МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждения образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных Технологий

Кафедра Программной инженерии

Специальность 1-40 05 01 Информационные системы и технологии (по направлениям)

Направление специальности 1-40 05 01-03 Информационные системы и технологии (издательско-полиграфический комплекс)

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**КУРСОВОГО ПРОЕКТ:**

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

Тема Программное средство «Криптокошелёк»

Исполнитель

студент (ка) 2 курса группы 1 Парибок Илья Александрович

(Ф.И.О.)

Руководитель работы ассистент Панченко О. Л.

(учен. степень, звание, должность, подпись, Ф.И.О.)

Курсовой проект защищен с оценкой

Председатель Пацей Н.В

(подпись)

Минск 2022

**Содержание**

[ВВЕДЕНИЕ 4](#_Toc105663551)

[1. Аналитический обзор литературы и формирование требований 5](#_Toc105663552)

[1.1. Анализ прототипов 5](#_Toc105663553)

[1.1.1. Exodus 5](#_Toc105663554)

[1.1.2. Atomicwallet 6](#_Toc105663555)

[1.1.3. Trust Wallet 7](#_Toc105663556)

[1.2. Требования к проекту 8](#_Toc105663557)

[2. Анализ требований к программному средству и разработка функциональных требований 9](#_Toc105663558)

[2.1. Описание средств разработки 9](#_Toc105663559)

[2.2. Спецификация функциональных требований к программному средству 11](#_Toc105663560)

[2.3. Спецификация функциональных требований 12](#_Toc105663561)

[3. Проектирование программного средства 13](#_Toc105663562)

[3.1. Общая структура 13](#_Toc105663563)

[3.2. Взаимоотношения между классами 15](#_Toc105663564)

[3.3. Модель базы данных 15](#_Toc105663565)

[3.4. Проектирование архитектуры проекта 17](#_Toc105663566)

[3.5. Проектирование последовательностей проекта 18](#_Toc105663567)

[4. Реализация программного средства 18](#_Toc105663568)

[4.1. Основные классы программного средства 18](#_Toc105663569)

[4.2. Описание классов и методов программного средства 19](#_Toc105663570)

[4.2.1. Выполнение входа 19](#_Toc105663571)

[4.2.2. Регистрация 19](#_Toc105663572)

[4.2.3. Просмотр баланса 19](#_Toc105663573)

[4.2.4. Просмотр истории транзакций 20](#_Toc105663574)

[4.2.5. Просмотр реквизитов 20](#_Toc105663575)

[4.2.6. Панель администратора 20](#_Toc105663576)

[4.2.7. Восстановление кошелька 20](#_Toc105663577)

[4.2.8. Просмотр баланса кошельков 21](#_Toc105663578)

[5. Тестирование, проверка работоспособности и анализ полученных результатов 22](#_Toc105663579)

[5.1. Тестирование авторизации и регистрации 22](#_Toc105663580)

[5.2. Тестирование создания транзакций 25](#_Toc105663581)

[5.3. Тестирование панели администратора 25](#_Toc105663582)

[6. Руководство по установке и использованию 26](#_Toc105663583)

[Заключение 30](#_Toc105663584)

[Список литературы 31](#_Toc105663585)

[ПРИЛОЖЕНИЕ А 32](#_Toc105663586)

[ПРИЛОЖЕНИЕ Б 33](#_Toc105663587)

[ПРИЛОЖЕНИЕ В 34](#_Toc105663588)

[ПРИЛОЖЕНИЕ Г 35](#_Toc105663589)

## **ВВЕДЕНИЕ**

Спрос на криптовалюту растет, как и необходимость в надежных сервисах для ее хранения. Сейчас есть много способов, как хранить свой цифровой капитал. Крипто может иметь вид отдельного программного обеспечения, облачного сервиса, аппаратного устройства.

Целью данной работы является создания приложения для удобного хранения и управления криптовалютами. Данное приложение предполагает регистрацию пользователя и разделение прав администратора и пользователя.

Главная задача данного курсового проектирования – это разработка программного средства, которое реализует все вышеперечисленные функции и решает поставленные задачи. Язык разработки проекта – C#. При выполнении курсового проекта будут использованы принципы и приемы ООП, база данных MS SQL Server, и технология Windows Presentation Foundation (WPF).

## **Аналитический обзор литературы и формирование требований**

## **Анализ прототипов**

Были проанализированы цели и задачи, поставленные в данном курсовом проекте, а также рассмотрены аналогичные примеры их решений. На основании анализа всех достоинств и недостатков данных альтернативных решений были сформулированы требования к данному программному средству.

# **Exodus**

В качестве первого аналога выступает десктопное приложение «Exodus». Данное приложение доступно Windows, Linux, MacOS, Android и IOS. Имеет следующие возможности:

- Восстановить кошелек с помощью секретной фразы

- Просмотр баланса портфолио

- Просмотр графика изменения цены портфолио

- Отправить криптовалюту

- Чтение qr-кодов для отправки, с помощью камеры или экрана

- Принять криптовалюту

- Предоставление qr-кода для приема криптовалюты

- Экспортирование адресов актива и истории транзакций

- Обменять криптовалюту

- Установить пароль для входа в приложения

- Просмотр цена на криптовалюту

- Добавление нескольких кошельков

- Просмотр транзакций пользовательских активов

- Поиск по активам

- Возможность включать/отключать активы

- Возможность получить приватные ключи

- Просмотр краткой информации, видео и новостях о криптовалюте

Главное окно приложения представлено на рисунке 1.1.

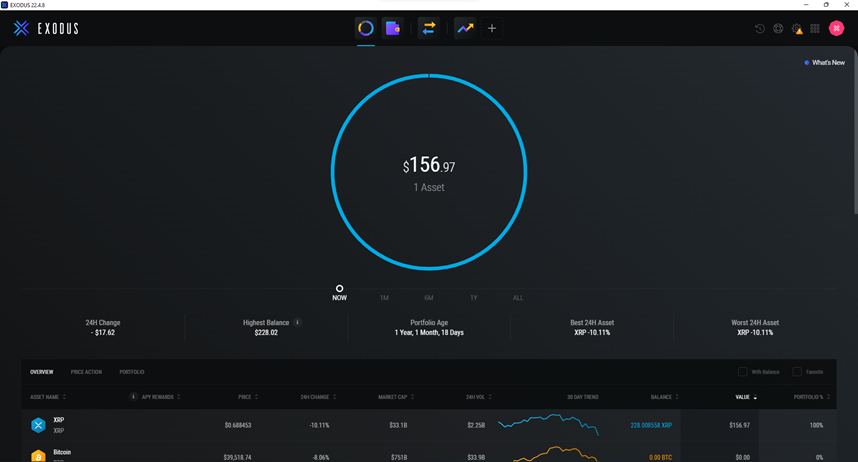


Рисунок 1.1 – Интерфейс приложения EXODUS

# **Atomicwallet**

В качестве следующего аналога выступает десктопное приложение «Atomic wallet». Данное приложение доступно Windows, Linux, MacOS, Android и IOS. Имеет следующие возможности:

- Восстановить кошелек с помощью секретной фразы

- Создание кошельков

- Просмотр баланса портфолио

- Отправить криптовалюту

- Принять криптовалюту

- Предоставление qr-кода для приема криптовалюты

- Экспортирование адресов актива и истории транзакций

- Обменять криптовалюту

- Установить пароль для входа в приложения

- Просмотр цена на криптовалюту

- Просмотр транзакций пользовательских активов

- Поиск по активам

- Возможность включать/отключать активы

- Возможность покупки криптовалюты

- Возможность установки дополнений для приложения

Недостатки:

- Нет русской локализации

- Просмотр графика изменения цены портфолио

- Чтение qr-кодов для отправки, с помощью камеры или экрана

- Добавление нескольких кошельков

- Возможность получить приватные ключи

Главное окно приложения представлено на рисунке 1.2.

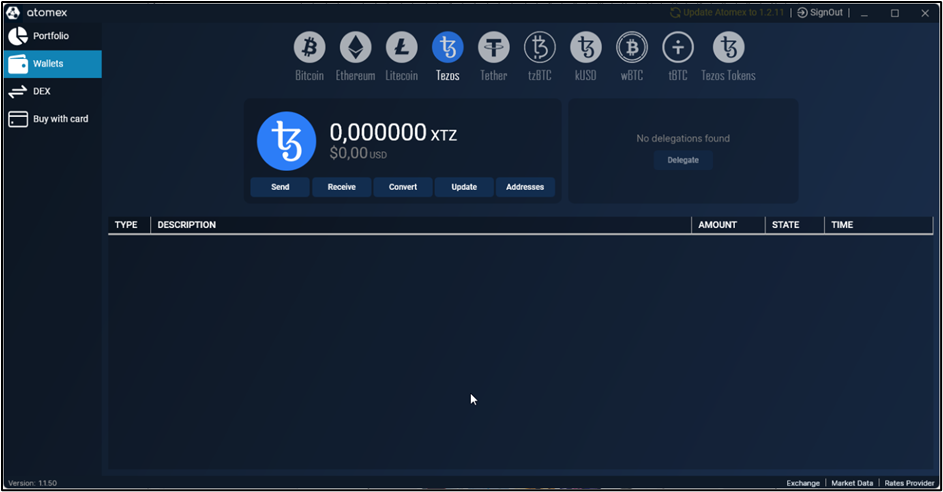


Рисунок 1.2 – Интерфейс криптокошелька «Atomicwallet»

# **Trust Wallet**

Заключительным приложением является мобильное приложение «Trust Wallet». Данное приложение доступно Android и IOS.

Имеет следующие возможности:

- Восстановить кошелек с помощью секретной фразы

- Создание кошельков

- Просмотр баланса портфолио

- Отправить криптовалюту

- Чтение qr-кодов для отправки, с помощью камеры

- Принять криптовалюту

- Получение уведомлений о цене

- Предоставление qr-кода для приема криптовалюты

- Обменять криптовалюту

- Установить пароль для входа в приложения

- Просмотр цена на криптовалюту

- Просмотр транзакций пользовательских активов

- Поиск по активам

- Возможность включать/отключать активы

- Возможность покупки криптовалюты

- Просмотр NFTs

- Браузер для децентрализованных приложений

- Русская локализация

Недостатки:

- Нет экспортирования адресов актива и истории транзакций

- Нет русской локализации

- Нет просмотра графика изменения цены портфолио

- Нет возможности добавление нескольких кошельков

- Нет возможности получить приватные ключи

Интерфейс программного обеспечения представлен на рисунке 1.3.

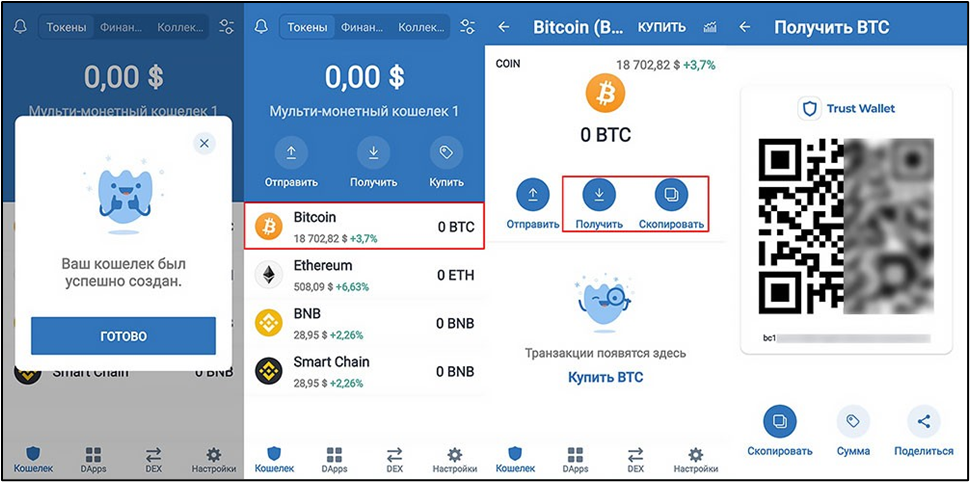


Рисунок 1.3 – Интерфейс криптокошелька «Trust Wallet»

## **Требования к проекту**

Обзор вышеперечисленных известных аналогов позволяет проанализировать все преимущества и недостатки альтернативных возможностей и позволяет сформулировать список требований, предъявляемых к программному средству, разрабатываемому в данном курсовом проекте. Программное средство должно обеспечивать возможность выполнения перечисленных ниже функций:

* управление администратором базой данных;
* возможность пользователю зарегистрироваться или войти в существующую учетную запись;
* возможность авторизированным пользователям просмотра информации о балансе своих активов;
* возможность создание транзакций;
* предоставление данных кошелька в виде qr кода.

## **Анализ требований к программному средству и разработка функциональных требований**

# **Описание средств разработки**

При разработке приложения были использованы:

* интегрированная среда разработки Microsoft Visual Studio 2022;
* программная платформа .NET Framework 6.0;
* язык программирования C#;
* расширяемый язык разметки XAML;
* технология WPF;
* библиотека NBitcoin;
* библиотека QRCoder;
* технология Entity Framework Core;
* MS SQL Server.

1. **Microsoft Visual Studio 2022**

Microsoft Visual Studio 2022 — это интегрированная среда разработки для написания, отладки и сборки кода, а также последующей публикации приложений. Данный продукт позволяет разрабатывать не только консольные, но и десктопные приложения, с использованиям таких технологий, как WinForms или WPF [1].

1. **Программная платформа .NET Framework 6.0**

Платформа .NET Framework — это созданная в 2002 году технология Microsoft, которая поддерживает создание и выполнение веб-служб и приложений Windows, обеспечивающая согласованную объектно-ориентированной среду программирования для локального сохранения и выполнения объектного кода, локального выполнения кода, распределенного в Интернете, либо для его удаленного выполнения. Основой платформы является общеязыковая среда Common Language Runtime, выполняющая код на разных поддерживаемых языках программирования [2].

1. **Язык программирования C#**

В качестве языка программирования используется C# – основной язык разработки в .NET Framework. Язык объектно-ориентированный, имеет строгую статическую типизацию, поддерживает перегрузку операторов, указатели на функции-члены классов, атрибуты, события, свойства, исключения. Используется как основной язык в технологии WPF [3].

1. **Технология WPF**

Для предоставления пользовательского интерфейса и разграничения дизайна и бизнес-логики используется технология Microsoft WPF – аналог WinForms, система для построения клиентских приложений Windows с возможностями взаимодействия с пользователем и графическая подсистема в составе .NET Framework, использующая язык разметки XAML [4].

1. **Расширяемый язык разметки XAML**

WPF предоставляет средства для создания визуального интерфейса, включая язык XAML (eXtensible Application Markup Language элементы управления, привязку данных, макеты, двухмерную и трёхмерную графику, анимацию, стили, шаблоны, документы, текст, мультимедиа и оформление. XAML представляет собой язык декларативного описания интерфейса, основанный на XML.

1. **Технология Entity Framework Core**

Для взаимодействия с базой данных в приложении WPF используется технология Entity Framework Core — кроссплатформерная и расширяемая версия технологии Entity Framework с открытым исходным кодом, которая в текущий момент развивается и дополняется.

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. Entity Framework Core – это последняя версия Entity Framework 7, которая совместима с .NET Core и на данный момент продолжает развиваться. Entity Framework Core предполагает три возможных способа взаимодействия с базой данных:

* Database First: Entity Framework Core создает набор классов, которые отражают модель конкретной базы данных;
* Model First: сначала разработчик создает модель базы данных, по которой затем Entity Framework Core создает реальную базу данных на сервере;
* Code First: разработчик создает класс модели данных, которые будут храниться в базе данных, а затем Entity Framework Core по этой модели генерирует базу данных и ее таблицы [5].

1. MS SQL Server

Для организации баз данных MS SQL Server использует реляционную модель, которая предполагает хранение данных в виде таблиц, каждая из которых состоит из строк и столбцов. Каждая строка хранит отдельный объект, а в столбцах размещаются атрибуты этого объекта. Для взаимодействия с базой данных применяется язык SQL (Structured Query Language). Клиент (например, внешняя программа) отправляет запрос 10 на языке SQL должным образом интерпретирует и выполняет запрос, а затем посылает клиенту результат выполнения. Основной используемый язык запросов — Transact-SQL — реализован на структурированном языке запросов (SQL) с расширениями [6].

1. Библиотека NBitcoin

NBitcoin — самая полная библиотека биткойнов для платформы .NET. Он реализует все наиболее актуальные предложения по улучшению биткойнов (BIP). Он также обеспечивает низкоуровневый доступ к примитивам Биткойн, так что вы можно легко создавать свое приложение поверх него.

1. Библиотека QRCoder

QRCoder — это простая библиотека, написанная на C#.NET, которая позволяет создавать QR-коды. Он не зависит от других библиотек и доступен как версия .NET Framework и .NET Core PCL на NuGet.

# **Спецификация функциональных требований к программному средству**

Программное средство должно предоставлять следующие функциональные возможности:

Для пользователя:

* регистрация;
* создание кошельков с помощью мнемонической фразы;
* авторизация;
* создание транзакций;
* просмотр транзакций;
* просмотр списка валют;
* просмотр баланса кошельков;
* восстановление кошелька;
* возможность предоставления адреса кошелька в виде qr кода;
* оповещение о транзакции по email;

Для администратора:

* авторизация;
* добавление криптовалют (их цена, название, сайт );
* просмотр транзакций пользователей;
* регистрация пользователей;
* редактирование криптовалют;
* удаление криптовалют;

# **Спецификация функциональных требований**

Для функциональности ПС необходимо создание базы данных для хранения информации приложения. Подробно база данных описано в следующем разделе.

В программном средстве необходимо реализовать регистрацию и авторизацию пользователей для доступа ко всем возможностям приложения. Для авторизации входными параметрами являются логин и пароль пользователя, которые содержатся в базе данных. Для регистрации входными данными являются имя, фамилия, логин, пароль, e-mail и мнемоническая фраза. Введенные данные, успешно прошедшие валидацию, заносятся в базу данных.

Для каждого пользователя создается его профиль, в котором он может просмотреть информацию о кошельке и о созданных транзакциях. Все сведения пользователе должны автоматически обновляться в базе данных.

Администратор обладает теми же возможностями, что и пользователь, а также обладает возможностями добавления, просмотра, редактирования и удаления информации о криптовалют, фильтрация и поиск транзакций, регистрация новых пользователей.

## **Проектирование программного средства**

# **Общая структура**

Программное средство «Криптокошелёк» имеет следующую структуру, представленную на рисунке 3.1.

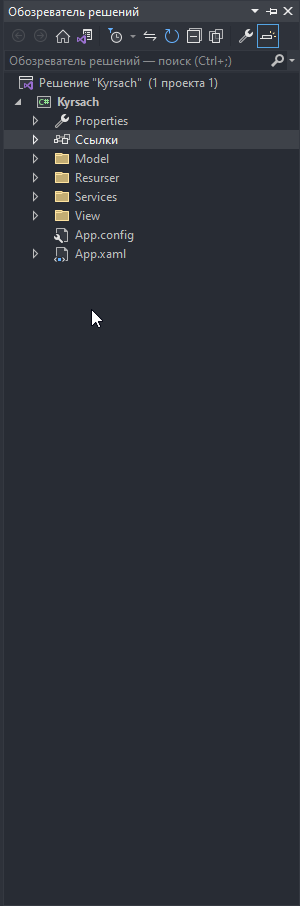


Рисунок 3.1 – Структура проекта

Описание структуры основных папок и файлов проекта представлено в таблице 3.1.

Таблица 3.1 – Описание структуры папок и файлов проекта

|  |  |
| --- | --- |
| Имя файла | Содержание |
| App.config | Файл с параметрами проекта. |
| Model | Папка содержащая классы для работы с бд |
| Папка Resources | Папка, содержащая стили. |
| Папка View | Папка, содержащая страницы и окна приложения |
| Папка Services | Папка, содержащая вспомогательные классы проекта. |

Более подробная структура содержимого папок программного средства описана на рисунке 3.2 и в таблице 3.2.

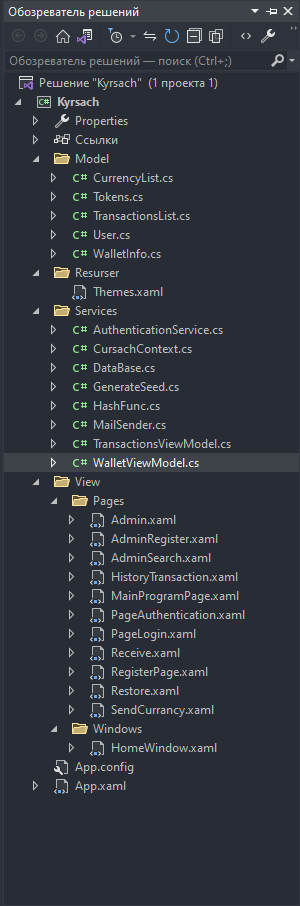


Рисунок 3.2 – Подробная структура проекта

Таблица 3.2 – Подробное описание структуры файлов проекта

|  |  |
| --- | --- |
| Имя файла | Содержание |
| CurrencyList.cs | Класс сущности CurrencyList. |
| Tokens.cs | Класс сущности Tokens. |
| TransactionsList.cs | Класс сущности TransactionsList. |
| User.cs | Класс сущности User |
| WalletInfo.cs | Вспомогательный класс для хранения информации о кошельках пользователя |
| Themes.xaml | Главная тема приложения |
| AuthenticationService.сs | Вспомогательный класс для регистрации и авторизации |
| DataBase.cs | Вспомогательный класс для для работы с бд. |
| GenerateSeed.cs | Вспомогательный класс для генерации мнемонической фразы. |
| HashFunc.cs | Вспомогательный класс для хеширования. |

Продолжение таблицы 3.2

|  |  |
| --- | --- |
| MailSender.cs | Класс для верификации почты пользователя |
| TransactionsViewModel | Вспомогательный класс для отображения транзакций |
| WalletViewModel | Вспомогательный класс для отображения кошельков |
| Папка Pages | Папка с страницами. |
| Admin.xaml | Страница администратора. |
| AdminRegister.xaml | Страница администратора для регистрации пользователей |
| AdminSearch.xaml | Страница администратора c транзакциями |
| HistoryTransaction.xaml | Страница с историей транзакций кошелька пользователя. |
| PageAuthentication.xaml | Страница авторизации почты |
| PageLogin.xaml | Страница авторизации пользователя |
| RegisterPage.xaml | Страница регистрации |
| Restore.xaml | Страница восстановление кошелька |
| SendCurrancy.xaml | Страница создания транзакций. |
| HomeWindow.xaml | Окно приложения |

# **Взаимоотношения между классами**

Для визуализации взаимосвязей между классами используется диаграмма UML – графическое представление набора элементов, изображаемое чаще всего в виде связанного графа с вершинами (сущностями) и ребрами (отношениями).

Для представления внутренней структуры программы в виде классов и связей между ними используется диаграмма классов. Приложение спроектировано таким образом, что каждый класс выполняет свои функции и практически не зависит от других. Диаграмма классов представлена в Приложении А.

# **Модель базы данных**

Для реализации поставленной задачи была создана база данных “Cursach”. Для ее создания использовалась система управления реляционными базами данных MSQL Server. База данных состоит из таблиц, представленных на рисунке 3.3. Скрипт для создания базы данных представлен в приложении Г.

На рисунке 3.4 проиллюстрирована структура таблицы «User», которая содержит информацию о пользователях. В данной таблице поле ID\_user, представленное является первичным ключом. Поле User\_name хранит логин пользователя, Password – пароль, Mail – адрес электронной почты, Mnemomic — мнемоническую фразу пользователя, Аdmin – роль пользователя (1 для администратора, 0 для пользователя).

На рисунке 3.5 изображена структура таблицы «Tokens», содержащая информацию о кошельках пользователей. Первичный ключ Adress хранит уникальный идентификатор адресс кошелька, поле ID\_Currency, ссылающееся на поле ID\_Currency таблицы «Currency\_list», содержит ID криптовалюты, Balance – баланс кошелька, Tx\_lsit – хэш кошелька, ID\_wallet – номер кошелька.

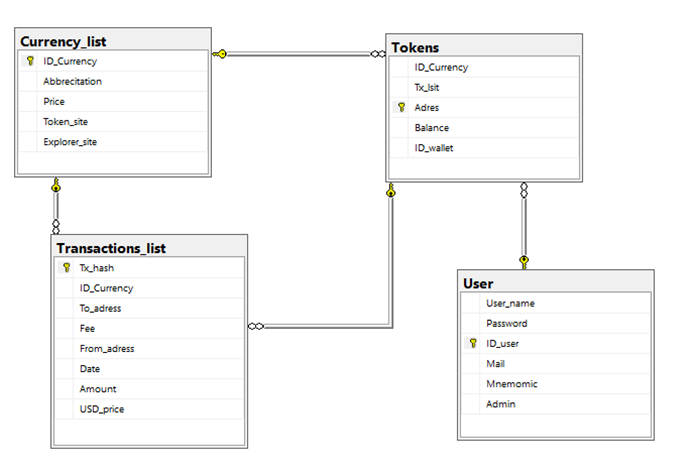


Рисунок 3.3 – База данных Cursach

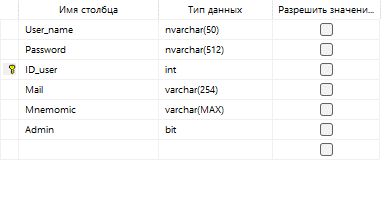


Рисунок 3.4 – Структура таблицы «User»

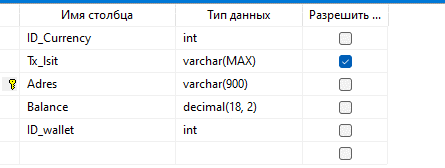


Рисунок 3.5 – Структура таблицы «Tokens»

На рисунке 3.6 показана структура таблицы «Transactions\_list», содержащая данные о совершённых транзакциях. В данной таблице первичный ключ Tx\_hash является идентификатором транзакции, поле ID\_Currency, ссылающееся на поле ID\_Currency таблицы «Currency\_list», содержит ID криптовалюты, совершившего заказ, поле To\_adress – адрес получателя транзации, Fee – комиссия транзакции. From\_adress – адрес отправителя, связанный с полем Adres таблицы Tokens. Date– дата транзакции, Amount – количество отсылаемых активов, USD\_price – цена криптовалюты.

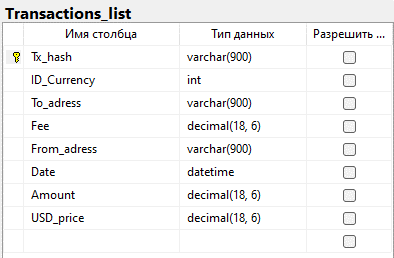


Рисунок 3.6 – Структура таблицы «Transactions\_list»

На рисунке 3.7 показана структура таблицы «Currency\_list», содержащая данные о криптовалютах. В данной таблице первичный ключ ID\_Currency является номером криптовалюты, поле Abbrecitation – Название криптовалюты в виде аббревиатуры, поле Price – цена криптовалюты, Token\_site – сайт криптовалюты, Explorer\_site – сайт проводника криптовалюты.

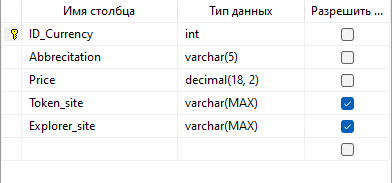


Рисунок 3.7 – Структура таблицы «Currency\_list»

# **Проектирование архитектуры проекта**

Для общего представления функционального назначения системы используется диаграмма использования, описывающая, какой функционал разрабатываемой программной системы доступен каждой группе пользователей. На диаграмме использования применяются два типа основных сущностей: варианты использования и группы пользователей.

Разные группы пользователей в диаграмме называются актёрами, и обозначают любые сущности, использующие систему. Любая функция системы называется вариантом использования. Каждый вариант использования обозначает набор действий, который может быть использован актёром для взаимодействии с системой, и определяет набор действий, выполняемых этой системой.

Диаграмма использования представлена в приложении Б.

# **Проектирование последовательностей проекта**

При проектировании приложения были рассмотрены различные сценарии использования. Диаграмма последовательностей представлена в приложении В получения транзакций и добавления в новых.

Из диаграммы видно, что сначала пользователь выбирает адрес, приложение делает запрос в базу данных, получая все связанные с адресом транзакции, отображает их. Затем пользователь создает собсвтенную транзакцию, тем саммым добавляя её в список всех транзакций.

## **Реализация программного средства**

# **Основные классы программного средства**

Для выполнения технических задач программного средства «Криптокошелёк» должны быть реализованы следующие функции и соответствующие им классы и методы:

* Выполнение входа;
* Регистрация;
* Восстановление кошелька;
* Создание транзакций;
* Просмотр транзакций;
* Просмотр списка валют;
* Просмотр баланса кошельков;
* Функционирование панелей администратора.

Далее подробно рассмотрены каждые из необходимых для выполнения технических задач функции, а также созданные для их выполнения классы и методы и их функционал и реализация.

# **Описание классов и методов программного средства**

## **Выполнение входа**

Необходимый функционал для возможности входа пользователя в аккаунт, а также для проверки корректности данных, реализован в классе Login.

Основным методом класса является метод signInButton\_Click(), выполняющийся при нажатия пользователя на кнопку «Войти».

При успешной валидации введённых пользователем входных данных выполняется вызов функции Login() класса AuthenticationService – проверка на существование введённого логина в базе данных, и при нахождении значения выполняется проверка на соответствие введённого пароля и хранящегося в базе данных пароля. При успешном прохождении всех проверок пользователю приходит письмо на почту с помощью функции SendEmail() класса MailSender, и направляется на страницу с проверкой кода. При вводе кода подтверждения его через метод AcessBTN, выполняющийся при нажатии пользователя на кнопку «Подтвердить», пользователь попадает на главную страницу.

Для перехода к странице регистрации со страницы входа реализован метод RegisterClick(), выполняющийся при нажатии пользователя на кнопку «Регистрация» на странице входа, и перенаправляющий пользователя со страницы входа на страницу регистрации.

Листинг реализации класса AuthenticationService представлен в приложении Г.

## **Регистрация**

Для совершения пользователем регистрации нового аккаунта реализован метод RegisterInDataBase класса DataBase.

При нажатии пользователя на кнопку «Зарегистрироваться» вызывается основной метод RegisterClick который пересылает на страницу регистрации. В начале все введённые пользователем данные проходят проверку: для валидации e-mail адреса вызывается метод IsValidEmail(), поле c мнемонической фразой проверятся свойством .IsValidChecksum класса Mnemonic библиотеки NBitcoin, остальные поля проверяются на пустоту.

После успешного прохождения вышеописанных проверок происходит попытка добавить пользователя в базу данных и перенаправление пользователя на страницу входа. Если логин уже зарегистрирован в базе данных, то пользователь получает соответствующее уведомление об ошибке.

Листинг реализации класса DataBase представлен в приложении Г.

## **Просмотр баланса**

После успешной авторизации пользователь перенаправляется на основную страницу программного средства реализованная классом WalletViewModel. Где он может просмотреть баланс своих кошельков.

В классе WalletViewModel храниться коллекция элементов WalletInfo, c кошельками пользователя, и текущий (выбранный) кошелёк. Выбор кошелька происходит через ЭУ combox.

Листинг реализации класса WalletViewModel представлен в приложении Г..

## **Просмотр истории транзакций**

Просмотр сеансов и подробной информации о фильме реализован в классе TransactionsViewModel, который инициализируется после вышеописанного нажатия на кнопку «История». В данном классе содержится основной метод WalletViewModel (), выполняющийся при нажатии на кнопку «История» или кнопки «Обновить» на истории.

Листинг реализации класса TransactionsViewModel представлен в приложении Г.

1. **Просмотр реквизитов**

При нажатие пользователем кнопки «Получить», вызывается метод Receive\_Click, который отправляет пользователя на страницу с реквизитами его кошелька. Страница реализует класс Receive\_Click, конструктор которого инициализирует основные данные, а метод QRcodeGenerate библиотеки QRCoder, представляет наши реквизиты в виде qr кода.

Листинг реализации класса Receive представлен в приложении Г.

## **Панель администратора**

Все необходимые администратору методы инкапсулированы в классе Admin, имеющем методы для добавления, удаления, и редактирования фильмов и их расписания.

Метод btnAdd\_Click() запускаются при нажатии на кнопку «Добавить» и добавляет введенные пользователем данные в бд.

Метод btnDelete\_Click () запускаются при нажатии на кнопку «Удалить» и удаляет выбранные пользователем данные из бд.

Метод btnEdit\_Click () запускаются при нажатии на кнопку «Редактировать» и редактирует выбранные пользователем данные в бд.

Метод btnCancel\_Click () запускаются при нажатии на кнопку «Отмена» и вызывает ClearAll(), который очищает поля данных которые ввёл пользователь.

Метод btnExit\_Click () функционирует после нажатия администратором на кнопку «Exit» и закрывает окно.

Листинг реализации класса Admin представлен в приложении Г.

## **Восстановление кошелька**

При нажатие пользователем кнопки «Восстановить», вызывается метод ReciveBySeed, который проверяет Seed фразу пользователя свойством IsValidChecksum класса Mnemonic библиотеки NBitcoin. Если фраза проходит валидацию, происходит запрос в бд, на поиск аккаунта. Если пользователь найден, проявляется поле для ввода нового пароля и кнопка «Обновить пароль», при нажатие которого вызывается метод UpdatePassword() который меняет пароль пользователя в бд.

Листинг реализации класса Restore представлен в приложении Г.

## **Просмотр баланса кошельков**

Все необходимые администратору методы инкапсулированы в классе AdminSearch. При нажатии на радиокнопки «ByAmount» и/или «ByFee» вызываются методы «CountVisible» и/или «FeeVisible» которые открывают и закрывают возможность ввода данных для фильтрации списка транзакций. При нажатие кнопки «FindTransaction» вызывается метод FindTransaction() который делает запрос в бд, а после чего фильтрует результаты на основе ограничений введённых пользователем.

Листинг реализации класса AdminSearch представлен в приложении Г

## **Тестирование, проверка работоспособности и анализ полученных результатов**

## **Тестирование авторизации и регистрации**

В момент авторизации и регистрации возможна ситуация, когда пользователь вводит некорректные данные, например, неверный пароль, незарегистрированный логин, некорректный адрес электронной почты. Такие исключения обрабатываются программным средством с помощью показа пользователю всплывающих сообщений с текстом ошибки. Примеры обработки разных видов исключительных ситуаций представлены на рисунках 5.1 – 5.8.

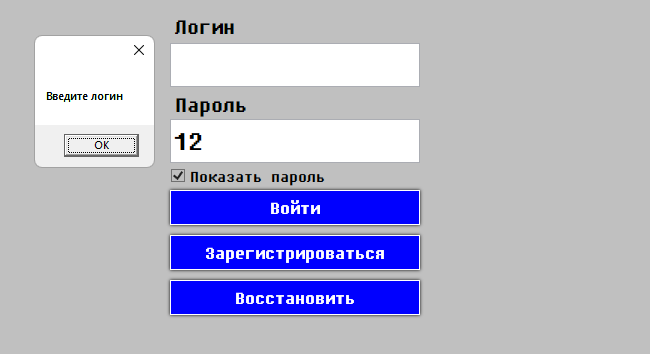


Рис 5.1 – Обработка пустого поля логина

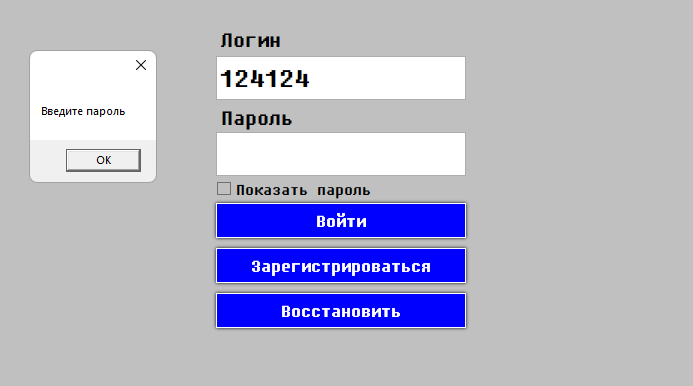


Рис 5.2 – Обработка пустого поля пароля

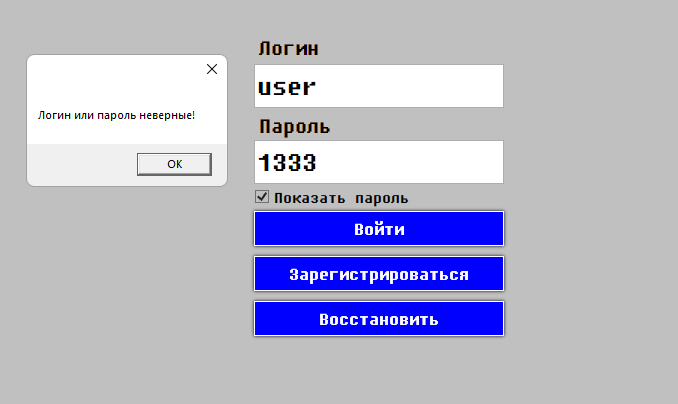


Рис 5.3 – Обработка неверного пароля при корректном логине

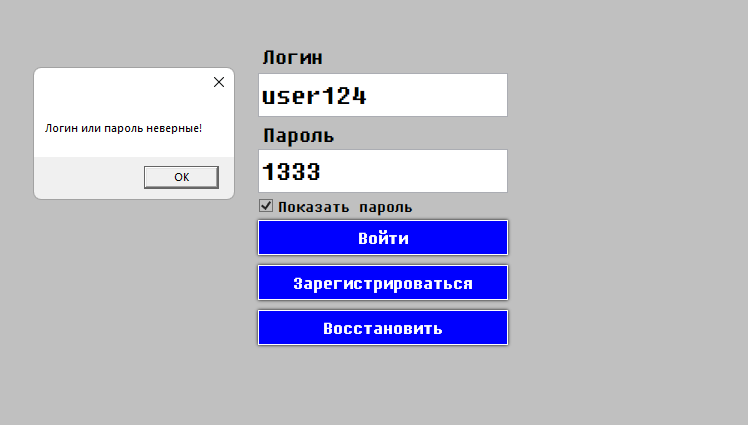


Рис 5.4 – Обработка ввода несуществующего логина

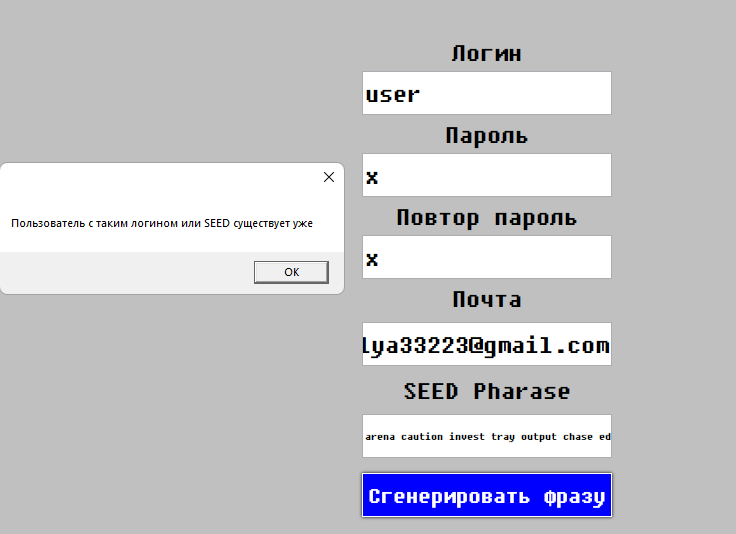


Рис 5.5 – Обработка регистрации с занятым логином

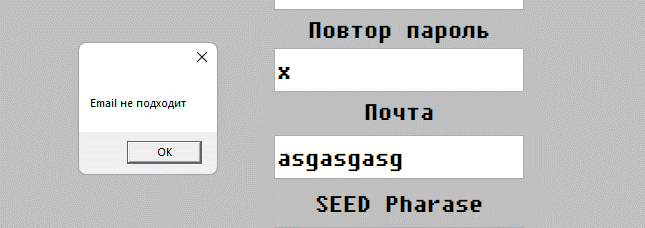


Рис 5.6 – Обработка регистрации с некорректным e-mail

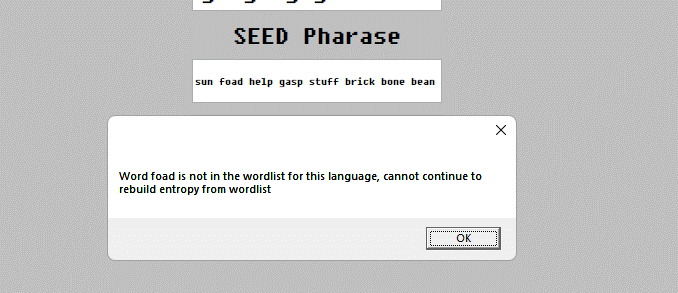


Рис 5.7 – Обработка регистрации с некорректной мнемонической фразой

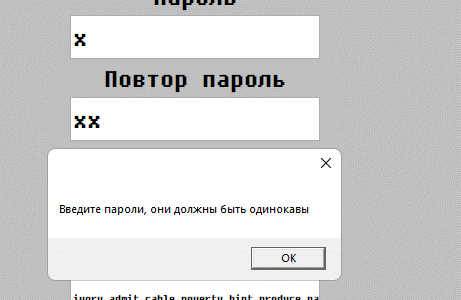


Рис 5.8 – Обработка регистрации с некорректным паролями

## **Тестирование создания транзакций**

При заполнении формы транзакции. При возникновении таких исключений пользователь уведомляется всплывающим окном. Примеры обработки исключительных ситуаций представлены на рисунках 5.9-10.

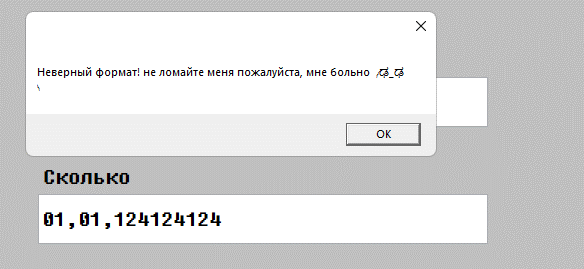


Рис 5.9 – Обработка ввода некорректных данных.

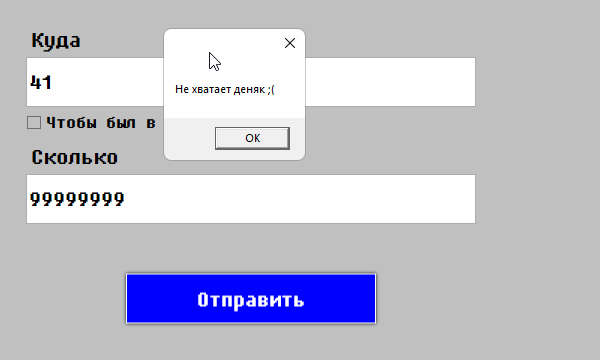


Рис 5.10 – Обработка отправки средств больше чем на счёте

## **Тестирование панели администратора**

В панели администратора возможно генерирование многих исключительных ситуаций, таких как: нажатие на кнопку «Удалить» или «Редактировать» при отсутствии выбранного значения в списке, ввод нецелочисленных либо выходящих за допустимые границы значений в поля. Все вышеописанные исключительные ситуации генерируют всплывающее окно с текстом ошибки, представленные на рисунках 5.11 – 5.12.

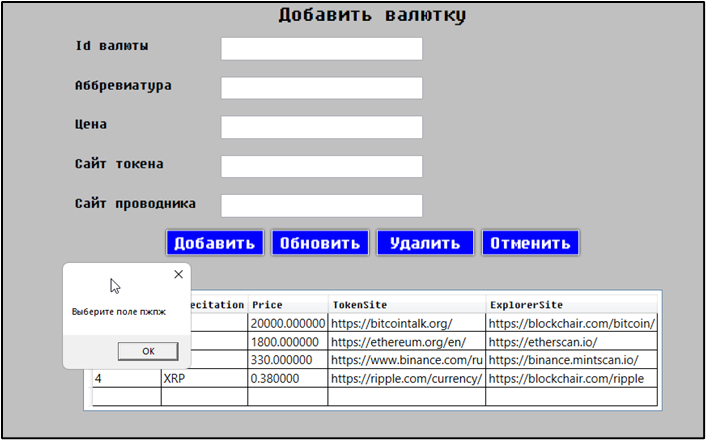


Рис 5.11 – Обработка нажатия на кнопку «Удалить» при отсутствии выбранного элемента в списке

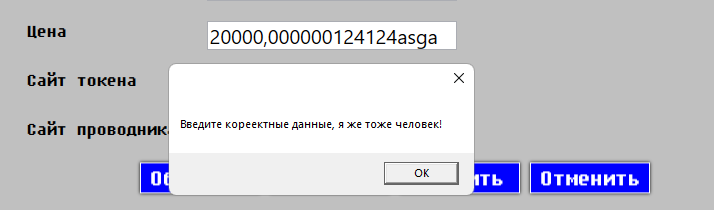


Рис 5.12 – Обработка ввода значений выходящего за границы допустимых значений

## **Руководство по установке и использованию**

При запуске программного средства «Криптокошелёк» пользователь попадает на страницу авторизации, содержащая формы входа и регистрации.

Стартовая страница позволяет зарегистрированному пользователю совершить вход в аккаунт, а не зарегистрированному – перейти на страницу регистрации нажатием на кнопку «Регистрация» и зарегистрироваться в системе, после чего пользователь будет перенаправлен на страницу авторизации, с которой он может совершить вход в систему, так же есть возможность восстановить аккаунт с помощью мнемонической фразы нажав кнопку «Восстановить».

По умолчанию новому зарегистрированному клиенту присваивается роль «Пользователь». Роль «Администратор» при регистрации не присваивается. Окно авторизации представлено на рисунках 6.1, 6.2.

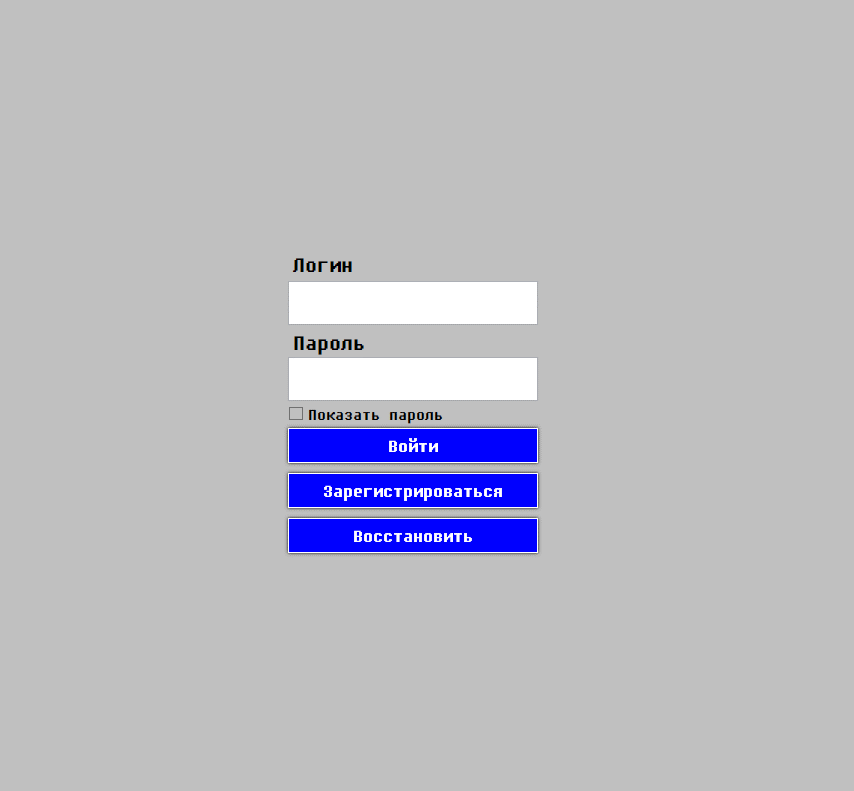


Рисунок 6.1 – Страница авторизации

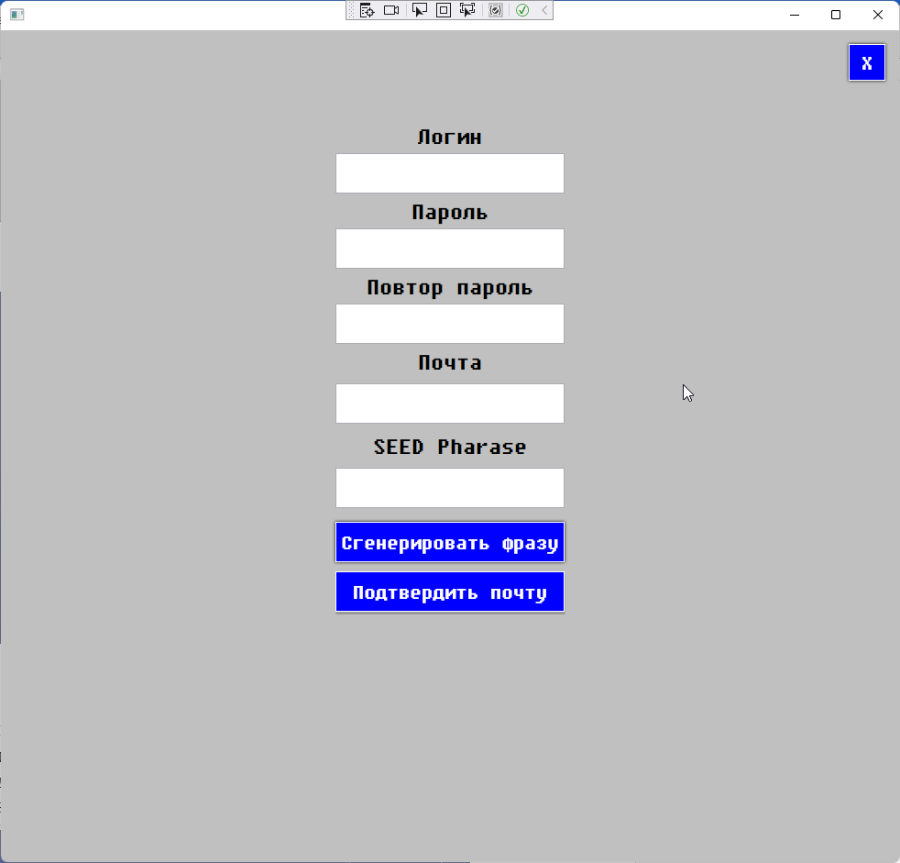


Рисунок 6.2 – Страница регистрации

После выполнения авторизации пользователь перенаправляется на основную страницу программного средства.

Пользователю доступен просмотр балансов всех кошельков, смена кошелька.

Основная страница изображена на рисунке 6.3.

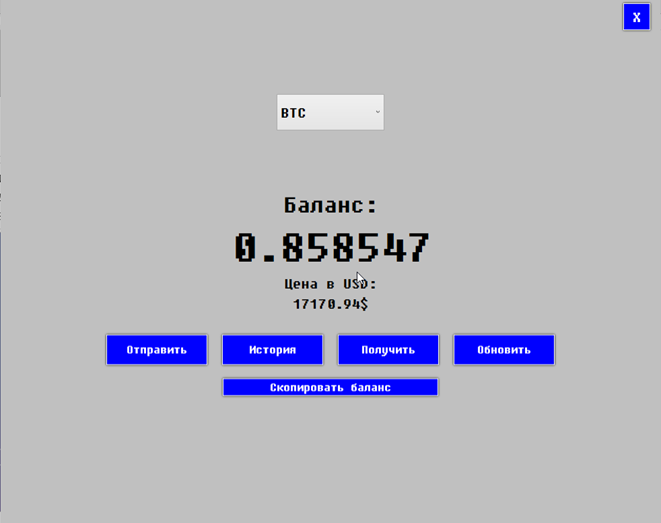


Рисунок 6.3 – Основная страница

При нажатии на кнопку «Отправить» пользователь перенаправляется на страницу с формой для отправки средств. Пользователю надо заполнить форму и нажать отправить.

Страница оформления транзакции проиллюстрирована на рисунке 6.4.

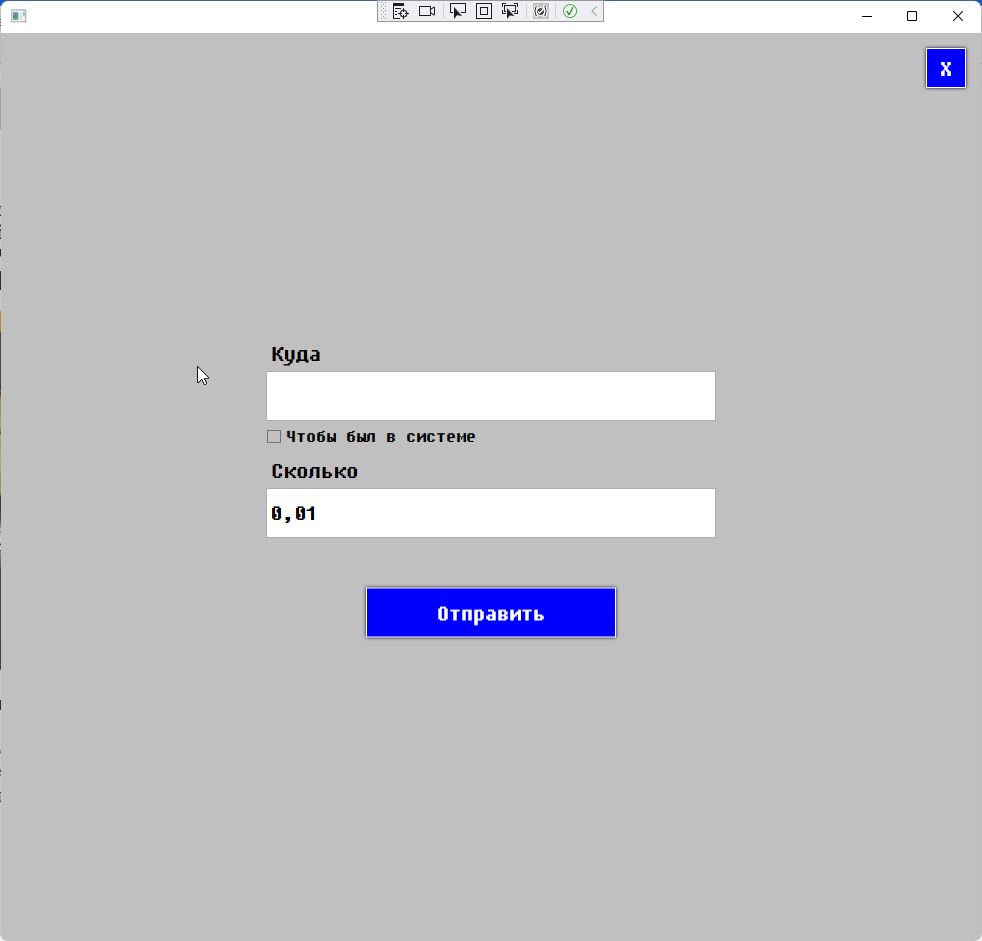


Рисунок 6.4 – Страница оформления транзакции

При нажатии на кнопку «Получить» пользователь перенаправляется на страницу с реквизитами своего кошелька.

Страница оформления реквизитов проиллюстрирована на рисунке 6.5



Рисунок 6.5 – Окно реквизитов

При нажатии на кнопку «История» пользователь перенаправляется на страницу с историей транзакций выбранного им в главном меню кошелька.

Страница оформления История транзакций на рисунке 6.6.

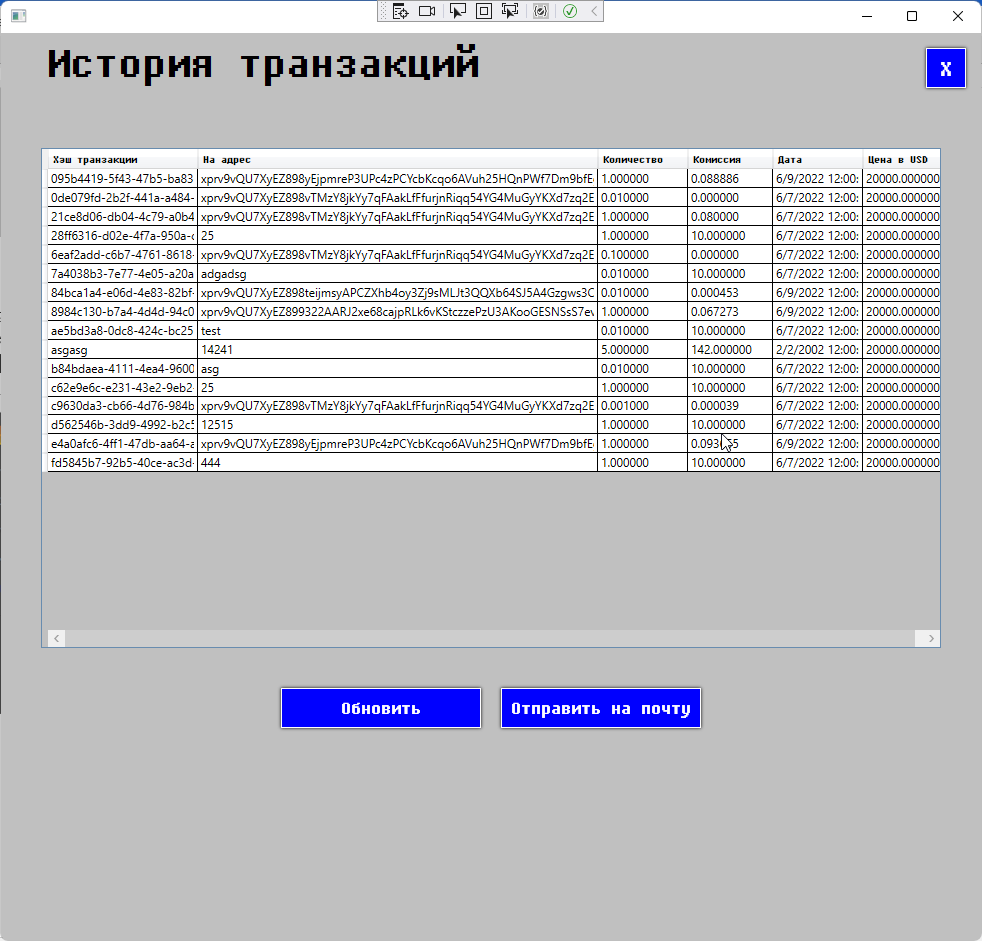


Рисунок 6.6 – История транзакций

# **Заключение**

В ходе выполнения курсового проекта было разработано программное средство «Криптокошелёк» на языка C# с использованием технологий Entity Framework Core, WPF.

При разработке программного средства были выполнены все пункты из указанного списка предполагаемого основного функционала приложения, а именно:

* создана база данных;
* реализована возможность пользователю зарегистрироваться или войти в существующую учетную запись;
* реализована возможность восстановления аккаунта;
* реализована возможность создания транзакций;
* реализована возможность просмотра баланса кошельков;
* реализована возможность просмотра реквизитов;
* реализовано уведомление пользователя о совершении транзакции.

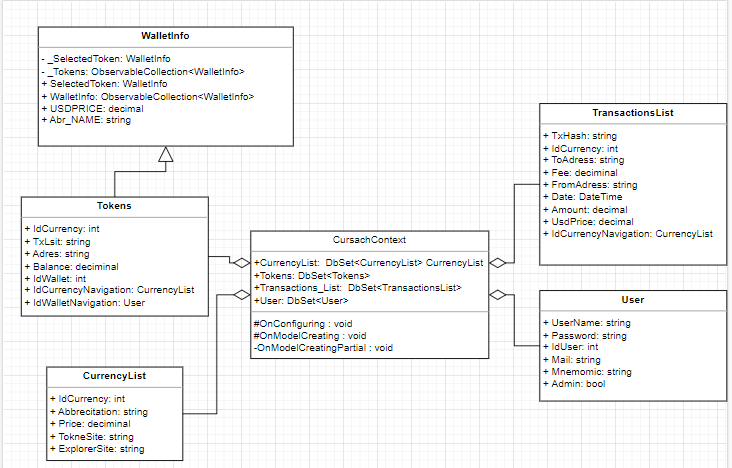
Тестирование программы показало, что она работает корректно и выполняет все свои функции.

В соответствии с полученным результатом работы программы можно сделать вывод, что разработанная программа работает верно, а требования технического задания выполнены в полном объеме.

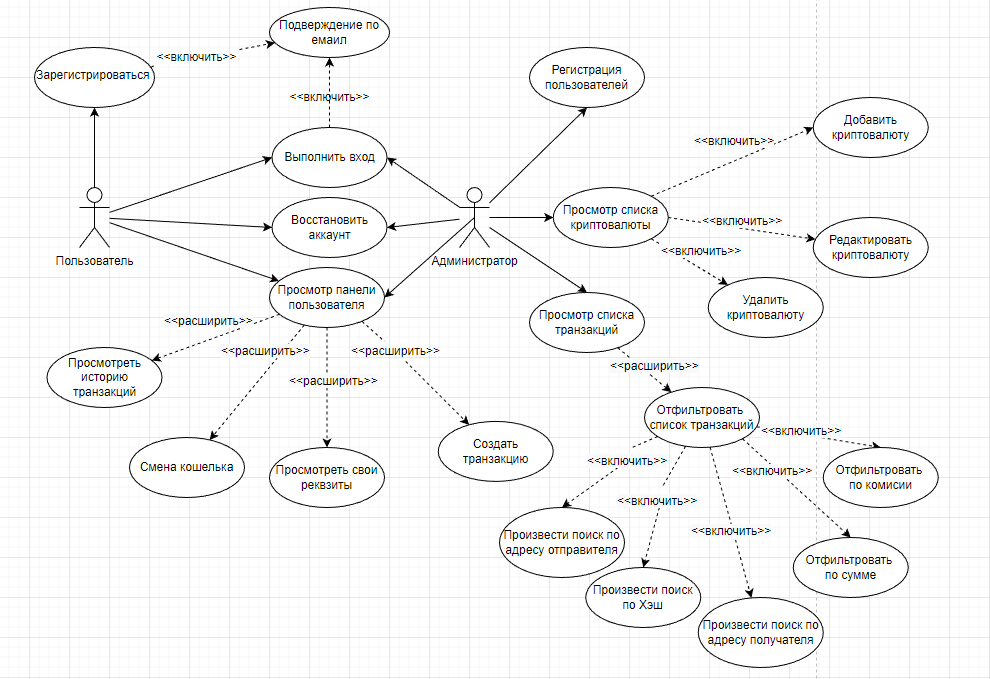
# **Список литературы**

1. Microsoft Visual Studio [Электронный ресурс] – https://ru.wikipedia.org/wiki/Microsoft\_Visual\_Studio – Дата доступа 23.02.2022
2. Полное руководство по языку программирования C# 7.0 и платформе .NET 4.7. Режим доступа: https://metanit.com/sharp/tutorial/ – Дата доступа: 14.03.2022
3. Пацей, Н.В. Курс лекций по языку программирования C# / Н. В. Пацей. – Минск: БГТУ, 2018. – 175 с.
4. Руководство по WPF // [Электронный ресурс]. – Режим доступа: https://metanit.com/sharp/wpf/ – Дата доступа: 28.03.2022
5. Работа с Entity Framework Core [Электронный ресурс] – https://professorweb.ru/my/entity-framework/6/level1/ – Дата доступа 03.05.2022
6. Блинова, Е.А. Курс лекций по Базам данным / Е.А. Блинова. – Минск: БГТУ, 2019. – 175 с.

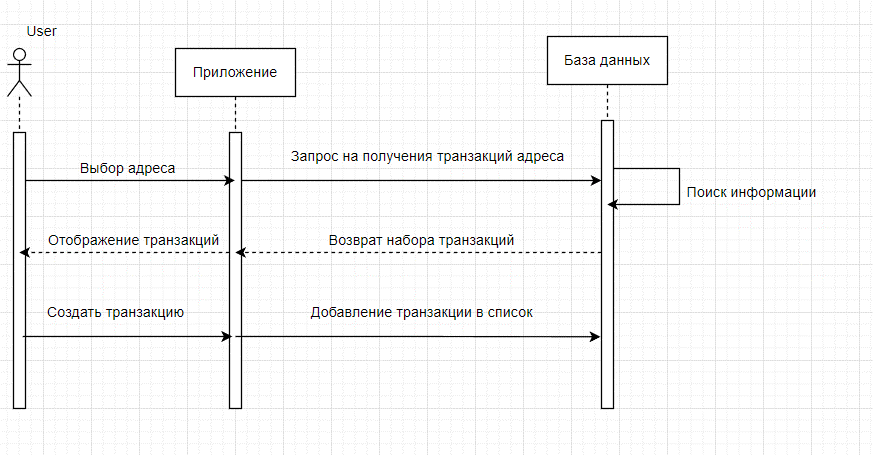
# **ПРИЛОЖЕНИЕ А**



# **ПРИЛОЖЕНИЕ Б**

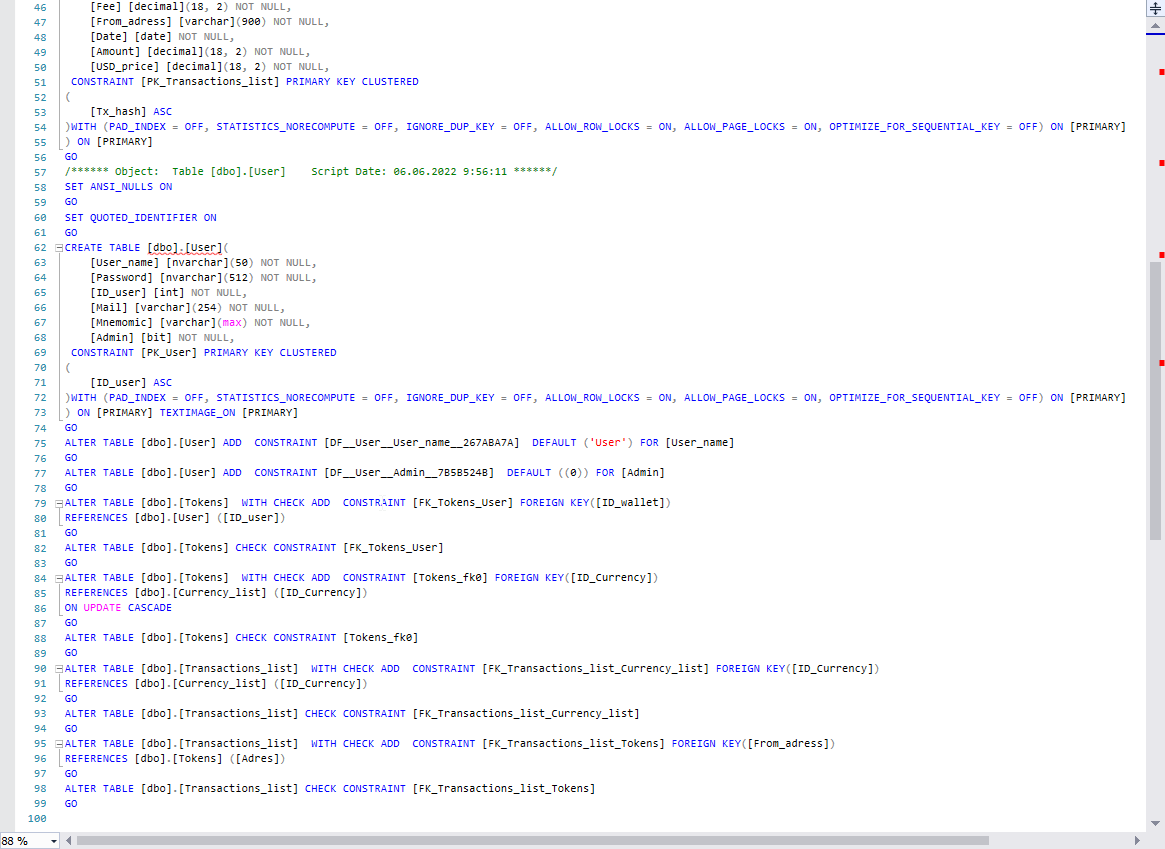


# **ПРИЛОЖЕНИЕ В**



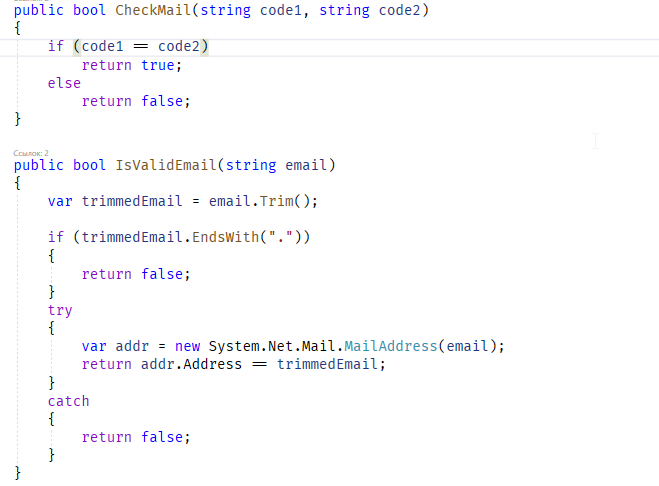
# **ПРИЛОЖЕНИЕ Г**

Скрипт для создания базы данных

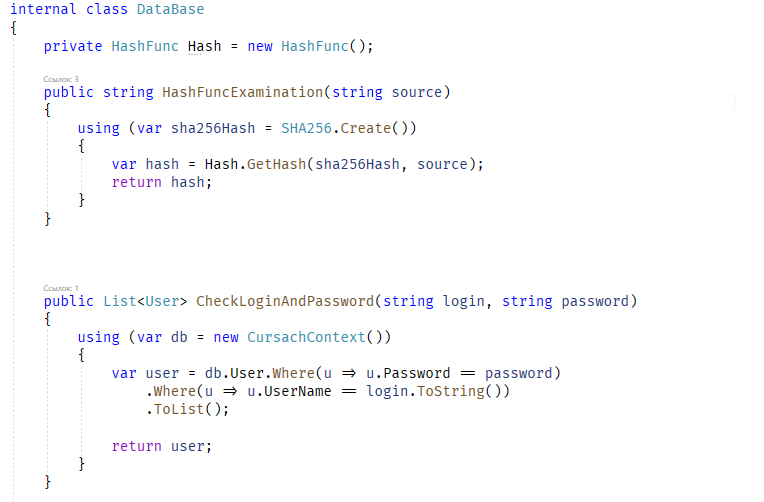


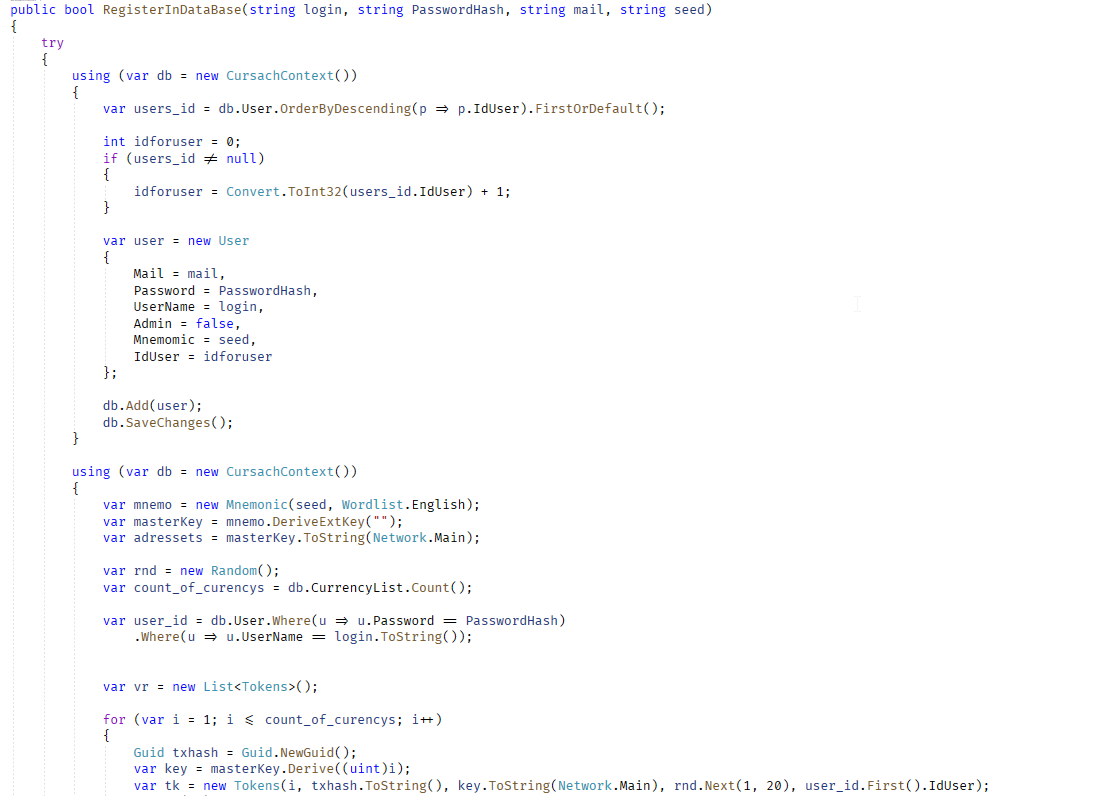
Листинг 1: Класс AuthenticationService

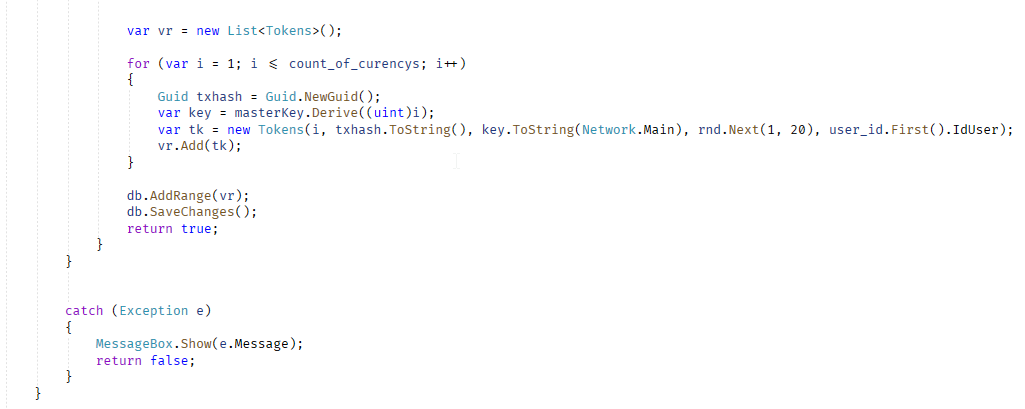




Листинг 2: Класс Database

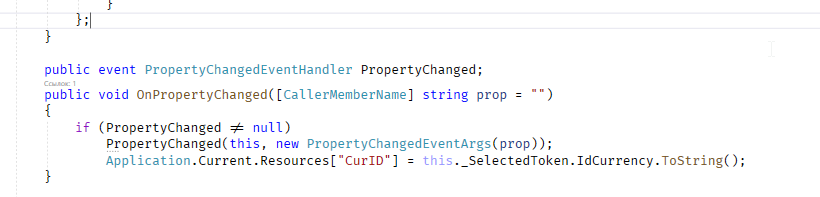






Листинг 3: Класс WalletViewModel

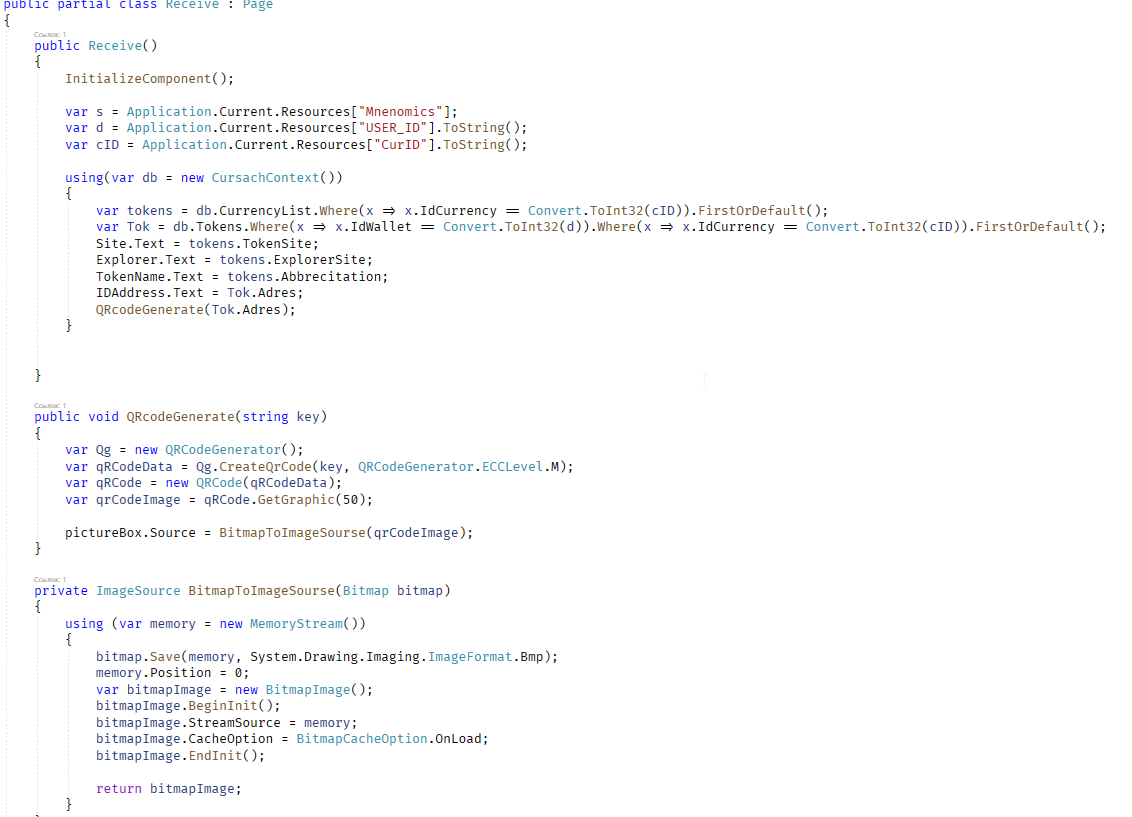


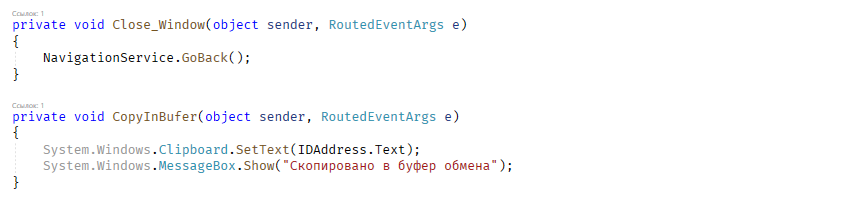


Листинг 4: Класс TransactionsViewModel

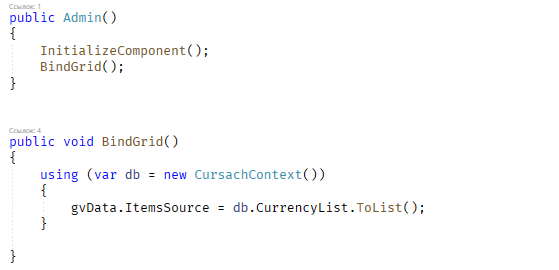


Листинг 5: Класс Receive



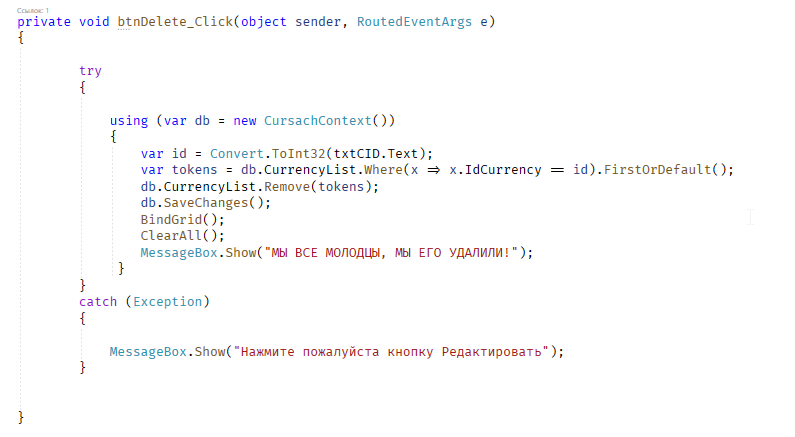


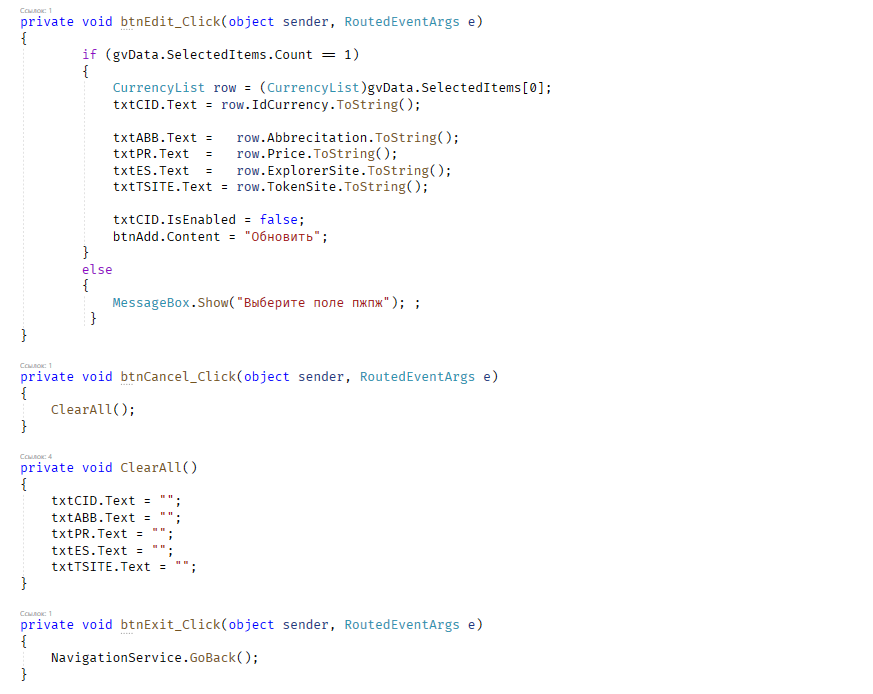
Листинг 6: Класс Admin



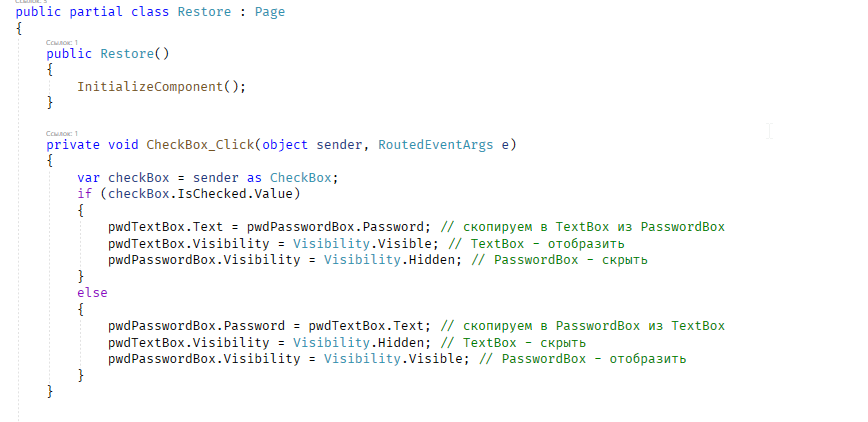


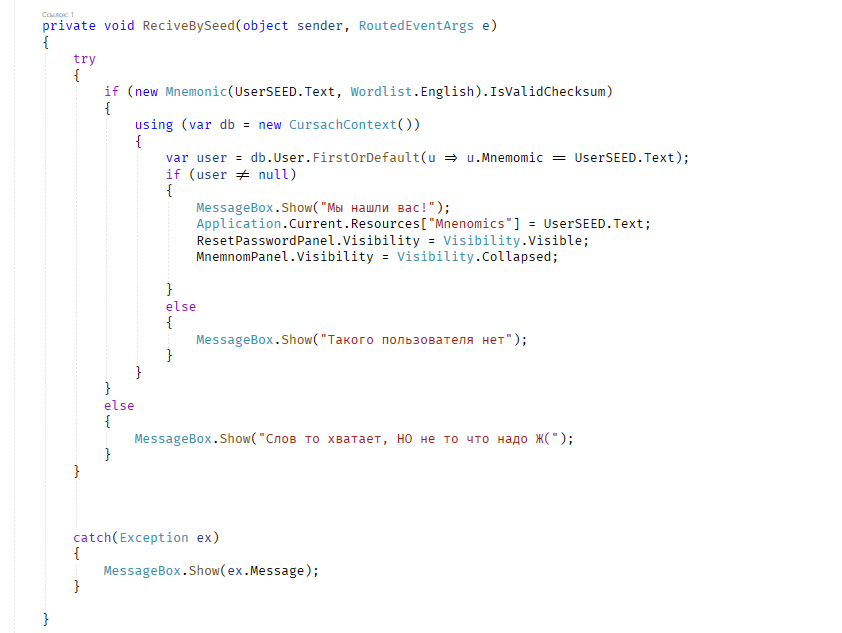






Листинг 7: Класс Restore







Листинг 8: Класс AdminSearch

