Министерство образования и науки Республики Башкортостан

Государственное автономное профессиональное образовательное учреждение

Уфимский колледж статистики, информатики и вычислительной техники

|  |  |
| --- | --- |
|  | УТВЕРЖДАЮ  Заместитель директора  по учебной работе  З.З. Курмашева  « » 2024 г. |

ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ И РАЗРАБОТКА ПРИЛОЖЕНИЯ ДЛЯ ВЕДЕНИЯ СПРАВОЧНИКА ИГР

Пояснительная записка к курсовому проекту

МДК 11.01 Технология разработки и защиты баз данных

|  |  |
| --- | --- |
|  | Руководитель проекта  Р.Ф. Каримова  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 г. |
|  | Студент гр. 21П-1  Н.С. Петухов  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 г. |

2024

Министерство образования и науки Республики Башкортостан

Государственное автономное профессиональное образовательное учреждение

Уфимский колледж статистики, информатики и вычислительной техники

|  |  |
| --- | --- |
|  | УТВЕРЖДАЮ  Заместитель директора  по учебной работе  З.З. Курмашева  «\_\_\_» 2024 г. |

ЗАДАНИЕ

на курсовой проект студенту дневного отделения, группы 21П-1, специальности 09.02.07 Информационные системы и программирование

Фамилия, имя, отчество: Петухов Никита Сергеевич

Тема курсового проекта: «Проектирование базы данных и разработка приложения для ведения справочника игр».

Текст задания:

при выполнении курсового проекта должны быть решены следующие задачи:

1. спроектирована структура базы данных;
2. разработана структура программы;
3. реализованы функции поиска и просмотра игр, просмотра основной информации игр и отзывов, добавление, изменения и удаления отзывов.

В результате выполнения курсового проекта должны быть представлены:

1. пояснительная записка, состоящая из следующих разделов:

Введение

1 Постановка задачи

2 Экспериментальный раздел

Заключение

Приложения

Список использованных источников

1. электронный носитель, содержащий разработанный программный продукт;
2. презентация курсового проекта в электронном виде.

Список рекомендуемых источников:

1. Фуфаев, Э.В. Базы данных [Текст]: учеб. пособ. для студ. учрежд. сред. проф. образования / Э.В. Фуфаев, Д.Э. Фуфаев. - 6-е изд., стер. - М.: Издательский центр «Академия», 2012.- 320 с.- (Среднее профессиональное образование).
2. Википедия [Электронный ресурс] // Свободная энциклопедия. – Режим доступа: <http://ru.wikipedia.org/wiki/>, свободный.

Задание к выполнению получил «31» января 2024 г.

Студент Петухов Никита Сергевич

Срок окончания «31» мая 2024 г.

Руководитель курсового проекта Р.Ф. Каримова

Задание рассмотрено на заседании цикловой комиссии информатики

«11» января 2024 г.

Председатель цикловой комиссии информатики О.В. Фатхулова

Министерство образования и науки Республики Башкортостан

Государственное автономное профессиональное образовательное учреждение

Уфимский колледж статистики, информатики и вычислительной техники

ЗАКЛЮЧЕНИЕ

на курсовой проект

Студент Петухов Никита Сергеевич

Группа 21П-1

Специальность 09.02.07 Информационные системы и программирование

Тема Проектирование базы данных и разработка приложения для ведения справочника игр

Объем курсового проекта:

количество листов пояснительной записки

количество листов графической части

Заключение о степени соответствия заданию на курсовое проектирование

Характеристика качеств, проявленных студентом при работе над проектом: самостоятельность, дисциплинированность, умение планировать работу и пользоваться литературным материалом и т.д.

Положительные стороны курсового проекта

Недостатки курсового проекта

Характеристика общетехнической и специальной подготовки студента

Заключение и предлагаемая оценка за курсовой проект

Руководитель курсового проекта Каримова Резида Флюновна

«\_\_\_\_» 2024 г.

Подпись

АННОТАЦИЯ

Пояснительная записка к курсовому проекту содержит постановку и программу решения задачи «Проектирование базы данных и разработка приложения для ведения справочника игр».

Программа Reviews.exe написана на языке C# в среде программирования Visual Studio 2022 с использованием системы управления базой данных MySQL Workbench предназначена для работы в операционной системе MS Windows 10 и выше, отлажена на данных контрольного примера.

СОДЕРЖАНИЕ

|  |  |
| --- | --- |
|  | лист |
| Введение | 4 |
| 1 Постановка задачи | 5 |
| 1.1 Описание предметной области | 5 |
| 1.2 Описание входной информации | 7 |
| 1.3 Описание выходной информации | 7 |
| 1.4 Концептуальное моделирование | 7 |
| 1.5 Логическое моделирование | 8 |
| 1.6 Описание структуры базы данных | 9 |
| 1.7 Контрольный пример | 11 |
| 1.8 Общие требования к программному продукту | 11 |
| 2 Экспериментальный раздел | 12 |
| 2.1 Описание программы | 12 |
| 2.2 Протокол тестирования программного продукта | 18 |
| 2.3 Руководство пользователя | 33 |
| Заключение | 49 |
| Приложения | 50 |
| Список источников | 92 |

ВВЕДЕНИЕ

В современном мире компьютерные игры являются одним из самых популярных видов развлечений. Ежегодно выпускаются сотни новых игр, и рынок игровой индустрии продолжает расти. Однако, несмотря на разнообразие предложений, выбор подходящей игры может быть сложным процессом, особенно для начинающих игроков.

Актуальность данной работы обусловлена тем, что у людей могут возникать трудности в поиске нужной информации, следовательно у игроков могут возникать трудности с поиском подходящей игры для развлечений.

Цель курсового проекта – разработка приложения для упрощения поиска основной информации об играх, в котором пользователи смогут узнать подробные данные об играх и прочитать мнения других игроков, что поможет им сделать выбор игры.

Задачами курсового проекта являются:

* описать предметную область;
* разработать структуру базы данных;
* разработать приложение;
* провести тестирование приложения.

1. Постановка задачи

* 1. Описание предметной области

Требуется разработать информационную систему для автоматизации поиска основной информации об играх, добавления новых отзывов и вычисления рейтинга игр по отзывам.

Данная информационная система предполагает наличие двух групп пользователей: авторизованный пользователь и неавторизованный пользователь.

Авторизованный пользователь имеет возможность:

* просматривать список игр;
* выполнять поиск игр по названию, жанру и фильтру;
* просматривать основную информацию и отзывы об игре;
* редактировать данные своего аккаунта;
* удалить свой аккаунт;
* просматривать свои отзывы;
* удалять свои отзывы;
* выполнить выход на окно авторизации;
* добавлять по одному отзыву для каждой игры;
* изменять свои отзывы.

Неавторизованный пользователь имеет следующие возможности:

* просматривать список игр;
* выполнять поиск игр по названию, жанру и фильтру;
* просматривать основную информацию и отзывы об игре;
* выполнить выход на окно авторизации.

В базе данных должны храниться следующие справочники: разработчики, игры-жанры, игры, жанры, издатели, отзывы, пользователи.

В таблице «Разработчики» содержится следующая информация:

* идентификатор разработчика;
* название.

У каждой игры может быть несколько жанров. В таблице «Игры-Жанры»

содержится следующая информация:

* идентификатор игры;
* идентификатор жанра.

В таблице «Игры» содержится следующая информация:

* идентификатор игры;
* название;
* описание;
* идентификатор разработчика;
* идентификатор издателя;
* дата релиза;
* картинка.

В таблице «Жанры» содержится следующая информация:

* идентификатор жанра;
* название.

В таблице «Издатели» содержится следующая информация:

* идентификатор издателя;
* название.

Для того, чтобы добавлять, изменять и удалять свои отзывы, пользователь должен сначала пройти авторизацию. Каждый авторизованный пользователь может добавить только один отзыв для каждой игры. В таблице «Отзывы» содержится следующая информация:

* идентификатор отзыва;
* текст;
* оценка;
* идентификатор игры;
* идентификатор пользователя;
* дата.

В таблице «Пользователи» содержится следующая информация:

* идентификатор пользователя;
* логин;
* пароль;
* имя.

1.2 Описание входной информации

Входной информацией являются:

* справочники: разработчики, игры-жанры, игры, жанры, издатели, отзывы, пользователи;
* данные пользователя и отзыва.

1.3 Описание выходной информации

Выходной информацией являются:

* списки игр и отзывов;
* информация об игре, отзыве и пользователе.

1.4. Концептуальное моделирование

Концептуальная модель базы данных — это абстрактное представление данных организации на высоком уровне. Она фокусируется на сборе сущностей, их атрибутов и отношений без указания деталей реализации. Основная цель концептуального моделирования данных — чётко понять требования и облегчить общение между заинтересованными сторонами.

Концептуальная модель базы данных представлена в рисунке 1.4.1.

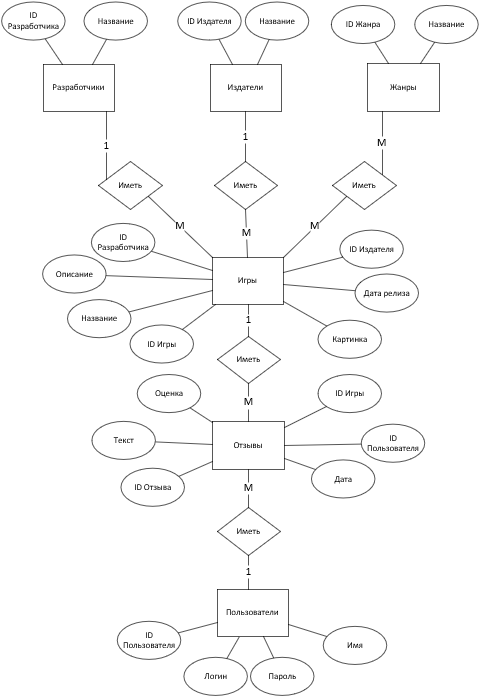


Рисунок 1.4.1 – Концептуальная модель БД

1.5. Логическое моделирование

Логическое моделирование базы данных — это процесс создания детального представления структуры и взаимосвязей между сущностями в базе данных. Оно основано на концептуальной модели данных и добавляет дополнительную информацию об атрибутах, типах данных и ограничениях. Цель логического моделирования — определить точную структуру и отношения между сущностями, сохраняя определённую степень абстракции от фактической реализации.

Схема данных — это структура, описывающая содержание, структуру и ограничения целостности базы данных. Она включает в себя таблицы, поля и ограничения целостности, которые определяют отношения между данными.

Схема данных представлена на рисунке 1.5.1.



Рисунок 1.5.1 – Схема данных БД

1.6. Описание структуры базы данных

Описание структуры базы данных представлено в таблице 1.6.1.

Таблица 1.6.1 - Описание структуры базы данных

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поля | Описание поля | Тип данных | Размер поля | Тип ключа[[1]](#footnote-1) |
| 1 | 2 | 3 | 4 | 5 |
| developers (Разработчики) | | | | |
| DeveloperId | ID разработчика | INT |  | PK |
| DeveloperName | Название | VARCHAR | 50 |  |
| gamegenres (Игры-Жанры) | | | | |
| GameId | ID игры | INT |  | PK |
| GenreId | ID жанра | INT |  | PK |
| games (Игры) | | | | |
| GameId | ID игры | INT |  | PK |
| GameName | Название | VARCHAR | 50 |  |
| GameDescription | Описание | TEXT |  |  |
| GameDeveloperId | ID разработчика | INT |  | FK |
| GamePublisherId | ID издателя | INT |  | FK |
| GameDateRelease | Дата релиза | DATE |  |  |
| GamePicture | Картинка | VARCHAR | 50 |  |
| GameRating | Рейтинг | DOUBLE |  |  |
| GameCommentsCount | Кол-во отзывов | INT |  |  |
| genres (Жанры) | | | | |
| GenreId | ID жанра | INT |  | PK |
| GenreName | Название | VARCHAR | 50 |  |
| publishers (Издатели) | | | | |
| PublisherId | ID издателя | INT |  | PK |
| PublisherName | Название | VARCHAR | 50 |  |

Продолжение таблицы 1.6.1

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | | 3 | | 4 | | 5 | |
| reviews (Отзывы) | | | | | | | | |
| ReviewId | | ID отзыва | | INT | |  | | PK |
| ReviewText | | Текст | | TEXT | |  | |  |
| ReviewEstimate | | Оценка | | INT | |  | |  |
| ReviewGameId | | ID игры | | INT | |  | | FK |
| ReviewUserId | | ID пользователя | | INT | |  | | FK |
| ReviewDate | | Дата | | DATETIME | |  | |  |
| users (Пользователи) | | | | | | | | |
| UserId | | ID пользователя | | INT | |  | | PK |
| UserLogin | | Логин | | VARCHAR | | 50 | |  |
| UserPassword | | Пароль | | VARCHAR | | 50 | |  |
| UserNickname | | Имя | | VARCHAR | | 50 | |  |

1.7. Контрольный пример

Контрольный пример программы — это набор входных данных и ожидаемых результатов, который используется для проверки правильности работы программы. Он помогает выявить ошибки и убедиться, что программа соответствует требованиям заказчика.

Входная информация контрольных примеров показаны в приложении А.

Выходная информация контрольных примеров показаны в приложении Б.

1.8 Общие требования к программному продукту

Пользователи должны иметь базовые навыки пользования персональным компьютером.

Минимальные требования к техническому обеспечению программного продукта следующие:

* ОС: Windows 7 Service Pack1/ Windows 7 64Bit Service Pack1/ Windows 8.1 64Bit / Windows 10 64Bit / Mac OS;
* процессор: 2.40 Ггц (четырехъядерный) / AMD Phenom 9850 (четырехъядерный) @ 2.5 Ггц;
* оперативная память: 512МБ (Win 7/Win8 и выше);
* видеокарта: NVIDIA 9800 GT c 512 МБ видеопамяти/ AMD HD 4870 с 1 Гб видеопамяти (DX 9, 10, 10.1);
* жесткий диск: 10 гигабайт свободного места;
* Microsoft DirectX версия 9.0c.

Функциональные возможности программного продукта:

* приложение должно формировать и отображать выходные данные пользователю;
* в приложении должен быть обеспечен просмотр таблиц (справочников) базы данных с возможность добавления, редактирования, удаления данных.

Требования к надежности:

* приложение должно обрабатывать ошибочные действия пользователя и сообщать ему об этом;
* приложение должно обеспечивать контроль входной и выходной информации.

Требования к информационной и программной совместимости: обеспечить работу приложения с таблицами СУБД MySQL Workbench.

Экспериментальный раздел

2. Экспериментальный раздел

2.1 Описание программы

Программа имеет модульную структуру. При ее запуске выполняется проект на WPF Reviews.exe. Программа Reviews.exe написана на языке C# в среде программирования Visual Studio 2022 с использованием системы управления базой данных MySQL Workbench.

Описание модулей и методов представлено в таблице 2.1.1.

Таблица 2.1.1 - Описание модулей

|  |  |
| --- | --- |
| Методы | Назначение |
| 1 | 2 |
| App.xaml.cs | |
| public static void UpdateRating() | Обновление рейтинга и кол-ва отзывов игр |
| MainWindow.xaml.cs | |
| public MainWindow() | Переход на окно авторизации |
| AddReview.xaml.cs | |
| public AddReview() | Закрытие предыдущего окна и сохранение текущего окна, заполнение данных комментария, если пользователь выбрал изменение отзыва и скрытие кнопки удаления, если пользователь выбрал добавление отзыва. |
| private void bBack\_Click(object sender, RoutedEventArgs e) | Переход на окно с основной информацией игры |
| private void bSave\_Click(object sender, RoutedEventArgs e) | Изменение или добавление отзыва |

Продолжение таблицы 2.1.1

|  |  |
| --- | --- |
| 1 | 2 |
| private string  StringFromRichTextBox(RichTextBox rtb) | Получение строки из RichTextBox |
| private void bDelete\_Click(object sender, RoutedEventArgs e) | Удаление отзыва |
| Auth.xaml.cs | |
| public Auth() | Закрытие предыдущего окна и сохранение текущего окна |
| private void bVisitor\_Click(object sender, RoutedEventArgs e) | Переход на окно просмотра игр без авторизации |
| private void bReg\_Click(object sender, RoutedEventArgs e) | Переход на окно регистрации |
| private void bAuth\_Click(object sender, RoutedEventArgs e) | Авторизация и переход на окно просмотра игр |
| GameInfo.xaml.cs | |
| public GameInfo() | Закрытие предыдущего окна и сохранение текущего окна, заполнение данных игры, скрытие кнопки добавления отзыва, если пользователь не авторизовался |

Продолжение таблицы 2.1.1

|  |  |
| --- | --- |
| 1 | 2 |
| private void bBack\_Click(object sender, RoutedEventArgs e) | Переход на окно просмотра игр или переход на окно просмотра отзывов пользователя, если пользователь выполнил переход из окна просмотра своих отзывов |
| private void bAddReview\_Click(object sender, RoutedEventArgs e) | Переход на окно добавления, изменения или удаления отзыва |
| GamesList.xaml.cs | |
| public GamesList() | Закрытие предыдущего окна и сохранение текущего окна, заполнение ListView играми, скрытие кнопки профиля, если пользователь не авторизовался |
| private void bExit\_Click(object sender, RoutedEventArgs e) | Переход на окно авторизации |
| private void bProfile\_Click(object sender, RoutedEventArgs e) | Переход на окно профиля |
| private void tbSearch\_TextChanged(object sender, TextChangedEventArgs e) | Вызов метода private void Refresh(string? search, int? idGenre, int? idFilter) |

Продолжение таблицы 2.1.1

|  |  |
| --- | --- |
| 1 | 2 |
| private void cbGenre\_DropDownClosed(object sender, EventArgs e) | Вызов метода private void Refresh(string? search, int? idGenre, int? idFilter) |
| private void cbFilter\_DropDownClosed(object sender, EventArgs e) | Вызов метода private void Refresh(string? search, int? idGenre, int? idFilter) |
| private void Refresh(string? search, int? idGenre, int? idFilter) | Заполнение ListView играми, которые соответствуют названию, жанру и фильтру, выбранными пользователем |
| Profile.xaml.cs | |
| public Profile() | Закрытие предыдущего окна и сохранение текущего окна, заполнение данных профиля |
| private void bBack\_Click(object sender, RoutedEventArgs e) | Переход на окно просмотра игр |
| private void bReviews\_Click(object sender, RoutedEventArgs e) | Переход на окно просмотра отзывов пользователя |
| private void bSave\_Click(object sender, RoutedEventArgs e) | Изменение данных аккаунта |
| private void bDelete\_Click(object sender, RoutedEventArgs e) | Удаление аккаунта |

Продолжение таблицы 2.1.1

|  |  |
| --- | --- |
| 1 | 2 |
| Reg.xaml.cs | |
| public Reg() | Закрытие предыдущего окна и сохранение текущего окна |
| private void bVisitor\_Click(object sender, RoutedEventArgs e) | Переход на окно просмотра игр без регистрации |
| private void bAuth\_Click(object sender, RoutedEventArgs e) | Переход на окно авторизации |
| private void bReg\_Click(object sender, RoutedEventArgs e) | Регистрация и переход на окно просмотра игр |
| UserReviews.xaml.cs | |
| public UserReviews() | Закрытие предыдущего окна и сохранение текущего окна, заполнение ListView отзывами пользователя |
| private void bBack\_Click(object sender, RoutedEventArgs e) | Переход на окно профиля |
| GameCard.xaml.cs | |
| public GameCard(Game game) | Заполнение карточки игры данными игры |

Продолжение таблицы 2.1.1

|  |  |
| --- | --- |
| 1 | 2 |
| private void UserControl\_MouseDoubleClick(object sender, MouseButtonEventArgs e) | Переход на окно основной информации игры |
| ReviewCard.xaml.cs | |
| public ReviewCard(Review review) | Заполнение карточки отзыва данными отзыва |
| private void UserControl\_MouseDoubleClick(object sender, MouseButtonEventArgs e) | Переход на окно основной информации игры, если текущее окно – окно просмотра отзывов пользователя |

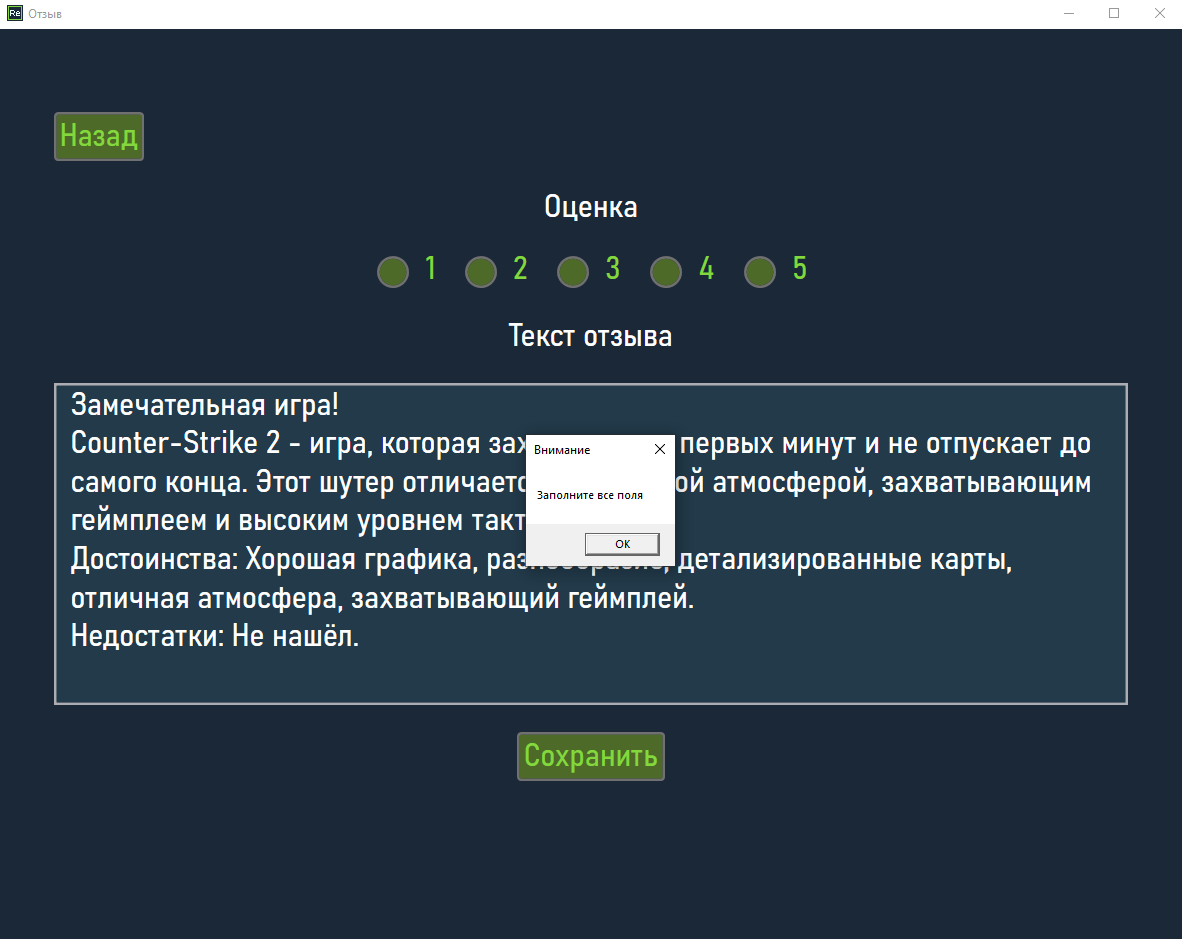
Код программы представлен в приложении В.

2.2 Протокол тестирования программного продукта

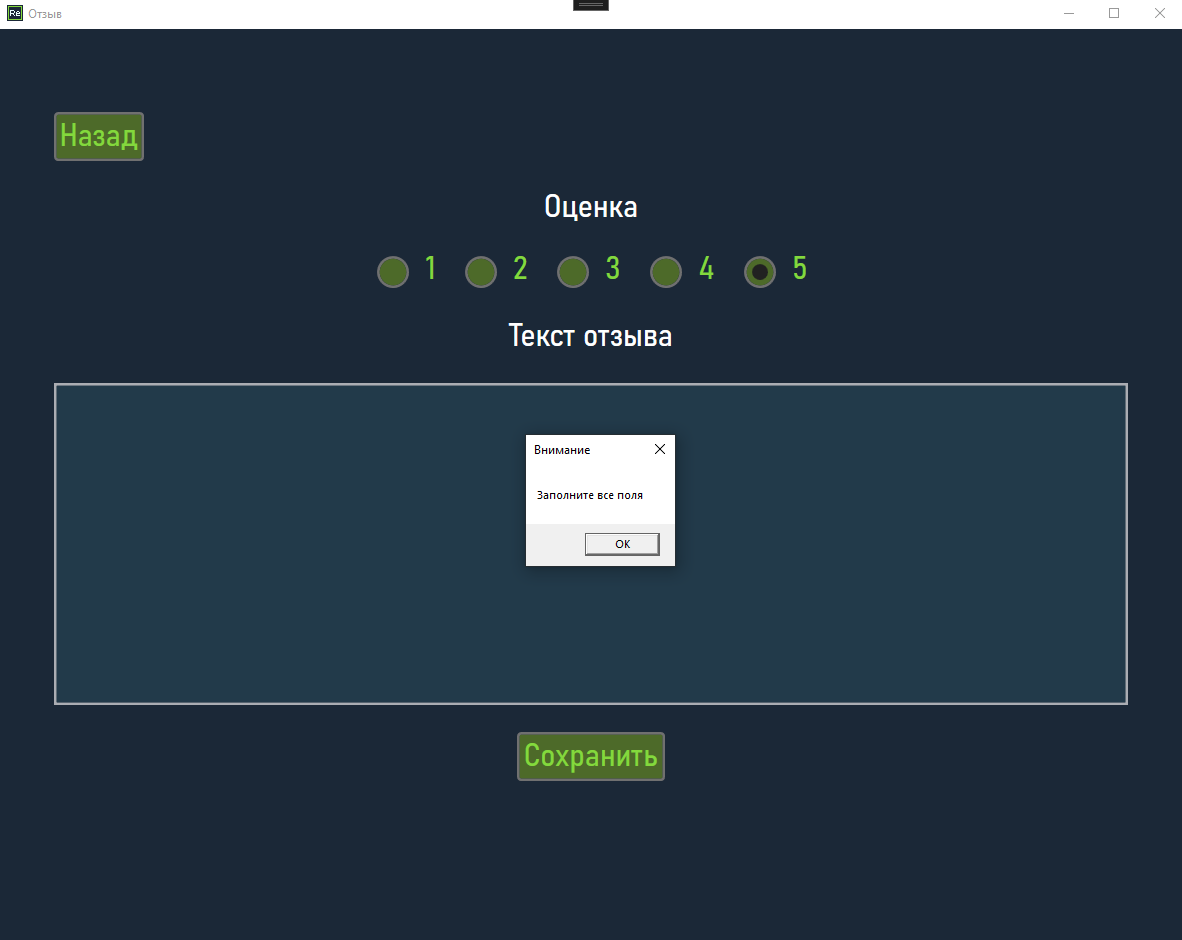
В протоколе тестирования отражаются:

* тестирование на корректных данных;
* тестирование на некорректных данных;
* тестировании продукта на данных контрольного примера.

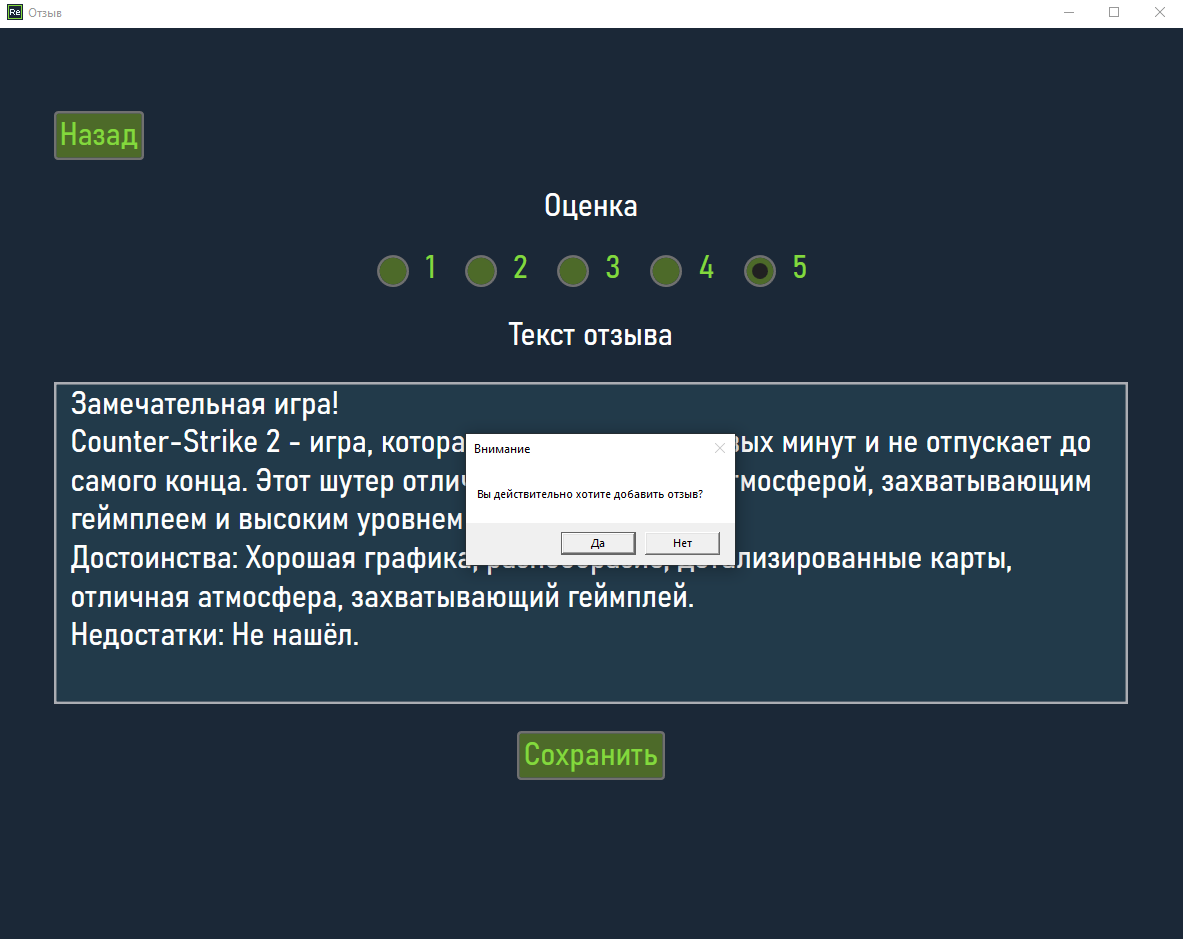
Тестирование на добавление нового отзыва об игре без указания оценки, ожидаемое сообщение «Заполните все поля» (рисунок 2.2.1).

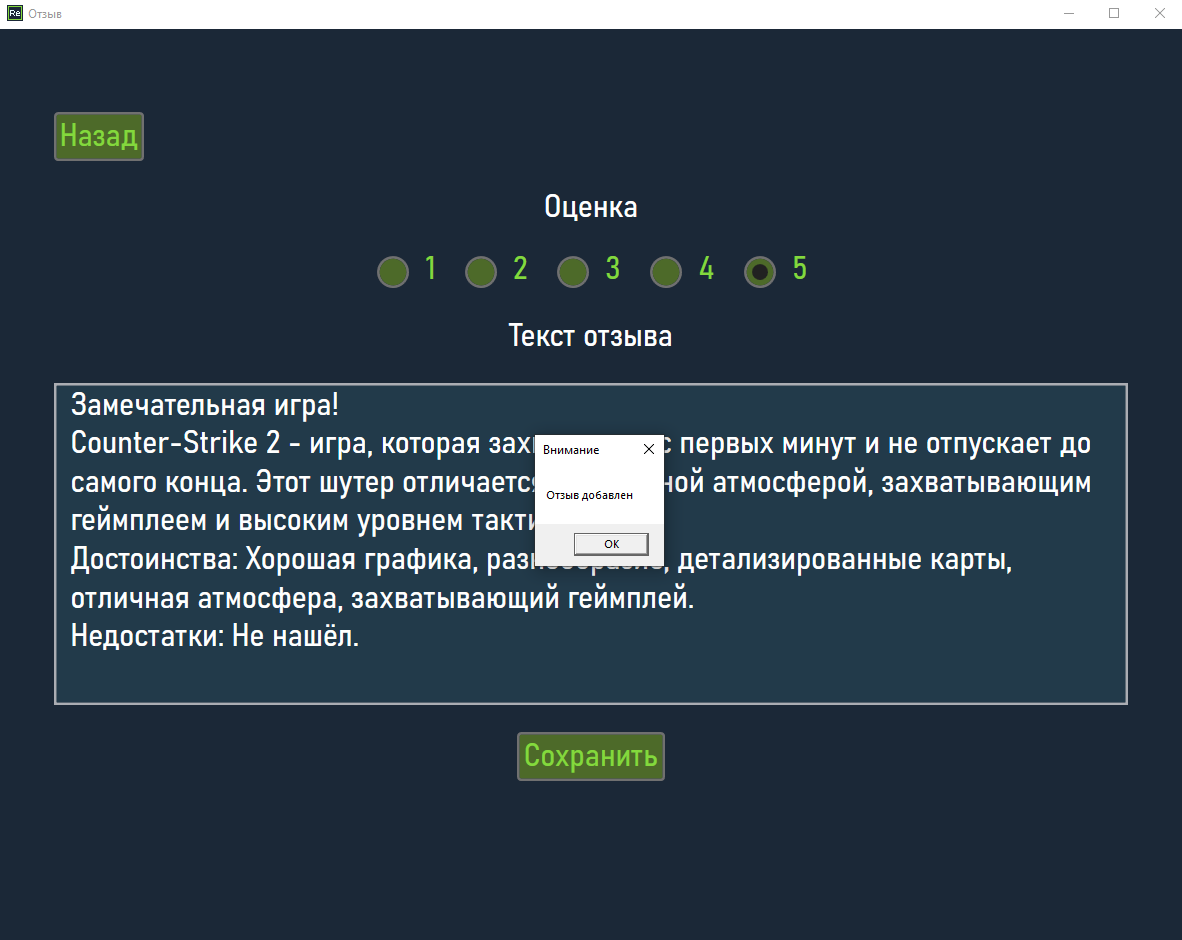
Рисунок 2.2.1 – Сообщение с просьбой заполнить все поля

Тестирование на добавление нового отзыва об игре без указания текста отзыва, ожидаемое сообщение «Заполните все поля» (рисунок 2.2.2).

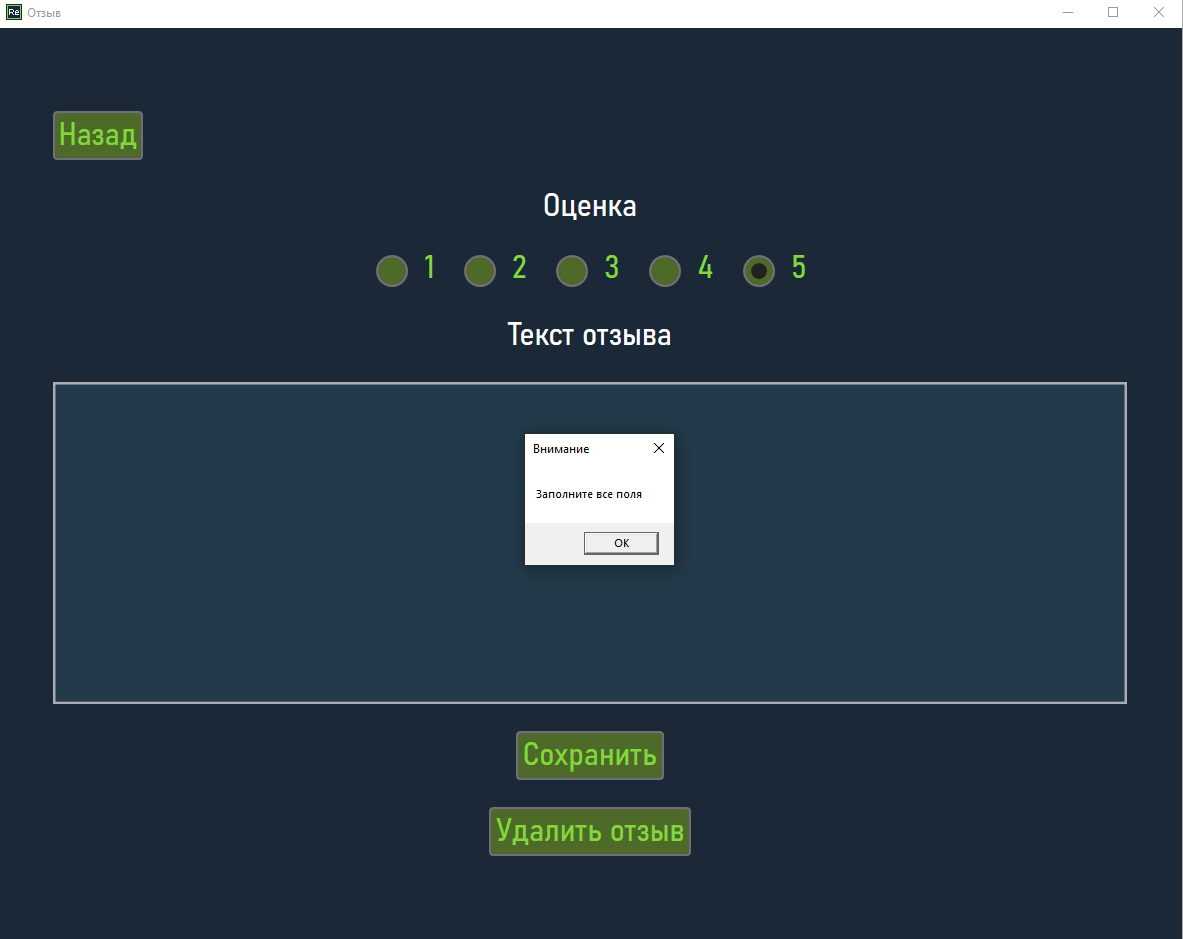
Рисунок 2.2.2 – Сообщение с просьбой заполнить все поля

Тестирование на добавление нового отзыва об игре, ожидаемое сообщение «Отзыв добвлен» (рисунки 2.2.3 – 2.2.4).

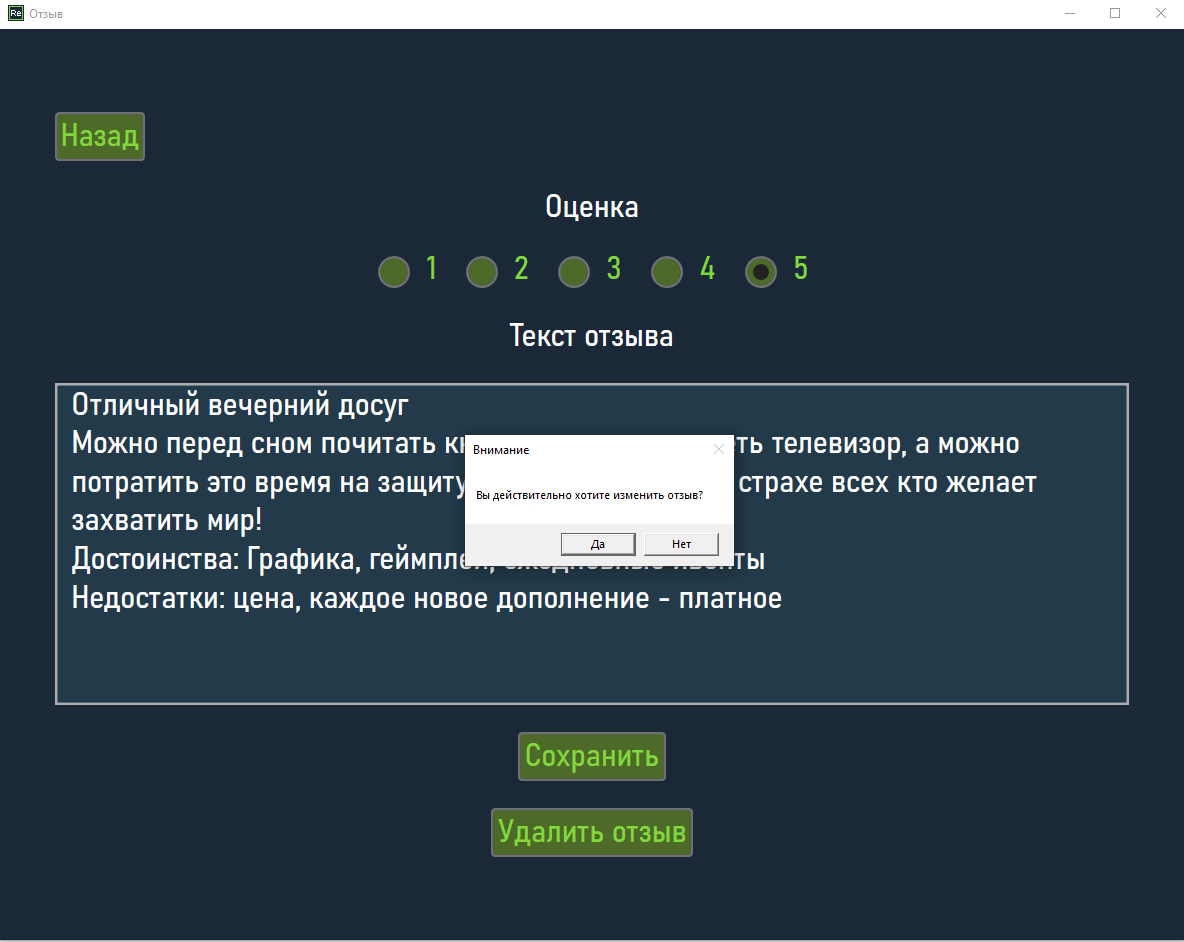
Рисунок 2.2.3 – Сообщение с подтверждением добавления отзыва

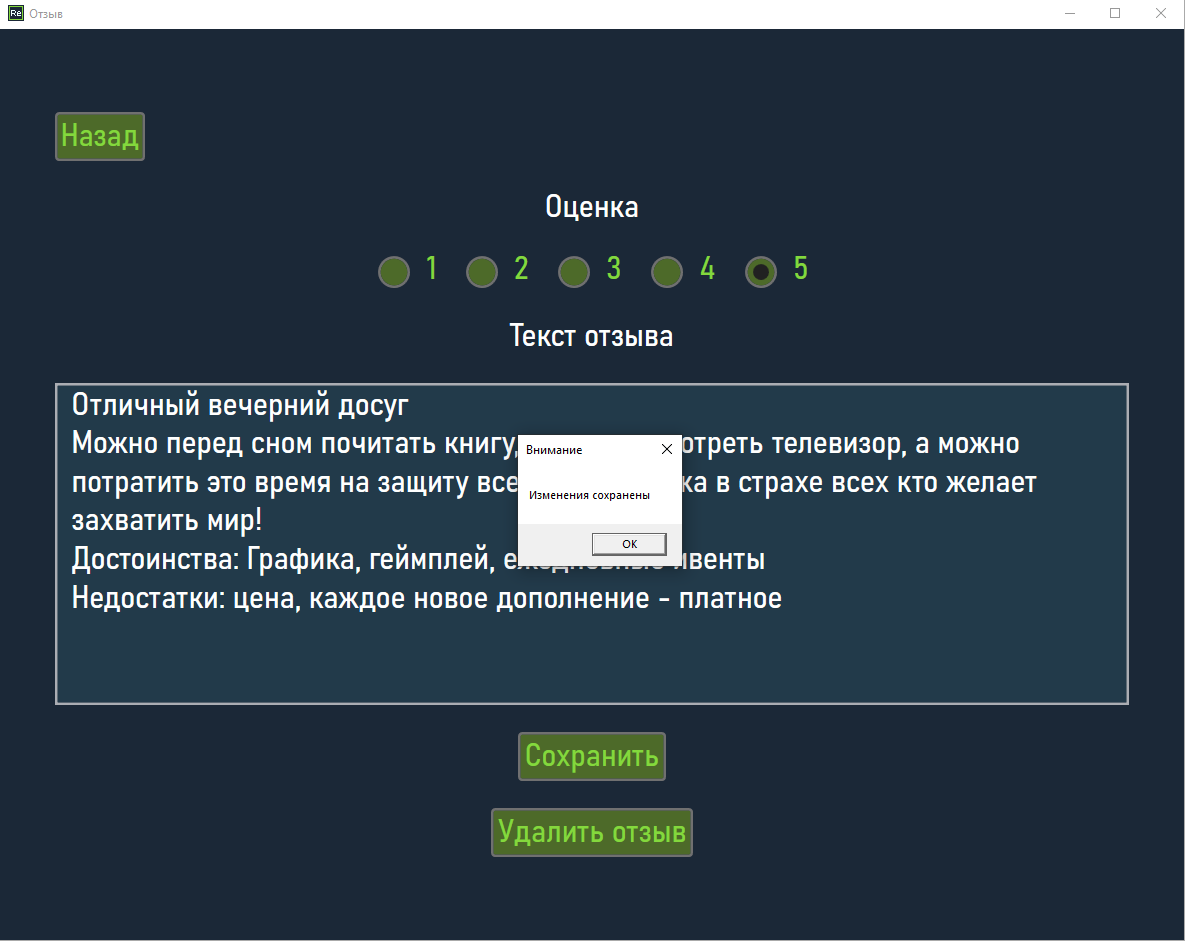
Рисунок 2.2.4 – Сообщение о добавлении отзыва

Тестирование на изменение отзыва об игре без указания текста отзыва, ожидаемое сообщение «Заполните все поля» (рисунок 2.2.5).

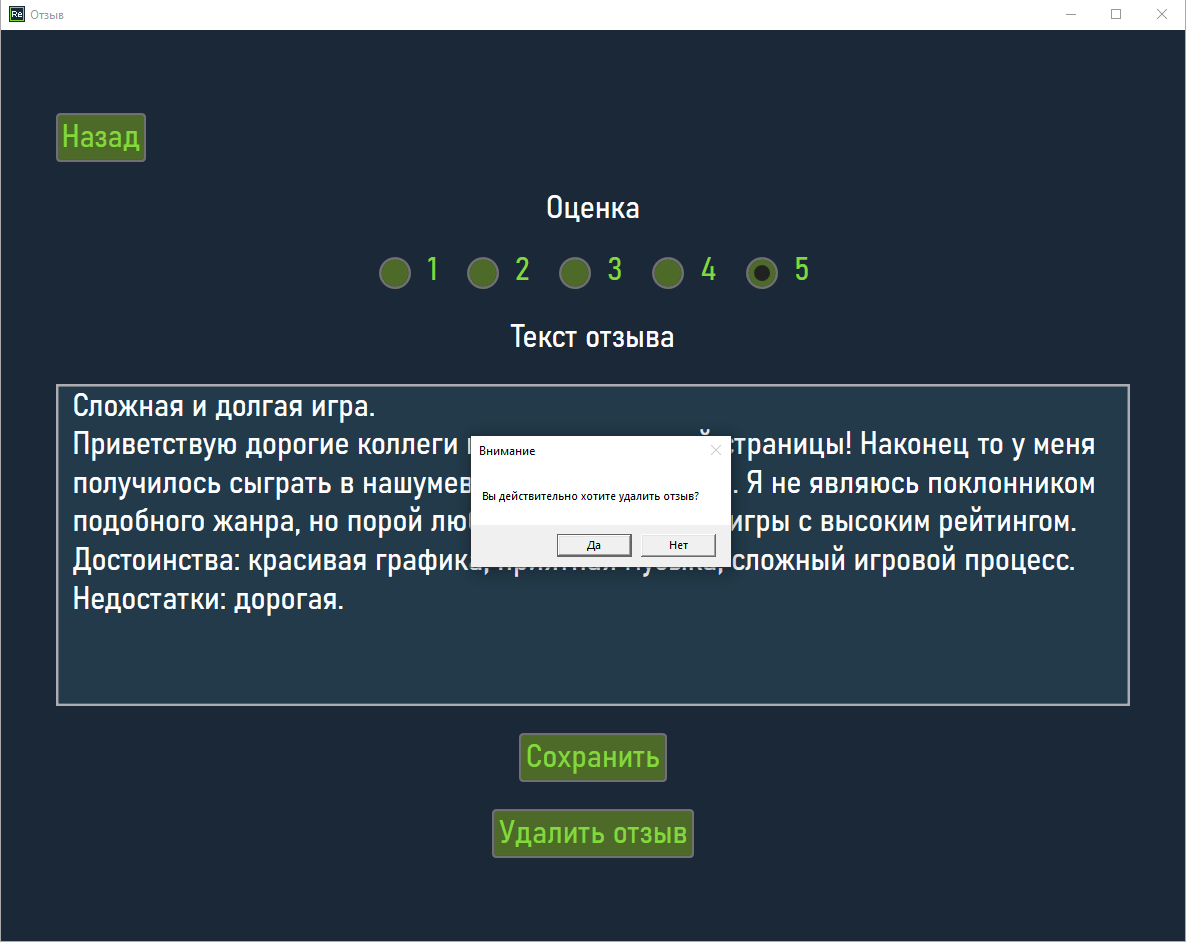
Рисунок 2.2.5 – Сообщение с просьбой заполнить все поля

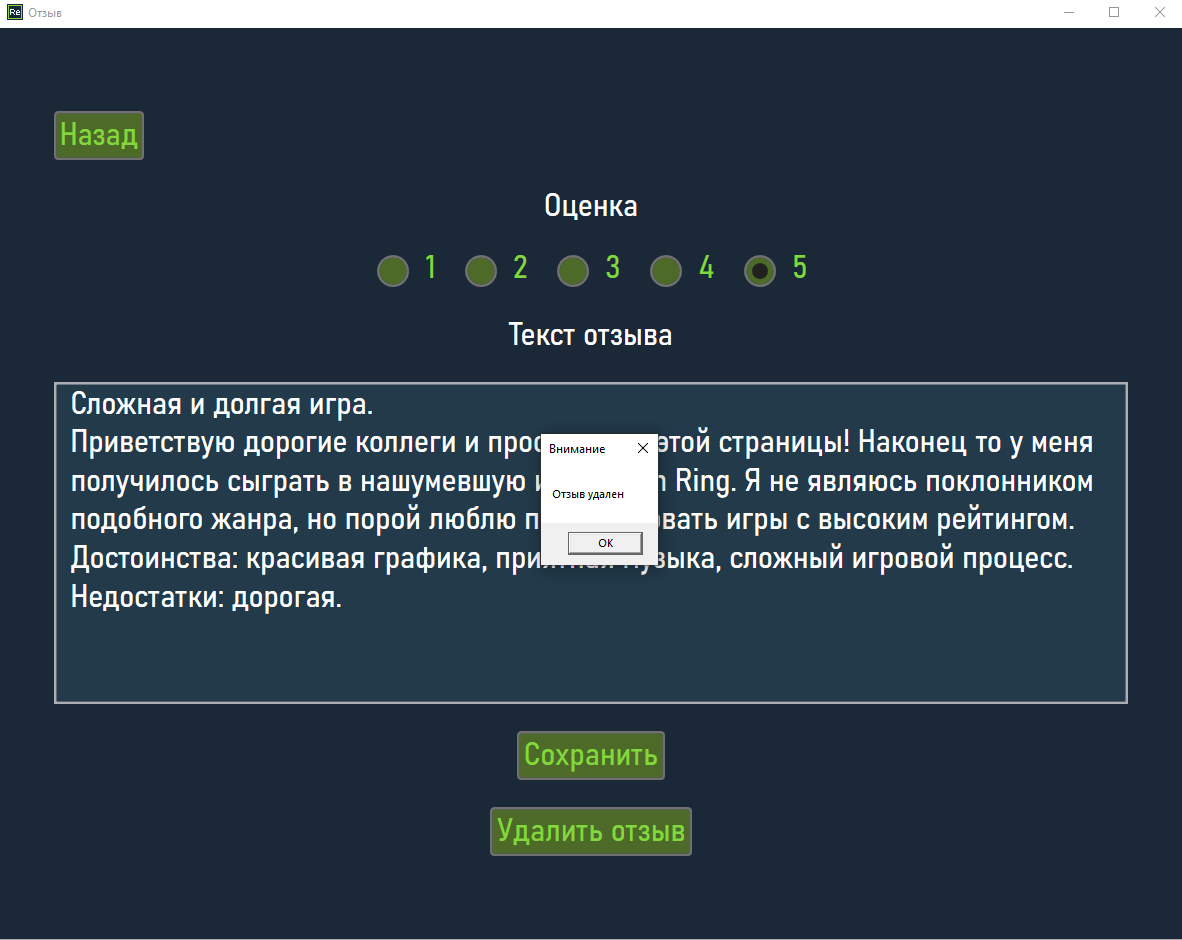
Тестирование на изменение отзыва об игре, ожидаемое сообщение «Изменения сохранены» (рисунки 2.2.6 – 2.2.7).

Рисунок 2.2.6 – Сообщение с подтверждением изменения отзыва

Рисунок 2.2.7 – Сообщение об изменении отзыва

Тестирование на удаление отзыва, ожидаемое сообщение «Отзыв удален» (рисунки 2.2.8 – 2.2.9).

Рисунок 2.2.8 – Сообщение с подтверждением удаления отзыва

Рисунок 2.2.9 – Сообщение об удалении отзыва

При тестировании продукта на данных контрольного примера, результаты полностью совпадают с ожидаемыми (рисунки 2.2.10 – 2.2.15).

Рисунок 2.2.10 – Информация игры до добавления нового отзыва

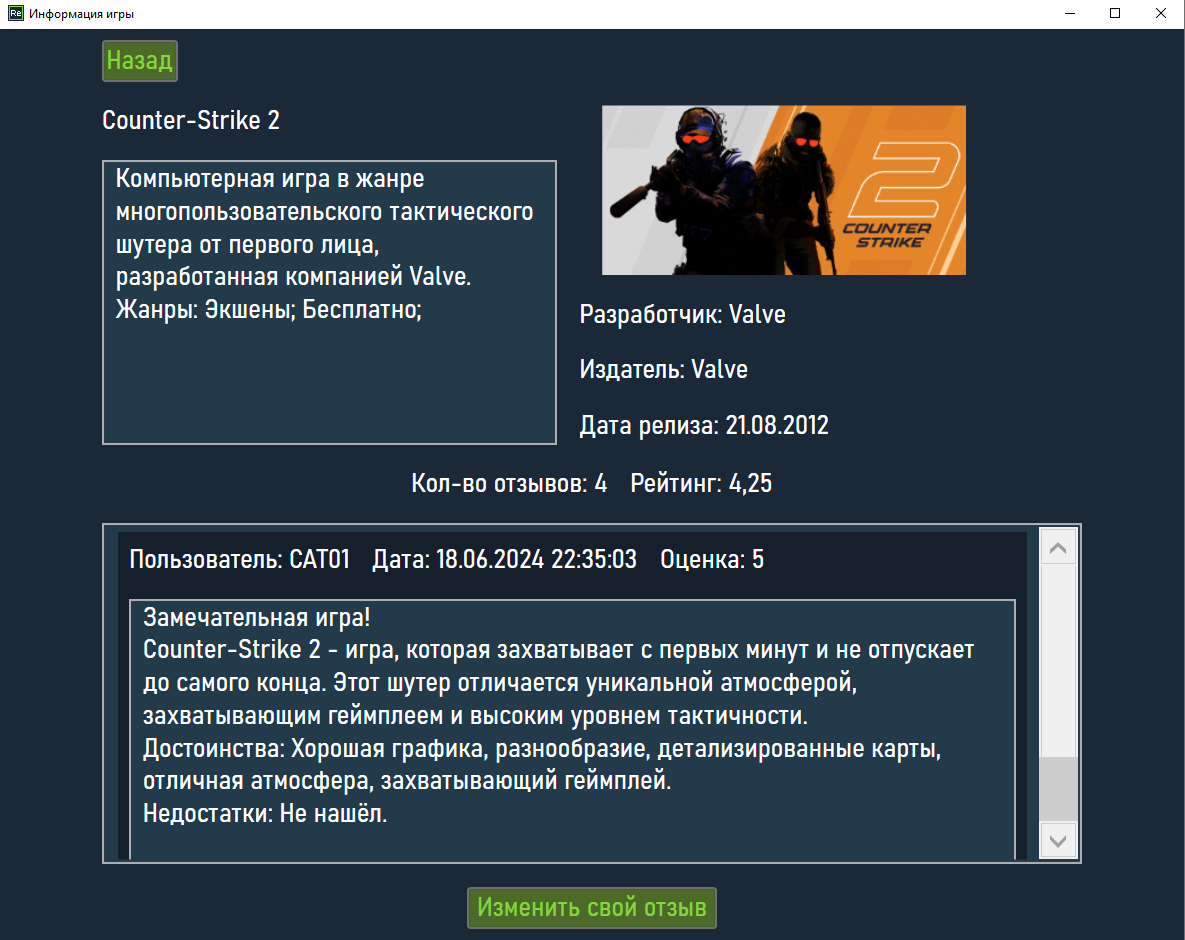
Рисунок 2.2.11 – Информация игры после добавления нового отзыва

Рисунок 2.2.12 – Информация игры до изменения отзыва

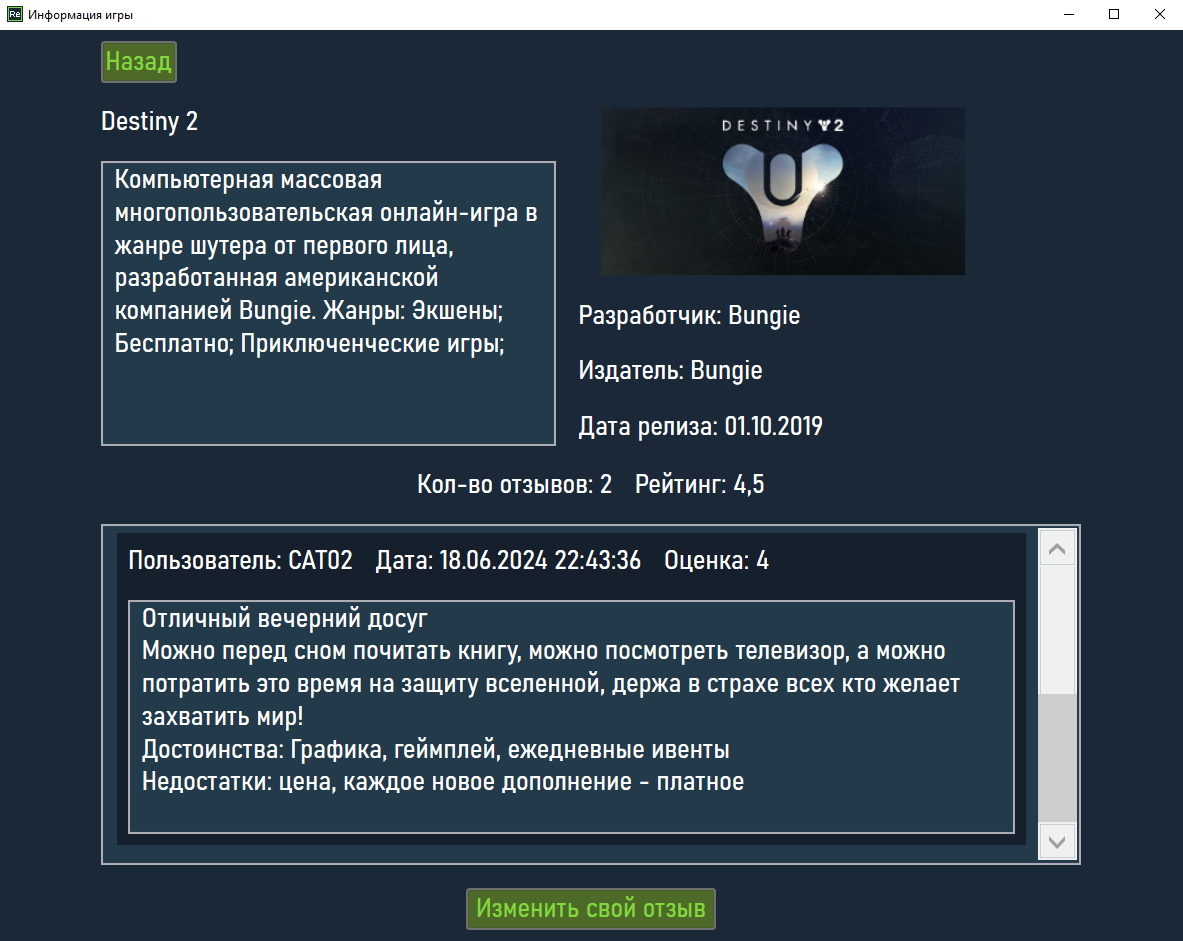


Рисунок 2.2.13 – Информация игры после изменения отзыва

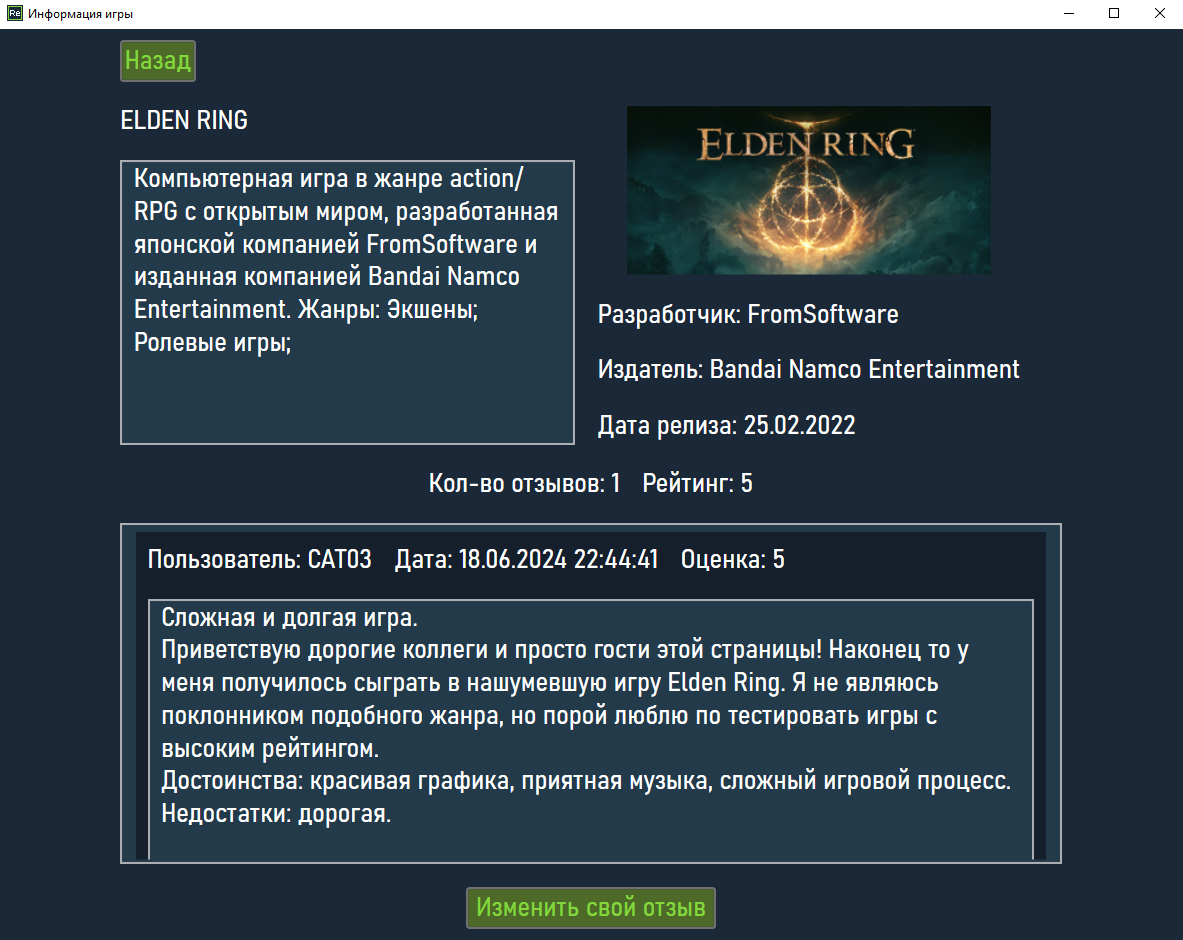


Рисунок 2.2.14 – Информация игры до удаления отзыва

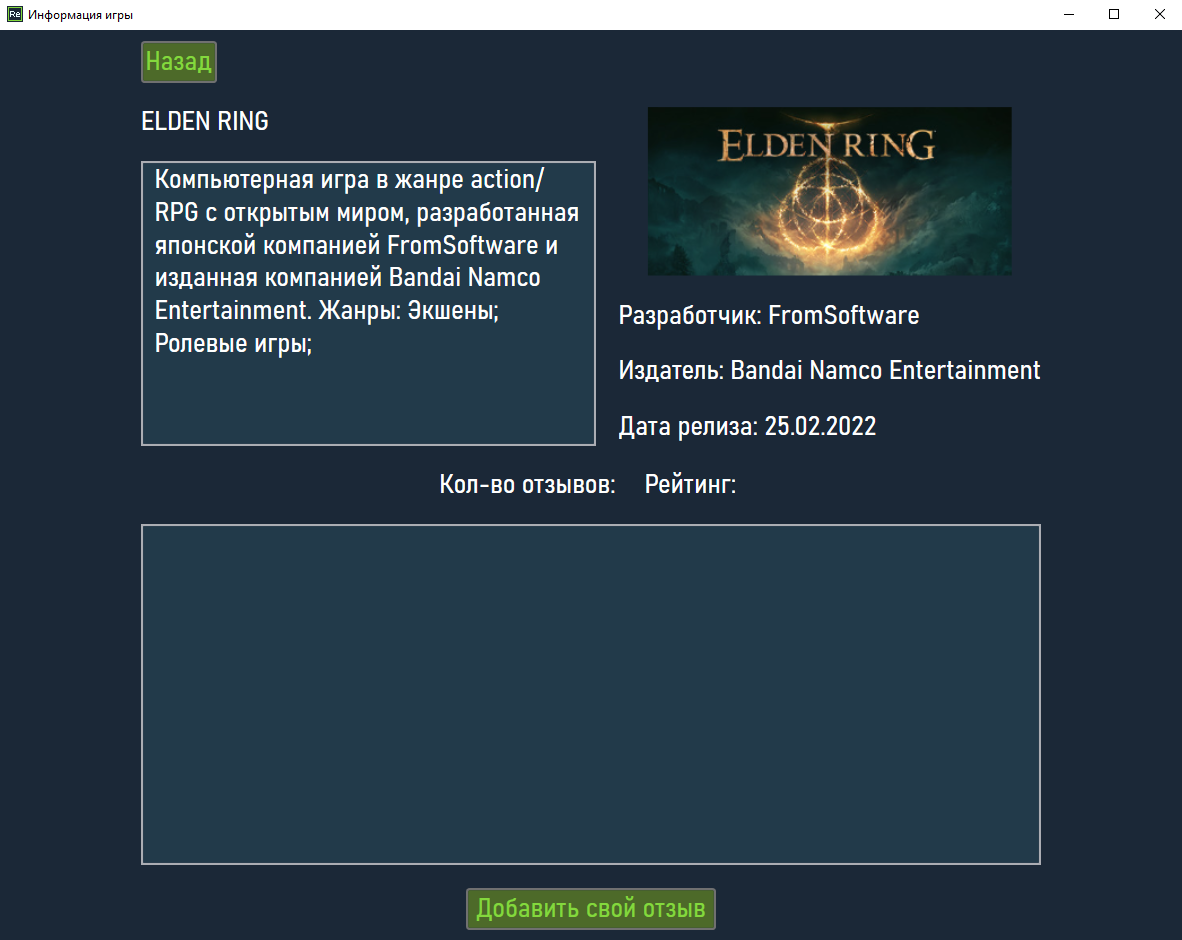


Рисунок 2.2.15 – Информация игры после удаления отзыва

2.3 Руководство пользователя

Назначение системы

Программа «Reviews» предназначена для упрощения поиска и просмотра пользователем основной информации игр, а также просмотра отзывов других пользователей.

Основной целью данной информационной системы является отображение списков игр и отзывов, а также основной информации игры.

Условия применения системы

Программное обеспечение разрабатывается для персональной вычислительной техники со следующими характеристиками:

* Microsoft Windows 7 / 8 / 10;
* процессор 1 ГГц;
* 128 МБ ОЗУ;
* 60 МБ свободного пространства на диске;
* разрешение экрана монитора не менее 1920 × 1080;
* дополнительное оборудование;
* принтер.

Программа «Reviews» предназначена для пользователей, имеющих как минимум первоначальные навыки работы с графической операционной системой, на которой будет запускаться данная программа.

Подготовка системы к работе

Для запуска программы необходимо запустить приложение Reviews.exe из каталога, в котором установлен данный программный продукт. После этого открывается окно авторизации для входа в программу (рисунок 2.3.1).

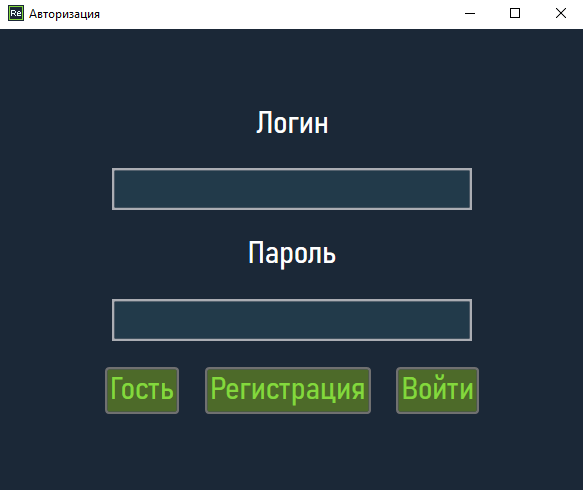
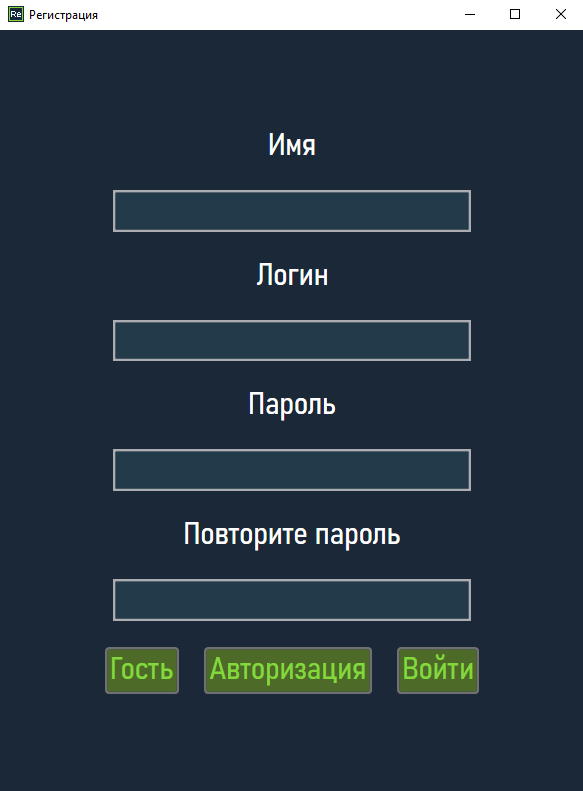
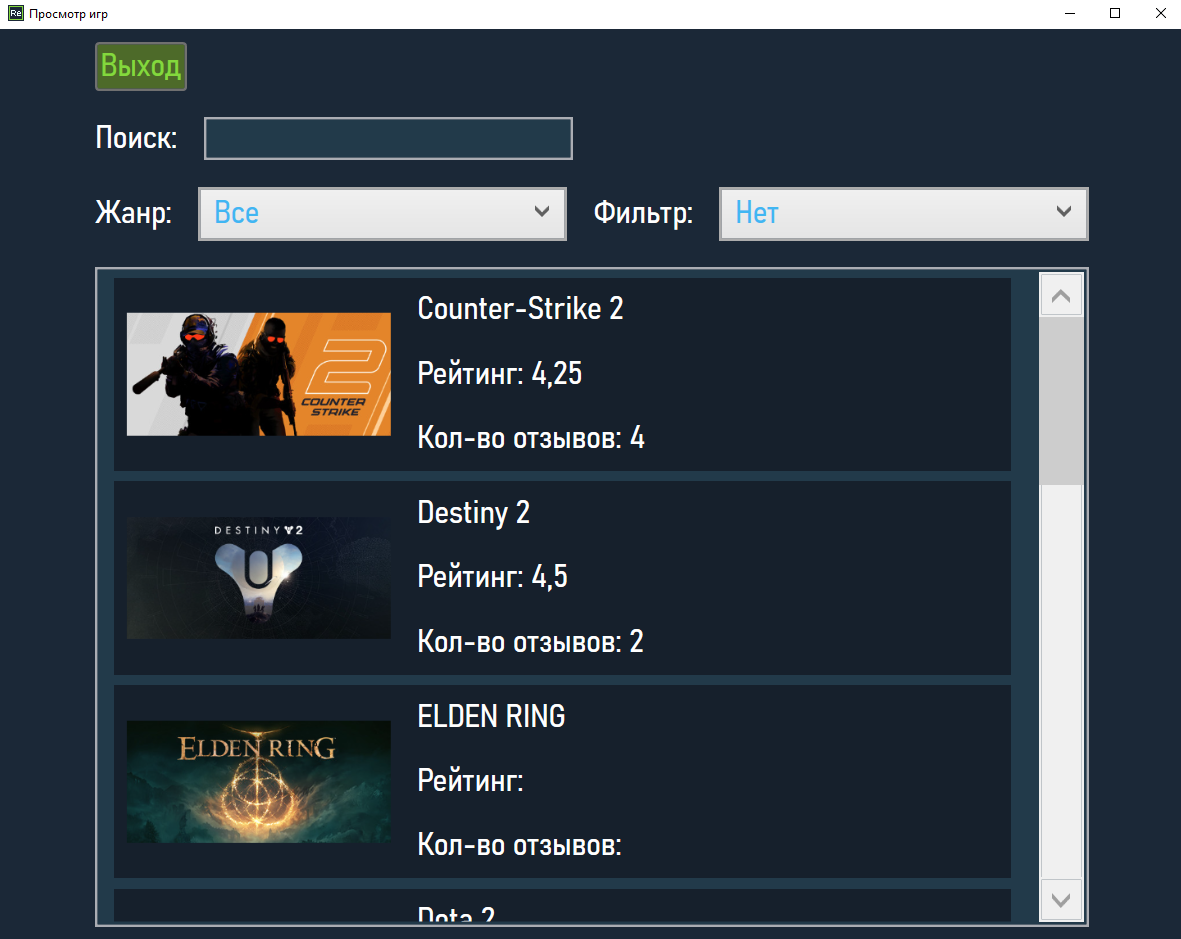


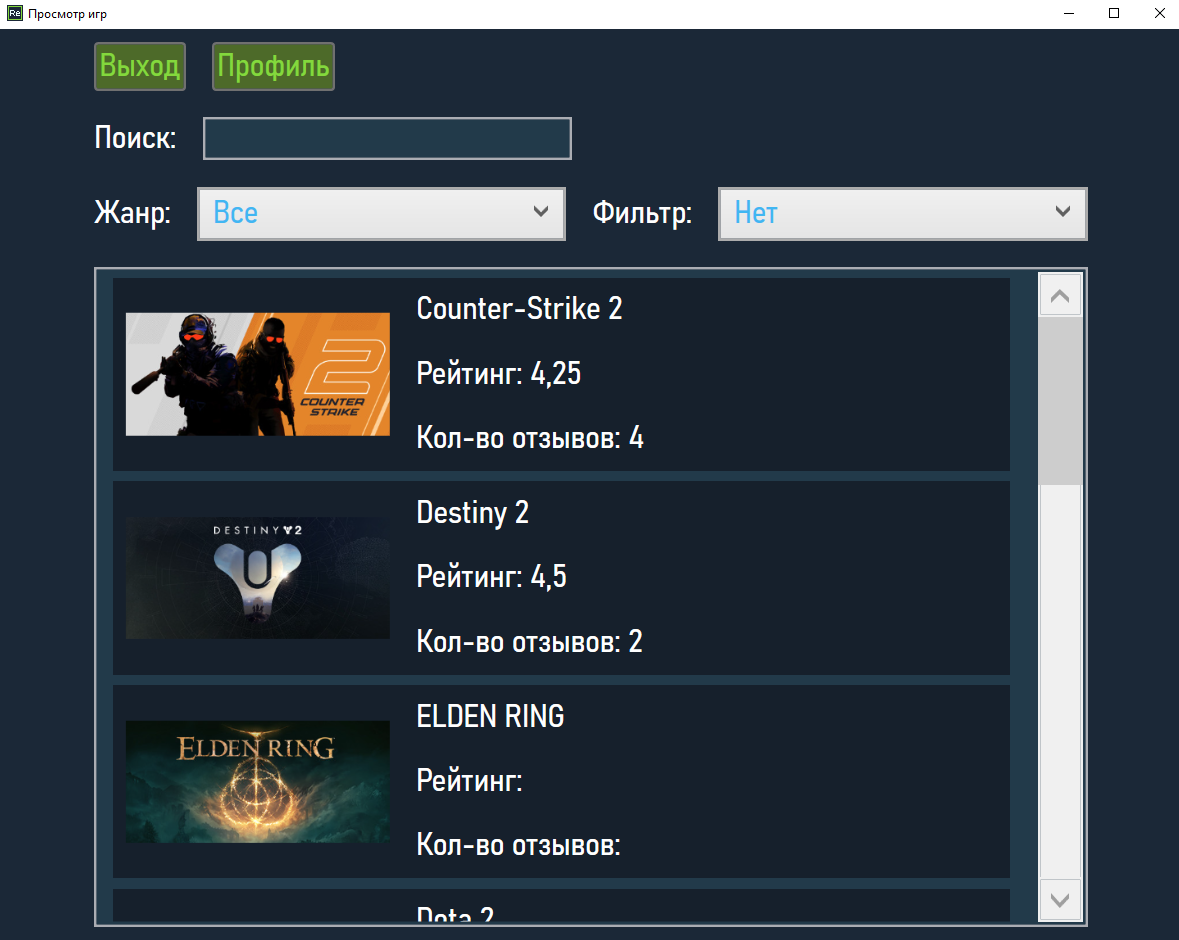
Рисунок 2.3.1 – Окно авторизации для входа в программу

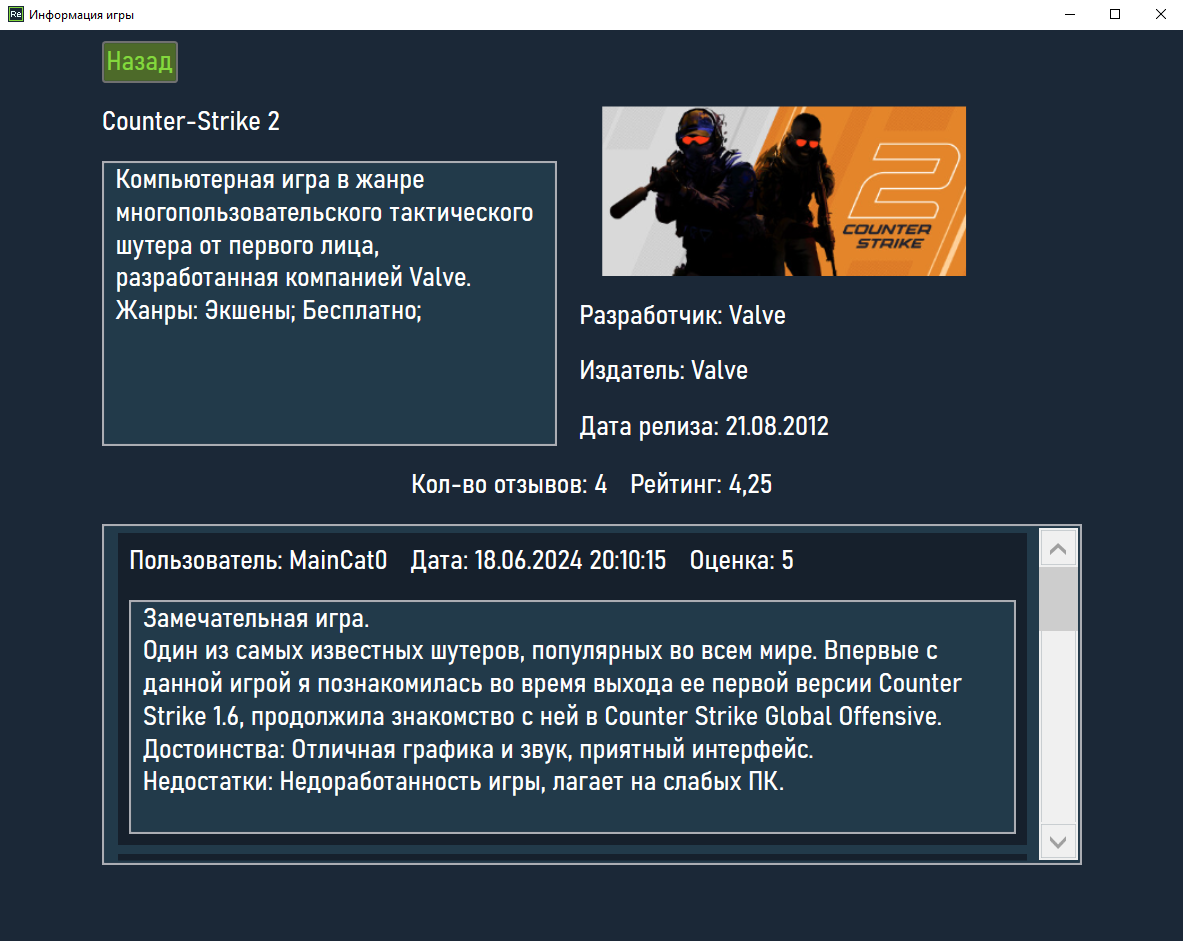
При нажатии на кнопку «Регистрация» открывается окно регистрации для входа в программу (рисунок 2.3.2).

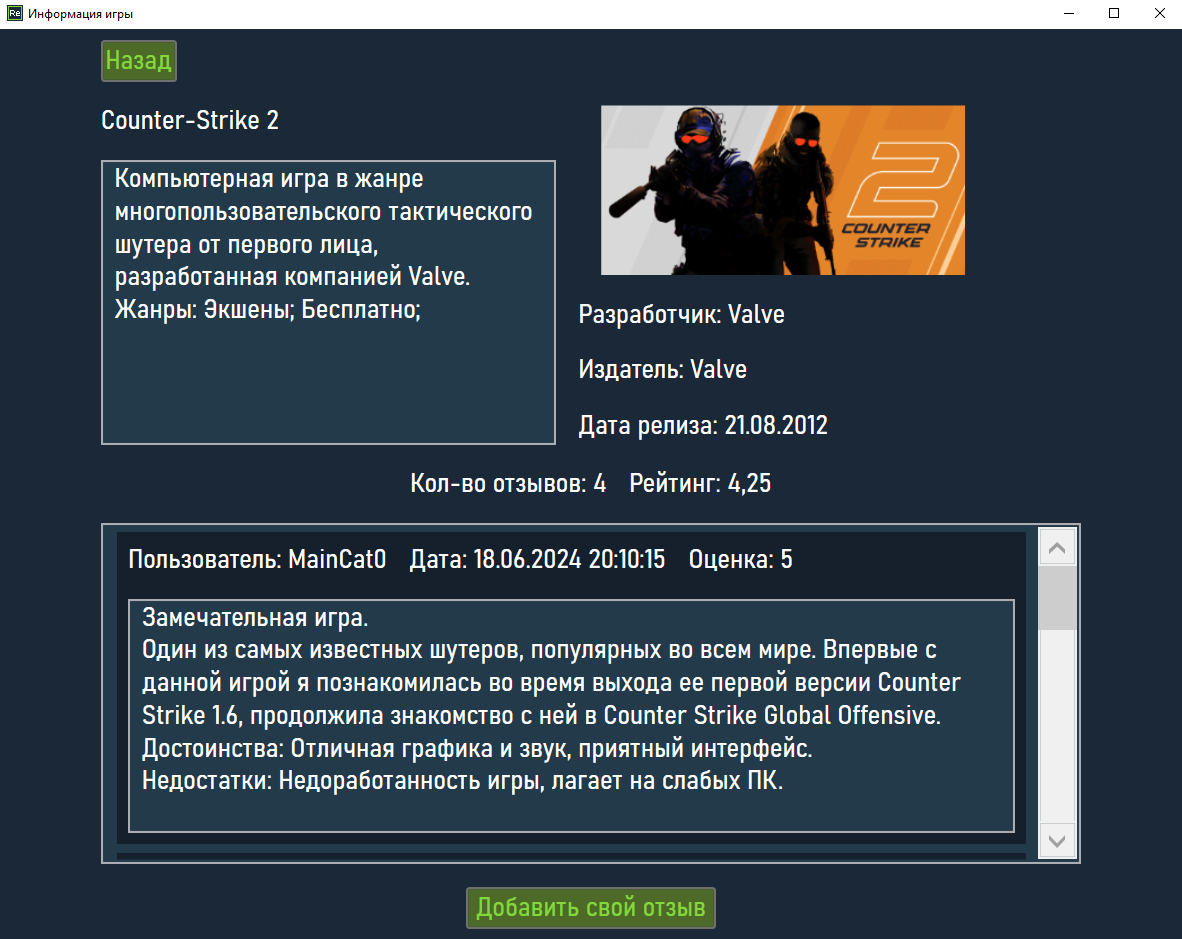
Рисунок 2.3.2 – Окно регистрации для входа в программу

При нажатии на кнопку «Гость» на окне авторизации или регистрации, открывается окно просмотра игр, где пользователь сможет выполнить поиск игр по названию, жанру и фильтру, а также выйти на окно авторизации, нажав на кнопку «Выход», при этом неавторизованному пользователю будут недоступны функции, которые доступны только для авторизованных пользователей (рисунки 2.3.3 – 2.3.6).

Рисунок 2.3.3 – Окно просмотра игр для неавторизованного пользователя

Рисунок 2.3.4 – Окно просмотра игр для авторизованного пользователя

Рисунок 2.3.5 – Окно основной информации игры для неавторизованного пользователя

Рисунок 2.3.6 – Окно основной информации игры для авторизованного пользователя

После правильного ввода данных и при нажатии на кнопку «Войти» на окне авторизации или регистрации, открывается окно просмотра игр, где пользователь сможет выполнить поиск игр по названию, жанру и фильтру, а также выйти из аккаунта на окно авторизации, нажав на кнопку «Выход» (рисунки 2.3.7 – 2.3.8).

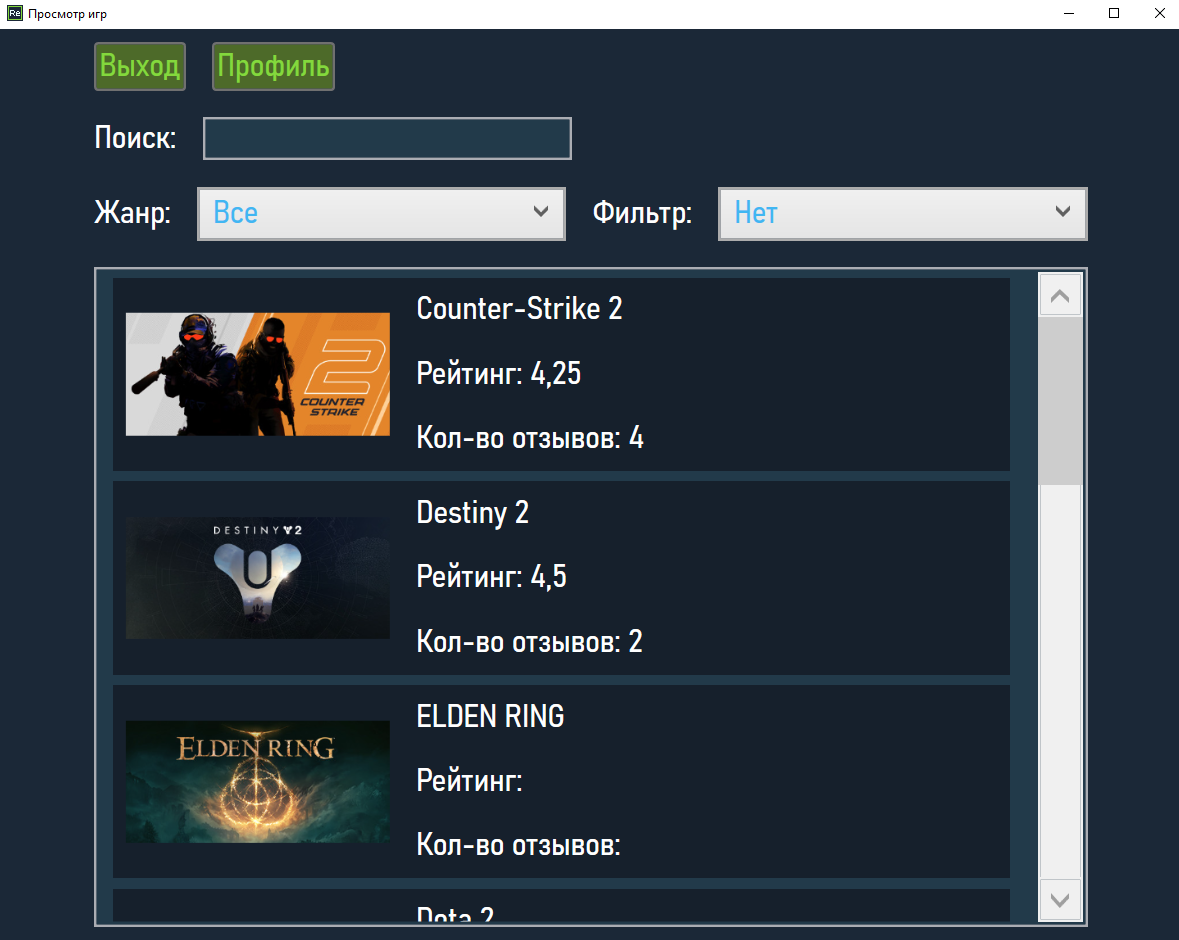


Рисунок 2.3.7 – Окно просмотра игр

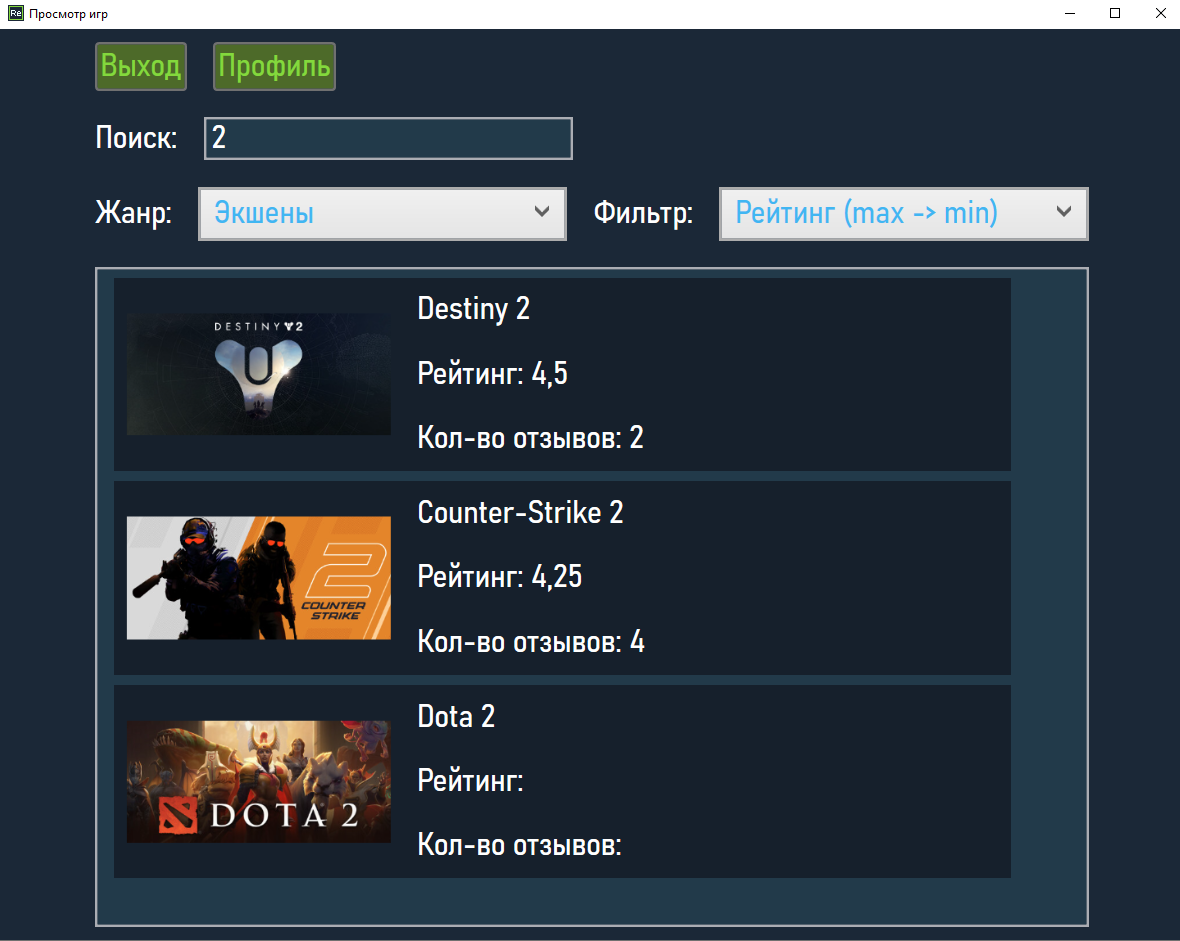
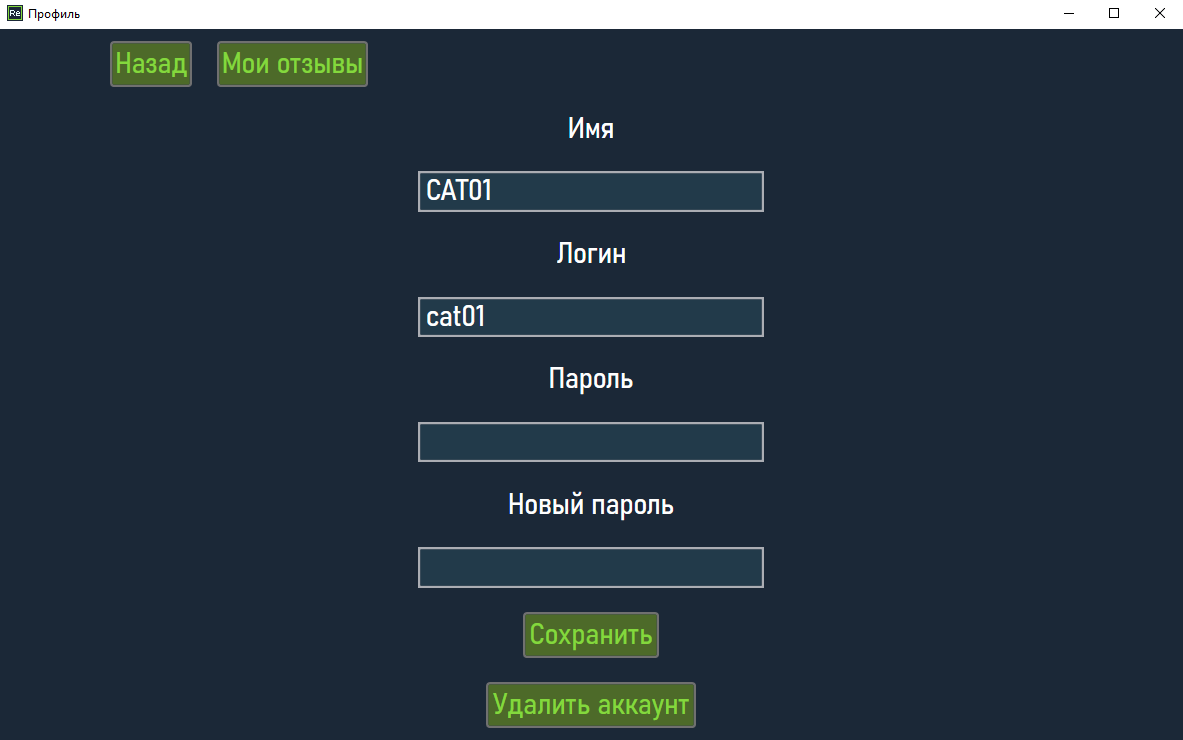
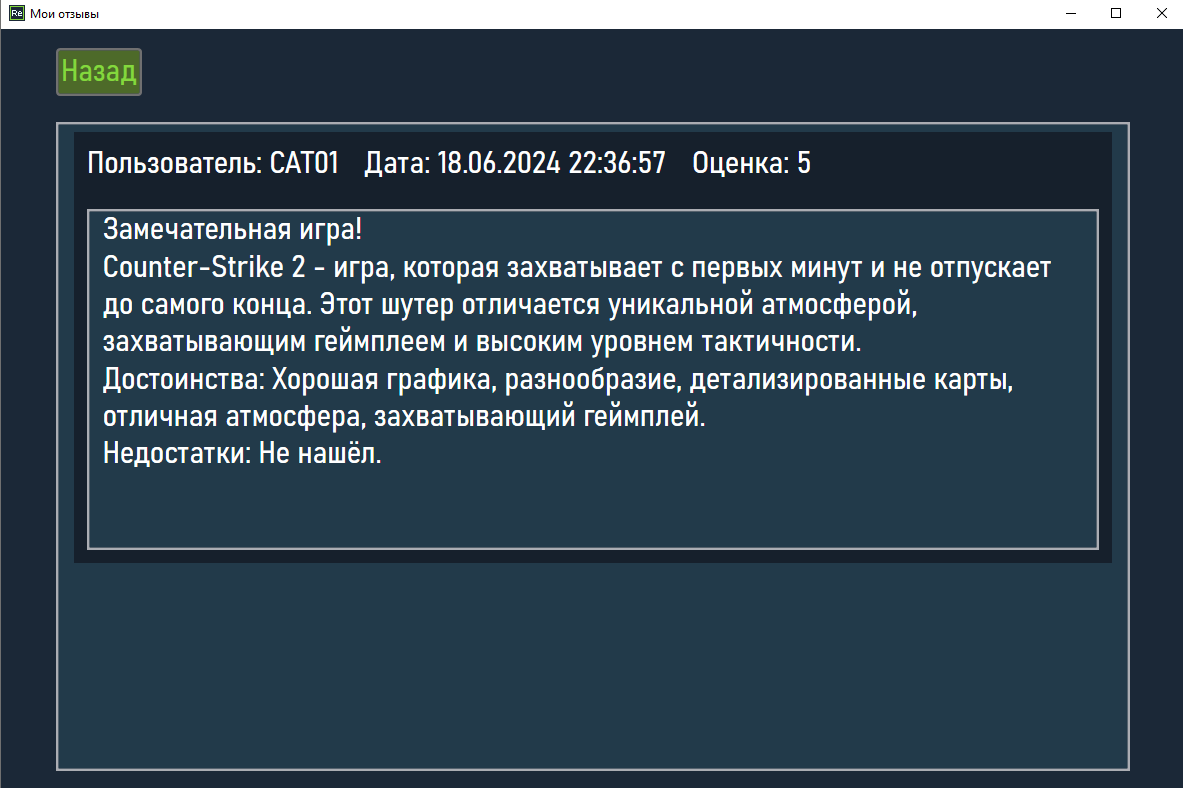


Рисунок 2.3.8 – Окно просмотра игр с выполненным поиском

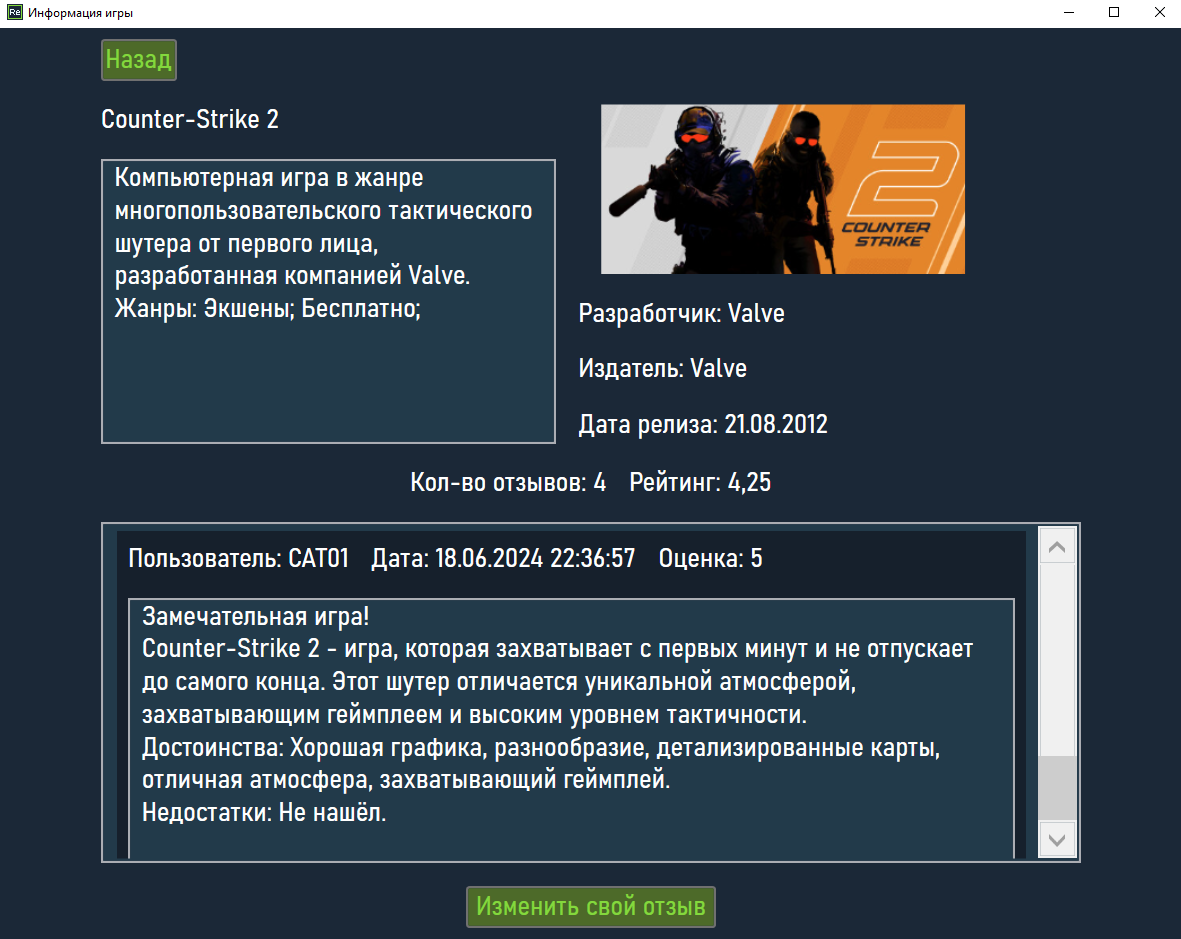
При нажатии на кнопку «Профиль» на окне просмотра игр, открывается окно профиля с данными пользователя, где возможно их изменить при нажатии на кнопку «Сохранить», заполнив поле пароля, а также удалить аккаунт при нажатии на кнопку «Удалить аккаунт», заполнив поле пароля, вернуться на окно просмотра игр, нажав на кнопку «Назад». При удалении аккаунта также удаляются все отзывы пользователя (рисунок 2.3.9).

Рисунок 2.3.9 – Окно профиля пользователя

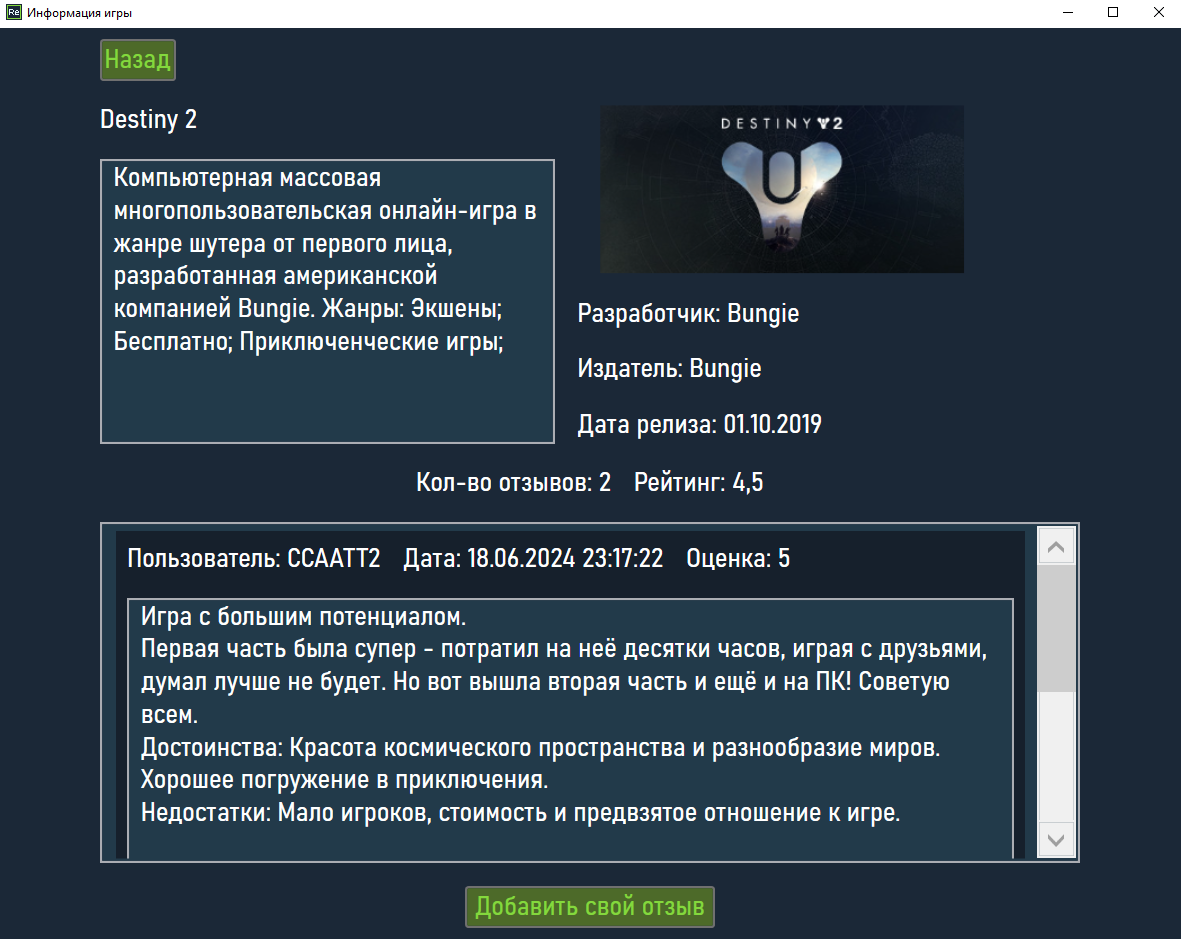
При нажатии на кнопку «Мои отзывы» на окне профиля пользователя, открывается окно просмотра отзывов пользователя, где возможно вернуться на окно просмотра профиля, нажав на кнопку «Назад» (рисунок 2.3.10).

Рисунок 2.3.10 – Окно просмотра отзывов пользователя

При двойном нажатии на карточку отзыва на окне просмотра отзывов пользователя, открывается окно просмотра основной информации игры, к которой был оставлен этот отзыв, где возможно вернуться на окно просмотра отзывов пользователя, нажав на кнопку «Назад» (рисунок 2.3.11).

Рисунок 2.3.11 – Окно просмотра основной информации игры

При двойном нажатии на карточку игры на окне просмотра игр, открывается окно просмотра основной информации игры, где возможно вернуться на окно просмотра игр, нажав на кнопку «Назад» (рисунки 2.3.12 – 2.3.13).

Рисунок 2.3.12 – Окно просмотра основной информации игры, если у пользователя нет отзыва на выбранную игру

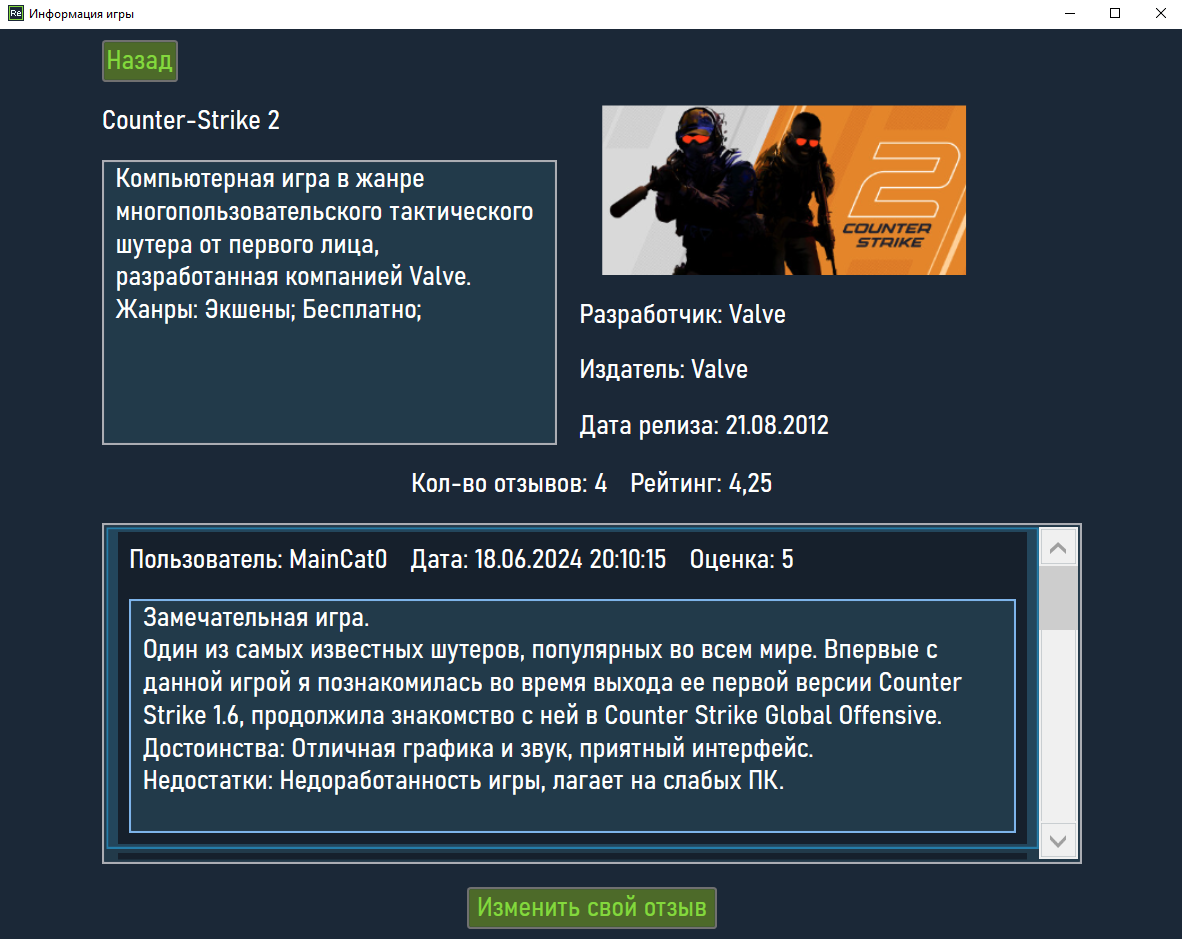
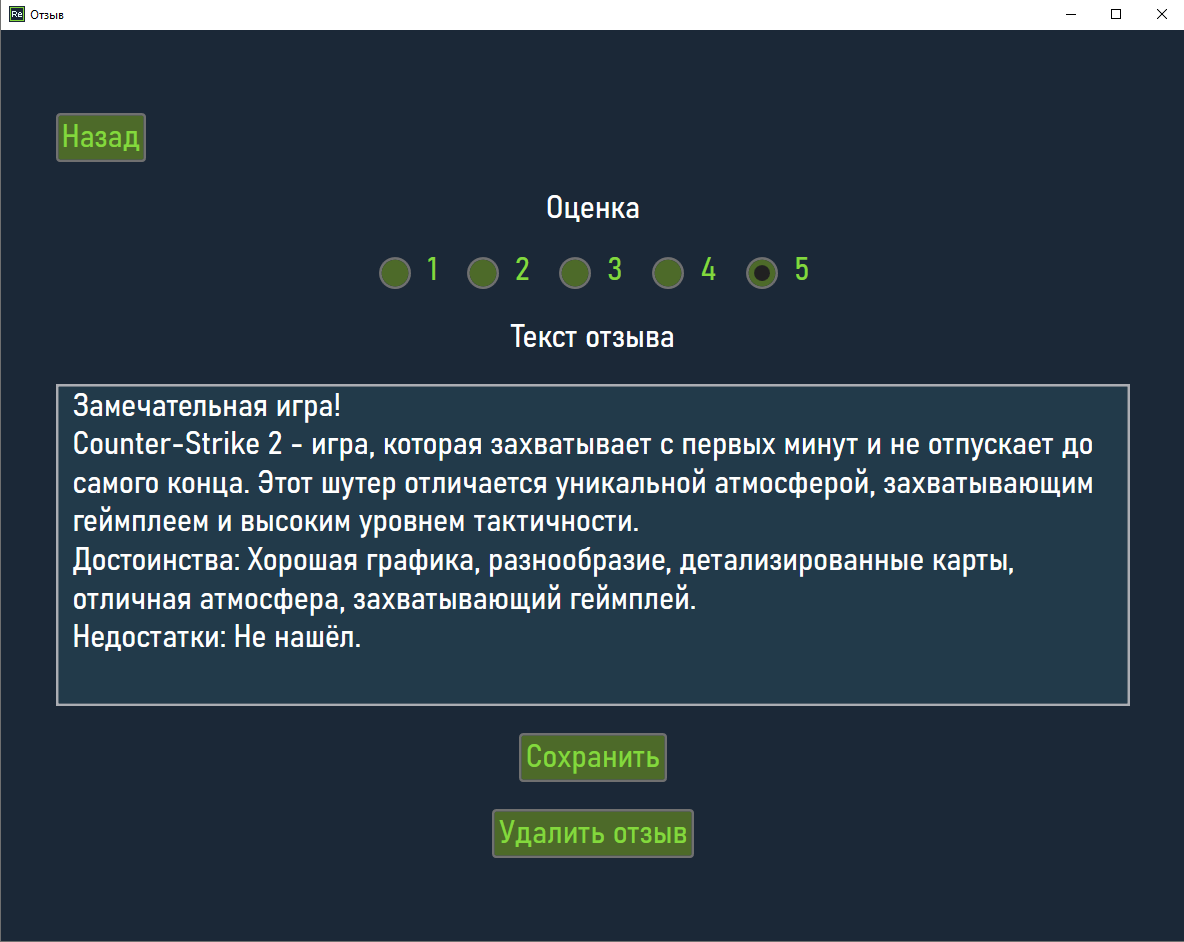
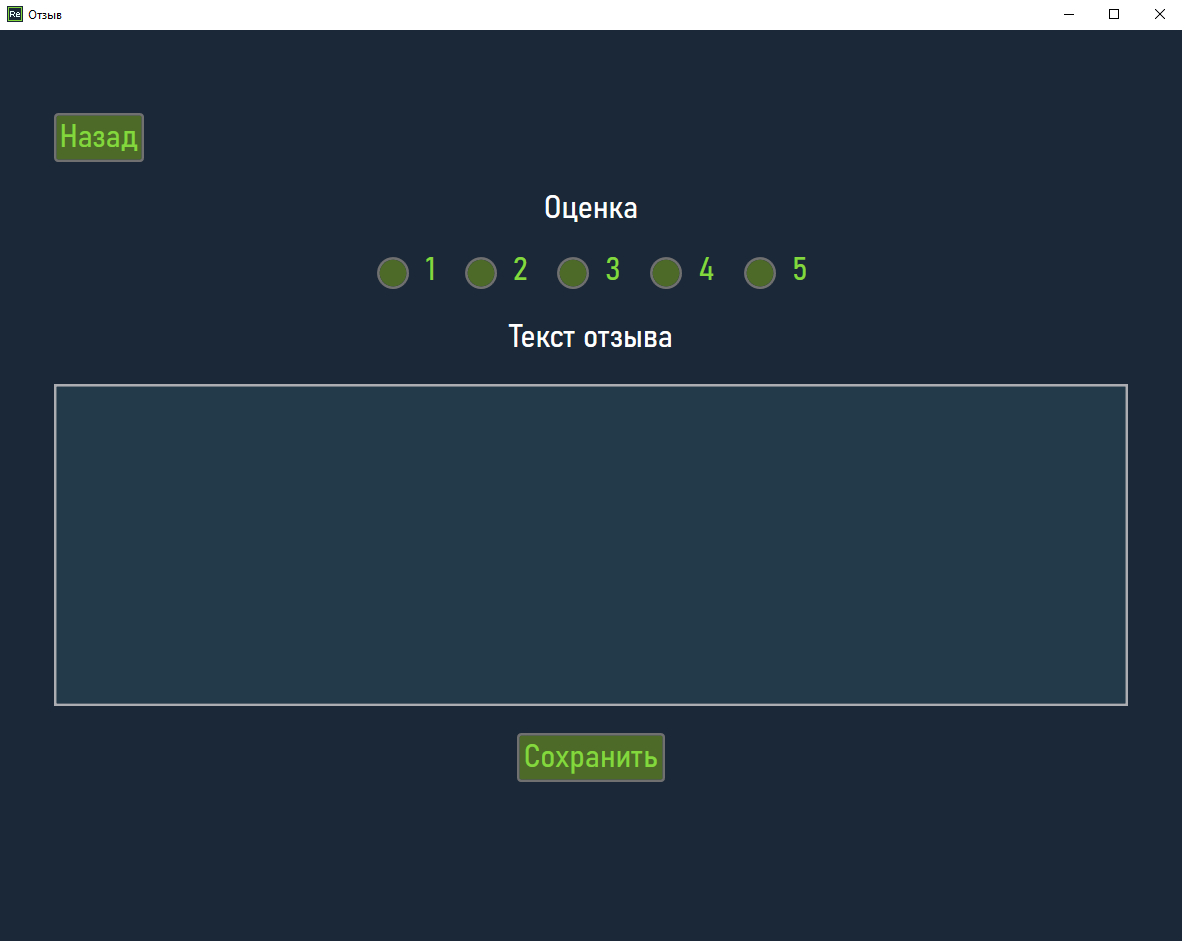


Рисунок 2.3.13 – Окно просмотра основной информации игры, если у пользователя есть отзыв на выбранную игру

При нажатии на кнопку «Добавить свой отзыв» или «Изменить свой отзыв» на окне просмотра основной информации игры, открывается окно отзыва, где возможно добавить или изменить отзыв, нажав на кнопку «Сохранить», а также удалить отзыв, нажав на кнопку «Удалить отзыв», вернуться на окно просмотра основной информации игры, нажав на кнопку «Назад» (рисунки 2.3.14 – 2.3.15).

Рисунок 2.3.14 – Окно просмотра отзыва, если у пользователя есть отзыв на выбранную игру

Рисунок 2.3.15 – Окно просмотра отзыва, если у пользователя нет отзыва на выбранную игру

ЗАКЛЮЧЕНИЕ

В процессе выполнения курсового проекта были разработаны структура и алгоритм работы WPF-приложения «Reviews».

При этом были изучены особенности реализации компонентов WPF для построения клиентских приложений с визуально привлекательными возможностями взаимодействия с пользователем.

Результатом работы стало создание WPF-приложения для сокращения времени на поиск основной информации об играх.

WPF-приложение написано на языке C# в среде разработки Visual Studio 2022 с использованием языка разметки XAML и системы управления базой данных MySQL Workbench.

Были проведены опытная эксплуатация и отладочное тестирование WPF приложения. По результатам отладочного тестирования недостатки не обнаружены.

С помощью приложения на основании данных контрольного примера были получены результаты, которые полностью совпадают с выходной информацией контрольного примера.

Входные данные контрольного примера

Таблица А.1 – Справочник «Разработчики»

|  |  |
| --- | --- |
| ID Разработчика | Название |
| 1 | Valve |
| 2 | Bungie |
| 3 | FromSoftware |

Таблица А.2 – Справочник «Игры-жанры»

|  |  |
| --- | --- |
| ID игры | ID жанра |
| 1 | 1 |
| 2 | 1 |
| 3 | 1 |
| 1 | 2 |
| 2 | 2 |
| 2 | 3 |
| 3 | 4 |

Таблица А.3 – Справочник «Игры»

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ID игры | Название | Описание | ID разработчика | ID издателя | Дата релиза | Картинка | Рейтинг | Кол-во отзывов |
| 1 | Counter-Strike 2 | Компьютерная игра в жанре многопользовательского тактического шутера от первого лица, разработанная компанией Valve. | 1 | 1 | 2012-08-21 | CounterStrike2.png | 4 | 3 |
| 2 | Destiny 2 | Компьютерная массовая многопользовательская онлайн-игра в жанре шутера от первого лица, разработанная американской компанией Bungie. | 2 | 2 | 2019-10-01 | Destiny2.png | 5 | 2 |
| 3 | ELDEN RING | Компьютерная игра в жанре action/RPG с открытым миром, разработанная японской компанией FromSoftware и изданная компанией Bandai Namco Entertainment. | 3 | 3 | 2022-02-25 | ELDENRING.png | 5 | 1 |

Таблица А.4 – Справочник «Жанры»

|  |  |
| --- | --- |
| ID жанра | Название |
| 1 | Экшены |
| 2 | Бесплатно |
| 3 | Приключенческие игры |
| 4 | Ролевые игры |

Таблица А.5 – Справочник «Издатели»

|  |  |
| --- | --- |
| ID издателя | Название |
| 1 | Valve |
| 2 | Bungie |
| 3 | Bandai Namco Entertainment |

Таблица А.6 – Справочник «Отзывы»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID отзыва | Текст | Оценка | ID игры | ID пользователя | Дата |
| 1 | 2 | 3 | 4 | 5 | 6 |
| 2 | Замечательная игра. Один из самых известных шутеров, популярных во всем мире. Впервые с данной игрой я познакомилась во время выхода ее первой версии Counter Strike 1.6, продолжила знакомство с ней в Counter Strike Global Offensive. Достоинства: Отличная графика и звук, приятный интерфейс. Недостатки: Недоработанность игры, лагает на слабых ПК. | 5 | 1 | 1 | 2024-06-18 20:10:15 |
| 3 | Хороший шутер. Контра одна из полулярных игр на данный момент, особенно после глобального обновления стала ещё и красивой. После смены движка и других изменений картинка стала более насыщенной и красивой, изменились некоторые эффекты например дымовая граната. Достоинства: Качество игры, геймплей, графика. Недостатки: Злые школьники. | 4 | 1 | 2 | 2024-06-18 20:10:50 |
| 5 | Counter Strike 2 cплошное разочарование. CSGO 2 оказалась полным разочарованием по сравнению с первой частью. Разработчики, кажется, забыли, что делает Counter-Strike настоящей легендой. Графика может быть улучшена, но это не компенсирует отсутствие внимания к геймплею. Переработка карт выглядит как неудачная. Достоинства: красивая графика, возможность покидать в руках гранаты/бомбу через осмотр. Недостатки: отсутствие старых карт, другая стрельба, плохая оптимизация. | 3 | 1 | 3 | 2024-06-18 20:11:23 |

Продолжение таблицы А.6

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 |
| 7 | Сложная и долгая игра. Приветствую дорогие коллеги и просто гости этой страницы! Наконец то у меня получилось сыграть в нашумевшую игру Elden Ring. Я не являюсь поклонником подобного жанра, но порой люблю по тестировать игры с высоким рейтингом. Достоинства: красивая графика, приятная музыка, сложный игровой процесс. Недостатки: дорогая. | 5 | 3 | 12 | 2024-06-18 20:29:14 |
| 20 | Игра с большим потенциалом. Первая часть была супер - потратил на неё десятки часов, играя с друзьями, думал лучше не будет. Но вот вышла вторая часть и ещё и на ПК! Советую всем. Достоинства: Красота космического пространства и разнообразие миров. Хорошее погружение в приключения. Недостатки: Мало игроков, стоимость и предвзятое отношение к игре. | 5 | 2 | 3 | 2024-06-18 20:21:09 |
| 22 | Шикарная игра! Игра просто на высшем уровне, графика заставляет полностью погрузиться в мир космоса. С каждым новым обновлением история все интереснее и интереснее, можно даже целый фильм сделать. Так же очень класнно проходить налеты и всякие соревнования. Достоинства: Отличная графика, классный сюжет, интересная система прокачки. Недостатки: Нету | 5 | 2 | 11 | 2024-06-18 20:17:32 |

Таблица А.7 – Справочник «Пользователи»

|  |  |  |  |
| --- | --- | --- | --- |
| ID пользователя | Логин | Пароль | Имя |
| 10 | cat01 | 01 | CAT01 |
| 11 | cat02 | 02 | CAT02 |
| 12 | cat03 | 03 | CAT03 |

Выходные данные контрольного примера

Таблица Б.1 – Справочник «Игры»

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ID игры | Название | Описание | ID разработчика | ID издателя | Дата релиза | Картинка | Рейтинг | Кол-во отзывов |
| 1 | Counter-Strike 2 | Компьютерная игра в жанре многопользовательского тактического шутера от первого лица, разработанная компанией Valve. | 1 | 1 | 2012-08-21 | CounterStrike2.png | 4 | 4 |
| 2 | Destiny 2 | Компьютерная массовая многопользовательская онлайн-игра в жанре шутера от первого лица, разработанная американской компанией Bungie. | 2 | 2 | 2019-10-01 | Destiny2.png | 5 | 2 |
| 3 | ELDEN RING | Компьютерная игра в жанре action/RPG с открытым миром, разработанная японской компанией FromSoftware и изданная компанией Bandai Namco Entertainment. | 3 | 3 | 2022-02-25 | ELDENRING.png | NULL | NULL |

Таблица Б.2 – Справочник «Отзывы»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID отзыва | Текст | Оценка | ID игры | ID пользователя | Дата |
| 1 | 2 | 3 | 4 | 5 | 6 |
| 2 | Замечательная игра. Один из самых известных шутеров, популярных во всем мире. Впервые с данной игрой я познакомилась во время выхода ее первой версии Counter Strike 1.6, продолжила знакомство с ней в Counter Strike Global Offensive. Достоинства: Отличная графика и звук, приятный интерфейс. Недостатки: Недоработанность игры, лагает на слабых ПК. | 5 | 1 | 1 | 2024-06-18 20:10:15 |
| 3 | Хороший шутер. Контра одна из полулярных игр на данный момент, особенно после глобального обновления стала ещё и красивой. После смены движка и других изменений картинка стала более насыщенной и красивой, изменились некоторые эффекты например дымовая граната. Достоинства: Качество игры, геймплей, графика. Недостатки: Злые школьники. | 4 | 1 | 2 | 2024-06-18 20:10:50 |
| 5 | Counter Strike 2 cплошное разочарование. CSGO 2 оказалась полным разочарованием по сравнению с первой частью. Разработчики, кажется, забыли, что делает Counter-Strike настоящей легендой. Графика может быть улучшена, но это не компенсирует отсутствие внимания к геймплею. Переработка карт выглядит как неудачная. Достоинства: красивая графика, возможность покидать в руках гранаты/бомбу через осмотр. Недостатки: отсутствие старых карт, другая стрельба, плохая оптимизация. | 3 | 1 | 3 | 2024-06-18 20:11:23 |

Продолжение таблицы Б.2

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 |
| 28 | Замечательная игра! Counter-Strike 2 - игра, которая захватывает с первых минут и не отпускает до самого конца. Этот шутер отличается уникальной атмосферой, захватывающим геймплеем и высоким уровнем тактичности. Достоинства: Хорошая графика, разнообразие, детализированные карты, отличная атмосфера, захватывающий геймплей. Недостатки: Не нашёл. | 5 | 1 | 10 | 2024-06-18 22:36:57 |
| 22 | Отличный вечерний досуг Можно перед сном почитать книгу, можно посмотреть телевизор, а можно потратить это время на защиту вселенной, держа в страхе всех кто желает захватить мир! Достоинства: Графика, геймплей, ежедневные ивенты Недостатки: цена, каждое новое дополнение - платное | 4 | 2 | 11 | 2024-06-18 22:43:36 |
| 20 | Игра с большим потенциалом. Первая часть была супер - потратил на неё десятки часов, играя с друзьями, думал лучше не будет. Но вот вышла вторая часть и ещё и на ПК! Советую всем. Достоинства: Красота космического пространства и разнообразие миров. Хорошее погружение в приключения. Недостатки: Мало игроков, стоимость и предвзятое отношение к игре. | 5 | 2 | 3 | 2024-06-18 20:17:32 |

Код программы

Форма App.xaml.cs

using System;

using System.Collections.Generic;

using System.Configuration;

using System.Data;

using System.Linq;

using System.Threading.Tasks;

using System.Windows;

namespace Reviews

{

/// <summary>

/// Interaction logic for App.xaml

/// </summary>

public partial class App : Application

{

public static int? UserId { get; set; }

public static Window? ActiveWindow { get; set; }

public static string? BackWindow { get; set; }

public static int? GameId { get; set; }

public static int? ReviewId { get; set; }

public static void UpdateRating()

{

using (GamereviewsContext db = new GamereviewsContext())

{

var games = db.Games;

var reviews = db.Reviews.ToList();

foreach (var game in games)

{

int count = 0;

double sum = 0;

foreach (var review in reviews)

{

if (review.ReviewGameId == game.GameId)

{

sum = sum + review.ReviewEstimate;

count = count + 1;

}

}

if (count != 0 && sum != 0)

{

double rating = sum / count;

rating = Math.Round(rating, 2);

game.GameRating = rating;

game.GameCommentsCount = count;

}

else

{

game.GameRating = null;

game.GameCommentsCount = null;

}

}

db.SaveChanges();

}

}

}

}

Форма MainWindow.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using Reviews.Wndws;

namespace Reviews

{

/// <summary>

/// Interaction logic for MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

App.ActiveWindow = this;

Auth auth = new Auth();

auth.Show();

}

}

}

Форма AddReview.xaml.cs (Отзыв)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using Microsoft.EntityFrameworkCore;

namespace Reviews.Wndws

{

/// <summary>

/// Логика взаимодействия для AddReview.xaml

/// </summary>

public partial class AddReview : Window

{

public AddReview()

{

InitializeComponent();

if (App.ActiveWindow != null) App.ActiveWindow.Close();

App.ActiveWindow = this;

using (GamereviewsContext db = new GamereviewsContext())

{

var reviews = db.Reviews.ToList();

var review = reviews.FirstOrDefault(i => i.ReviewUserId == App.UserId && i.ReviewGameId == App.GameId);

if (review != null)

{

App.ReviewId = review.ReviewId;

switch (review.ReviewEstimate)

{

case 1:

rbOne.IsChecked = true;

break;

case 2:

rbTwo.IsChecked = true;

break;

case 3:

rbThree.IsChecked = true;

break;

case 4:

rbFour.IsChecked = true;

break;

case 5:

rbFive.IsChecked = true;

break;

}

rtbReview.AppendText(review.ReviewText);

}

else

{

bDelete.Visibility = Visibility.Hidden;

}

}

}

private void bBack\_Click(object sender, RoutedEventArgs e)

{

GameInfo gameInfo = new GameInfo();

gameInfo.Show();

}

private void bSave\_Click(object sender, RoutedEventArgs e)

{

if (StringFromRichTextBox(rtbReview) != "\r\n" && (rbOne.IsChecked == true || rbTwo.IsChecked == true || rbThree.IsChecked == true || rbFour.IsChecked == true || rbFive.IsChecked == true))

{

using (GamereviewsContext db = new GamereviewsContext())

{

var reviews = db.Reviews;

var review = reviews.FirstOrDefault(i => i.ReviewUserId == App.UserId && i.ReviewGameId == App.GameId);

if (review != null)

{

var result = MessageBox.Show("Вы действительно хотите изменить отзыв?", "Внимание", MessageBoxButton.YesNo);

switch (result)

{

case MessageBoxResult.Yes:

if (rbOne.IsChecked == true) review.ReviewEstimate = 1;

else if (rbTwo.IsChecked == true) review.ReviewEstimate = 2;

else if (rbThree.IsChecked == true) review.ReviewEstimate = 3;

else if (rbFour.IsChecked == true) review.ReviewEstimate = 4;

else if (rbFive.IsChecked == true) review.ReviewEstimate = 5;

review.ReviewText = StringFromRichTextBox(rtbReview);

review.ReviewDate = DateTime.Now;

db.SaveChanges();

MessageBox.Show("Изменения сохранены", "Внимание");

break;

default:

break;

}

}

else

{

var result = MessageBox.Show("Вы действительно хотите добавить отзыв?", "Внимание", MessageBoxButton.YesNo);

switch (result)

{

case MessageBoxResult.Yes:

int gameId = 0;

if (App.GameId != null) gameId = (int)App.GameId;

int userId = 0;

if (App.UserId != null) userId = (int)App.UserId;

int estimate = 0;

if (rbOne.IsChecked == true) estimate = 1;

else if (rbTwo.IsChecked == true) estimate = 2;

else if (rbThree.IsChecked == true) estimate = 3;

else if (rbFour.IsChecked == true) estimate = 4;

else if (rbFive.IsChecked == true) estimate = 5;

Review newReview = new Review(StringFromRichTextBox(rtbReview), estimate, gameId, userId, DateTime.Now);

reviews.Add(newReview);

db.SaveChanges();

MessageBox.Show("Отзыв добавлен", "Внимание");

break;

default:

break;

}

}

App.UpdateRating();

GameInfo gameInfo = new GameInfo();

gameInfo.Show();

}

}

else

{

MessageBox.Show("Заполните все поля", "Внимание");

}

}

private string StringFromRichTextBox(RichTextBox rtb)

{

TextRange textRange = new TextRange(

rtb.Document.ContentStart,

rtb.Document.ContentEnd

);

return textRange.Text;

}

private void bDelete\_Click(object sender, RoutedEventArgs e)

{

var result = MessageBox.Show("Вы действительно хотите удалить отзыв?", "Внимание", MessageBoxButton.YesNo);

switch (result)

{

case MessageBoxResult.Yes:

using (GamereviewsContext db = new GamereviewsContext())

{

var reviews = db.Reviews;

reviews.Remove(db.Reviews.First(i => i.ReviewId == App.ReviewId));

db.SaveChanges();

App.UpdateRating();

MessageBox.Show("Отзыв удален", "Внимание");

GameInfo gameInfo = new GameInfo();

gameInfo.Show();

}

break;

default:

break;

}

}

}

}

Форма Auth.xaml.cs (Авторизация)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace Reviews.Wndws

{

/// <summary>

/// Логика взаимодействия для Auth.xaml

/// </summary>

public partial class Auth : Window

{

public Auth()

{

InitializeComponent();

if (App.ActiveWindow != null) App.ActiveWindow.Close();

App.ActiveWindow = this;

}

private void bVisitor\_Click(object sender, RoutedEventArgs e)

{

App.UserId = 0;

GamesList gamesList = new GamesList();

gamesList.Show();

}

private void bReg\_Click(object sender, RoutedEventArgs e)

{

Reg reg = new Reg();

reg.Show();

}

private void bAuth\_Click(object sender, RoutedEventArgs e)

{

if (tbLogin.Text != "" && pbPassword.Password != "")

{

using (GamereviewsContext db = new GamereviewsContext())

{

var users = db.Users;

var user = users.FirstOrDefault(i => i.UserLogin == tbLogin.Text && i.UserPassword == pbPassword.Password);

if (user != null)

{

App.UserId = user.UserId;

GamesList gamesList = new GamesList();

gamesList.Show();

}

else

{

MessageBox.Show("Неверный логин или пароль", "Внимание");

}

}

}

else

{

MessageBox.Show("Заполните все поля", "Внимание");

}

}

}

}

Форма GameInfo.xaml.cs (Основная информация игры)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using Microsoft.EntityFrameworkCore;

using Reviews.UserControls;

namespace Reviews.Wndws

{

/// <summary>

/// Логика взаимодействия для GameInfo.xaml

/// </summary>

public partial class GameInfo : Window

{

public GameInfo()

{

InitializeComponent();

if (App.ActiveWindow != null) App.ActiveWindow.Close();

App.ActiveWindow = this;

if (App.UserId == 0)

{

bAddReview.Visibility = Visibility.Hidden;

}

using (GamereviewsContext db = new GamereviewsContext())

{

var games = db.Games.ToList();

var reviews = db.Reviews.ToList();

var game = games.First(i => i.GameId == App.GameId);

lName.Content = game.GameName;

iImage.Source = new BitmapImage(new Uri($"C://Users/ADMIN/OneDrive/Рабочий стол/Работы в VS/Reviews/Reviews/Resources/{game.GamePicture}"));

lDate.Content = lDate.Content + game.GameDateRelease.ToString();

lRating.Content = lRating.Content + game.GameRating.ToString();

lCount.Content = lCount.Content + game.GameCommentsCount.ToString();

games = db.Games.Include(i => i.Genres).ToList();

game = games.First(i => i.GameId == App.GameId);

rtbDescription.AppendText(game.GameDescription + " Жанры: ");

foreach (var genre in game.Genres)

{

rtbDescription.AppendText(genre.GenreName + "; ");

}

games = db.Games.Include(i => i.GameDeveloper).ToList();

game = games.First(i => i.GameId == App.GameId);

lDeveloper.Content = lDeveloper.Content + game.GameDeveloper.DeveloperName;

games = db.Games.Include(i => i.GamePublisher).ToList();

game = games.First(i => i.GameId == App.GameId);

lPublisher.Content = lPublisher.Content + game.GamePublisher.PublisherName;

var review = reviews.FirstOrDefault(i => i.ReviewUserId == App.UserId && i.ReviewGameId == App.GameId);

if (review != null)

{

bAddReview.Content = "Изменить свой отзыв";

}

reviews = db.Reviews.Include(i => i.ReviewUser).ToList();

foreach (var rReview in reviews)

{

if (rReview.ReviewGameId == App.GameId)

{

ReviewCard reviewCard = new ReviewCard(rReview);

lvReviews.Items.Add(reviewCard);

}

}

}

}

private void bBack\_Click(object sender, RoutedEventArgs e)

{

if (App.BackWindow == "Просмотр игр")

{

GamesList gamesList = new GamesList();

gamesList.Show();

}

else

{

UserReviews userReviews = new UserReviews();

userReviews.Show();

}

}

private void bAddReview\_Click(object sender, RoutedEventArgs e)

{

AddReview addReview = new AddReview();

addReview.Show();

}

}

}

Форма GamesList.xaml.cs (Просмотр игр)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using Microsoft.EntityFrameworkCore;

using Reviews.UserControls;

namespace Reviews.Wndws

{

/// <summary>

/// Логика взаимодействия для GamesList.xaml

/// </summary>

public partial class GamesList : Window

{

public GamesList()

{

InitializeComponent();

if (App.ActiveWindow != null) App.ActiveWindow.Close();

App.ActiveWindow = this;

if (App.UserId == 0)

{

bProfile.Visibility = Visibility.Hidden;

}

App.UpdateRating();

using (GamereviewsContext db = new GamereviewsContext())

{

var games = db.Games.Include(i => i.Genres).ToList();

var genres = db.Genres.ToList();

foreach (var genre in genres)

{

cbGenre.Items.Add(genre.GenreName);

}

foreach (var game in games)

{

GameCard gameCard = new GameCard(game);

lvGames.Items.Add(gameCard);

}

}

}

private void bExit\_Click(object sender, RoutedEventArgs e)

{

var result = MessageBox.Show("Вы действительно хотите выйти?", "Внимание", MessageBoxButton.YesNo);

switch (result)

{

case MessageBoxResult.Yes:

App.UserId = null;

App.BackWindow = null;

App.GameId = null;

App.ReviewId = null;

Auth auth = new Auth();

auth.Show();

break;

default:

break;

}

}

private void bProfile\_Click(object sender, RoutedEventArgs e)

{

Profile profile = new Profile();

profile.Show();

}

private void tbSearch\_TextChanged(object sender, TextChangedEventArgs e)

{

Refresh(tbSearch.Text, cbGenre.SelectedIndex, cbFilter.SelectedIndex);

}

private void Refresh(string? search, int? idGenre, int? idFilter)

{

using (GamereviewsContext db = new GamereviewsContext())

{

var games = db.Games.Include(i => i.Genres).ToList();

var genres = db.Genres.ToList();

if (search != null) games = games.Where(i => i.GameName.Contains(search)).ToList();

if (idGenre != 0)

{

var genre = genres.FirstOrDefault(i => i.GenreId == idGenre);

if (genre != null) games = games.Where(i => i.Genres.Contains(genre)).ToList();

}

if (idFilter != 0)

{

switch(idFilter)

{

case 1:

games = games.OrderBy(i=> i.GameCommentsCount).ToList();

break;

case 2:

games = games.OrderByDescending(i => i.GameCommentsCount).ToList();

break;

case 3:

games = games.OrderBy(i => i.GameRating).ToList();

break;

case 4:

games = games.OrderByDescending(i => i.GameRating).ToList();

break;

}

}

lvGames.Items.Clear();

foreach (var game in games)

{

GameCard gameCard = new GameCard(game);

lvGames.Items.Add(gameCard);

}

}

}

private void cbGenre\_DropDownClosed(object sender, EventArgs e)

{

Refresh(tbSearch.Text, cbGenre.SelectedIndex, cbFilter.SelectedIndex);

}

private void cbFilter\_DropDownClosed(object sender, EventArgs e)

{

Refresh(tbSearch.Text, cbGenre.SelectedIndex, cbFilter.SelectedIndex);

}

}

}

Форма Profile.xaml.cs (Профиль)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace Reviews.Wndws

{

/// <summary>

/// Логика взаимодействия для Profile.xaml

/// </summary>

public partial class Profile : Window

{

public Profile()

{

InitializeComponent();

if (App.ActiveWindow != null) App.ActiveWindow.Close();

App.ActiveWindow = this;

using (GamereviewsContext db = new GamereviewsContext())

{

var users = db.Users.ToList();

var user = users.FirstOrDefault(i => i.UserId == App.UserId);

if (user != null)

{

tbLogin.Text = user.UserLogin;

tbNickname.Text = user.UserNickname;

}

}

}

private void bBack\_Click(object sender, RoutedEventArgs e)

{

GamesList gamesList = new GamesList();

gamesList.Show();

}

private void bReviews\_Click(object sender, RoutedEventArgs e)

{

UserReviews userReviews = new UserReviews();

userReviews.Show();

}

private void bSave\_Click(object sender, RoutedEventArgs e)

{

if (tbLogin.Text != "" && tbNickname.Text != "" && pbPassword.Password != "")

{

using (GamereviewsContext db = new GamereviewsContext())

{

var users = db.Users;

var user = users.First(i => i.UserId == App.UserId);

if (pbPassword.Password == user.UserPassword)

{

var userSearch = users.FirstOrDefault(i => (i.UserLogin == tbLogin.Text || i.UserNickname == tbNickname.Text) && i.UserId != App.UserId);

if (userSearch == null)

{

var result = MessageBox.Show("Вы действительно хотите изменить данные аккаунта?", "Внимание", MessageBoxButton.YesNo);

switch (result)

{

case MessageBoxResult.Yes:

user.UserLogin = tbLogin.Text;

user.UserNickname = tbNickname.Text;

if (pbRpassword.Password != "")

{

user.UserPassword = pbRpassword.Password;

}

db.SaveChanges();

MessageBox.Show("Изменения сохранены", "Внимание");

Profile profile = new Profile();

profile.Show();

break;

default:

break;

}

}

else

{

MessageBox.Show("Пользователь с таким логином или именем уже существует", "Внимание");

}

}

else

{

MessageBox.Show("Неверный пароль", "Внимание");

}

}

}

else

{

MessageBox.Show("Заполните все поля", "Внимание");

}

}

private void bDelete\_Click(object sender, RoutedEventArgs e)

{

if (pbPassword.Password != "")

{

using (GamereviewsContext db = new GamereviewsContext())

{

var users = db.Users;

var user = users.First(i => i.UserId == App.UserId);

if (user.UserPassword == pbPassword.Password)

{

var result = MessageBox.Show("Вы действительно хотите удалить аккаунт?", "Внимание", MessageBoxButton.YesNo);

switch (result)

{

case MessageBoxResult.Yes:

var reviews = db.Reviews;

foreach (var review in reviews)

{

if (review.ReviewUserId == App.UserId)

{

reviews.Remove(review);

}

}

db.SaveChanges();

App.UpdateRating();

users.Remove(db.Users.First(i => i.UserId == App.UserId));

db.SaveChanges();

MessageBox.Show("Аккаунт удален", "Внимание");

App.UserId = null;

App.BackWindow = null;

App.GameId = null;

App.ReviewId = null;

Auth auth = new Auth();

auth.Show();

break;

default:

break;

}

}

else

{

MessageBox.Show("Неверный пароль", "Внимание");

}

}

}

else

{

MessageBox.Show("Заполните поле пароля", "Внимание");

}

}

}

}

Форма Reg.xaml.cs (Регистрация)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace Reviews.Wndws

{

/// <summary>

/// Логика взаимодействия для Reg.xaml

/// </summary>

public partial class Reg : Window

{

public Reg()

{

InitializeComponent();

if (App.ActiveWindow != null) App.ActiveWindow.Close();

App.ActiveWindow = this;

}

private void bVisitor\_Click(object sender, RoutedEventArgs e)

{

App.UserId = 0;

GamesList gamesList = new GamesList();

gamesList.Show();

}

private void bAuth\_Click(object sender, RoutedEventArgs e)

{

Auth auth = new Auth();

auth.Show();

}

private void bReg\_Click(object sender, RoutedEventArgs e)

{

if (tbLogin.Text != "" && tbNickname.Text != "" && pbPassword.Password != "" && pbRpassword.Password != "")

{

using (GamereviewsContext db = new GamereviewsContext())

{

var users = db.Users;

var user = users.FirstOrDefault(i => i.UserLogin == tbLogin.Text || i.UserNickname == tbNickname.Text);

if (user != null)

{

MessageBox.Show("Пользователь с таким логином или именем уже существует", "Внимание");

}

else

{

if (pbPassword.Password == pbRpassword.Password)

{

User newUser = new User(tbLogin.Text, pbPassword.Password, tbNickname.Text);

users.Add(newUser);

db.SaveChanges();

users = db.Users;

user = users.First(i => i.UserLogin == tbLogin.Text || i.UserNickname == tbNickname.Text);

App.UserId = user.UserId;

GamesList gamesList = new GamesList();

gamesList.Show();

}

else

{

MessageBox.Show("Пароли должны совпадать", "Внимание");

}

}

}

}

else

{

MessageBox.Show("Заполните все поля", "Внимание");

}

}

}

}

Форма UserReviews.xaml.cs (Отзывы пользователя)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using Microsoft.EntityFrameworkCore;

using Reviews.UserControls;

namespace Reviews.Wndws

{

/// <summary>

/// Логика взаимодействия для UserReviews.xaml

/// </summary>

public partial class UserReviews : Window

{

public UserReviews()

{

InitializeComponent();

if (App.ActiveWindow != null) App.ActiveWindow.Close();

App.ActiveWindow = this;

using (GamereviewsContext db = new GamereviewsContext())

{

var reviews = db.Reviews.Include(i => i.ReviewUser).ToList();

foreach (var review in reviews)

{

if (review.ReviewUserId == App.UserId)

{

ReviewCard reviewCard = new ReviewCard(review);

lvReviews.Items.Add(reviewCard);

}

}

}

}

private void bBack\_Click(object sender, RoutedEventArgs e)

{

Profile profile = new Profile();

profile.Show();

}

}

}

Форма GameCard.xaml.cs (Карточка игры)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using Microsoft.EntityFrameworkCore;

using Reviews.Wndws;

namespace Reviews.UserControls

{

/// <summary>

/// Логика взаимодействия для GameCard.xaml

/// </summary>

public partial class GameCard : UserControl

{

private Game game1;

public GameCard(Game game)

{

InitializeComponent();

iImage.Source = new BitmapImage(new Uri($"C://Users/ADMIN/OneDrive/Рабочий стол/Работы в VS/Reviews/Reviews/Resources/{game.GamePicture}"));

lName.Content = game.GameName;

lRating.Content = lRating.Content + game.GameRating.ToString();

lCount.Content = lCount.Content + game.GameCommentsCount.ToString();

game1 = game;

}

private void UserControl\_MouseDoubleClick(object sender, MouseButtonEventArgs e)

{

App.BackWindow = "Просмотр игр";

App.GameId = game1.GameId;

GameInfo gameInfo = new GameInfo();

gameInfo.Show();

}

}

}

Форма ReviewCard.xaml.cs (Карточка отзыва)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using Reviews.Wndws;

namespace Reviews.UserControls

{

/// <summary>

/// Логика взаимодействия для ReviewCard.xaml

/// </summary>

public partial class ReviewCard : UserControl

{

private Review review1;

public ReviewCard(Review review)

{

InitializeComponent();

lNickname.Content = lNickname.Content + review.ReviewUser.UserNickname;

lDate.Content = lDate.Content + review.ReviewDate.ToString();

lEstimate.Content = lEstimate.Content + review.ReviewEstimate.ToString();

rtbReview.AppendText(review.ReviewText);

review1 = review;

}

private void UserControl\_MouseDoubleClick(object sender, MouseButtonEventArgs e)

{

if (App.ActiveWindow != null && App.ActiveWindow.Title == "Мои отзывы")

{

App.BackWindow = "Мои отзывы";

App.GameId = review1.ReviewGameId;

GameInfo gameInfo = new GameInfo();

gameInfo.Show();

}

}

}

}

Класс GamereviewsContext.cs (Работа с базой данных)

using System;

using System.Collections.Generic;

using Microsoft.EntityFrameworkCore;

namespace Reviews;

public partial class GamereviewsContext : DbContext

{

public GamereviewsContext()

{

}

public GamereviewsContext(DbContextOptions<GamereviewsContext> options)

: base(options)

{

}

public virtual DbSet<Developer> Developers { get; set; }

public virtual DbSet<Game> Games { get; set; }

public virtual DbSet<Genre> Genres { get; set; }

public virtual DbSet<Publisher> Publishers { get; set; }

public virtual DbSet<Review> Reviews { get; set; }

public virtual DbSet<User> Users { get; set; }

protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)

//#warning To protect potentially sensitive information in your connection string, you should move it out of source code. You can avoid scaffolding the connection string by using the Name= syntax to read it from configuration - see https://go.microsoft.com/fwlink/?linkid=2131148. For more guidance on storing connection strings, see http://go.microsoft.com/fwlink/?LinkId=723263.

=> optionsBuilder.UseMySql("server=localhost;user=root;password=1234;database=gamereviews", Microsoft.EntityFrameworkCore.ServerVersion.Parse("8.0.37-mysql"));

protected override void OnModelCreating(ModelBuilder modelBuilder)

{

modelBuilder

.UseCollation("utf8mb4\_0900\_ai\_ci")

.HasCharSet("utf8mb4");

modelBuilder.Entity<Developer>(entity =>

{

entity.HasKey(e => e.DeveloperId).HasName("PRIMARY");

entity.ToTable("developers");

entity.Property(e => e.DeveloperName).HasMaxLength(50);

});

modelBuilder.Entity<Game>(entity =>

{

entity.HasKey(e => e.GameId).HasName("PRIMARY");

entity.ToTable("games");

entity.HasIndex(e => e.GameDeveloperId, "GameDeveloperId");

entity.HasIndex(e => e.GamePublisherId, "GamePublisherId");

entity.Property(e => e.GameDescription).HasColumnType("text");

entity.Property(e => e.GameName).HasMaxLength(50);

entity.Property(e => e.GamePicture).HasMaxLength(50);

entity.HasOne(d => d.GameDeveloper).WithMany(p => p.Games)

.HasForeignKey(d => d.GameDeveloperId)

.OnDelete(DeleteBehavior.ClientSetNull)

.HasConstraintName("games\_ibfk\_1");

entity.HasOne(d => d.GamePublisher).WithMany(p => p.Games)

.HasForeignKey(d => d.GamePublisherId)

.OnDelete(DeleteBehavior.ClientSetNull)

.HasConstraintName("games\_ibfk\_2");

entity.HasMany(d => d.Genres).WithMany(p => p.Games)

.UsingEntity<Dictionary<string, object>>(

"Gamegenre",

r => r.HasOne<Genre>().WithMany()

.HasForeignKey("GenreId")

.OnDelete(DeleteBehavior.ClientSetNull)

.HasConstraintName("gamegenres\_ibfk\_2"),

l => l.HasOne<Game>().WithMany()

.HasForeignKey("GameId")

.OnDelete(DeleteBehavior.ClientSetNull)

.HasConstraintName("gamegenres\_ibfk\_1"),

j =>

{

j.HasKey("GameId", "GenreId")

.HasName("PRIMARY")

.HasAnnotation("MySql:IndexPrefixLength", new[] { 0, 0 });

j.ToTable("gamegenres");

j.HasIndex(new[] { "GenreId" }, "GenreId");

});

});

modelBuilder.Entity<Genre>(entity =>

{

entity.HasKey(e => e.GenreId).HasName("PRIMARY");

entity.ToTable("genres");

entity.Property(e => e.GenreName).HasMaxLength(50);

});

modelBuilder.Entity<Publisher>(entity =>

{

entity.HasKey(e => e.PublisherId).HasName("PRIMARY");

entity.ToTable("publishers");

entity.Property(e => e.PublisherName).HasMaxLength(50);

});

modelBuilder.Entity<Review>(entity =>

{

entity.HasKey(e => e.ReviewId).HasName("PRIMARY");

entity.ToTable("reviews");

entity.HasIndex(e => e.ReviewGameId, "ReviewGameId");

entity.HasIndex(e => e.ReviewUserId, "ReviewUserId");

entity.Property(e => e.ReviewDate).HasColumnType("datetime");

entity.Property(e => e.ReviewText).HasColumnType("text");

entity.HasOne(d => d.ReviewGame).WithMany(p => p.Reviews)

.HasForeignKey(d => d.ReviewGameId)

.OnDelete(DeleteBehavior.ClientSetNull)

.HasConstraintName("reviews\_ibfk\_1");

entity.HasOne(d => d.ReviewUser).WithMany(p => p.Reviews)

.HasForeignKey(d => d.ReviewUserId)

.OnDelete(DeleteBehavior.ClientSetNull)

.HasConstraintName("reviews\_ibfk\_2");

});

modelBuilder.Entity<User>(entity =>

{

entity.HasKey(e => e.UserId).HasName("PRIMARY");

entity.ToTable("users");

entity.Property(e => e.UserLogin).HasMaxLength(50);

entity.Property(e => e.UserNickname).HasMaxLength(50);

entity.Property(e => e.UserPassword).HasMaxLength(50);

});

OnModelCreatingPartial(modelBuilder);

}

partial void OnModelCreatingPartial(ModelBuilder modelBuilder);

}

Класс Developer.cs (Разработчик)

using System;

using System.Collections.Generic;

namespace Reviews;

public partial class Developer

{

public int DeveloperId { get; set; }

public string DeveloperName { get; set; } = null!;

public virtual ICollection<Game> Games { get; set; } = new List<Game>();

}

Класс Game.cs (Игра)

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations.Schema;

namespace Reviews;

public partial class Game

{

public int GameId { get; set; }

public string GameName { get; set; } = null!;

public string GameDescription { get; set; } = null!;

public int GameDeveloperId { get; set; }

public int GamePublisherId { get; set; }

public DateOnly GameDateRelease { get; set; }

public string GamePicture { get; set; } = null!;

public double? GameRating { get; set; }

public int? GameCommentsCount { get; set; }

public virtual Developer GameDeveloper { get; set; } = null!;

public virtual Publisher GamePublisher { get; set; } = null!;

public virtual ICollection<Review> Reviews { get; set; } = new List<Review>();

public virtual ICollection<Genre> Genres { get; set; } = new List<Genre>();

}

Класс Genre.cs (Жанр)

using System;

using System.Collections.Generic;

namespace Reviews;

public partial class Genre

{

public int GenreId { get; set; }

public string GenreName { get; set; } = null!;

public virtual ICollection<Game> Games { get; set; } = new List<Game>();

}

Класс Publisher.cs (Издатель)

using System;

using System.Collections.Generic;

namespace Reviews;

public partial class Publisher

{

public int PublisherId { get; set; }

public string PublisherName { get; set; } = null!;

public virtual ICollection<Game> Games { get; set; } = new List<Game>();

}

Класс Review.cs (Отзыв)

using System;

using System.Collections.Generic;

namespace Reviews;

public partial class Review

{

public int ReviewId { get; set; }

public string ReviewText { get; set; } = null!;

public int ReviewEstimate { get; set; }

public int ReviewGameId { get; set; }

public int ReviewUserId { get; set; }

public DateTime ReviewDate { get; set; }

public virtual Game ReviewGame { get; set; } = null!;

public virtual User ReviewUser { get; set; } = null!;

public Review(string reviewText, int reviewEstimate, int reviewGameId, int reviewUserId, DateTime reviewDate)

{

ReviewText = reviewText;

ReviewEstimate = reviewEstimate;

ReviewGameId = reviewGameId;

ReviewUserId = reviewUserId;

ReviewDate = reviewDate;

}

}

Класс User.cs (Пользователь)

using System;

using System.Collections.Generic;

namespace Reviews;

public partial class User

{

public int UserId { get; set; }

public string UserLogin { get; set; } = null!;

public string UserPassword { get; set; } = null!;

public string UserNickname { get; set; } = null!;

public virtual ICollection<Review> Reviews { get; set; } = new List<Review>();

public User (string userLogin, string userPassword, string userNickname)

{

UserLogin = userLogin;

UserPassword = userPassword;

UserNickname = userNickname;

}

}

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. [ГОСТ 28195-89](https://gost.ruscable.ru/Index/11/11212.htm" \o "https://gost.ruscable.ru/Index/11/11212.htm) Оценка качества программных средств. Общие положения: межгосударственный стандарт: дата введения 1990-07-01 - <https://docs.cntd.ru/document/1200009135?ysclid=lroox3amxp882669222> – Текст: электронный.
2. [ГОСТ 28806-90](https://gost.ruscable.ru/Index/10/10605.htm) Качество программных средств. Термины и определения: межгосударственный стандарт: дата введения 1992-01-01 - <https://docs.cntd.ru/document/1200009077?ysclid=lrq6to4u1z728775522> – Текст: электронный.
3. [ГОСТ 34.321-96](https://gost.ruscable.ru/Index/6/6808.htm) Информационные технологии. Система стандартов по базам данных. Эталонная модель управления данными: межгосударственный стандарт: дата введения 2001-07-01 - <https://docs.cntd.ru/document/1200017662?ysclid=lrq6wn65z5753511468> – Текст: электронный.
4. [ГОСТ Р ИСО/МЭК 12207-99. Информационная технология. Процессы жизненного цикла программных средств](https://docs.cntd.ru/document/1200009075#7D20K3): межгосударственный стандарт: дата введения 2000-07-01 -https://docs.cntd.ru/document/1200009075?ysclid=lrool2t1vo124212660. – Текст: электронный.
5. Metanit [Электронный ресурс] / Руководство по языку C#. – Режим доступа <https://metanit.com/sharp/tutorial/> , свободный.
6. Metanit [Электронный ресурс] / Руководство по WPF. – Режим доступа <https://metanit.com/sharp/wpf/> , свободный.
7. Microsoft [Электронный ресурс] / Руководство по языку C#. – Режим доступа <https://docs.microsoft.com/ru-ru/dotnet/csharp/> , свободный.

1. PK-первичный ключ

   FK-внешний ключ [↑](#footnote-ref-1)