МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных Технологий

Кафедра Программной инженерии

Специальность 1-40 01 01 «Программное обеспечение информационных

технологий»

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА  
КУРСОВОГО ПРОЕКТА:**

по дисциплине             «Объектно-ориентированные технологии программирования и стандарты проектирования»

Тема «Система Управления Складом»

Исполнитель

студент 2 курса группы 4                                                  Кравченко Н.В.

(Ф.И.О.)

Руководитель работы                                                 ассистент         Северинчик Н. А.

(учен. степень, звание, должность, подпись, Ф.И.О.)  
  
Курсовой проект защищен с оценкой

Председатель Пацей Н.В. (подпись)

Минск 2021

Содержание

[Введение 3](#_Toc42775075)

[1 Постановка задачи 4](#_Toc42775076)

[1.1 Средства разработки 4](#_Toc42775077)

[1.2 Обзор прототипов и аналогов 5](#_Toc42775078)

[2 Разработка архитектуры проекта 7](#_Toc42775079)

[2.1 Обобщенная структура 7](#_Toc42775080)

[2.2 Диаграмма UML 8](#_Toc42775081)

[2.3 Диаграмма классов 8](#_Toc42775082)

[2.4 Проектирование базы данных 11](#_Toc42775083)

[3 Разработка функциональной модели 14](#_Toc42775084)

[3.1 Описание функций приложения 14](#_Toc42775085)

[3.2 Классы в папках Classes 14](#_Toc42775086)

[3.3 Основные окна приложения 15](#_Toc42775087)

[3.3.1 Главное окно агрегатора 15](#_Toc42775088)

[3.3.2 Главное окно игры 15](#_Toc42775089)

[3.4 Основные страницы приложения 15](#_Toc42775090)

[3.4.1 Страница авторизации 16](#_Toc42775091)

[3.4.2 Страница регистрации 16](#_Toc42775092)

[3.4.3 Страница профиля 17](#_Toc42775093)

[3.4.4 Страница новостей 17](#_Toc42775094)

[3.4.5 Страница магазина 18](#_Toc42775095)

[3.4.6 Страница покупки золота 18](#_Toc42775096)

[3.5 Основные игровые элементы управления 20](#_Toc42775097)

[3.5.1 Статистика 20](#_Toc42775098)

[3.5.2 Связь с автором 20](#_Toc42775099)

[4 Руководство пользователя 21](#_Toc42775100)

[5 Тестирование программного средства 23](#_Toc42775101)

[Заключение 25](#_Toc42775102)

[Список используемых источников 26](#_Toc42775103)

[Приложение A. UML-диаграмма проекта «WpfApp» 27](#_Toc42775104)

[Приложение B. UML-диаграмма проекта «WpfGame\_00» 28](#_Toc42775105)

Введение

С каждым годом различные отрасли промышленности растут в достаточно быстрой прогрессии. В целом система управления товарами используется в различных отраслях промышленности, от производства до коммунальных услуг, здравоохранения, образования, государственного управления и т.д.

Моё приложение инвентаризации будет процессом организации и хранения достаточного количества продуктов на складе для удовлетворения спроса, а также контроля потока и поддержания запасов, чтобы гарантировать, что нужное количество инвентаря доступно в нужное время и нужного качества.

Это приложение инвентаризации позволяет пользователю управлять подсчетом товаров и их движением, заказами на их покупку и продажу, импортом и экспортом, управлять поставщиками и клиентами в любом бизнесе.

Это поможет пользователю сделать многие из этих процессов автоматическими без необходимости запоминать детали, связанные с заказом, обработкой, комплектованием и упаковкой.

Целью курсового проекта является: развитие практических навыков вёрстки приложений на базе WPF с использованием языка разметки — XAML, и языка программирования — С#, показать навыки разработки дизайна, макета для приложения и умение грамотно разделить информацию для пользователя с использованием интересных цветовых сочетаний.

# Постановка задачи

Основной задачей курсового проекта является разработка приложения с возможностью управлять подсчетом товаров и их движением, заказами на их покупку и продажу, импортом и экспортом, управлять поставщиками и клиентами, что поможет пользователям сделать многие из этих процессов автоматическими без необходимости запоминать детали, связанные с заказом.

Функционально данное программное средство должно выполнять следующие задачи:

* выполнять регистрацию и авторизацию;
* поддерживать работу с базой данных;
* управлять клиентами, товарами, заказами, пользователями
* отслеживать количество товаров на складе, их названия, описание, фотографий, категорий, стоимости;
* добавлять новые товары, клиентов, пользователей
* обновлять товары и их данные
* осуществлять удаление товаров
* осуществлять поиск товаров

## Средства разработки

При разработке приложения были использованы:

* среда разработки Microsoft Visual Studio 2019;
* технология WPF;
* объектно-ориентированная технология Entity Framework на базе фреймворка .NET
* система управления Microsoft SQL Server.

Microsoft Visual Studio — это стартовая площадка для написания, отладки и сборки кода, а также последующей публикации приложений. Интегрированная среда разработки (IDE) представляет собой многофункциональную программу, которую можно использовать для различных аспектов разработки программного обеспечения. Помимо стандартного редактора и отладчика, которые существуют в большинстве сред IDE, Visual Studio включает в себя компиляторы, средства автозавершения кода, графические конструкторы и многие другие функции для упрощения процесса разработки.

Windows Presentation Foundation (WPF) — это система следующего поколения для построения клиентских приложений Windows с визуально привлекательными возможностями взаимодействия с пользователем. C WPF можно создавать широкий спектр как автономных приложений, так и приложений, размещенных в веб-обозревателе. в основе WPF лежит векторная система визуализации, не зависящая от разрешения и созданная с расчетом на возможности современного графического оборудования. WPF расширяет базовую систему полным набором функций разработки приложений, элементами управления, привязкой данных, макетом, графикой, анимацией, стилями, шаблонами, документами, мультимедиа, текстом и оформлением. Именно использование WPF позволило гибко управлять дизайном интерфейса, также стало возможным подключение различных сторонних пакетов и использование паттернов.

Entity Framework представляет специальную объектно-ориентированную технологию на базе фреймворка .NET для работы с данными. Если традиционные средства ADO.NET позволяют создавать подключения, команды и прочие объекты для взаимодействия с базами данных, то Entity Framework представляет собой более высокий уровень абстракции, который позволяет абстрагироваться от самой базы данных и работать с данными независимо от типа хранилища. Если на физическом уровне мы оперируем таблицами, индексами, первичными и внешними ключами, но на концептуальном уровне, который нам предлагает Entity Framework, мы уже работает с объектами.

Microsoft SQL Server — система управления реляционными БД. SQL Server Management Studio – утилита из Microsoft SQL Server, необходимая для управления и администрирования всех компонентов Microsoft SQL Server. Она в себя включает скриптовый редактор и графическую программу, которая работает с объектами и настройками сервера. SQL является, прежде всего, информационно-логическим языком, предназначенным для описания, изменения и извлечения данных, хранимых в реляционных базах данных.

## Обзор прототипов и аналогов

В интернете есть десятки программ, которые позволяют предпринимателям вести учет в рознице, но большинство из них являются «сырыми» или мало функциональными.

Покупая случайную учетную программу для склада, есть риск, что она:

* неудобна для освоения;
* не поддерживает работу с имеющимся на складе оборудованием;
* не содержит критичного для работы функционала;
* не имеет круглосуточной службы поддержки
* раскрывает достаточные возможности только на максимальном тарифе.

Ниже перечислены критерии выбора программ учёта, которые нужно учесть предпринимателю:

1. Перечень поддерживаемых операций. Кому-то нужно знать просто приход/расход, а для кого-то важен дополнительно ценовой учет и аналитика продаж.

2. Стоимость внедрения и сопровождения. Нет смысла подробно вникать в обзор программы складского учета, если предприниматель не готов платить минимальный ежемесячный платеж.

3. Круглосуточная техническая поддержка.

4. Наличие дополнительных опциональных модулей (CMS, бухгалтерских, логистических). При планировании дисконтной системы CMS просто необходима.

5. Сетевые возможности. Например, для распределенных в пространстве складов будет актуальна только облачная программа складского учета.

6. Простота освоения. Новый сотрудник должен за несколько часов освоить основные возможности программы.

7. Стабильность работы. Программа не должна подвисать и перезагружаться, потому что это может приводить к потере последних введенных данных.

8. Наличие полнофункциональной демо-версии. Гораздо проще выбрать программу, загрузив ее полнофункциональную версию и испытав возможности

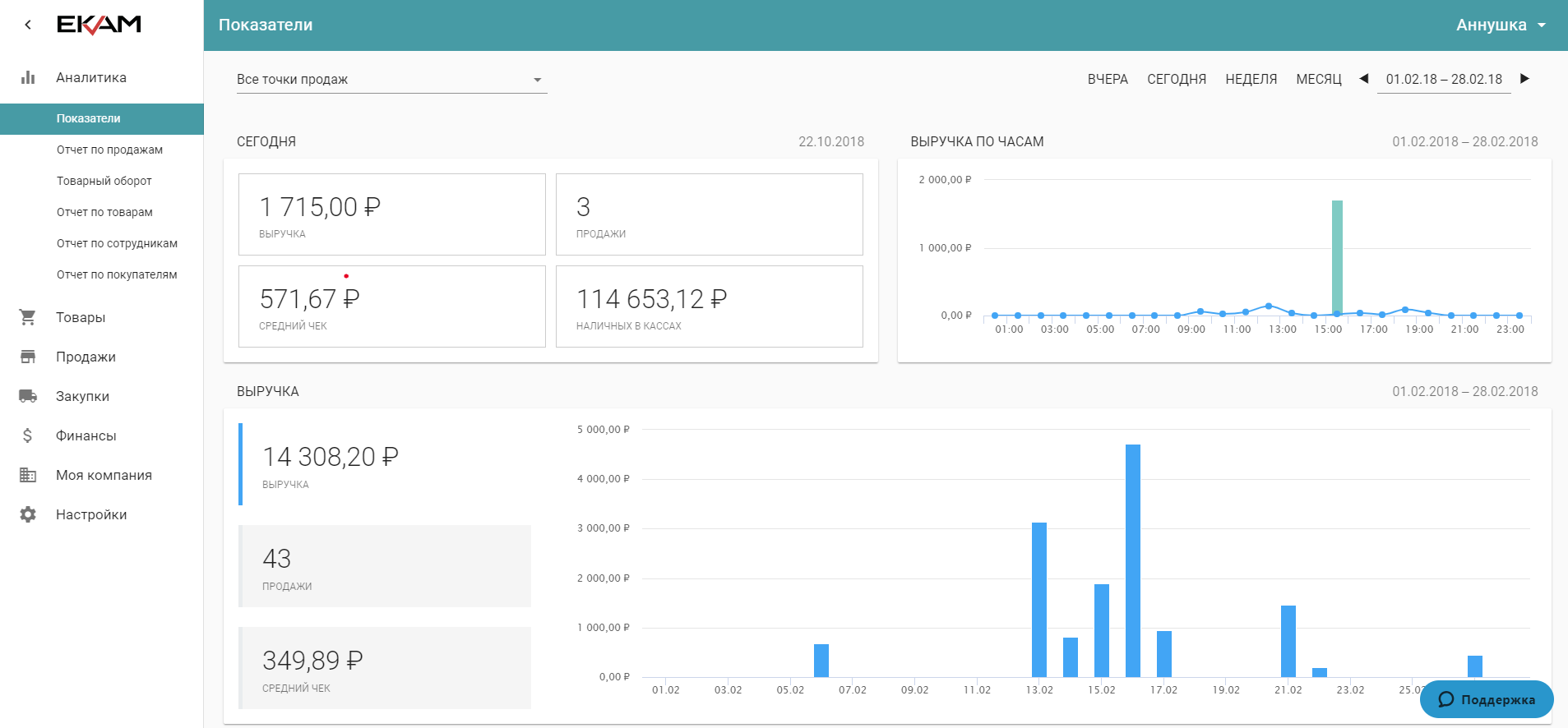
9. Открытый API, позволяющий дорабатывать программу под индивидуальные потребности клиента.

10.Удобный интерфейс. Переключение между меню во время работы должно занимать у персонала минимум времени.

Далее я приведу примеры программ для розничной торговли в разрезе соответствия их перечисленным критериям.

**«ЕКАМ»**

Товароучетная система «ЕКАМ» является одним из лидеров на СНГ рынке в своей нише и занимает высокие позиции в рейтингах.



Интерфейс программы складского учета «ЕКАМ» (рис. 1.1)

Программа работает через «облако» и предназначается для комплексной автоматизации малого и среднего бизнеса в сфере торговли и оказания услуг. [«ЕКАМ»](https://www.ekam.ru/page/sklad-programma) позволяет автоматизировать следующие операции:

* складской учет;
* продажи;
* закупки;
* управленческая отчетность;
* создание программ лояльности и дисконтных программ;
* прием и выполнение заказов в кафе и ресторанах;
* движение денег.

Также программа складского учета «ЕКАМ» интегрируется с бухгалтерскими приложениями, упрощая составление налоговой отчетности.

**Плюсы программы складского учета «ЕКАМ»:**

1. Оперативный и точный контроль за складскими остатками.
2. Автоматическая установка продажной цены, исходя из заданной наценки.
3. Удобное и быстрое проведение инвентаризации.
4. Интеллектуальная система закупок.
5. Удаленная работа с программой.
6. Большое разнообразие настраиваемых аналитических, финансовых, товарных и управленческих отчетов.
7. Возможность одновременного учета товаров в нескольких структурных подразделениях.
8. Загрузка прайсов поставщиков в программу, облегчающее создание номенклатуры.
9. Мультиплатформенность: программой можно пользоваться на компьютерах через браузер и на мобильных гаджетах через специальное приложение.
10. Приемлемые для малого бизнеса тарифные планы.
11. Подключение интернет-магазина.
12. Дружелюбный, понятный интерфейс, позволяющий самостоятельно проводить настройку программы и быстро обучать работе с программой новых сотрудников..
13. Наличие практичного модуля клиентской базы с возможностью привязки дисконтных карт.
14. Наличие полнофункциональной демо-версии, позволяющей 14 дней пользоваться всеми возможностями программы.
15. Наличие телефона круглосуточной технической поддержки.
16. Автоматическая передача данных о продажах в 1С.
17. Составление технологических карт для общепита.
18. Высокая стабильность работы
19. Открытый API для индивидуальной настройки под требования клиента.

**Минусы «ЕКАМ»:**

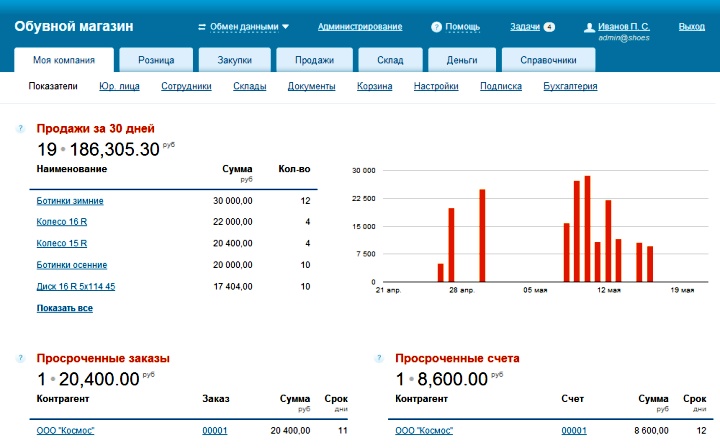
1. Отсутствует функционал по управлению доставкой.
2. Нет блока календаря и задач.

Одним из достоинств «ЕКАМ» является оперативная и профессиональная техническая поддержка. Её специалисты решают проблему как подсказками, так и при помощи удаленного подключения к компьютеру пользователя.

Настройка «ЕКАМ» производится сотрудниками компании. После этого предприниматели могут пользоваться всеми преимуществами приложения.

**«МойСклад»**

Рейтинг облачной программы складского учета «МойСклад» среди бизнесменов довольно высокий. Это обусловлено стабильностью работы приложения и достаточностью его функций для мелких предпринимателей.



Интерфейс программы складского учета «МойСклад» (рис. 1.2)

Кроме того, разработчики приложения не стали ограничиваться поддержкой торговли и добавили в функционал производственные операции.

Плюсы программы «МойСклад»:

1. Широкий функционал, подходящий для розничной, оптовой торговли, сферы общепита и небольших производств.

2. Поддержка дисконтных карт, создание клиентской базы, формирование воронки продаж.

3. Стабильная работа.

4. Дружелюбный интерфейс и легкость освоения программы новыми сотрудниками.

5.Наличие демо-версии с полным функционалом.

6.Мультиплатформенность: программой можно пользоваться на ОС Windows, macOS, Android, Linux, iOS.

7. Открытый API.

Минусы «МойСклад»:

1. Стоимость ежемесячной абонплаты выше среднерыночной.

2. Отсутствие шаблонов для продаж.

3. Тарифные планы программы складского учета МойСклад

4. Тарифные планы программы складского учета «МойСклад»

Можно сказать, что программа «МойСклад» создавалась для небольших магазинов и компаний, а приобретают ее и представители более крупного бизнеса. Но учитывать пожелания и замечания клиентов разработчики не спешат, что приводит к недовольству технической поддержкой и программой в целом.

Проанализировав плюсы и минусы данных товароучетных систем, которые являются одними из лидеров на СНГ рынке в своей нише и занимает высокие позиции в рейтингах я попытался реализовать плюсы и избежать минусы этих приложений.

# Разработка архитектуры проекта

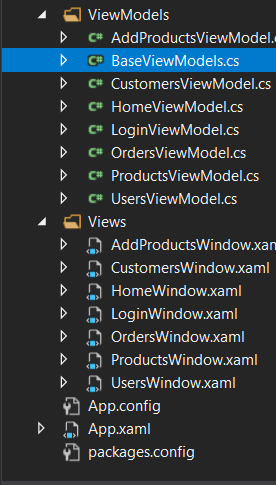
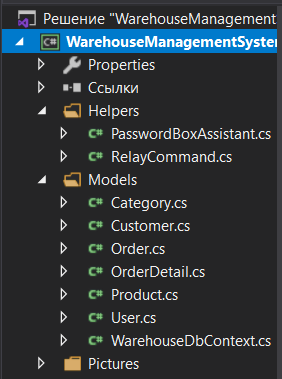
## Обобщенная структура

Решение представлено проектом «WarehouseManagementSystem» с имеющей структурой (Рисунок 2.1.).

Рисунок 2.1 – Обобщенная структура проекта

Описание обобщенной структуры проекта «WarehouseManagementSystem» представлено в таблице 2.1.

Таблица 2.1 – Обобщенная структура проекта



|  |  |
| --- | --- |
| Объект | Описание |
| Properties | свойства проекта, содержит информацию о сборке, используемых ресурсах и настройках |
| Ссылки | перечень сборок, используемых в проекте |
| Папка Helpers | привязка(соединение) команды с имеющей к ней прикладной логикой, отвечающей за обслуживание интерфейса |
| Папка Models | описывает используемые в приложении данные |
| Папка Pictures | содержит основные картинки проекта |
| Папка ViewModels | Связывает модель и представление, определяет логику по обновлению данных в модели |
| Папка Views | определяет визуальный интерфейс, через который пользователь взаимодействует с приложением |
| App.config | файл с параметрами проекта |
| App.xaml | класс Application |
| packages.config | файл для поддержки списка пакетов, на которые ссылается проект |

## Диаграмма UML

UML — унифицированный язык моделирования (Unified Modeling Language) — язык графического описания для объектного моделирования в области разработки программного обеспечения, для моделирования системного проектирования и отображения организационных структур.

В данной программе существует две разновидности ролей пользователя. На диаграмме вариантов использования, представлены роли администратора и пользователя (Рисунок 2.2.).

Пользователю предоставляется выполнять регистрацию и авторизацию (с помощью адреса электронной почты и пароля), управление клиентами, товарами, заказами, пользователями, отслеживание количества товаров на складе, их названия, категорий, стоимости, описания, фотографий, добавление нового товара, категории, клиента, создание нового пользователя, обновление товара, удаление товара, поиск товара.

Администратору предоставляется поддерживать работу c базой данных и регистрировать пользователей приложения.

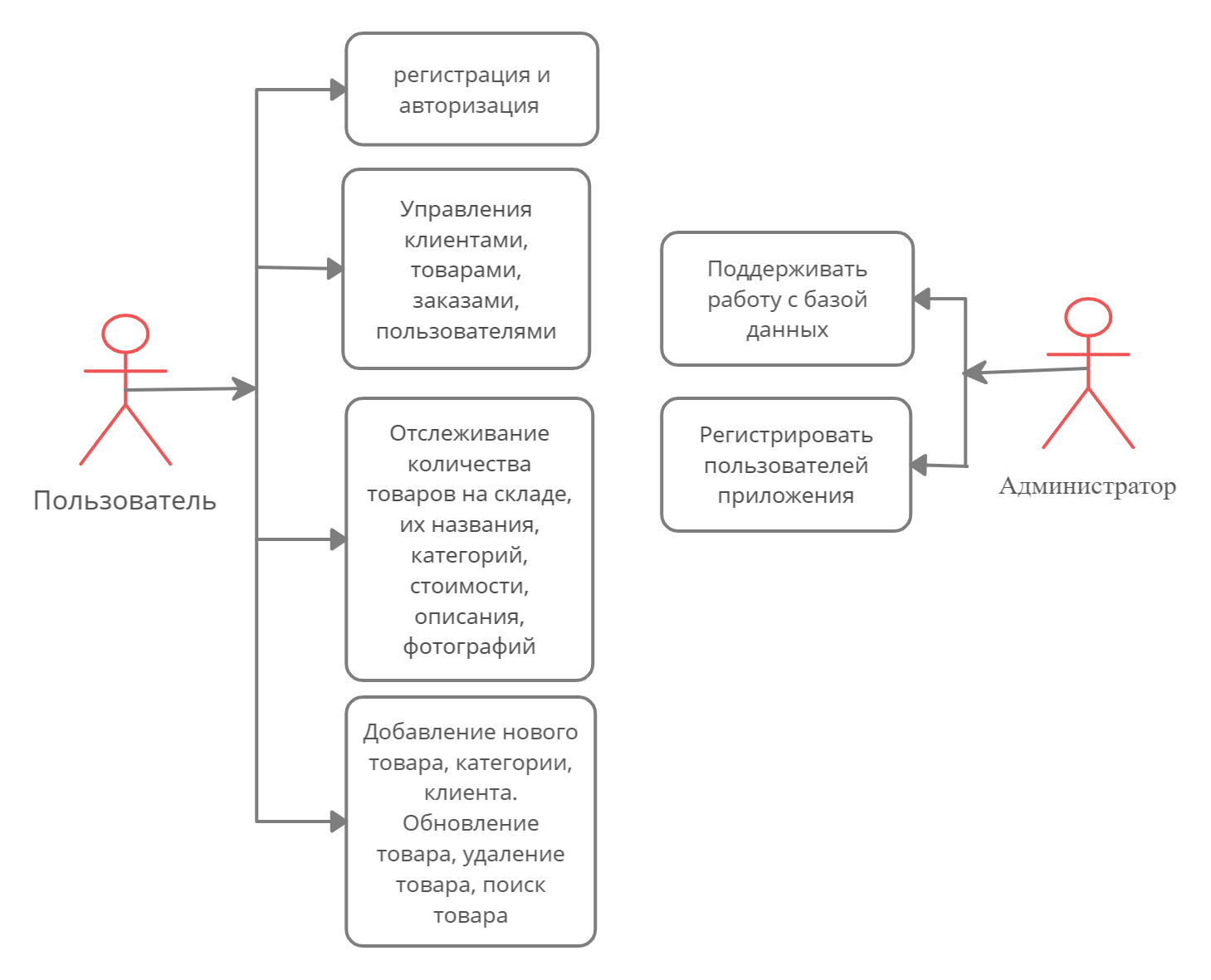


Рисунок 2.2 – Диаграмма вариантов использования

## Диаграмма классов

Структурная диаграмма языка моделирования UML, демонстрирующая общую структуру иерархии классов системы, их коопераций, атрибутов, методов, интерфейсов и взаимосвязей между ними. Широко применяется не только для документирования и визуализации, но также для конструирования посредством прямого или обратного проектирования.

На данной диаграмме классов изображены классы программы, методы, поля и свойства классов. Диаграмма классов служит для представления статической структуры модели системы в терминологии классов объектно-ориентированного программирования. Диаграмма классов может отражать различные взаимосвязи между отдельными сущностями предметной области, такими как объекты и подсистемы, а также описывает их внутреннюю структуру (поля, методы) и типы отношений (наследование, реализация интерфейсов).

Диаграмма классов моделей представления (связывает модель и представление через механизм привязки данных) представлена на рисунке 2.3.

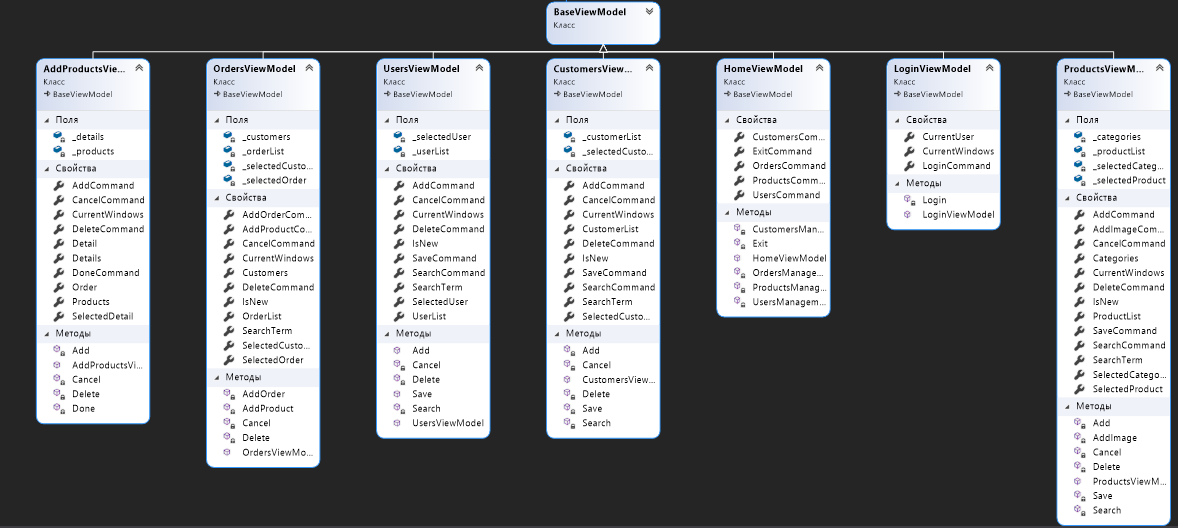


Рисунок 2.3 – Диаграмма классов модели представления.

Диаграмма классов представления (определяют визуальный интерфейс) представлена на рисунке 2.4.

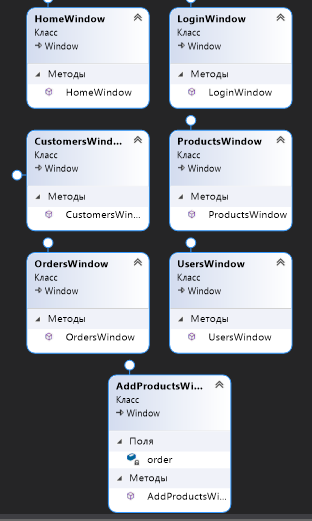


Рисунок 2.4 – Диаграмма классов представления.

Диаграмма классов моделей (описывает используемые в приложении данные) представлена на рисунке 2.5.

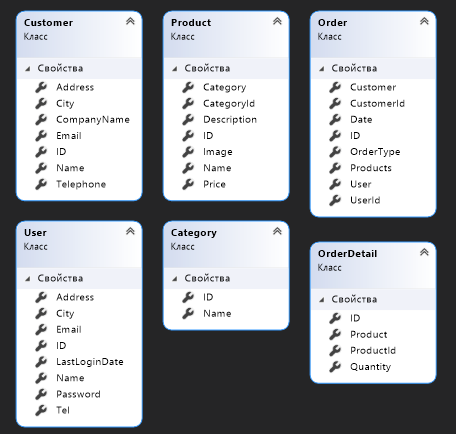


Рисунок 2.5 – Диаграмма классов моделей.

Диаграмма классов ассистирования пароля и привязки команды с имеющей к ней прикладной логикой, отвечающей за обслуживание интерфейса представлена на рисунке 2.6.

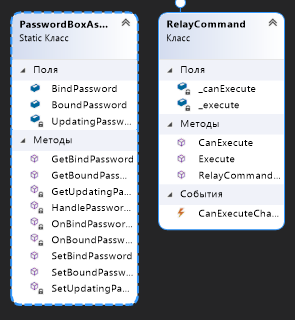


Рисунок 2.6 – Диаграмма классов ассистирования пароля и привязки команды с прикладной логикой.Диаграмма пользовательских элементов управления представлена на рисунке 2.7.

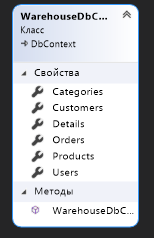


Рисунок 2.7 – Диаграмма пользовательских элементов управления.

Диаграмма пользовательских элементов управления представлена на рисунке 2.8.

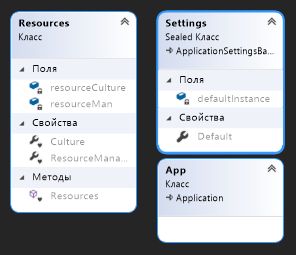


Рисунок 2.8 – Диаграмма пользовательских элементов управления.

## Проектирование базы данных

Данные для работы будут храниться в базе данных. Её создание происходит с помощью Microsoft SQL Server — это система управления реляционными базами данных, которая подключается к приложению.

При разработке приложения сначала разрабатывалась база данных, которая должна содержать всю нужную информацию для приложения, после программное обеспечение. База данных состоит из 5 таблиц, между которыми установлены связи один ко многим. (Рисунок 2.9.)

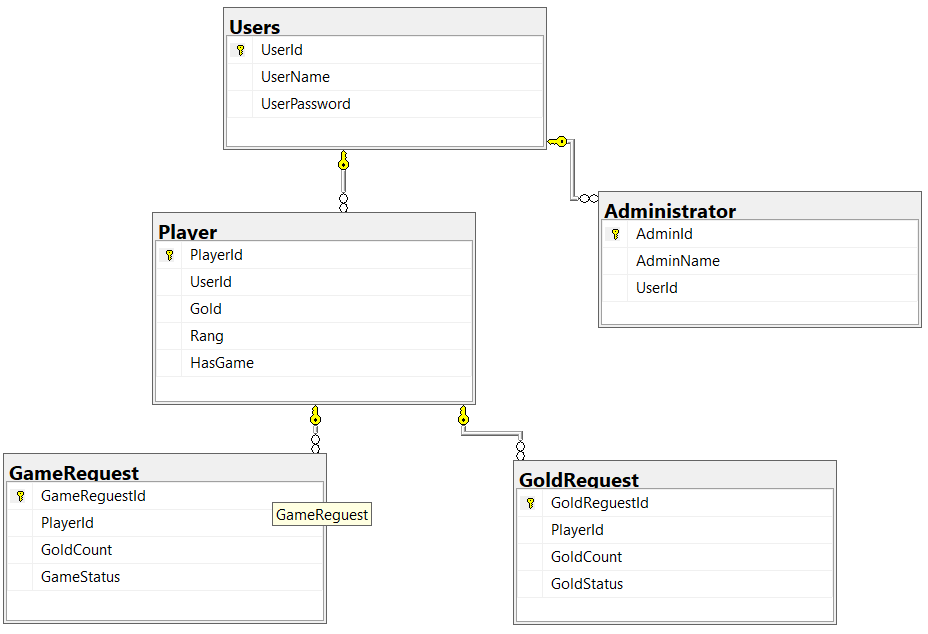


Рисунок 2.9 – Диаграмма базы данных.

Таблица Users предназначена для хранения информации о зарегистрированных пользователях, состоит из 3 столбцов:

* UserName – логин, уникальный для каждого пользователя;
* UserId – уникальный идентификатор пользователя;
* UserPassword – пароль пользователя.

Таблица Administrator служит для хранения информации об администраторах, состоит из 3 столбцов:

* AdminId – уникальный идентификатор администратора;
* AdminName – логин администратора;
* UserId – уникальный идентификатор пользователя;

Таблица Player предназначена для хранения информации о игроках, состоит из 5 столбцов:

* PlayerId – уникальный идентификатор игрока;
* UserId – уникальный идентификатор пользователя;
* Gold – количество золота на аккаунте;
* Rang – звание игрока в игре;
* HasGame – идентификатор игрока показывающий наличие игры.

Таблица GameRequest служит для хранения информации о статусе покупки игроком игры, состоит из 4 столбцов:

* GameRequestId – содержит уникальное значение, отвечающее за номер запроса на покупку игры;
* PlayerId – уникальный идентификатор игрока;
* GoldCount – количество имеющегося у игрока золота;
* GameStatus – статус обработки заказа администратором.

Таблица GoldRequest служит для хранения информации о статусе покупки игроком золота, состоит из 4 столбцов:

* GoldRequestId – содержит уникальное значение, отвечающее за номер запроса на покупку золота;
* PlayerId – уникальный идентификатор игрока;
* GoldCount – количество имеющегося у игрока золота;

GoldStatus – статус обработки заказа администратором.

Так же в моей базе данных присутствуют процедуры, а именно:

Процедура, подтверждающая покупку игры пользователем (Рисунок 2.10).

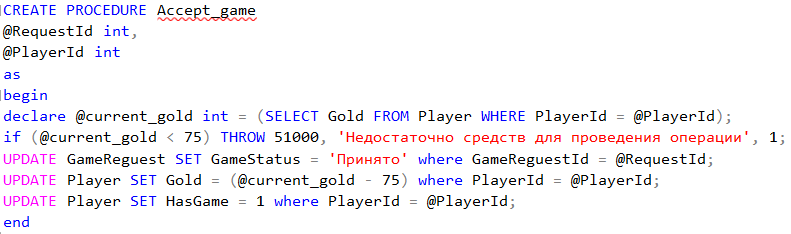


Рисунок 2.10 – процедура покупки игры.

Процедура, подтверждающая покупку игрового золота (Рисунок 2.11).

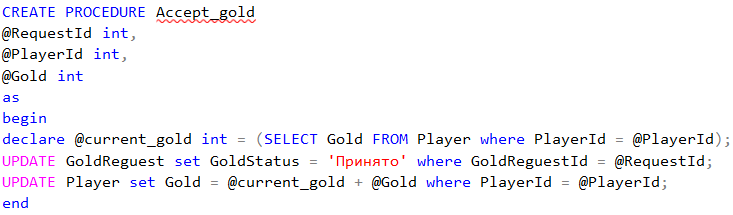


Рисунок 2.11 – процедура покупки золота.

Процедура, с помощью которой происходит регистрация пользователя (Рисунок 2.12).

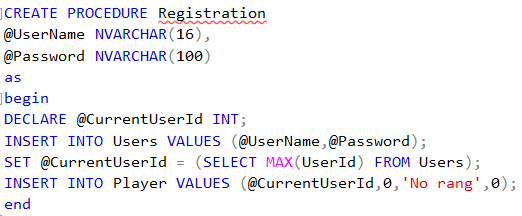


Рисунок 2.12 – процедура регистрации.

Процедура, с помощью которой сохраняется статистика (Рисунок 2.13).

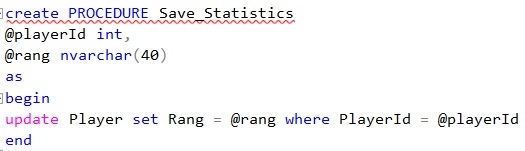


Рисунок 2.13 – процедура сохранения статистики.

Процедура, с помощью которой добавляется золото (Рисунок 2.14).



Рисунок 2.14 – процедура добавления золота.

# Разработка функциональной модели

## Описание функций приложения

Данное приложение позиционирует себя, как приложение, позволяющее:

* регистрацию пользователя с последующей авторизацией;
* авторизацию;
* просмотр новостей;
* возможность сделать заказ на некоторую сумму золота;
* возможность сделать заказ на игру;
* просмотр общего количества золота на аккаунте;
* возможность удалить аккаунт;
* возможность играть в игру.

В процессе создания приложения был реализован паттерн одиночка (Singleton, Синглтон) — порождающий паттерн, который гарантирует, что для определенного класса будет создан только один объект, а также предоставит к этому объекту точку доступа.

Синглтон позволяет создать объект только при его необходимости. Если объект не нужен, то он не будет создан, в этом отличие синглтона от глобальных переменных. В нашем проекте синглтон был реализован в классах Admin, Player, User. Они позволяют создать только один объект на время одной сессии.

## Классы в папках Classes

В папках Classes созданы сущности, которые представляют собой классы C# (Рисунок 3.1.).

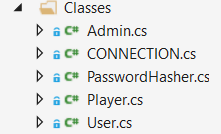
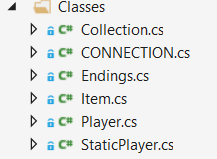
 

Рисунок 3.1 – Сущности и контексты в папках Classes.

## Основные окна приложения

### Главное окно агрегатора

MainWindow — главное окно агрегатора. Оно содержит кнопки навигации (Рисунок 3.2).

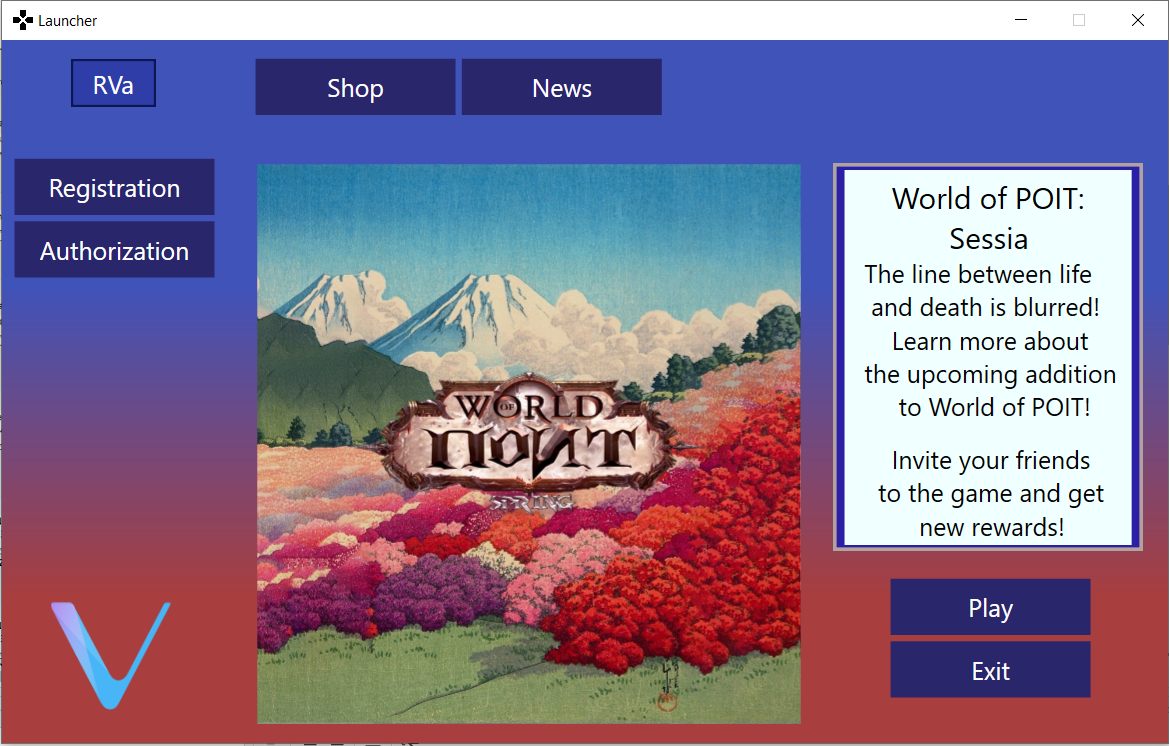


Рисунок 3.2 – главное окно агрегатора.

### Главное окно игры

MainWindow — основное окно игры. Оно содержит кнопки навигации, а также объект отображения страниц (Рисунок 3.3).

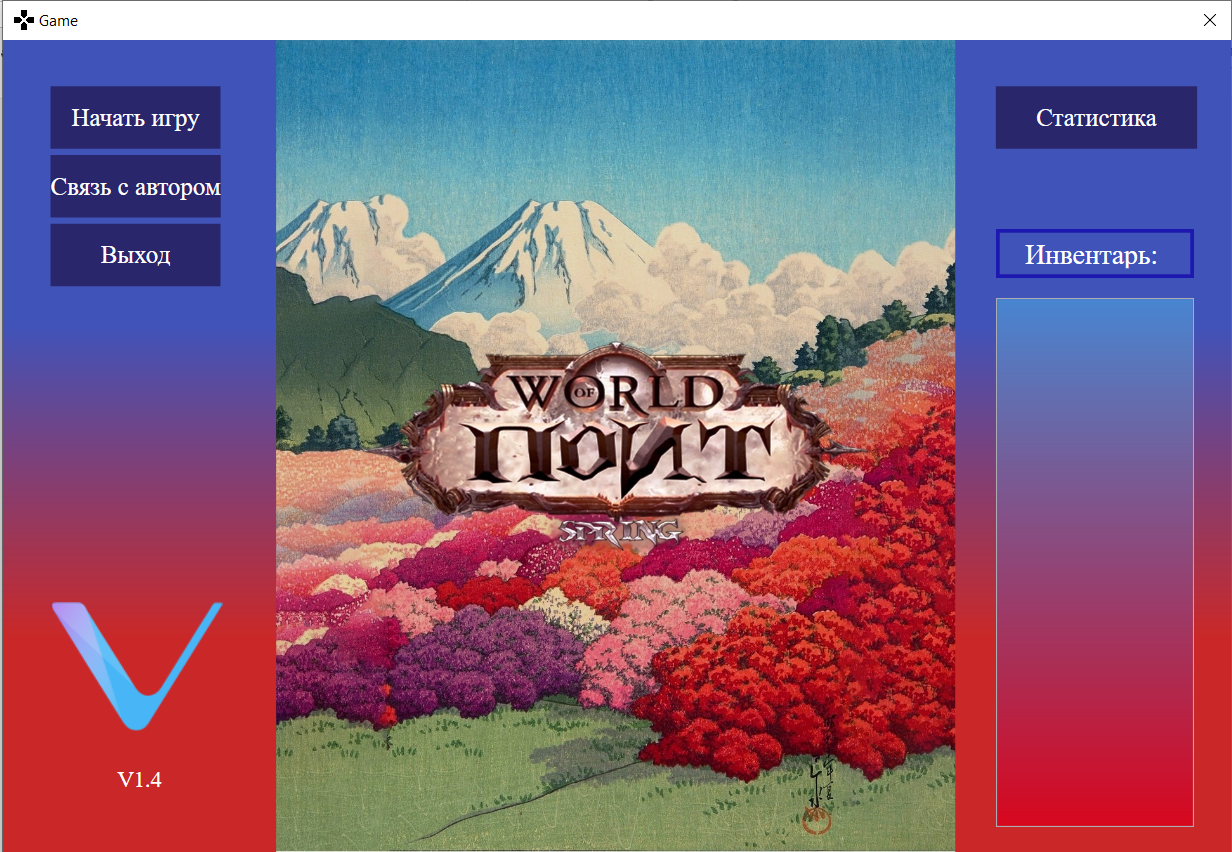


Рисунок 3.3 – главное окно игры.

## Основные страницы приложения

Навигация по приложению осуществляется с помощью окна MainWindow к которому прикреплён Frame. Свойство NavigationService связано с Frame.Content. При использовании кнопки перехода меняется страница (Рисунок 3.4.).

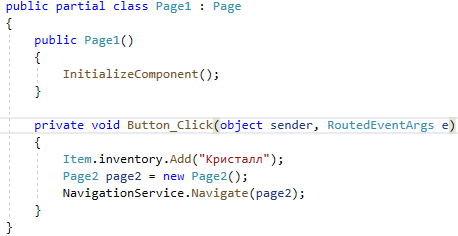


Рисунок 3.4 – Свойство первой страницы.

### Страница авторизации

Страница авторизации содержит поля Name и Password. Кнопка авторизации не сработает до тех пор, пока не будут введены корректные данные в поля. На этом этапе происходит проверка пользователя на наличие прав администратора. За это отвечает функция «IsAdmin» (Рисунок 3.5).

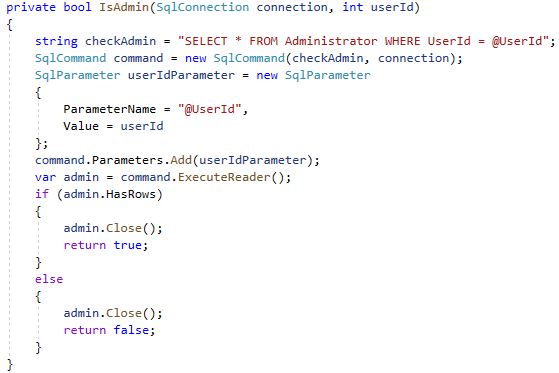


Рисунок 3.5 – Реализация функции «IsAdmin» .

### Страница регистрации

Страница регистрации содержит поля Name и Password. Кнопка регистрации не сработает до тех пор, пока не будут введены корректные данные в поля. На этом этапе происходит проверка на существование пользователя. За это отвечает функция «UserExists» (Рисунок 3.6).

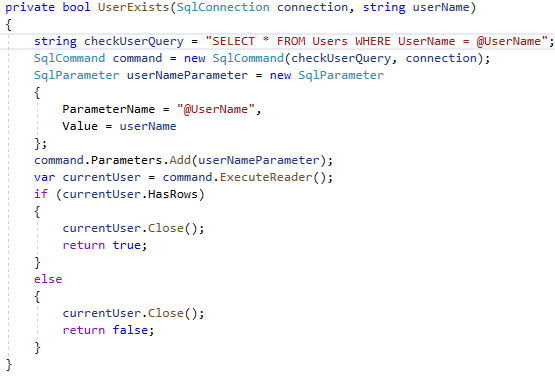
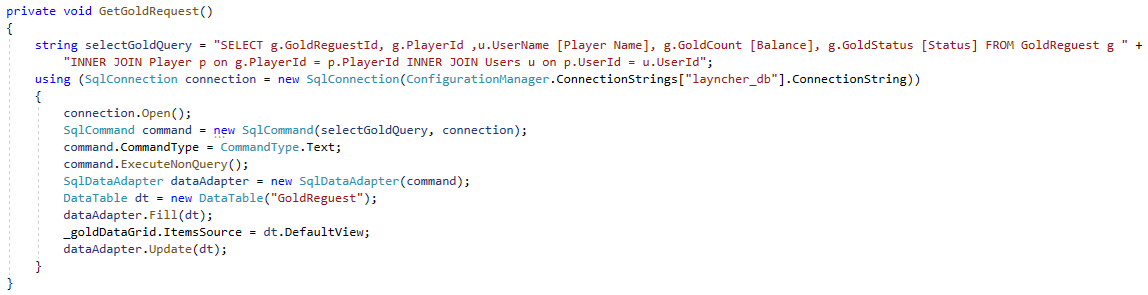


Рисунок 3.6 – Реализация функции «UserExists».

### Страница профиля

Страница профиля содержит основную информацию.

Пользователям, имеющим доступ «Admin», появляется возможность обработки заказов игрока. Получение заказов осуществляется с помощью вызова функций «GetGoldRequest» и «GetGameRequest» (Рисунок 3.7).



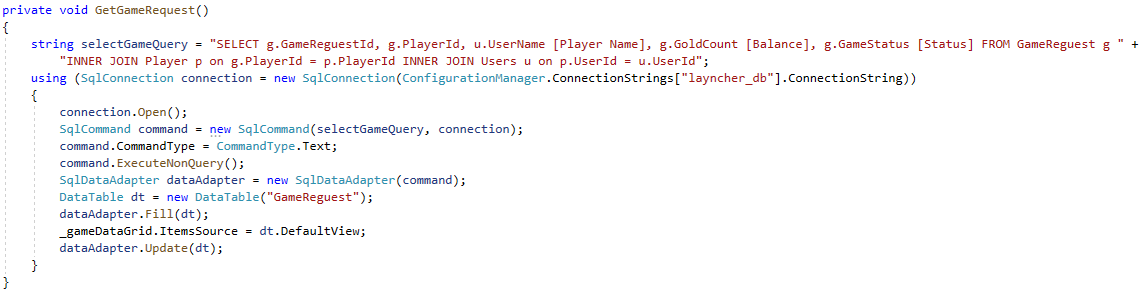


Рисунок 3.7 – Реализация функций «GetGoldRequest» и «GetGameRequest».

### Страница новостей

Страница новостей содержит сводную информацию обо всех изменениях внутри агригатора.

### Страница магазина

Страница магазина содержит кнопки, позволяющие отправить запрос на покупку игры, а также возможность удалить свой аккаунт. Событие удаления аккаунта «BanButtonClick» (Рисунок 3.8) и событие покупки игры «GameButtonClick» (Рисунок 3.9).



Рисунок 3.8 – Реализация события «\_banButton\_Click».



Рисунок 3.9 – Реализация события «\_buyGamesButton\_Click».

### Страница покупки золота

Страница покупки золота содержит поле для ввода суммы золота, которое хочет получить пользователь, а также кнопку отправляющую запрос администратору (Рисунок 3.10).

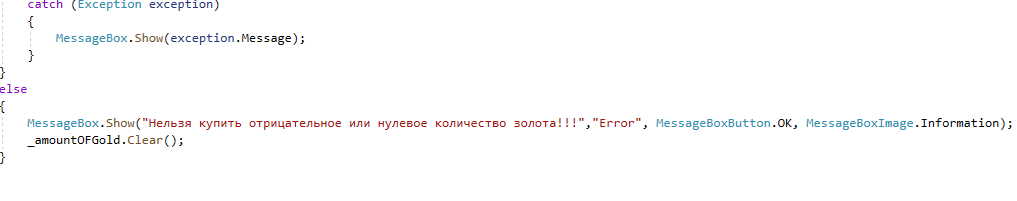
 

Рисунок 3.10 – Реализация события «\_buyButton\_Click».

## Основные игровые элементы управления

### Статистика

Статистика содержит информацию об игроке, а именно: имя игрока, титул игрока и его ранг (Рисунок 3.11).



Рисунок 3.11 – Реализация конструктора класса «StaticticsWindow1».

### Связь с автором

Игра содержит кнопку связи с автором. За работу данных кнопок отвечают события: «\_vk\_Click», «\_git\_Click» (Рисунок 3.12).

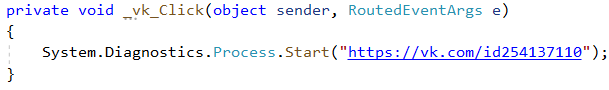
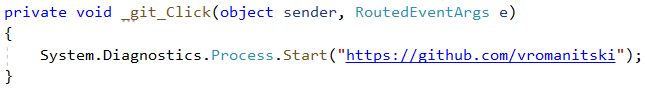
 

Рисунок 3.12 – Реализация событий «\_vk\_Click» и «\_git\_Click».

# Руководство пользователя

Перед установкой программы прежде всего нужно установить Microsoft SQL Server. Далее открыть SQL Server Management Studio, произвести все необходимые настройки. Открыть текстовый документ «scriptThatWorks», далее используя код внутри этого файла создать базу данных «layncher\_db» и все пять таблиц, а также все процедуры (Рисунок 4.1).

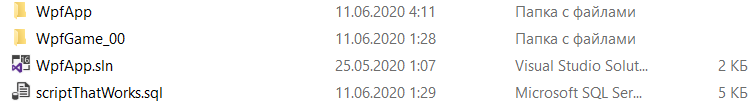
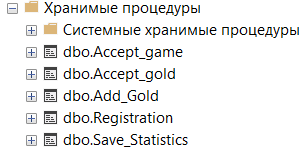
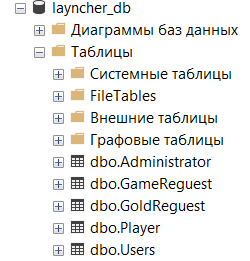
 

Рисунок 4.1 – Запуск БД.

Начальная страница для пользователя и админа одинаковые (Рисунок 4.2). На начальной странице необходимо ввести учетные данные, для дальнейшего входа в программу. С начальной страницы пользователь может перейти на страницу регистрации для создание учетной записи.

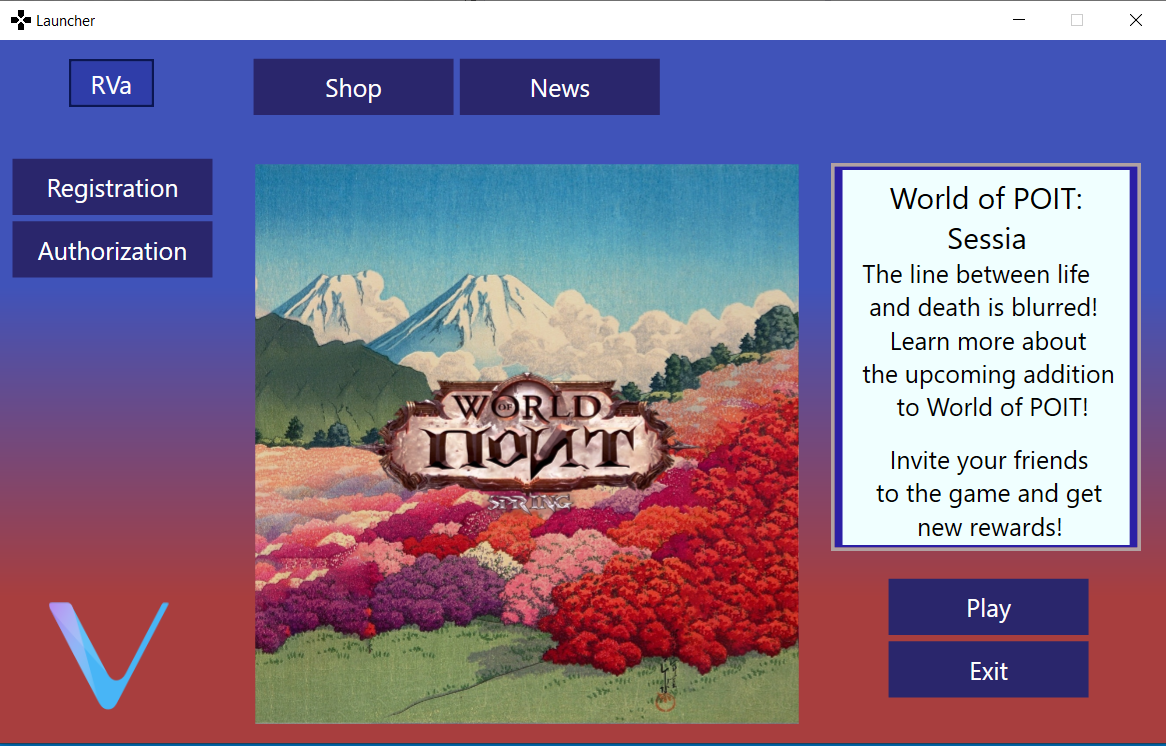


Рисунок 4.2 – Начальная страница.

На странице регистрации необходимо ввести общую информации для регистрации (Рисунок 4.3) и дальнейшей авторизации в программе.

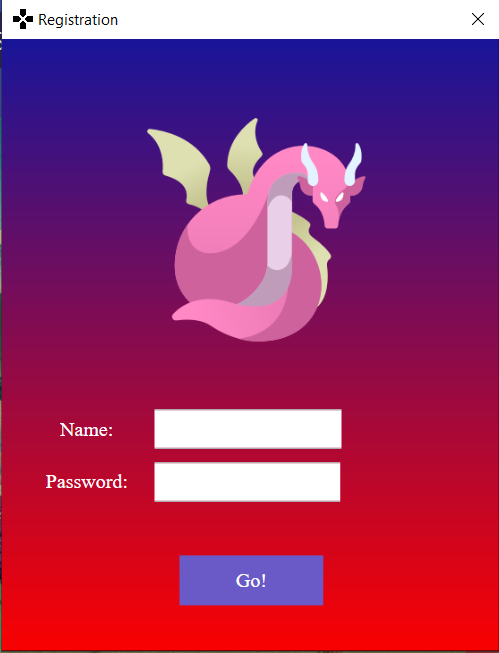


Рисунок 4.3 – Страница регистрации.

Страница профиля в зависимость от уровня доступа для различных пользователей разная: пользователь может запустить игру (при условии, что игра куплена) (Рисунок 4.4), а для администратора появляется таблица с запросами.

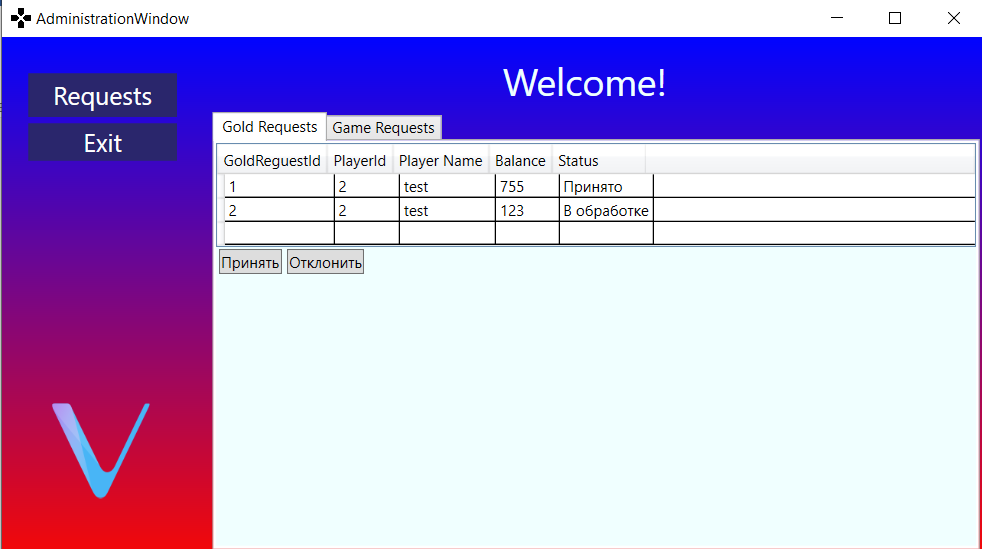
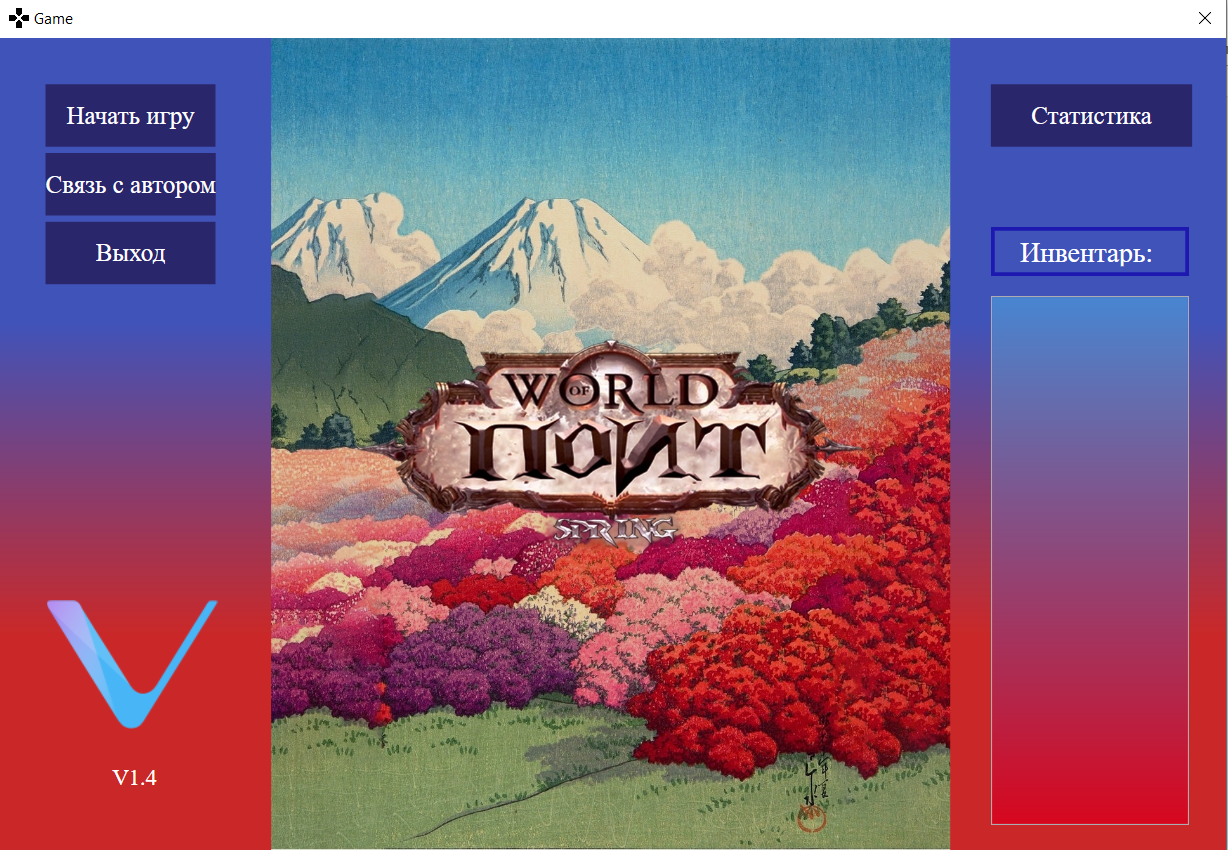


Рисунок 4.4 – Страница профиля для пользователя и администратора.

# Тестирование программного средства

Программное средство проходило тестирование. Сначала происходила отладка программы в процессе написания кода, после завершения основного кода, во время тестирования на работоспособность, и при наличии ошибок, исправлялся. Таким образом добивалась максимальная работоспособность программы. Пользователю нужно указать на неправильно введенные данные или недочеты, чтобы он мог их исправить.

Результат обработки исключения при авторизации, если не введён UserName или Password (Рисунок 5.1).

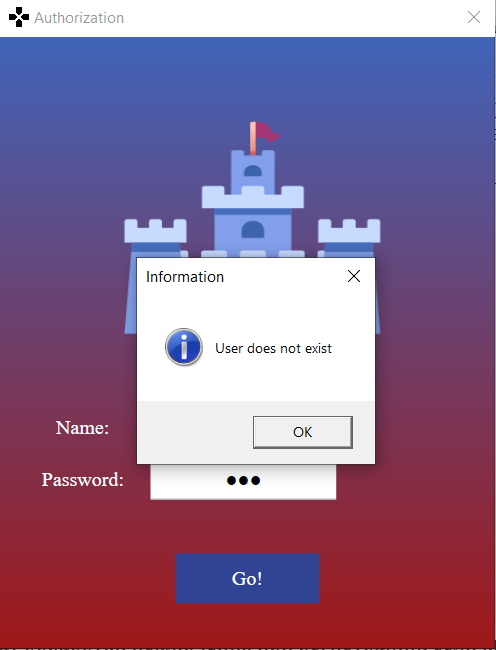


Рисунок 5.1 – Пример проверки авторизации.

Результат обработки исключения при регистрации, если введен UserName, зарегистрированный в системе ранее (Рисунок 5.2).

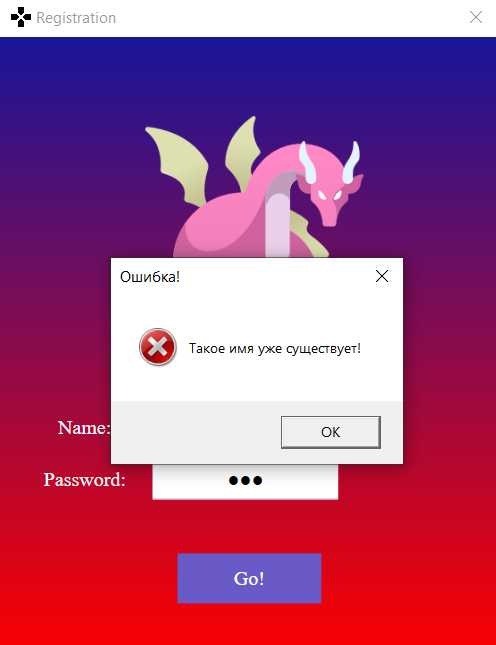


Рисунок 5.2 – Пример проверки регистрации.

Результат обработки исключения при попытке принять запрос, если не выбрана ячейка таблицы (Рисунок 5.3).

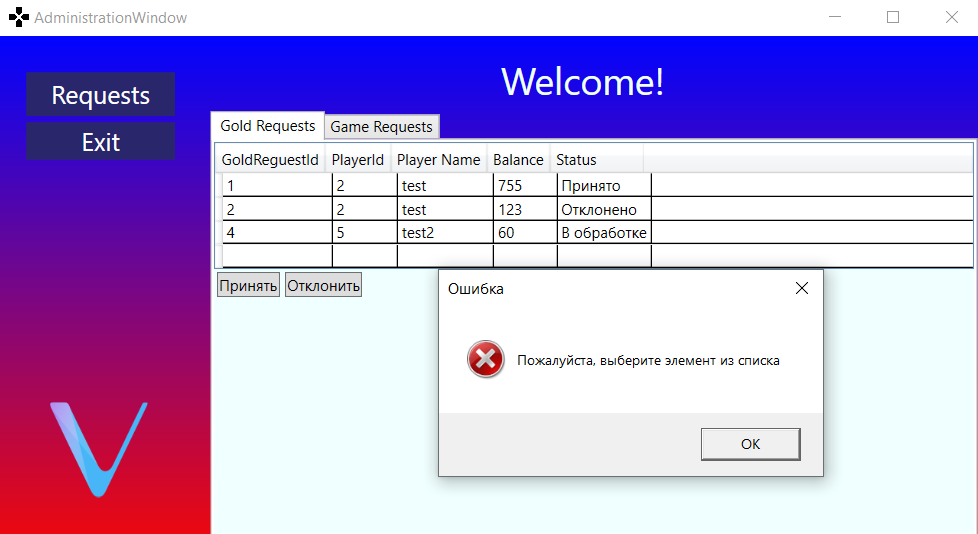


Рисунок 5.3 – Пример теста принять «нулевой» запрос.

Результат обработки исключения при попытке принять запрос, который уже был (Рисунок 5.4).

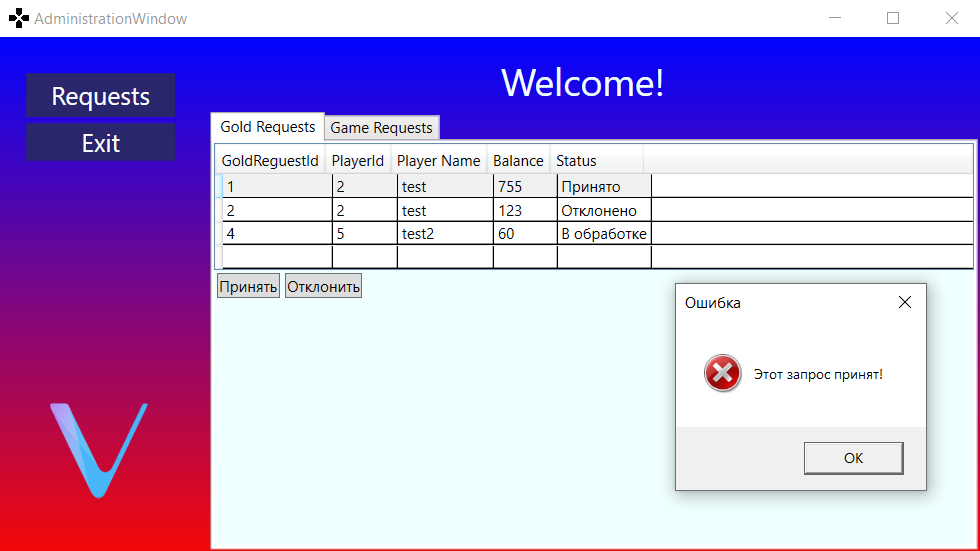


Рисунок 5.4 – Пример теста принять «принятый» запрос.

Результат обработки исключения при попытке купить игру с недостаточной суммой золота (Рисунок 5.5).



Рисунок 5.5 – Пример теста купить игру без нужной суммы золота.

# Заключение

Итогом курсового проекта является рабочее приложение типа «агрегатор» с возможностью играть в игру.

В ходе выполнения курсового проекта было выполнено следующее:

* создание базы данных для хранения информации;
* разработка архитектуры приложения;
* разработка функциональной части приложения;
* разработка пользовательского интерфейса;
* написание исходного кода приложения;
* тестирование приложения.

В соответствии с полученным результатом работы программы можно сделать вывод, что требования технического задания выполнены в полном объеме, интерфейс получился достаточно понятный и интересный, были использованы графические элементы.

За время выполнения курсового проекта по созданию «агрегатора» были изучены способы работы с «SQL Server», принципы создания приложения с использованием .NET и WPF, изучен новый язык верстки XAML для построения интерфейса и работы с ним. Использовался ADO.NET для работы с базой данный, а само приложение написано на языке С#.

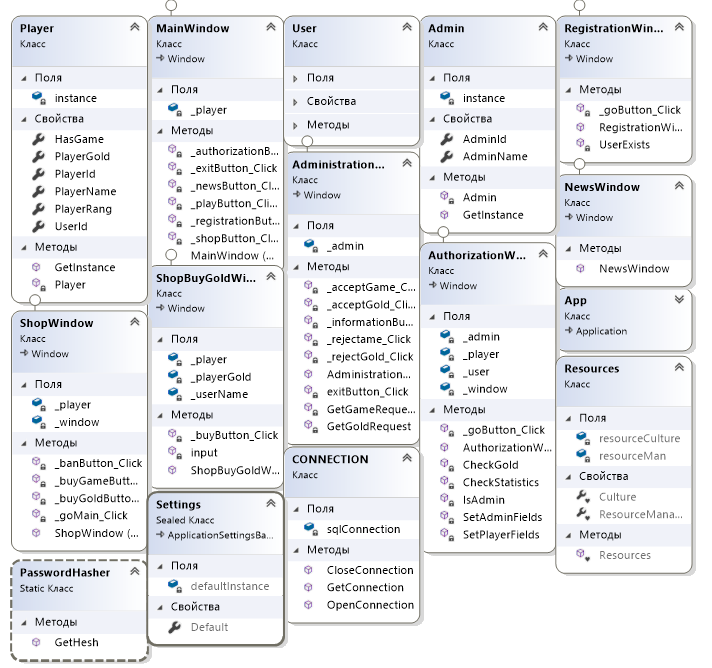
# Список используемых источников

1. Общие сведения о Microsoft Visual Studio [Электронный ресурс]. / Режим доступа <https://docs.microsoft.com/ru-ru/visualstudio/get-started/visual-studio-ide?view=vs-2019>/
2. Windows Presentation Foundation [Электронный ресурс]. / Режим доступа: <https://ru.wikipedia.org/wiki/Windows_Presentation_Foundation>
3. Введение в ADO.NET [Электронный ресурс]. / Режим доступа:

<https://metanit.com/sharp/adonet/1.1.php>/

1. Учебник по языку SQL [Электронный ресурс]. / Режим доступа: <https://habr.com/ru/post/255361/>

# Приложение A. UML-диаграмма проекта «WpfApp»



# Приложение B. UML-диаграмма проекта «WpfGame\_00»

