Приобрести игру;
* Добавить игру в избранное;
* Удалить игру из избранного.
  1. Требования к функционалу сотрудника
* Войти в аккаунт;
* Выйти из аккаунта;
* Добавлять игру;

1. Технические средства разработки

В таблице 2 представлен обзор популярных СУБД.

1. Краткий обзор СУБД

|  |  |
| --- | --- |
| Название | Преимущества |
| MS SQL | Доступность;  простой понятный интерфейс;  надежность;  позволяет синхронизироваться с другими продуктами компании Microsoft. |
| MySQL | Бесплатная система;  возможность синхронизации с другими БД. |
| PostgreSQL | Распространяется бесплатно;  Наиболее удобна для веб-разработки. |
| Oracle | Удобна при администрировании БД с огромным объемом данных. |

В таблице 3 представлен обзор популярных технологий и интерфейсов доступа к данным.

1. Краткий обзор технологий и интерфейсов доступа к данным

|  |  |
| --- | --- |
| Название | Описание |
| ODBC | «Open Database Connectivity» - технология компании Microsoft, созданная в начале 90-х с целью решения проблемы обмена данными между приложением и несколькими источниками данных (одно приложение может обращаться к разным СУБД с помощью ODBC)  Использует SQL в качестве стандарта для доступа к данным.  Существуют OBDC-драйверы для нереляционных данных (электронные таблица, текст, XML файлы) |

Продолжение таблицы 3

|  |  |
| --- | --- |
| ADO | «Active Data Objects» - разработка компании Microsoft, которая позволяет представлять данные из различных источников (реляционных БД, текстовых файлов) в объектно-ориентированном виде. |
| ВDE | Технология компании Borland, позволяющая подключиться к локальным БД, получить доступ к SQL-северам (Oracle, MS SQL Server, InterBase) с помощью Borland SQL Links, извлекать данные из любых источников данных |

Выбранные технологии:

* C# – удобный и понятный язык программирования и наиболее подходящий для разработки приложения Windows Forms;
* Microsoft SQL Server – объектно-реляционная СУБД. Удобна при работе с Visual Studio.

1. Диаграмма прецендентов использования

На рисунке 1 изображена uml-диаграмма претендентов, отражающая распределение доступа к функционалу приложения в зависимости от роли авторизированного пользователя приложения. Игрок – обычный авторизированный пользователь системы. Разработчик – добавленный администратором пользователь системы.



1. Роли пользователей
2. Сущности и атрибуты сущностей
3. Сущности

|  |  |
| --- | --- |
| Сущность | Назначение сущности |
| Пользователи | Таблица, содержащая данные для входа |
| Магазин/Игры | Таблица, содержащая данные об играх, выпущенных компанией |
| Библиотека | Таблица, содержащая данные о купленных играх пользователями |
| Избранные | Таблица, содержащая данные об играх, которые пользователи поместили в избранные |
| Жанры | Таблица, содержащая данные о жанрах всех игр игровой компании |

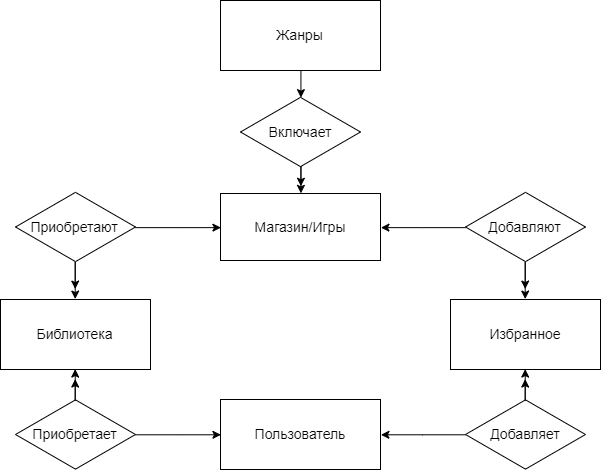
1. Атрибуты сущностей

|  |  |  |  |
| --- | --- | --- | --- |
| Сущность | Атрибуты | Назначение атрибута | Ключевое поле |
| Пользователи | Id | Уникальный код пользователя | ПК |
| Email | Email пользователя |  |
| Пароль | Пароль пользователя |  |
| Роль | Роль пользователя |  |
| Никнейм | Никнейм пользователя |  |
| Магазин/Игры | Id | Уникальный код игры | ПК |
| Название | Название игры |  |
| Цена | Цена игры |  |
| Жанр | Указатель на уникальный код жанра | ВК |
| Библиотека | Id\_Игры | Указатель на уникальный код игры | ВК |
| Id\_Пользователя | Указатель на уникальный код пользователя | ВК |
| Наиграно часов | Количество часов проведённые в игре |  |
| Избранное | Id\_Игры | Указатель на уникальный код игры | ВК |
| Id\_Пользователя | Указатель на уникальный код пользователя | ВК |
| Жанры | Id | Уникальный код жанра | ПК |
| Название | Название жанра |  |
| Количество игр | Количество игр жанра |  |

1. Инфологическая модель
2. Описание связей между сущностями

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Тип связи | Сущности | Назначение |
| Приобретают | 1 – много | Магазин/Игры – Библиотека | Одну игру приобретают множество игроков. Одну и ту же игру приобретают только один раз. |
| Приобретает | 1 – много | Пользователь – Библиотека | Один игрок приобретает множество игр. Одну и ту же игру один игрок приобретает только один раз. |
| Добавляют | 1 – много | Магазин/Игры – Избранное | Одну игру добавляют в избранное множество игроков. Одну и ту же игру добавляют в избранное только один раз. |
| Добавляет | 1 – много | Пользователи – Избранное | Один игрок добавляет в избранное множество игр. Один игрок добавляет одну игру только один раз. |
| Включает | 1 – много | Жанр – Магазин/Игры | Один жанр включает в себя множество игр. Одна игра включает в себя только один жанр. |

На рисунке 2 изображена Инфологическая модель.



1. Инфологическая модель
2. Даталогическая модель
3. Сущность «Пользователи»

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Тип | Обязательно к заполнению | Ограничение |
| Id | int | Да | ПК |
| Email | nvarchar(255) | Да |  |
| Пароль | nvarchar(255) | Да |  |
| Роль | nvarchar(31) | Да |  |
| Никнейм | nvarchar(255) | Нет |  |

1. Сущность «Магазин/Игры»

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Тип | Обязательно к заполнению | Ограничение |
| Id | int | Да | ПК |
| Название | nvarchar(255) | Да |  |
| Цена | int | Да |  |
| Жанр | int | Нет | ВК |

1. Сущность «Библиотека»

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Тип | Обязательно к заполнению | Ограничение |
| Id\_Игры | int | Да | ВК |
| Id\_Пользователя | int | Да | ВК |
| Наиграно часов | int | Нет |  |

1. Сущность «Избранное»

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Тип | Обязательно к заполнению | Ограничение |
| Id\_Игры | int | Да | ВК |
| Id\_Пользователя | int | Да | ВК |

1. Сущность «Жанры»

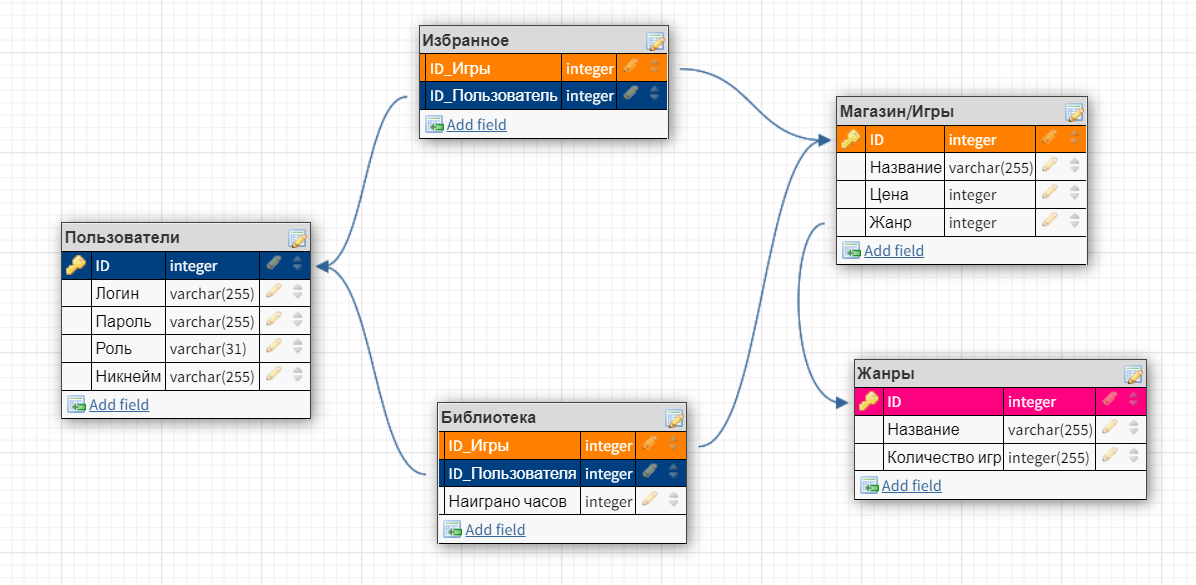
|  |  |  |  |
| --- | --- | --- | --- |
| Название | Тип | Обязательно к заполнению | Ограничение |
| Id | int | Да | ПК |

Продолжение таблицы 11

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Тип | Обязательно к заполнению | Ограничение |
| Название | nvarchar(255) | Да |  |
| Количество игр | int | Нет |  |

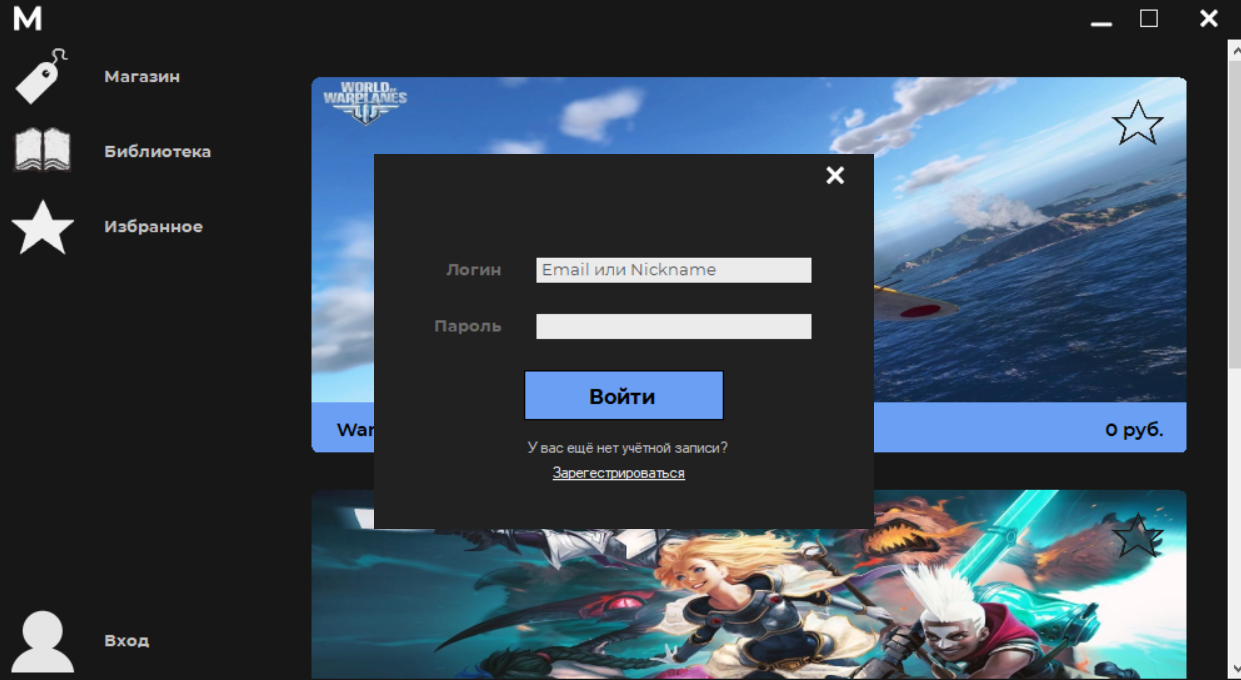
1. ER-диаграмма

На рисунке 3 изображена ER-диаграмма разрабатываемой БД.



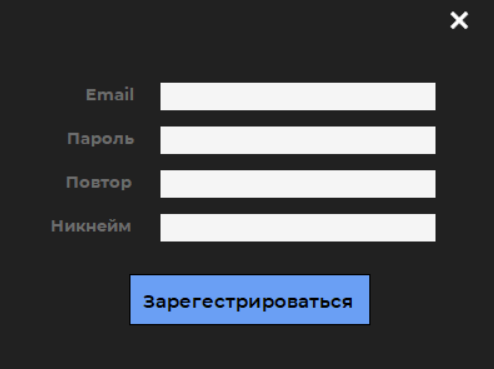
1. ER-диаграмма
2. Проект основных дисплейных фрагментов
   1. Форма авторизации

На рисунке 4 изображена страница авторизации.



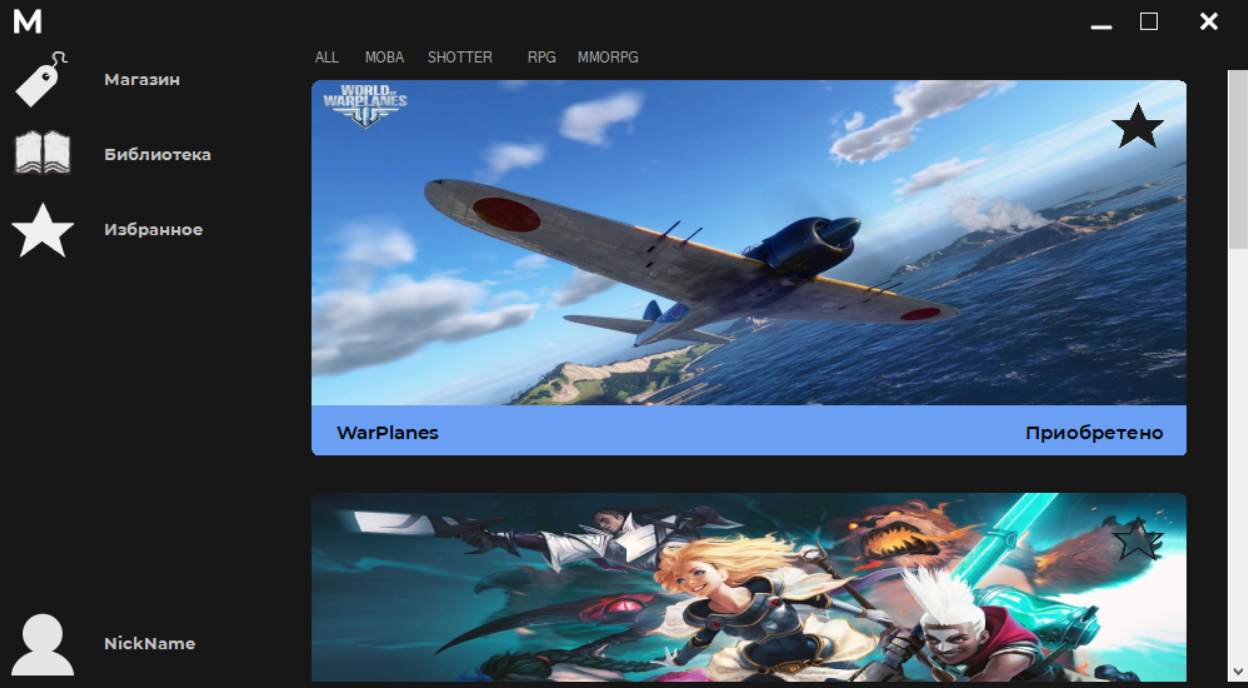
1. Форма авторизации
   1. Форма регистрации

На рисунке 5 изображена страница регистрации.



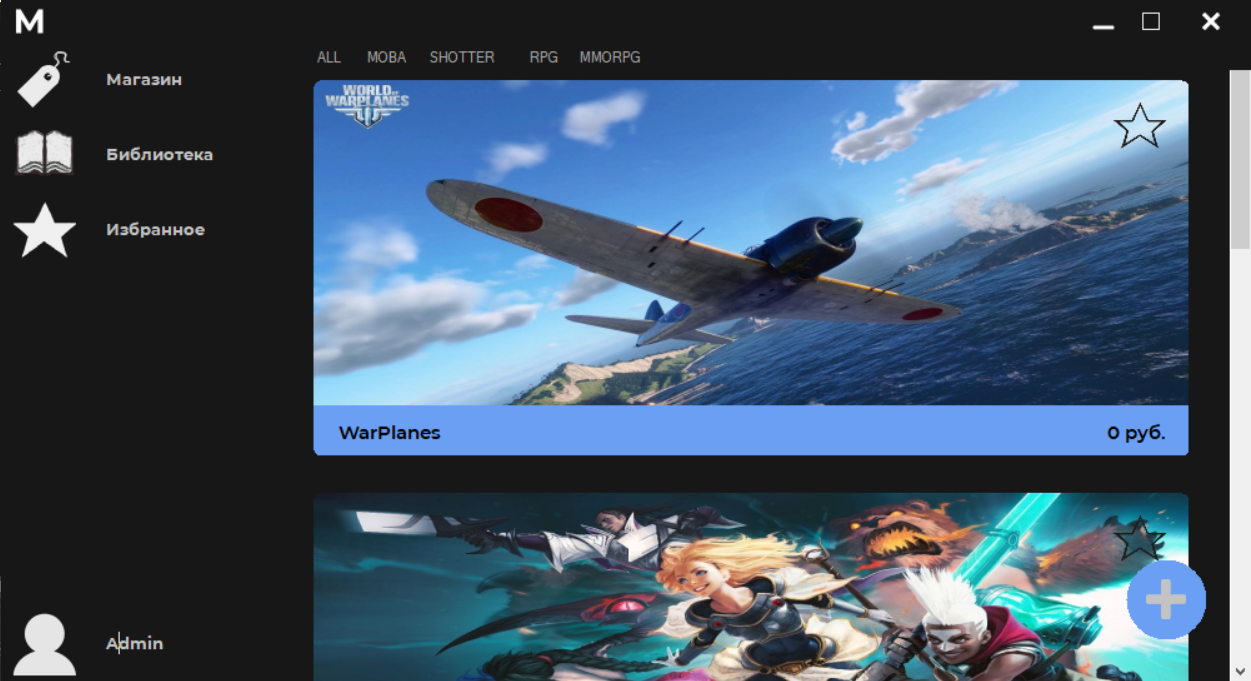
1. Форма регистрации
   1. Основная форма приложения
      1. Вкладка «Магазин» основной формы

На рисунке 6 изображена вкладка «Магазин» основной формы приложения для игрока.



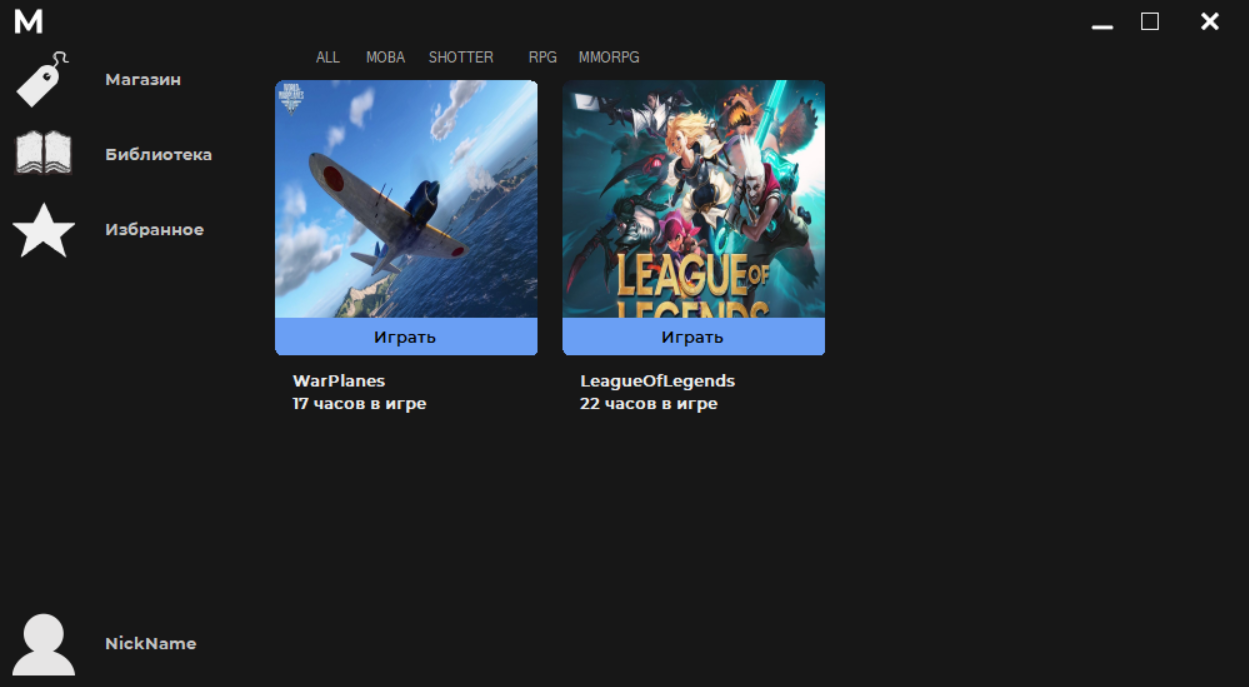
1. Вкладка «Магазин» для игрока

На рисунке 7 изображена в вкладка «Магазин» основной формы приложения разработчика.



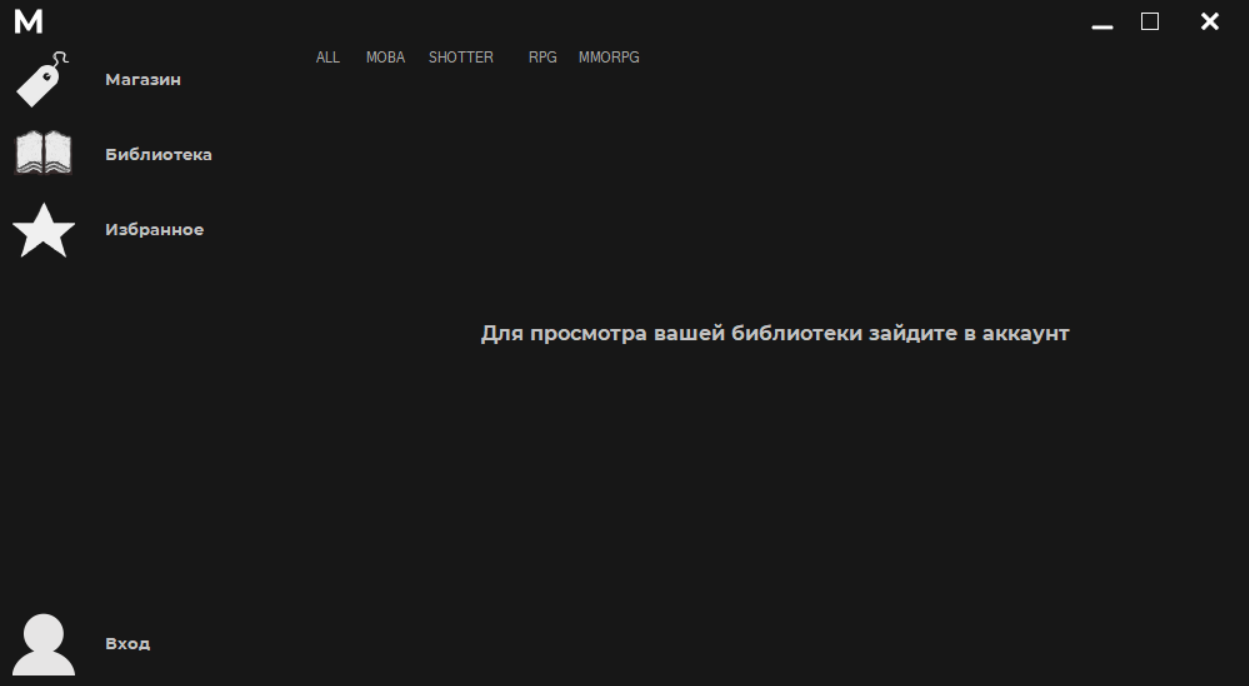
1. Вкладка «Магазин» для разработчика
   * 1. Вкладка «Библиотека» основной формы

На рисунке 8 изображена вкладка «Библиотека» основной формы приложения для авторизированного пользователя.



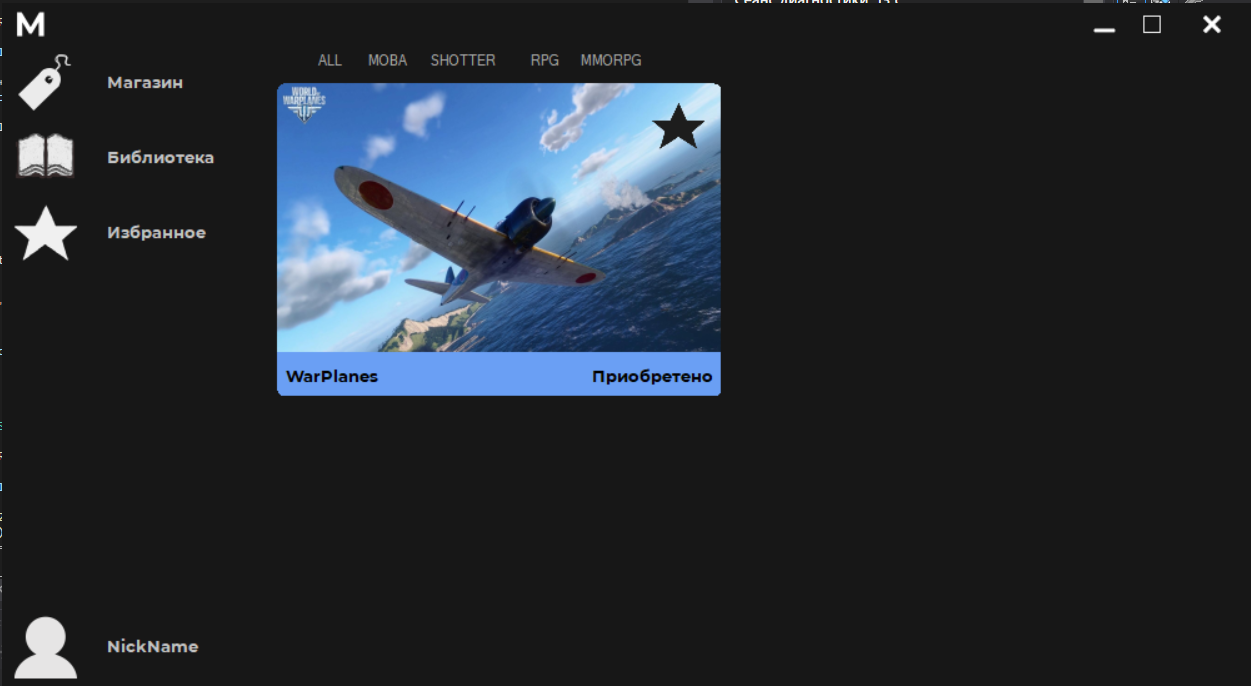
1. Вкладка «Библиотека» для авторизированного пользователя

На рисунке 9 изображена вкладка «Библиотека» основной формы приложения для неавторизированного пользователя.



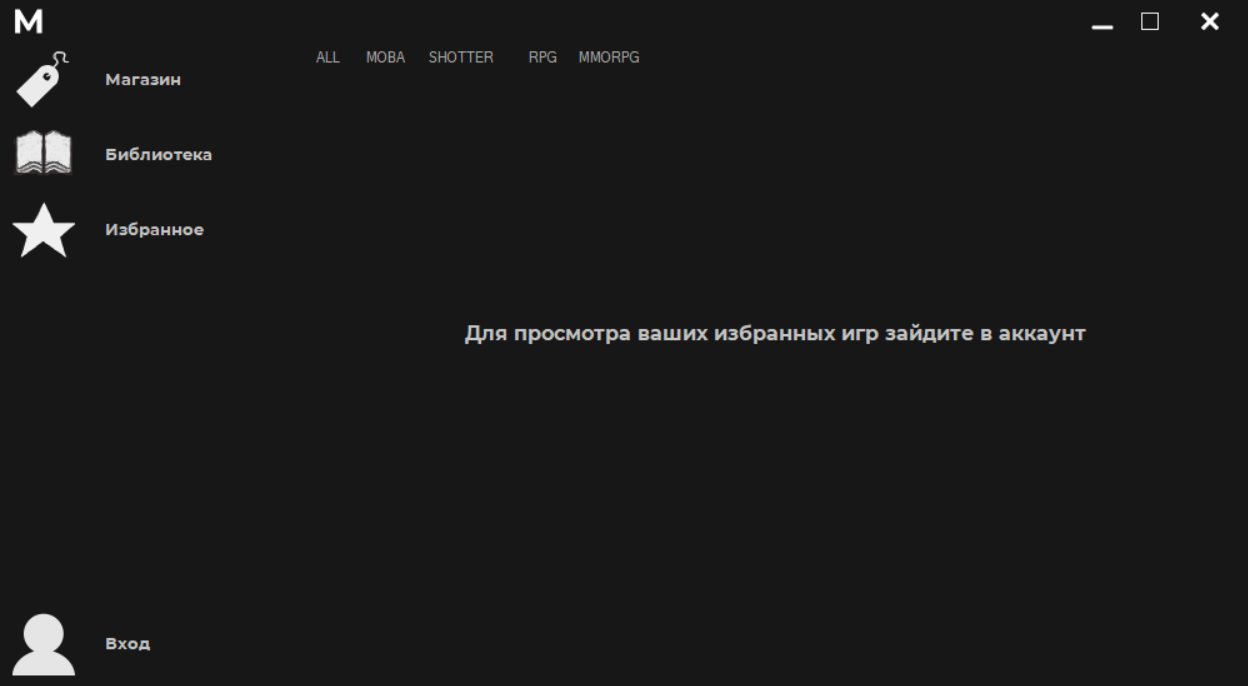
1. Вкладка «Библиотека» для неавторизированного пользователя
   * 1. Вкладка «Избранное» основной формы

На рисунке 10 изображена вкладка «Избранное» основной формы приложения для авторизированного пользователя.



1. Вкладка «Избранное» для авторизированного пользователя

На рисунке 11 изображена вкладка «Избранное» основной формы приложения для авторизированного пользователя.



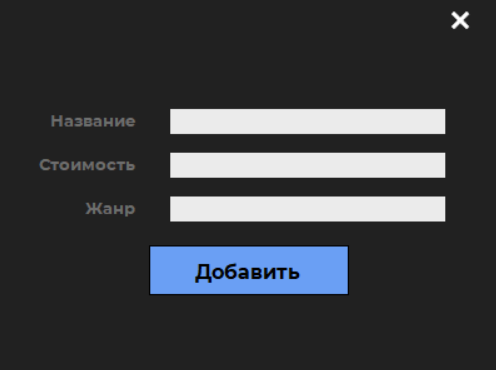
1. Вкладка «Избранное» для неавторизированного пользователя
   1. Форма авторизированного пользователя

На рисунке 12 изображена форма авторизированного пользователя.



1. Форма авторизированного пользователя
   1. Форма добавления игры

На рисунке 13 изображена форма добавления игры разработчиком.



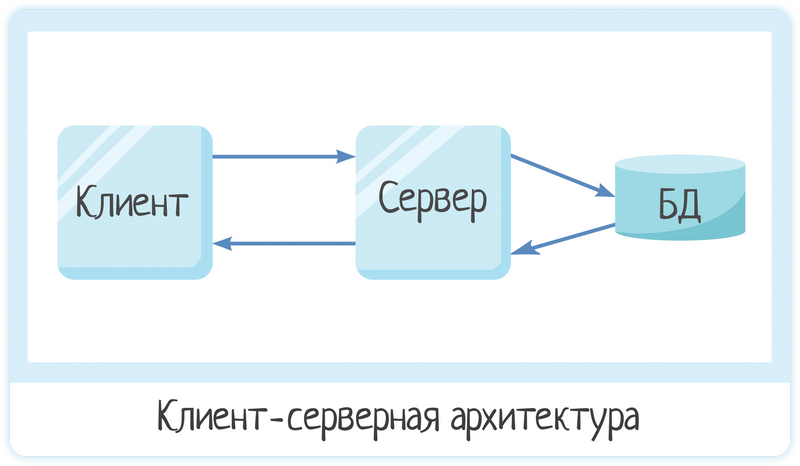
1. Форма добавление игры разработчиком
2. Исключительные ситуации
3. Исключительные ситуации

|  |  |  |
| --- | --- | --- |
| Форма | Ситуация | Следствие |
| Форма регистрации (Sign in) | Одно из полей ничего не содержит. | Всплывает подсказка «Поля должны быть заполнены» |
| Пароли не совпадают | Всплывает подсказка «Пароли не совпадают» |
| Введённый логин уже существует в БД | Всплывает подсказка «Пользователь с таким Email уже зарегистрирован» |
| Форма авторизации (Sign up) | Не удается зайти в систему, если:   * Поле «Логин» пусто; * Поле «Пароль» пусто; * Введённого логина нет в БД; * Пароль не соответствует существующему в БД. | Если введённого логина нет в БД или пароль не соответствует существующему в БД, то всплывает подсказка «Неверный логин или пароль». Если поля «Логин» или «Пароль» пустые, то кнопка Войти не активирована |
| Форма добавления игры | Жанр, к которому относят новую игру, отсутствует в БД | Жанр добавляется в БД |
| Одно из полей, кроме поля «Жанр», ничего не содержит | Всплывает подсказка «Необходимо ввести данные об игре» |

1. Реализация добавления, удаления, обновления

Добавление, удаление и обновление осуществляется согласно архитектуре «Клиент-Сервер», которая изображена на рисунке 14.

Клиент передает данные на сервер. Далее, сервер отправляет запрос в БД. БД же возвращает запрашиваемые данные серверу, а тот в свою очередь передает их клиенту. Передача, добавление, удаление и обновление данных осуществляется через взаимодействие пользователя с интерфейсом, триггеры которого, срабатывая, отправляют SQL запросы на сервер через код C#, а также ожидают ответного результата.



1. Схема архитектуры «Клиент-Сервер»
2. Разработанные запросы отражены в таблице 12.
3. Разработанные запросы

|  |  |
| --- | --- |
| Код запроса | Назначение |
| Select Count(\*) from Пользователи where Email = @email | Запрос для подсчёта количества пользователей с введенным email, для регистрации |
| Select \* from dbo.[GetGames](@email) | Выборка информации об играх, чтобы установить приобретены ли игры у пользователя или добавлены ли в избранное |
| Insert into Пользователи values (@email, @password, 'user', @nickname) | Добавление нового игрока в таблицу "Пользователи" |

Продолжение таблицы 13

|  |  |
| --- | --- |
| Select Count(\*) from Genres where Название = @genre | Проверка на наличие жанра в БД |
| Insert into Genres values (@genre, 0) | Добавления нового жанра в БД |
| Insert into [Магазин/Игры] values (@genre, @price, (Select Id from Genres where Название = @genre) | Добавление игры в БД |
| Select Count(\*) from Пользователи where (Email = @login And Пароль = @password) OR (Никнейм = @login And Пароль = @password) | Проверка на наличие пользователя в соответствии с введёнными данными |
| Select \* from Пользователи where (Email = @login And Пароль = @password) OR (Никнейм = @login And Пароль = @password) | Выборка данных об авторизированном пользователе |
| Insert into Библиотека values ((select ID from [Магазин/Игры] where Название = @gameName), (select ID from Пользователи where Email = @email), 0) | Добавление игры в библиотеку авторизированного пользователя |
| Delete from Избранное where ID\_Игры = (select ID from [Магазин/Игры] where Название = @gameName) AND ID\_Пользователя = (select ID from Пользователи where Email = @email) | Удаление игры из избранного авторизированного пользователя |
| Insert into Избранное values ((select ID from [Магазин/Игры] where Название = @gameName), (select ID from Пользователи where Email = @email)) | Добавление игры в избранное авторизированного пользователя |
| Select Никнейм from Пользователи where Email = @email | Выборка никнейма авторизированного пользователя |
| Select Роль from [Пользователи] where Email = @email | Выборка роли авторизированного пользователя |
| Select Название from Genres | Получение названия всех жанров, чтобы установить динамически их на панель |
| Select \* from dbo.[GetGames](@email) where Genre = (select id from Genres where Название = @gameName) | Выборка информации об играх, чтобы установить приобретены ли игры у пользователя или добавлены ли в избранное, но по определённому жанру |

Продолжение таблицы 13

|  |  |
| --- | --- |
| Update Библиотека set [Наиграно часов] = (select [Наиграно часов] from Библиотека where ID\_Игры = (select ID from [Магазин/Игры] where Название = @gameName) And ID\_Пользователя = (select ID from Пользователи where Email = @email))+1 where ID\_Игры = (select ID from [Магазин/Игры] where Название = @gameName) And ID\_Пользователя = (select ID from Пользователи where Email = @email) | Обновление количество часов в определённой игре у авторизированного пользователя |
| Select [Магазин/Игры].Название, Библиотека.[Наиграно часов] from [Магазин/Игры] Inner Join Библиотека ON [Магазин/Игры].ID = Библиотека.ID\_Игры where ID\_Пользователя = (select ID from Пользователи where Email = @email) | Выборка данных из сущности Библиотека, чтобы загрузить приобретённые игроком игры |

1. Обеспечение целостности БД
   1. Уникальность атрибутов

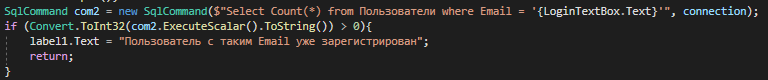
Во всех таблицах Пользователи, Магазин/Игры и Жанры присутствует уникальный атрибут, по которому связаны все отношения. В таблице 14 указаны уникальные поля в таблицах.

1. Уникальные поля

|  |  |
| --- | --- |
| Название таблицы | Уникальное поле |
| Пользователи | ID |
| Магазин/Игры | ID |
| Жанры | ID |

* 1. Проверка данных

У игроков не могут совпадать поле Email. Поэтому вводится дополнительная проверка при регистрации. Код изображён на рисунке 15



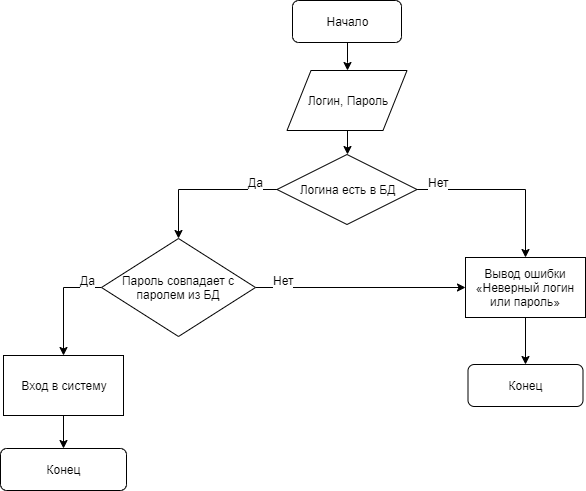
1. Ограничение повторяющихся значений email
2. Разграничение прав доступа и защита данных

hashbytes — хеш-функция формирования ключа, используемая для защищенного хранения паролей. С его помощью пароли всех пользователей хранятся в виде хеша, что позволяет хранить их в защищенном от кражи виде.

Так как пароли в базе данных хранятся в виде хеша, то на сервере происходит хеширование пароля, пришедшего с клиента. Затем Хеш-пароль сравнивается с полученным паролем из базы данных. По результатам проверки осуществляется отправка данных на клиентское приложение. Отправленные данные изменяют состояние клиентского приложения: либо появится ошибка о неправильно введенных данных, либо отобразится интерфейс в соответствии с ролью пользователя (см. рис. 1).

1. Разработка алгоритмов программных модулей
   1. Авторизация

На рисунке 16 изображена блок-схема процесса авторизации.



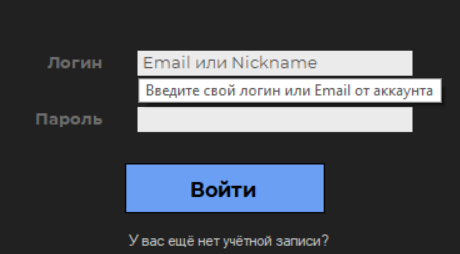
1. Процесс авторизации
   1. Регистрация

На рисунке 17 изображена блок-схема процесса регистрации пользователя.



1. Процесс регистрации
2. Справочная система

Для лучшего освоения пользователя в приложении на каждой форме были добавлены подсказки. Например, на рисунке 18 изображены подписи, которые указывают в какое поле какие данные нужно вводить.



1. Подсказки для ввода данных

Так же на форме авторизации находится сообщение для пользователя, который ещё не зарегистрирован, которое изображено на рисунке 19.

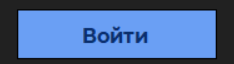


1. Сообщение для перехода на форму регистрации
2. Тестирование приложения

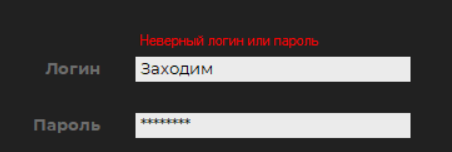
В таблице 15 описано поведение модуля авторизации при особых условиях.

1. Поведение модуля авторизации

|  |  |
| --- | --- |
| Действие | Результат |
| Все или одно поле - пустые. Нажать на кнопку «Войти». | Кнопка «Войти» не доступна для нажатия (рис. 20). |
| Ввести некорректный логин. Нажать на кнопку «Войти». | Появится ошибка «Неверный логин или пароль» (рис. 21). |
| Ввести некорректный пароль. Нажать на кнопку «Войти». | Появится ошибка «Неверный логин или пароль» (рис. 21). |
| Ввести корректные данные. Нажать на кнопку «Войти». | Откроется вкладка «Магазин» основной формы авторизированного пользователя (см. рис. 6) или форма разработчика (см рис. 7). |



1. Кнопка «Войти» недоступна

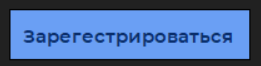


1. Ошибка «Неверный логин или пароль»

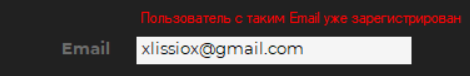
В таблице 16 описано поведение модуля регистрации при особых условиях.

1. Поведение модуля регистрации

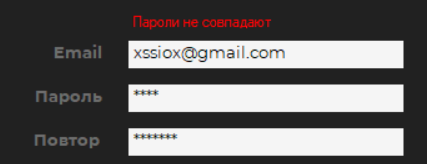
|  |  |
| --- | --- |
| Действие | Результат |
| Все или одно поле - пустые. Нажать на кнопку «Зарегистрироваться». | Кнопка «Зарегистрироваться» недоступна для нажатия (рис. 22). |
| Ввести существующий логин. Нажать «Зарегистрироваться». | Появится ошибка «Пользователь с таким Email уже зарегистрирован» (рис. 23). |
| Ввести разные символы в поля «Пароль» и «Повтор пароля»  Нажать «Зарегистрироваться». | Появится ошибка «Пароли не совпадают» (рис. 24). |
| Заполнить поля, ввести несуществующий в БД логин и нажать «Зарегистрироваться». | Будет создан новый пользователь ИС. Пользователь будет возвращён на форму авторизации (см. рис. 4) |



1. Кнопка «Зарегистрироваться» недоступна



1. Ошибка «Пользователь с таким Email уже зарегистрирован»

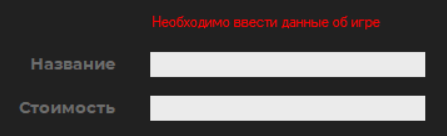


1. Ошибка «Пароли не совпадают»

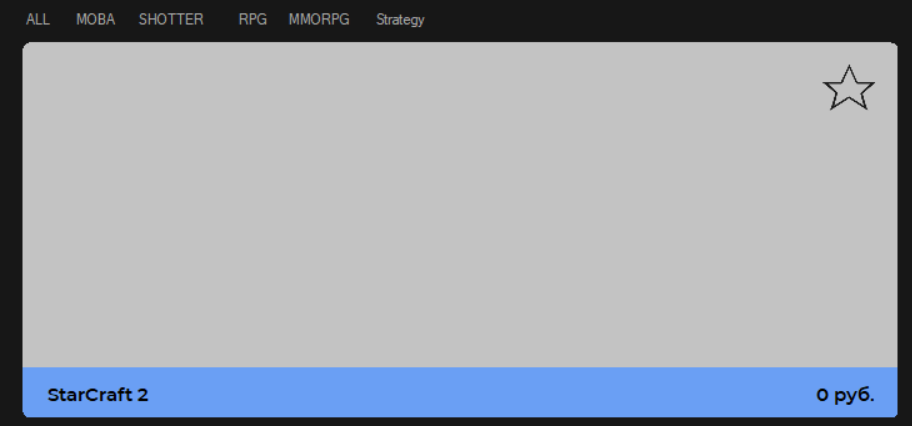
В таблице 17 описано поведение модуля приобретения пользователем игры.

1. Поведение модуля добавления игры

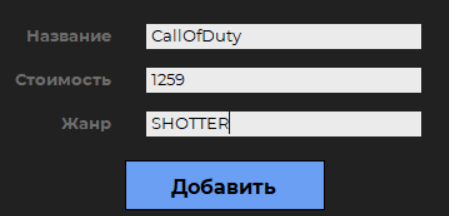
|  |  |
| --- | --- |
| Действие | Результат |
| Все или одно поле, кроме «Жанр» - пустые. Нажать на кнопку «Добавить». | Появится ошибка «Необходимо ввести данные об игре» (рис. 25). |
| Ввести в поле «Жанр» значение, которого нет в БД.  Заполнить поля «Название» и «Стоимость». Нажать на кнопку «Добавить» | Форма закроется, а жанр добавленной игры добавится в БД (рис. 26). |
| Ввести в поле «Жанр» существующим значением (рис. 27).  Заполнить поля «Название» и «Стоимость». Нажать на кнопку «Добавить» | Форма закроется, игра будет добавится как в жанр «ALL» (рис. 28), так и в жанр, который передали значению поля «Жанр». |



1. Ошибка «Необходимо ввести данные об игре»



1. Добавлен жанр «Strategy»



1. Добавление игры с существующим жанром в БД

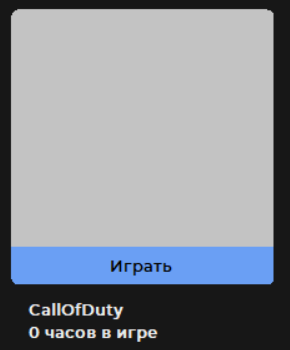


1. Игра добавлена в жанр «SHOTTER»

В таблице 18 описано поведение модуля приобретения игры (вкладка «Магазин» и «Избранное»).

1. Поведение модуля приобретения игры

|  |  |
| --- | --- |
| Действие | Результат |
| Нажать на ценник игры | Игра добавлена в библиотеку пользователя (рис. 29), а вместо ценника появляется сообщение «Приобретено» (рис. 30). |



1. Игра добавлена в библиотеку пользователя



1. Сообщение о приобретённой игре

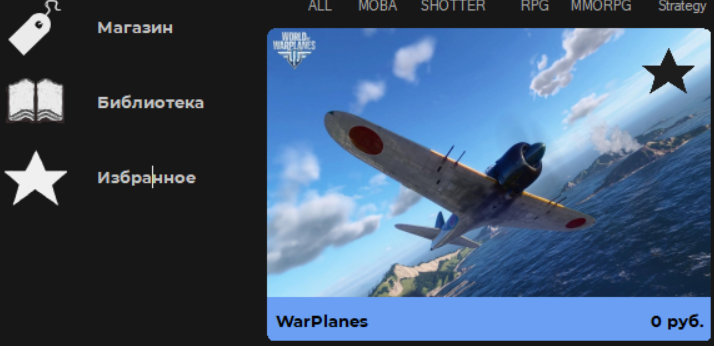
В таблице 19 описано поведение модуля добавления игры в избранное (вкладка «Магазин» и вкладка «Избранное»).

1. Поведение модуля приобретения игры

|  |  |
| --- | --- |
| Действие | Результат |
| Игра не находится в избранном пользователя. Нажать на значок «Звезда» на панели игры (рис. 31). | Игра добавлена в избранное пользователя (рис. 32), а «Звезда» заполняется цветом (рис. 33). |
| Игра находится в избранном пользователя. Нажать на значок «Звезда» на панели игры. | Игра удалена из избранного пользователя, а «Звезда» теряет заполнение (рис. 31). |



1. Значок «Звезда»



1. Игра добавлена в избранное пользователя



1. Значок «Звезда» при добавлении игры в избранное

ЗАКЛЮЧЕНИЕ

Выполняя данную курсовую работу, получил навыки проектирования, разработки и администрирования БД, а также проектирования, разработки клиентского приложения, навыки хеширования данных и навыки работы с инструментами Microsoft SQL Server и Visual Studio.

Для улучшения приложения рекомендую выводить количество игр жанра, который выбрал во вкладке «Магазин» пользователь. Также игры, приобретённые пользователем, должны запускаться через библиотеку. Помимо этого, было бы неплохо, если бы пользователь мог изменять пароль или Никнейм.

СПИСОК ИСПОЛЬЗВОАННЫХ ИСТОЧНИКОВ

1. Документация MS SQL: https://docs.microsoft.com/ru-ru/sql/?view=sql-server-ver15.
2. SQLCommand.ExecuteScalar Метод. ⎯ URL: https://docs.microsoft.com/ru-ru/dotnet/api/system.data.sqlclient.sqlcommand.executescalar?view=dotnet-plat-ext-5.
3. SQLCommand.ExecuteReader Метод. ⎯ URL: https://docs.microsoft.com/ru-ru/dotnet/api/system.data.sqlclient.sqlcomm-and.executereader?view=dotnet-plat-ext-5.0.
4. SQLCammand.ExecuteNonQuery метод. – URL: https://docs.microsoft.com/ru-ru/dotnet/api/system.data.sqlclient.sqlcommand.executenonquery?view=netframework-4.8.
5. Синтаксический анализ строк даты и времени в .NET. ⎯ URL: https://-docs.microsoft.com/ru-ru/dotnet/standard/base-types/parsing-datetime.
6. Анонимные методы на языке C#. – URL: https://metanit.com/sharp/tutorial-/3.15.php.
7. Пользовательский элемент управления Windows Forms. – URL: https://-docs.microsoft.com/ru-ru/dotnet/desktop/winforms/controls/custom?view=n-etdesktop-5.0.
8. Основы Linq в языке C#. - URL: https://metanit.com/sharp/tutorial/15.1.php.
9. Регулярные выражения в языке C#. – URL: https://docs.microsoft.com/ru-ru/dotnet/standard/base-types/regular-expressions.
10. Функции, используемых в базах данных SQL. – URL: https://docs.microsoft.com/ru-ru/sql/t-sql/functions/functions?view=sql-server-ver15.
11. Триггеры и их использование в базах данных SQL. – URL: https://docs.microsoft.com/ru-ru/sql/t-sql/statements/create-trigger-transact-sql?view=sql-server-ver15.
12. Свойства элементов управления Windows Forms. – URL: https://docs.microsoft.com/ru-ru/dotnet/desktop/winforms/controls/properties-in-windows-forms-controls?view=netframeworkdesktop-4.8.
13. Появление всплывающий подсказок Windows Forms. – URL:

https://www.cyberforum.ru/windows-forms/thread93002.html.

ПРИЛОЖЕНИЕ А Характеристики программного и аппаратного обеспечения

1. Аппаратное обеспечение

Для использования ИС потребуется следующие элементы аппаратного обеспечения:

* Компьютер;
* Мышь;
* Клавиатура.

1. Программное обеспечение

Для использования ИС потребуется следующие элементы программного обеспечения: фреймворк .NET Framework версии 4.7.2 и MS SQL Express.

ПРИЛОЖЕНИЕ Б Руководство системного программиста

СОГЛАСОВАНО

преподаватель ТРЗиБД

должность

А.О. Резанова

подпись, инициалы и фамилия

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г.

УТВЕРЖДЕНО

директор ИСПО Политехнического колледжа

должность

Р.А. Байбиков

подпись, инициалы и фамилия

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г.

MaFriGames

**Руководство системного программиста**

**ЛИСТ УТВЕРЖДЕНИЯ**

**Листов 9**

СОГЛАСОВАНО

Преподаватель ТРЗБД

должность

А. О. Резанова

подпись, инициалы и фамилия

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г.

Представители предприятия разработчика

должность

подпись, инициалы и фамилия

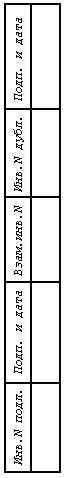
«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г.

должность

подпись, инициалы и фамилия

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г.

2021



MaFriGames

**Руководство системного программиста**

**Листов 9**

2021

**АННОТАЦИЯ**

В данном программном документе приведено руководство системного программиста по установке специального программного обеспечения «MaFriGames».

В данном программном документе, в разделе «Общие сведения о программе» указаны назначение и функции программы и сведения о технических и программных средствах, обеспечивающих выполнение данной программы, а также требования к персоналу.

В разделе «Структура программы» приведены сведения о структуре программы, ее составных частях, о связях между составными частями и о связях с другими программами.

В данном программном документе, в разделе «Настройка программы» приведено описание действий по настройке программы на условия конкретного применения (настройка на состав технических и программных средств, выбор функций и др.).

В разделе «Проверка программы» приведено описание способов проверки, позволяющих дать общее заключение о работоспособности программы (контрольные примеры, методы прогона, результаты).

В данном программном документе, в разделе «Сообщения системному программисту» указаны тексты сообщений, выдаваемых в ходе выполнения настройки, проверки программы, а также в ходе выполнения программы, описание их содержания и действий, которые необходимо предпринять по этим сообщениям.

Оформление программного документа «Руководство системного программиста» произведено по требованиям ЕСПД (ГОСТ 19.104-78, ГОСТ 91.106-78, ГОСТ 19.503-79)

1) ГОСТ 19.104-78 ЕСПД. Основные надписи

2) ГОСТ 19.106-78 ЕСПД. Общие требования к программным документам, выполненным печатным способом

3) ГОСТ 19.503-79 ЕСПД. Руководство системного программиста. Требования к содержанию и оформлению

**СОДЕРЖАНИЕ**

[1. ОБЩИЕ СВЕДЕНИЯ О ПРОГРАММЕ 4](#_Toc90160025)

[1.1 Назначение программы 4](#_Toc90160026)

[1.2 Функции программы 4](#_Toc90160027)

[1.3 Минимальный состав программных средств 4](#_Toc90160028)

[2. СТРУКТУРА ПРОГРАММЫ 5](#_Toc90160029)

[2.1 Сведения о структуре программы 5](#_Toc90160030)

[2.3 Сведения о связях между составными частями программы 5](#_Toc90160031)

[2.4 Сведения о связях с другими программами 5](#_Toc90160032)

[3. НАСТРОЙКА ПРОГРАММЫ 6](#_Toc90160033)

[3.1 Настройка на состав технических средств 6](#_Toc90160034)

[3.2 Настройка на состав программных средств 6](#_Toc90160035)

[4. УСТАНОВКА ПРОГРАММЫ 6](#_Toc90160036)

[4.1 Лицензионное соглашение 6](#_Toc90160037)

[4.2 Инсталляция MaFriGames 6](#_Toc90160038)

[5. ПРОВЕРКА ПРОГРАММЫ 7](#_Toc90160039)

[5.1 Описание способов проверки 7](#_Toc90160040)

[5.1.1 Проверка работоспособности программы 7](#_Toc90160041)

[6. СООБЩЕНИЕ СИСТЕМНОМУ ПРОГРАММИСТУ 7](#_Toc90160042)

1. **ОБЩИЕ СВЕДЕНИЯ О ПРОГРАММЕ**

**1.1 Назначение программы**

MaFriGames — это программное обеспечение, предназначенное для установки на компьютер, функционирующий под операционной системой Windows 10, которое позволяет пользователю приобретать игры одноименной компании.

* 1. **Функции программы**

1. регистрация
2. приобретение игр
3. настройки аккаунта
4. удаление игр из избранного
5. для разработчика: добавление игр
6. поддержка Windows 10
   1. **Минимальный состав программных средств**

Требования к OC:

* + - * Microsoft SQL Server 18.
      * Windows 10 (64-разрядная).

1. **СТРУКТУРА ПРОГРАММЫ**
   1. **Сведения о структуре программы**

Программа структурно состоит из:

* + 1. Ядро программы.
    2. Интерфейс программы.
    3. Модуля работы с базой данных программы.
  1. **Сведения о составных частях программы**

Ядро программы - механизмы работы с файлами конфигурации, механизмы сетевого взаимодействия и обмена данными, механизмы загрузки модулей устройств, передача информации между модулями, разграничение прав доступа к функциям модулей.

Интерфейс программы - механизмы взаимодействия пользователя системы: создание и редактирование векторных и растровых изображение, анимации, сканирование изображений, экспорт готовых изображений в наиболее популярные форматы.

Модуля работы с базой данных программы – механизмы работы с базой данных, внесения в нее новых данных, удаления и обновления существующих, а также механизм вывода данных базы данных в интерфейсах программы.

* 1. **Сведения о связях между составными частями программы**

Ядро программы получает от пользователя данные и передает их Модулю работы с базой данных.

Модуль базы данных преобразует данные в формат данных программы и после преобразования передает их Интерфейсу программы.

Интерфейс программы, получив команды от пользователя, передает эти данные Модулю баз данных, а также передает данные об изменениях в Ядро программы.

* 1. **Сведения о связях с другими программами**

Связь MaFriGames с другим программным обеспечением отсутствует.

1. **НАСТРОЙКА ПРОГРАММЫ**
   1. **Настройка на состав технических средств**

MaFriGames не требует каких-либо настроек на состав технических средств.

* 1. **Настройка на состав программных средств**

MaFriGames не требует каких-либо настроек на состав программных средств.

1. **УСТАНОВКА ПРОГРАММЫ**
   1. **Лицензионное соглашение**

Лицензионное соглашение отсутствует.

* 1. **Инсталляция MaFriGames**

Для запуска программного обеспечение «MaFriGames» не требуется инсталляция.

Для использования программного обеспечения необходимо запустить исполняемый файл mafrigames.exe.

1. **ПРОВЕРКА ПРОГРАММЫ**

Для проверки работоспособности программного обеспечения необходимо убедиться в том, что при вызове программы осуществилась ее загрузка, и отсутствуют сообщения операционной системы о невозможности выполнения программы.

**5.1 Описание способов проверки**

**5.1.1 Проверка работоспособности программы**

Работоспособность «MaFriGames» проверяется запуском программы. Запустить программу, откроется окно (рис.34).



Рисунок 34 – Окно при запуске приложения

1. **СООБЩЕНИЕ СИСТЕМНОМУ ПРОГРАММИСТУ**

Т.к. программное обеспечение MaFriGames не требует установки, то системному программисту не будет выведено ни одного сообщения.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ** | | | | | | | | | |
| Номера листов (страниц) | | | | | Всего листов (страниц) в докум. | № документа | Входящий  № сопрово дительного документа и дата | Подп. | Дата |
| Изм. | Изменен-ных | Заменен-ных | Новых | Анулиро-ванных |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

ПРИЛОЖЕНИЕ В Листинг SQL-скрипта

-- Создание БД

CREATE DATABASE [MaFriGames]

-- Создание таблиц БД

CREATE TABLE [dbo].[Genres](

[Id] [int] IDENTITY(1,1) NOT NULL primary key,

[Название] [nvarchar](128) NULL,

[Количество игр] [int] NULL

)

CREATE TABLE [dbo].[Пользователи](

[ID] [int] IDENTITY(1,1) NOT NULL primary key,

[Email] [nvarchar](255) NOT NULL,

[Пароль] [nvarchar](255) NOT NULL,

[Роль] [nvarchar](255) NOT NULL,

[Никнейм] [nvarchar](127) NULL

)

CREATE TABLE [dbo].[Магазин/Игры](

[ID] [int] IDENTITY(1,1) NOT NULL primary key,

[Название] [nvarchar](255) NOT NULL,

[Цена] [int] NOT NULL,

[Genre] [int] NULL foreign key references Genres(ID)

)

CREATE TABLE [dbo].[Избранное](

[ID\_Игры] [int] NOT NULL foreign key references [Магазин/Игры](ID),

[ID\_Пользователя] [int] NOT NULL foreign key references Пользователи(ID)

)

CREATE TABLE [dbo].[Библиотека](

[ID\_Игры] [int] NOT NULL foreign key references [Магазин/Игры](ID),

[ID\_Пользователя] [int] NOT NULL foreign key references Пользователи(ID),

[Наиграно часов] [int] NULL

)

-- Функция для выдачи информации о приобретённых и избранных играх пользователя  
create function GetGames(@email as varchar(255)) returns @result table(name varchar(255), cost varchar(255), isBuied bit, isFavourite bit, Genre int)

begin

declare cur cursor

for select ID, Название, Цена, Genre from [Магазин/Игры]

declare @id int, @Name varchar(255), @Price int, @Genre int

open cur

fetch next from cur into @id, @Name, @Price, @Genre

while(@@FETCH\_STATUS = 0)

begin

declare @isbuied bit, @isfavourite bit

if((Select Count(\*) from Библиотека where ID\_Пользователя = (select ID from Пользователи where Email = @email) AND ID\_Игры = @id) > 0)

set @isbuied = 1

else

set @isbuied = 0

if((Select Count(\*) from Избранное where ID\_Пользователя = (select ID from Пользователи where Email = @email) AND ID\_Игры = @id) > 0)

set @isfavourite = 1

else

set @isfavourite = 0

insert into @result values(@Name, @Price, @isbuied, @isfavourite, @Genre)

fetch next from cur into @id, @Name, @Price, @Genre

end

return

end

-- Триггер, который обновляют количество игр определённого жанра (жанра игры, которую добавили)

create trigger GenresTrigger On [Магазин/Игры]

after insert

as

declare @Id int

select @Id = Genre from inserted

Update Genres set [Количество игр] = (select COUNT(\*) from [Магазин/Игры] where Genre = @Id) where Id = @Id

ПРИЛОЖЕНИЕ Г Листинг приложения

1. LibraryComponent.cs

using System;

using System.Drawing;

using System.Data.SqlClient;

using System.IO;

using System.Windows.Forms;

namespace Kursach

{

public partial class LibraryComponent : UserControl

{

private string \_gameName

{

get

{

return GameName.Text;

}

set

{

GameName.Text = value;

}

}

private int timeInGame;

private string \_timeInGame

{

get

{

return TimeInGame.Text;

}

set

{

timeInGame = Convert.ToInt32(value.Substring(0, 1));

TimeInGame.Text = value;

}

}

private string backgroundPicture

{

set

{

try {

MainPanel.BackgroundImage = Image.FromFile(@"..\..\Resources\" + value + ".jpg");

}

catch

{

MainPanel.BackColor = Color.FromArgb(195, 195, 195);

}

}

}

public LibraryComponent(string gn, string tin)

{

InitializeComponent();

SetRoundedShape(MainPanel, 10, true);

\_gameName = gn;

\_timeInGame = tin;

backgroundPicture = gn;

}

private void ReadLogin(out string login)

{

using (StreamReader sr = new StreamReader(@"..\userData.txt"))

{

login = sr.ReadLine();

}

}

static void SetRoundedShape(Control control, int radius, bool isMainForm)

{

System.Drawing.Drawing2D.GraphicsPath path = new System.Drawing.Drawing2D.GraphicsPath();

if (isMainForm)

{

path.AddLine(radius, 0, control.Width - radius, 0);

path.AddArc(control.Width - radius, 0, radius, radius, 270, 90);

}

path.AddLine(control.Width, radius, control.Width, control.Height - radius);

path.AddArc(control.Width - radius, control.Height - radius, radius, radius, 0, 90);

path.AddLine(control.Width - radius, control.Height, radius, control.Height);

path.AddArc(0, control.Height - radius, radius, radius, 90, 90);

path.AddLine(0, control.Height - radius, 0, radius);

if (isMainForm)

{

path.AddArc(0, 0, radius, radius, 180, 90); ;

}

control.Region = new Region(path);

}

private void PlayButton\_Click(object sender, EventArgs e)

{

using (SqlConnection connection = new SqlConnection(GlobalFields.CONNECTION))

{

connection.Open();

string login;

ReadLogin(out login);

SqlCommand com1 = new SqlCommand($"Update Библиотека set [Наиграно часов] = (select [Наиграно часов] from Библиотека where ID\_Игры = (select ID from [Магазин/Игры] where Название = '{\_gameName}') And ID\_Пользователя = (select ID from Пользователи where Email = '{login}'))+1 where ID\_Игры = (select ID from [Магазин/Игры] where Название = '{\_gameName}') And ID\_Пользователя = (select ID from Пользователи where Email = '{login}')", connection);

com1.ExecuteNonQuery();

}

\_timeInGame = ++timeInGame + " часов в игре";

}

}

}

1. ShopAssortiment.cs

using System;

using System.Drawing;

using System.IO;

using System.Data.SqlClient;

using System.Windows.Forms;

namespace Kursach

{

public partial class ShopAssortiment : UserControl

{

private bool isBuied;

private bool \_isBuied

{

get

{

return isBuied;

}

set

{

if (value)

{

Price.Text = "Приобретено";

}

isBuied = value;

}

}

private bool isFavourite;

private bool \_isFavourite

{

get

{

return isFavourite;

}

set

{

if (value)

{

panel2.BackgroundImage = Image.FromFile(@"..\..\Resources\FavouriteMiniChecked.png");

}

isFavourite = value;

}

}

private string \_gameName {

get

{

return GameName.Text;

}

set

{

GameName.Text = value;

}

}

private string price;

private string \_price

{

get

{

return price;

}

set

{

if (!isBuied)

{

Price.Text = value;

}

}

}

private string backgroundPicture

{

set

{

try

{

MainPanel.BackgroundImage = Image.FromFile(@"..\..\Resources\" + value + ".jpg");

}

catch

{

MainPanel.BackColor = Color.FromArgb(195, 195, 195);

}

}

}

public ShopAssortiment(string gn, string pr, bool isbyed, bool isfavourite)

{

InitializeComponent();

panel2.BackColor = Color.Transparent;

SetRoundedShape(MainPanel, 10, true);

SetRoundedShape(panel1, 10, false);

\_isBuied = isbyed;

\_isFavourite = isfavourite;

\_gameName = gn;

\_price = pr;

price = pr;

backgroundPicture = gn;

}

private void ReadLogin(out string login)

{

using (StreamReader sr = new StreamReader(@"..\userData.txt"))

{

login = sr.ReadLine();

}

}

static void SetRoundedShape(Control control, int radius, bool isMainForm)

{

System.Drawing.Drawing2D.GraphicsPath path = new System.Drawing.Drawing2D.GraphicsPath();

if (isMainForm)

{

path.AddLine(radius, 0, control.Width - radius, 0);

path.AddArc(control.Width - radius, 0, radius, radius, 270, 90);

}

path.AddLine(control.Width, radius, control.Width, control.Height - radius);

path.AddArc(control.Width - radius, control.Height - radius, radius, radius, 0, 90);

path.AddLine(control.Width - radius, control.Height, radius, control.Height);

path.AddArc(0, control.Height - radius, radius, radius, 90, 90);

path.AddLine(0, control.Height - radius, 0, radius);

if (isMainForm)

{

path.AddArc(0, 0, radius, radius, 180, 90); ;

}

control.Region = new Region(path);

}

private void Price\_Click(object sender, EventArgs e)

{

if (!GlobalFields.isConnected)

{

Settings settings = new Settings();

settings.ShowDialog();

return;

}

using (SqlConnection connection = new SqlConnection(GlobalFields.CONNECTION))

{

connection.Open();

string login;

ReadLogin(out login);

SqlCommand com1 = new SqlCommand($"Insert into Библиотека values ((select ID from [Магазин/Игры] where Название = '{\_gameName}'), (select ID from Пользователи where Email = '{login}'), 0)", connection);

com1.ExecuteNonQuery();

}

\_isBuied = true;

}

private void Price\_MouseEnter(object sender, EventArgs e)

{

\_price = "Приобрести";

}

private void Price\_MouseLeave(object sender, EventArgs e)

{

\_price = price;

}

private void Panel2\_MouseEnter(object sender, EventArgs e)

{

if (isFavourite) { return; }

panel2.BackgroundImage = Image.FromFile(@"..\..\Resources\FavouriteMiniHov.png");

}

private void Panel2\_MouseLeave(object sender, EventArgs e)

{

if (isFavourite) { return; }

panel2.BackgroundImage = Image.FromFile(@"..\..\Resources\FavouriteMini.png");

}

private void Panel2\_Click(object sender, EventArgs e)

{

if (!GlobalFields.isConnected)

{

Settings settings = new Settings();

settings.ShowDialog();

return;

}

using (SqlConnection connection = new SqlConnection(GlobalFields.CONNECTION))

{

connection.Open();

string login;

ReadLogin(out login);

string sql = isFavourite ? $"Delete from Избранное where ID\_Игры = (select ID from [Магазин/Игры] where Название = '{\_gameName}') AND ID\_Пользователя = (select ID from Пользователи where Email = '{login}')" : $"Insert into Избранное values ((select ID from [Магазин/Игры] where Название = '{\_gameName}'), (select ID from Пользователи where Email = '{login}'))";

SqlCommand com1 = new SqlCommand(sql, connection);

com1.ExecuteNonQuery();

}

\_isFavourite = !\_isFavourite;

}

}

}

1. Settings.cs  
     
   using System;

using System.Drawing;

using System.IO;

using System.Data.SqlClient;

using System.Windows.Forms;

namespace Kursach

{

public partial class Settings : Form

{

Point lastPoint;

MainForm mainForm;

bool isPasswordChecked;

bool isLoginChecked;

public Settings()

{

InitializeComponent();

panel2.Visible = GlobalFields.isConnected;

mainForm = MainForm.Instantiate;

SetRoundedShape(IconPanel, 64);

if (GlobalFields.isConnected)

{

NickName.Text = MainForm.Instantiate.EnterTextBox.Text;

using (StreamReader sr = new StreamReader(@"..\userData.txt"))

{

for (int i = 1; !sr.EndOfStream ; i++)

{

if (i == 2)

{

IconPanel.BackgroundImage = Image.FromFile(sr.ReadLine());

continue;

}

sr.ReadLine();

}

}

}

}

static void SetRoundedShape(Control control, int radius)

{

System.Drawing.Drawing2D.GraphicsPath path = new System.Drawing.Drawing2D.GraphicsPath();

path.AddLine(radius, 0, control.Width - radius, 0);

path.AddArc(control.Width - radius, 0, radius, radius, 270, 90);

path.AddLine(control.Width, radius, control.Width, control.Height - radius);

path.AddArc(control.Width - radius, control.Height - radius, radius, radius, 0, 90);

path.AddLine(control.Width - radius, control.Height, radius, control.Height);

path.AddArc(0, control.Height - radius, radius, radius, 90, 90);

path.AddLine(0, control.Height - radius, 0, radius);

path.AddArc(0, 0, radius, radius, 180, 90);

control.Region = new Region(path);

}

private void UpsidePanel\_MouseMove(object sender, MouseEventArgs e)

{

if (e.Button == MouseButtons.Left)

{

this.Left += e.X - lastPoint.X;

this.Top += e.Y - lastPoint.Y;

}

}

private void UpsidePanel\_MouseDown(object sender, MouseEventArgs e)

{

lastPoint = new Point(e.X, e.Y);

}

private void CloseButton\_Click(object sender, EventArgs e)

{

Close();

}

private void Button1\_Click(object sender, EventArgs e)

{

EnterBD();

}

private void EnterBD()

{

using (SqlConnection connection = new SqlConnection(GlobalFields.CONNECTION))

{

connection.Open();

SqlCommand com1 = new SqlCommand($"Select Count(\*) from Пользователи where (Email = '{LoginTextBox.Text}' And Пароль = '{PasswordTextBox.Text}') OR (Никнейм = '{LoginTextBox.Text}' And Пароль = '{PasswordTextBox.Text}') ", connection);

if (Convert.ToInt32(com1.ExecuteScalar().ToString()) == 0)

{

issues.Text = "Неверный логин или пароль";

return;

}

SqlCommand com2 = new SqlCommand($"Select \* from Пользователи where (Email = '{LoginTextBox.Text}' And Пароль = '{PasswordTextBox.Text}') OR (Никнейм = '{LoginTextBox.Text}' And Пароль = '{PasswordTextBox.Text}') ", connection);

SqlDataReader Reader;

Reader = com2.ExecuteReader();

Reader.Read();

mainForm.EnterTextBox.Text = Reader[4].ToString();

using (StreamWriter sw = new StreamWriter(@"..\userData.txt"))

{

sw.WriteLine($"{Reader[1]}");

}

}

GlobalFields.isConnected = true;

mainForm.SetShopAssortiment();

Close();

}

private void Label2\_Click(object sender, EventArgs e)

{

Registration registration = new Registration(this);

registration.ShowDialog();

Close();

}

private void PasswordTextBox\_KeyDown(object sender, KeyEventArgs e)

{

if(e.KeyCode == Keys.Enter)

{

EnterBD();

}

}

private void Button2\_Click(object sender, EventArgs e)

{

using (StreamWriter sw = new StreamWriter(@"..\userData.txt"))

{

sw.WriteLine(" ");

}

GlobalFields.isConnected = false;

mainForm.EnterTextBox.Text = "Вход";

mainForm.SetShopAssortiment();

Close();

}

private void IconPanel\_Click(object sender, EventArgs e)

{

OpenFileDialog OPF = new OpenFileDialog();

OPF.Multiselect = true;

if (OPF.ShowDialog() == DialogResult.OK)

{

IconPanel.BackgroundImage = Image.FromFile(OPF.FileName);

}

using (StreamWriter sw = new StreamWriter(@"..\userData.txt", true))

{

sw.WriteLine(OPF.FileName);

}

}

private void LoginTextBox\_Enter(object sender, EventArgs e)

{

LoginTextBox.Text = "";

LoginTextBox.ForeColor = Color.FromArgb(0,0,0);

}

private void LoginTextBox\_Leave(object sender, EventArgs e)

{

if (LoginTextBox.Text == "") {

LoginTextBox.Text = "Email или Nickname";

LoginTextBox.ForeColor = Color.FromArgb(88, 88, 88);

}

}

private void LoginTextBox\_MouseEnter(object sender, EventArgs e)

{

ToolTip t = new ToolTip();

t.SetToolTip(LoginTextBox, "Введите свой логин или Email от аккаунта");

}

private void LoginTextBox\_TextChanged(object sender, EventArgs e)

{

if (string.IsNullOrWhiteSpace(LoginTextBox.Text))

{

isLoginChecked = false;

button1.Enabled = false;

return;

}

isLoginChecked = true;

if (isLoginChecked && isPasswordChecked)

{

button1.Enabled = true;

}

}

private void PasswordTextBox\_TextChanged(object sender, EventArgs e)

{

if (string.IsNullOrWhiteSpace(PasswordTextBox.Text))

{

isPasswordChecked = false;

button1.Enabled = false;

return;

}

isPasswordChecked = true;

if (isPasswordChecked && isLoginChecked)

{

button1.Enabled = true;

}

}

}

}

1. Registration.cs

using System;

using System.Data.SqlClient;

using System.Windows.Forms;

namespace Kursach

{

public partial class Registration : Form

{

Settings settings;

bool isPasswordChecked;

bool isLoginChecked;

bool isNickNameChecked;

public Registration(Settings set)

{

InitializeComponent();

settings = set;

}

private void Button1\_Click(object sender, EventArgs e)

{

if (!PasswordTextBox.Text.Equals(CheckPasswortTextBox.Text))

{

label1.Text = "Пароли не совпадают";

return;

}

if(string.IsNullOrWhiteSpace(PasswordTextBox.Text) || string.IsNullOrWhiteSpace(LoginTextBox.Text) || string.IsNullOrWhiteSpace(NicknameTextBox.Text))

{

label1.Text = "Поля должны быть заполнены";

return;

}

using (SqlConnection connection = new SqlConnection(GlobalFields.CONNECTION))

{

connection.Open();

SqlCommand com2 = new SqlCommand($"Select Count(\*) from Пользователи where Email = '{LoginTextBox.Text}'", connection);

if (Convert.ToInt32(com2.ExecuteScalar().ToString()) > 0){

label1.Text = "Пользователь с таким Email уже зарегистрирован";

return;

}

SqlCommand com1 = new SqlCommand($"Insert into Пользователи values ('{LoginTextBox.Text}', '{PasswordTextBox.Text}', 'user', '{NicknameTextBox.Text}') ", connection);

com1.ExecuteNonQuery();

}

Close();

settings.Show();

}

private void CloseButton\_Click(object sender, EventArgs e)

{

Close();

}

private void LoginTextBox\_TextChanged(object sender, EventArgs e)

{

if (string.IsNullOrWhiteSpace(LoginTextBox.Text))

{

isLoginChecked = false;

button1.Enabled = false;

return;

}

isLoginChecked = true;

if (isLoginChecked && isPasswordChecked && isNickNameChecked)

{

button1.Enabled = true;

}

}

private void PasswordTextBox\_TextChanged(object sender, EventArgs e)

{

if (string.IsNullOrWhiteSpace(PasswordTextBox.Text))

{

isPasswordChecked = false;

button1.Enabled = false;

return;

}

isPasswordChecked = true;

if (isPasswordChecked && isLoginChecked && isNickNameChecked)

{

button1.Enabled = true;

}

}

private void NicknameTextBox\_TextChanged(object sender, EventArgs e)

{

if (string.IsNullOrWhiteSpace(PasswordTextBox.Text))

{

isNickNameChecked = false;

button1.Enabled = false;

return;

}

isNickNameChecked = true;

if (isPasswordChecked && isLoginChecked && isNickNameChecked)

{

button1.Enabled = true;

}

}

}

}

1. GlobalFields.cs

namespace Kursach

{

class GlobalFields

{

public static readonly string CONNECTION = @"Data Source=LAPTOP-MTT4SSE6\SQLEXPRESS; Initial Catalog=Practica;Integrated Security=True";

public static bool isConnected;

}

}

1. Form1.cs

using System;

using System.Drawing;

using System.Data.SqlClient;

using System.IO;

using System.Windows.Forms;

namespace Kursach

{

public partial class MainForm : Form

{

Point lastPoint;

public static MainForm Instantiate;

public MainForm()

{

InitializeComponent();

CheckForEnter();

SetShopAssortiment();

SetRoundedShape(panel2, 64);

Instantiate = this;

}

private void ReadLogin(out string login)

{

using (StreamReader sr = new StreamReader(@"..\userData.txt"))

{

login = sr.ReadLine();

}

}

private void CheckForEnter()

{

string login;

ReadLogin(out login);

if (!login.Equals(" ")) {

GlobalFields.isConnected = true;

using (SqlConnection connection = new SqlConnection(GlobalFields.CONNECTION))

{

connection.Open();

SqlCommand com = new SqlCommand($"Select Никнейм from Пользователи where Email = '{login}'", connection);

EnterTextBox.Text = com.ExecuteScalar().ToString();

}

}

}

public void SetShopAssortiment()

{

Clear();

IssiuesTextBox.Visible = false;

panel2.Visible = false;

using (SqlConnection connection = new SqlConnection(GlobalFields.CONNECTION))

{

connection.Open();

string login;

ReadLogin(out login);

if (GlobalFields.isConnected)

{

SqlCommand com1 = new SqlCommand($"Select Роль from [Пользователи] where Email = '{login}'", connection);

if (com1.ExecuteScalar().ToString() == "admin")

{

panel2.Visible = true;

}

}

SqlCommand com2 = new SqlCommand($"Select \* from dbo.[GetGames]('{login}')", connection);

SqlDataReader Reader1;

Reader1 = com2.ExecuteReader();

for (int i = 0; Reader1.Read(); i++)

{

string gameId = Reader1[0].ToString();

ShopAssortiment shop = new ShopAssortiment(Reader1[0].ToString(), Reader1[1].ToString() + " руб.", Convert.ToInt32(Reader1[2]) == 1, Convert.ToInt32(Reader1[3]) == 1);

shop.Location = new Point(50, 30 + i \* 330);

CenterPanel.Controls.Add(shop);

}

Reader1.Close();

SqlCommand com3 = new SqlCommand("Select Название from Genres", connection);

Reader1 = com3.ExecuteReader();

for (int i = 0; Reader1.Read(); i++)

{

string Name = Reader1[0].ToString();

Label label = new Label();

label.Size = new Size(Name.Length\*10, 16);

label.Click += (sender, e) => {

using (SqlConnection connection2 = new SqlConnection(GlobalFields.CONNECTION))

{

connection2.Open();

ReadLogin(out login);

if (GlobalFields.isConnected)

{

SqlCommand com1 = new SqlCommand($"Select Роль from [Пользователи] where Email = '{login}'", connection2);

if (com1.ExecuteScalar().ToString() == "admin")

{

panel2.Visible = true;

}

}

Clear();

SqlCommand com4 = new SqlCommand($"Select \* from dbo.[GetGames]('{login}') where Genre = (select id from Genres where Название = '{Name}')", connection2);

if(Name == "ALL") { com4.CommandText = $"Select \* from dbo.[GetGames]('{login}')"; }

SqlDataReader Reader2;

Reader2 = com4.ExecuteReader();

for (int j = 0; Reader2.Read(); j++)

{

string gameId = Reader2[0].ToString();

ShopAssortiment shop = new ShopAssortiment(Reader2[0].ToString(), Reader2[1].ToString() + " руб.", Convert.ToInt32(Reader2[2]) == 1, Convert.ToInt32(Reader2[3]) == 1);

shop.Location = new Point(50, 30 + j \* 330);

CenterPanel.Controls.Add(shop);

}

}

};

label.Text = Name;

Genres.Controls.Add(label);

if(i == 0)

{

label.Location = new Point(50, 5);

continue;

}

label.Location = new Point(Genres.Controls[i-1].Location.X + Genres.Controls[i-1].Width + 10, 5);

}

}

}

private void SetLibrary()

{

Clear();

if (!GlobalFields.isConnected)

{

IssiuesTextBox.Visible = true;

IssiuesTextBox.Text = "Для просмотра вашей библиотеки зайдите в аккаунт";

return;

}

using (SqlConnection connection = new SqlConnection(GlobalFields.CONNECTION))

{

connection.Open();

string login;

ReadLogin(out login);

SqlCommand com2 = new SqlCommand($"Select [Магазин/Игры].Название, Библиотека.[Наиграно часов] from [Магазин/Игры] Inner Join Библиотека ON [Магазин/Игры].ID = Библиотека.ID\_Игры where ID\_Пользователя = (select ID from Пользователи where Email = '{login}')", connection);

SqlDataReader Reader1;

Reader1 = com2.ExecuteReader();

int j = -1;

for (int i = 0; Reader1.Read(); i++)

{

if(i % 3 == 0) { j++; }

LibraryComponent component = new LibraryComponent(Reader1[0].ToString(), Reader1[1].ToString() + " часов в игре");

component.Location = new Point(20+230\*(i%3), 30+330\*j);

CenterPanel.Controls.Add(component);

}

}

}

private void SetFavourite()

{

Clear();

if (!GlobalFields.isConnected)

{

IssiuesTextBox.Visible = true;

IssiuesTextBox.Text = "Для просмотра ваших избранных игр зайдите в аккаунт";

return;

}

using (SqlConnection connection = new SqlConnection(GlobalFields.CONNECTION))

{

connection.Open();

string login;

ReadLogin(out login);

SqlCommand com2 = new SqlCommand($"Select \* from dbo.[GetGames]('{login}')", connection);

SqlDataReader Reader1;

Reader1 = com2.ExecuteReader();

int j = -1;

for (int i = 0; Reader1.Read(); i++)

{

if (Convert.ToInt32(Reader1[3]) == 1) {

if (i % 2 == 0) { j++; }

string gameId = Reader1[0].ToString();

FavouriteGames shop = new FavouriteGames(Reader1[0].ToString(), Reader1[1].ToString() + " руб.", Convert.ToInt32(Reader1[2]) == 1, true);

shop.Location = new Point(20 + 375 \* (i % 2), 30 + j \* 280);

CenterPanel.Controls.Add(shop);

continue;

}

i--;

}

}

}

private void Clear()

{

CenterPanel.Controls.Clear();

}

static void SetRoundedShape(Control control, int radius)

{

System.Drawing.Drawing2D.GraphicsPath path = new System.Drawing.Drawing2D.GraphicsPath();

path.AddLine(radius, 0, control.Width - radius, 0);

path.AddArc(control.Width - radius, 0, radius, radius, 270, 90);

path.AddLine(control.Width, radius, control.Width, control.Height - radius);

path.AddArc(control.Width - radius, control.Height - radius, radius, radius, 0, 90);

path.AddLine(control.Width - radius, control.Height, radius, control.Height);

path.AddArc(0, control.Height - radius, radius, radius, 90, 90);

path.AddLine(0, control.Height - radius, 0, radius);

path.AddArc(0, 0, radius, radius, 180, 90);

control.Region = new Region(path);

}

#region TRIGGERS

private void upsidePanel\_MouseMove(object sender, MouseEventArgs e)

{

if(e.Button == MouseButtons.Left)

{

this.Left += e.X - lastPoint.X;

this.Top += e.Y - lastPoint.Y;

}

}

private void upsidePanel\_MouseDown(object sender, MouseEventArgs e)

{

lastPoint = new Point(e.X, e.Y);

}

private void button1\_Click(object sender, EventArgs e)

{

Application.Exit();

}

private void Hide\_Click(object sender, EventArgs e)

{

this.WindowState = FormWindowState.Minimized;

}

private void FullScreen\_Click(object sender, EventArgs e)

{

this.WindowState = FormWindowState.Maximized;

}

private void Panel7\_Click(object sender, EventArgs e)

{

Settings settings = new Settings();

settings.ShowDialog();

}

private void ShopPanel\_MouseEnter(object sender, EventArgs e)

{

ShopTextBox.ForeColor = Color.FromArgb(235, 235, 235);

panel4.BackgroundImage = Image.FromFile(@"..\..\Resources\PriceBright.png");

}

private void ShopPanel\_MouseLeave(object sender, EventArgs e)

{

ShopTextBox.ForeColor = Color.FromArgb(195, 195, 195);

panel4.BackgroundImage = Image.FromFile(@"..\..\Resources\Price.png");

}

private void LibraryPanel\_MouseEnter(object sender, EventArgs e)

{

LibraryTextBox.ForeColor = Color.FromArgb(235, 235, 235);

panel5.BackgroundImage = Image.FromFile(@"..\..\Resources\BookBright.png");

}

private void LibraryPanel\_MouseLeave(object sender, EventArgs e)

{

LibraryTextBox.ForeColor = Color.FromArgb(195, 195, 195);

panel5.BackgroundImage = Image.FromFile(@"..\..\Resources\Book.png");

}

private void FavouritePanel\_MouseEnter(object sender, EventArgs e)

{

FavouriteTextBox.ForeColor = Color.FromArgb(235, 235, 235);

panel6.BackgroundImage = Image.FromFile(@"..\..\Resources\FavouritBright.png");

}

private void FavouritePanel\_MouseLeave(object sender, EventArgs e)

{

FavouriteTextBox.ForeColor = Color.FromArgb(195, 195, 195);

panel6.BackgroundImage = Image.FromFile(@"..\..\Resources\Favourit.png");

}

private void LibraryPanel\_Click(object sender, EventArgs e)

{

SetLibrary();

}

private void ShopPanel\_Click(object sender, EventArgs e)

{

SetShopAssortiment();

}

private void ShopTextBox\_Click(object sender, EventArgs e)

{

SetShopAssortiment();

}

private void Panel4\_Click(object sender, EventArgs e)

{

SetShopAssortiment();

}

private void FavouritePanel\_Click(object sender, EventArgs e)

{

SetFavourite();

}

private void Panel7\_MouseEnter(object sender, EventArgs e)

{

EnterTextBox.ForeColor = Color.FromArgb(235,235,235);

panel8.BackgroundImage = Image.FromFile(@"..\..\Resources\LoginBright.png");

}

private void Panel7\_MouseLeave(object sender, EventArgs e)

{

EnterTextBox.ForeColor = Color.FromArgb(195, 195, 195);

panel8.BackgroundImage = Image.FromFile(@"..\..\Resources\Login.png");

}

#endregion

private void Panel2\_MouseEnter(object sender, EventArgs e)

{

panel2.BackgroundImage = Image.FromFile(@"..\..\Resources\AddGamesBright.png");

}

private void Panel2\_MouseLeave(object sender, EventArgs e)

{

panel2.BackgroundImage = Image.FromFile(@"..\..\Resources\AddGames.png");

}

private void Panel2\_Click(object sender, EventArgs e)

{

AddGame addGame = new AddGame();

addGame.ShowDialog();

}

}

}

1. FavouriteGames.cs

using System;

using System.Drawing;

using System.Data.SqlClient;

using System.IO;

using System.Windows.Forms;

namespace Kursach

{

public partial class FavouriteGames : UserControl

{

private bool isBuied;

private bool \_isBuied

{

get

{

return isBuied;

}

set

{

if (value)

{

Price.Text = "Приобретено";

}

isBuied = value;

}

}

private bool isFavourite;

private bool \_isFavourite

{

get

{

return isFavourite;

}

set

{

if (value)

{

FavouritePanel.BackgroundImage = Image.FromFile(@"..\..\Resources\FavouriteMiniChecked.png");

}

isFavourite = value;

}

}

private string \_gameName

{

get

{

return GameName.Text;

}

set

{

GameName.Text = value;

}

}

private string price;

private string \_price

{

get

{

return price;

}

set

{

if (!isBuied)

{

Price.Text = value;

}

}

}

private string backgroundPicture

{

set

{

try

{

MainPanel.BackgroundImage = Image.FromFile(@"..\..\Resources\" + value + ".jpg");

}

catch

{

MainPanel.BackColor = Color.FromArgb(195, 195, 195);

}

}

}

public FavouriteGames(string gn, string pr, bool isbyed, bool isfavourite)

{

InitializeComponent();

FavouritePanel.BackColor = Color.Transparent;

SetRoundedShape(MainPanel, 10, true);

SetRoundedShape(DownSidePanel, 10, false);

\_isBuied = isbyed;

\_isFavourite = isfavourite;

\_gameName = gn;

\_price = pr;

price = pr;

backgroundPicture = gn;

}

private void ReadLogin(out string login)

{

using (StreamReader sr = new StreamReader(@"..\userData.txt"))

{

login = sr.ReadLine();

}

}

static void SetRoundedShape(Control control, int radius, bool isMainForm)

{

System.Drawing.Drawing2D.GraphicsPath path = new System.Drawing.Drawing2D.GraphicsPath();

if (isMainForm)

{

path.AddLine(radius, 0, control.Width - radius, 0);

path.AddArc(control.Width - radius, 0, radius, radius, 270, 90);

}

path.AddLine(control.Width, radius, control.Width, control.Height - radius);

path.AddArc(control.Width - radius, control.Height - radius, radius, radius, 0, 90);

path.AddLine(control.Width - radius, control.Height, radius, control.Height);

path.AddArc(0, control.Height - radius, radius, radius, 90, 90);

path.AddLine(0, control.Height - radius, 0, radius);

if (isMainForm)

{

path.AddArc(0, 0, radius, radius, 180, 90); ;

}

control.Region = new Region(path);

}

private void Price\_Click(object sender, EventArgs e)

{

if (!GlobalFields.isConnected)

{

return;

}

using (SqlConnection connection = new SqlConnection(GlobalFields.CONNECTION))

{

connection.Open();

string login;

ReadLogin(out login);

SqlCommand com1 = new SqlCommand($"Insert into Библиотека values ((select ID from [Магазин/Игры] where Название = '{\_gameName}'), (select ID from Пользователи where Email = '{login}'), 0)", connection);

com1.ExecuteNonQuery();

}

\_isBuied = true;

}

private void Price\_MouseEnter(object sender, EventArgs e)

{

\_price = "Приобрести";

}

private void Price\_MouseLeave(object sender, EventArgs e)

{

\_price = price;

}

private void Panel2\_MouseEnter(object sender, EventArgs e)

{

if (isFavourite) { return; }

FavouritePanel.BackgroundImage = Image.FromFile(@"..\..\Resources\FavouriteMiniHov.png");

}

private void Panel2\_MouseLeave(object sender, EventArgs e)

{

if (isFavourite) { return; }

FavouritePanel.BackgroundImage = Image.FromFile(@"..\..\Resources\FavouriteMini.png");

}

private void Panel2\_Click(object sender, EventArgs e)

{

if (!GlobalFields.isConnected)

{

return;

}

using (SqlConnection connection = new SqlConnection(GlobalFields.CONNECTION))

{

connection.Open();

string login;

ReadLogin(out login);

string sql = isFavourite ? $"Delete from Избранное where ID\_Игры = (select ID from [Магазин/Игры] where Название = '{\_gameName}') AND ID\_Пользователя = (select ID from Пользователи where Email = '{login}')" : $"Insert into Избранное values ((select ID from [Магазин/Игры] where Название = '{\_gameName}'), (select ID from Пользователи where Email = '{login}'))";

SqlCommand com1 = new SqlCommand(sql, connection);

com1.ExecuteNonQuery();

}

\_isFavourite = !\_isFavourite;

}

}

}

1. AddGame.cs

using System;

using System.Data.SqlClient;

using System.Windows.Forms;

namespace Kursach

{

public partial class AddGame : Form

{

public AddGame()

{

InitializeComponent();

}

private void CloseButton\_Click(object sender, EventArgs e)

{

Close();

}

private void Button1\_Click(object sender, EventArgs e)

{

if (string.IsNullOrWhiteSpace(GameNameTextBox.Text) || string.IsNullOrWhiteSpace(PriceTextBox.Text))

{

label1.Text = "Необходимо ввести данные об игре";

return;

}

using (SqlConnection connection = new SqlConnection(GlobalFields.CONNECTION))

{

connection.Open();

SqlCommand com1 = new SqlCommand($"Select Count(\*) from Genres where Название = '{GenreTextBox.Text}'", connection);

if (Convert.ToInt32(com1.ExecuteScalar()) == 0)

{

SqlCommand com2 = new SqlCommand($"Insert into Genres values ('{GenreTextBox.Text}', 0)", connection);

com2.ExecuteNonQuery();

}

SqlCommand com3 = new SqlCommand($"Insert into [Магазин/Игры] values ('{GameNameTextBox.Text}', {PriceTextBox.Text}, (Select Id from Genres where Название = '{GenreTextBox.Text}')) ", connection);

com3.ExecuteNonQuery();

}

MainForm.Instantiate.SetShopAssortiment();

Close();

}

}

}