МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ ТАТАРСТАН

Государственное автономное профессиональное образовательное учреждение «Казанский колледж технологии и дизайна»

|  |
| --- |
| К ЗАЩИТЕ ДОПУЩЕН |
| Зам. директора по УР |
| \_\_\_\_\_\_\_Ф.М. Мухарлямова |
| «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

Разработка информационной системы «Криптокошелек»

ВКР 09.02.07 ДО 3А/20

|  |  |
| --- | --- |
| Руководитель проекта | Харитонова О.К. |
| Студент | Аксанов Д.Н. |

2024

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ ТАТАРСТАН

Государственное автономное профессиональное образовательное учреждение «Казанский колледж технологии и дизайна»

|  |
| --- |
| УТВЕРЖДАЮ  Зам. директора по учебной работе (зав.отделением)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

ЗАДАНИЕ НА ВКР (ДИПЛОМНУЮ РАБОТУ)

Студенту \_Карасеву Григорию Аркадьевичу\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Специальность 09.02.07 «Информационные системы и программирование»

Тема ВКР (дипломной работы) Разработка информационной системы «Криптокошелек» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Срок окончания ВКР (дипломной работы) \_01.06.2024\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Перечень вопросов, подлежащих разработке

1. Системотехнологическая реализация
2. Общая характеристика
3. Характеристика предприятия
4. Анализ предметной области
5. Постановка цели
6. Обоснование выбора СУБД
7. Обоснование среды разработки и языка программирования
8. Детальное проектирование
9. Контрольный пример
10. Компоненты, использованные в программе
11. Описание программы
12. Запуск информационной системы
13. Расчёт затрат на создание программного продукта
14. Расчет цены предложения и минимального количества копий тиражирования
15. Мероприятия по охране труда и технике безопасности
16. Мероприятия по технике безопасности

Специальное задание

Презентация ВКР (дипломной работы) в программе POWERPOINT

Руководитель ВКР (дипломной работы) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Харитонова О.К.

Рассмотрено методической цикловой комиссией: Математических, общепрофессиональных и дисциплин информационных систем

Протокол №5 \_от «17» мая 2024 г.

Председатель МЦК \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Харитонова О.К.

Дата выдачи дипломного задания «17» января 2024 г.

Содержание

[Введение 6](#_Toc168157726)

[Глава 1. Системотехнологическая реализация 8](#_Toc168157727)

[1.1 Общая характеристика 8](#_Toc168157728)

[1.2 Характеристика предприятия 9](#_Toc168157729)

[1.3 Анализ предметной области 11](#_Toc168157730)

[1.4 Постановка цели 12](#_Toc168157731)

[1.5 Детальное проектирование 19](#_Toc168157732)

[1.6 Контрольный пример 24](#_Toc168157733)

[1.5 Компоненты, использованные в программе 31](#_Toc168157734)

[1.7 Описание программы 35](#_Toc168157735)

[1.8 Запуск информационной системы 45](#_Toc168157736)

[1.9 Расчёт затрат на создание программного продукта 54](#_Toc168157737)

[Глава 2. Мероприятия по охране труда и технике безопасности 63](#_Toc168157738)

[2.1 Мероприятия по технике безопасности 63](#_Toc168157739)

[Заключение 65](#_Toc168157740)

[Библиографический список 67](#_Toc168157741)

[Код приложения 68](#_Toc168157742)

# Введение

В современном мире криптовалюты стали неотъемлемой частью финансовой системы, предоставляя уникальные возможности для хранения и передачи цифровых активов. С развитием криптовалютных технологий их использование становится все более распространенным, что подчеркивает важность безопасного и удобного хранения криптовалютных средств. В связи с этим разработка приложения криптокошелька представляет собой актуальную задачу, направленную на обеспечение надежного управления цифровыми активами.

Целью данной дипломной работы является разработка и реализация приложения криптокошелька, обеспечивающего удобный доступ к криптовалютным средствам, а также обеспечивающего высокий уровень безопасности и функциональности. В работе будут рассмотрены основные принципы работы криптокошельков, а также проведен анализ существующих решений в данной области.

Исследование и разработка приложения криптокошелька представляют собой важный шаг в обеспечении безопасного и удобного управления криптовалютными активами, что позволит пользователям эффективно управлять своими цифровыми финансами.

В разработку информационной системы входят следующие аспекты:

* управление каталогом товаров: создание, редактирование и удаление товаров, их описаний, изображений и цен;
* обработка заказов: прием заказов, обработку платежей, отслеживание статуса заказа и организацию пополнения;
* взаимодействие с клиентами: регистрацию пользователей, учет истории покупок, обратную связь, предоставление скидок и акций;
* аналитика и отчетность: сбор и анализ данных о продажах, поведении клиентов и эффективности маркетинговых кампаний

Таким образом, границы разработки охватывают все этапы процесса продаж компьютерной техники, начиная от представления товаров до обработки заказов и анализа эффективности бизнеса.

В современном мире информационные системы играют важную роль в различных сферах деятельности человека, в том числе и в бизнесе.

Информационная система – это взаимосвязанная совокупность средств, методов и персонала, используемых для хранения, обработки и выдачи информации в интересах достижения поставленной цели.

Ожидаемые результаты разработанной информационной системы по продаже криптовалют включают автоматизацию процессов продаж, улучшение обслуживания клиентов, эффективное управление финансами и предоставление аналитики о продажах. Область применения системы включает только онлайн, розничные и оптовые компании по продаже компьютерной техники.

# Глава 1. Системотехнологическая реализация

## 1.1 Общая характеристика

Криптокошелек (или цифровой кошелёк) представляет собой программное или аппаратное средство для хранения, отправки и получения криптовалюты. Он обеспечивает пользователя доступом к их криптовалютным активам, позволяя управлять различными видами криптовалют в одном месте. Криптокошельки могут быть разделены на горячие (онлайн) и холодные (оффлайн) в зависимости от их подключения к интернету. Горячие кошельки обеспечивают более легкий доступ к активам для проведения транзакций, тогда как холодные кошельки обеспечивают более высокий уровень безопасности за счет отсутствия подключения к интернету.

Криптокошельки обычно используются для хранения приватных ключей, которые необходимы для подписи транзакций и подтверждения владения криптовалютой. Они также обеспечивают доступ к публичным адресам для получения криптовалютных платежей. В зависимости от типа криптокошелька, пользователи могут иметь доступ к различным функциям, таким как просмотр баланса, создание новых адресов, выполнение транзакций и просмотр истории операций.

Основные характеристики криптокошельков включают безопасность, удобство использования, мультивалютность и возможность интеграции с другими сервисами. Они играют важную роль в экосистеме криптовалют, обеспечивая пользователям средства для управления своими цифровыми активами.

## 1.2 Характеристика предприятия

Характеристика предприятия, специализирующегося на криптокалюты, может включать следующие аспекты:

1. Направление деятельности:

предприятие по продаже криптовалюты является ключевым игроком на рынке потребительской валюты и информационных технологий. Оно предоставляет широкий выбор валют, необходимых как для повседневного использования, так и для профессиональной деятельности. Это включает в себя не только самые новые валюты, но так же, доллар, рубль и евро.

2.Тип предприятия

Существует несколько типов предприятий, занимающихся криптокошельками. Вот некоторые из них:

1. Централизованные криптокошельки: Это криптокошельки, управляемые централизованной компанией или сервисом. Пользователи хранят свои криптовалюты на серверах этой компании, что означает, что компания имеет контроль над доступом к средствам пользователей.

2. Децентрализованные криптокошельки: Эти криптокошельки позволяют пользователям иметь полный контроль над своими средствами. Приватные ключи хранятся непосредственно на устройствах пользователей, что обеспечивает большую степень безопасности.

3. Хардварные криптокошельки: Это физические устройства, специально разработанные для хранения криптовалюты. Они обеспечивают высокий уровень безопасности за счет изоляции приватных ключей от интернета.

4. Мобильные криптокошельки: Это приложения для мобильных устройств, которые позволяют пользователям управлять своими криптовалютными средствами через смартфон или планшет.

5. Веб-кошельки: Это онлайн-кошельки, доступные через веб-браузер. Пользователи могут получить доступ к своим средствам через интернет из любого устройства.

6. Мультивалютные криптокошельки: Эти кошельки поддерживают хранение различных криптовалют в одном месте, что удобно для пользователей, у которых есть различные активы.

Клиентская база:

* клиентами предприятия могут быть как обычные потребители, так и малые и средние предприятия, образовательные учреждения, государственные организации и другие компании, которые нуждаются в приобретении криптовалют. Каждый клиент получает индивидуальный подход и качественное обслуживание.

1. Услуги:

* в дополнение к продаже валют предприятие может предоставлять ряд дополнительных услуг, таких как консультации по выбору валют, также обучение пользователей основам работы с приложением.

1. Маркетинговые стратегии:

* предприятие может использовать различные маркетинговые стратегии для привлечения новых клиентов и удержания существующих. Это может включать в себя рекламные кампании в СМИ и интернете, акции и специальные предложения для покупателей, а также программы лояльности и бонусные системы.

## 1.3 Анализ предметной области

Анализ предметной области криптокошельков включает в себя изучение основных принципов и технологий, лежащих в их основе, а также анализ существующих решений на рынке. Вот несколько ключевых аспектов анализа предметной области:

1. Типы криптокошельков: Исследование различных типов криптокошельков, таких как холодные (offline) кошельки, горячие (online) кошельки, мобильные кошельки, бумажные кошельки и аппаратные кошельки. Каждый тип имеет свои преимущества и недостатки, которые важно учитывать при разработке приложения.

2. Безопасность: Анализ методов обеспечения безопасности криптокошельков, включая механизмы шифрования, многофакторную аутентификацию, использование хранилищ с открытым и закрытым ключом и защиту от кибератак.

3. Удобство использования: Оценка удобства использования криптокошельков для конечных пользователей, включая интерфейс пользователя, процесс регистрации и входа, управление активами и выполнение транзакций.

4. Мультивалютность: Рассмотрение возможности работы с различными криптовалютами в одном приложении, обеспечивая поддержку различных блокчейн-активов.

5. Интеграция с другими сервисами: Изучение возможностей интеграции криптокошелька с биржами криптовалют, системами оплаты и другими финансовыми сервисами.

Анализ предметной области позволит выявить требования к функциональности и безопасности приложения криптокошелька, а также определить основные преимущества и недостатки существующих решений на рынке.

1.4 Постановка цели

Создать инновационное приложение для управления финансами, которое предоставит пользователям удобный и безопасный способ отслеживать свои расходы, планировать бюджет и инвестировать средства. Цель проекта заключается в разработке интуитивно понятного интерфейса, обеспечивающего быстрый доступ к финансовой информации, а также в создании функционала для автоматического анализа трат и предоставления персонализированных рекомендаций по экономии и инвестированию.

Для получения необходимой информации в приложении, информационная система должна быть написана на языке программирования C# и языка разметки XAML, позволяющий вводить данные и получать необходимые для разработчика и пользователя результат.

Такая постановка цели позволяет четко определить направление исследования, его значимость и ожидаемые результаты. Она также указывает на то, какие аспекты рынка криптовалют будут рассмотрены в работе и какие выводы и рекомендации будут сделаны на основе анализа.

Для реализации поставленной цели надо решить следующие задачи:

* изучить предметную область;
* выбрать систему управления базами данных (СУБД);
* выбрать среду разработки;
* реализовать информационно-логическую модель с использованием СУБД (создать Базу Данных по информационно-логической модели);
* реализовать информационную систему с использованием среды разработки.

#### 1.4.1 Обоснование выбора СУБД

Выбор СУБД для продажи компьютерной техники зависит от ряда факторов, включая размер бизнеса, требования к производительности, безопасности, стоимости и масштабируемости.

SQL Server Management Studio (SSMS) — это интегрированная среда для управления любой инфраструктурой SQL, от SQL Server до баз данных SQL Azure. SSMS предоставляет средства для настройки, наблюдения и администрирования экземпляров SQL Server и баз данных. С помощью SSMS можно развертывать, отслеживать и обновлять компоненты уровня данных, используемые вашими приложениями, а также создавать запросы и скрипты.

MySQL - это уже среда, которая использует язык запросов SQL, и не имеет графического интерфейса. Управление осуществляется через командную строку и имеет свой синтаксис, а также реализует модель "клиент-сервер", ибо ваши БД (даже если вы используете один компьютер, т. е. локально) находятся на виртуальном сервере.

Основные достоинства SSMS:

– СУБД масштабируется, поэтому работать с ней можно на портативных ПК или мощной мультипроцессорной технике. Процессор может одновременно обрабатывать большой объем запросов;

– размер страниц – до 8 кб, поэтому данные извлекаются быстро, подробную и сложную информацию хранить удобнее. Система позволяет обрабатывать транзакции в интерактивном режиме, есть динамическая блокировка;

– рутинные административные задачи автоматизированы: это управление блокировками, памятью, редактура размеров файлов. У системы продуманы настройки, можно создать профили пользователей;

– реализован поиск по фразам, тексту, словам, можно создавать ключевые индексы;

– в SQL Server есть репликации через интернет, предусмотрена синхронизация. Есть полноценный веб-ассистент для форматирования страниц;

– в систему интегрирован сервер интерактивного анализа для принятия решений, создания корпоративных отчетов. Есть службы преобразования информации;

– СУБД поддерживает работу с другими продуктами Microsoft: Access, MS Excel.

Недостатки:

– зависимость от операционной среды: СУБД работает только с системой Windows.

Для создания базы данных магазина сувениров, который обычно не требует сложной структуры данных и высокой производительности, подходят более простые и бюджетные решения, такие как MS SQL.

#### 1.4.2 Обоснование среды разработки и языка программирования

Выбор языка программирования и среды разработки зависит от множества факторов, таких как тип проекта, опыт команды разработчиков, требования к производительности, масштабируемости, поддержке и бюджету.

Microsoft Visual Studio — линейка продуктов компании Microsoft, включающих интегрированную среду разработки (IDE) программного обеспечения и ряд других инструментов. Данные продукты позволяют разрабатывать как консольные приложения, так и игры и приложения с графическим интерфейсом, в том числе с поддержкой технологии Windows Forms, UWP а также веб-сайты, веб-приложения, веб-службы как в родном, так и в управляемом кодах для всех платформ, поддерживаемых Windows, Windows Mobile, Windows CE, .NET Framework, .NET Core, .NET, MAUI, Xbox, Windows Phone .NET Compact Framework и Silverlight. После покупки компании Xamarin корпорацией Microsoft появилась возможность разработки IOS и Android программ.

Visual Studio включает в себя редактор исходного кода с поддержкой технологии IntelliSense и возможностью простейшего рефакторинга кода. Встроенный отладчик может работать как отладчик уровня исходного кода, так и отладчик машинного уровня. Остальные встраиваемые инструменты включают в себя редактор форм для упрощения создания графического интерфейса приложения, веб-редактор, дизайнер классов и дизайнер схемы базы данных. Visual Studio позволяет создавать и подключать сторонние дополнения (плагины) для расширения функциональности практически на каждом уровне, включая добавление поддержки систем контроля версий исходного кода (как, например, Subversion и Visual SourceSafe), добавление новых наборов инструментов (например, для редактирования и визуального проектирования кода на предметно-ориентированных языках программирования) или инструментов для прочих аспектов процесса разработки программного обеспечения (например, клиент Team Explorer для работы с Team Foundation Server).

Достоинства Visual Studio:

* встроенный Web-сервер. Для обслуживания Web-приложения ASP.NET необходим Web-сервер, который будет ожидать Web-запросы и обрабатывать соответствующие страницы. Наличие в Visual Studio интегрированного Web-сервера позволяет запускать Web-сайт прямо из среды проектирования, а также повышает безопасность, исключая вероятность получения доступа к тестовомуWeb-сайту с какого-нибудь внешнего компьютера, поскольку тестовый сервер может принимать соединения только с локального компьютера;
* поддержка множества языков при разработке. Visual Studio позволяет писать код на своем языке или любых других предпочитаемых языках, используя все время один и тот же интерфейс (IDE). Более того, Visual Studio также еще позволяет создавать Web-страницы на разных языках, но помещать их все в одно и то же Web-приложение. Единственным ограничением является то, что в каждой Web-странице можно использовать только какой-то один язык (очевидно, что в противном случае проблем при компиляции было бы просто не избежать);
* меньше кода для написания. Для создания большинства приложений требуется приличное количество стандартного стереотипного кода, и Web-страницы ASP. NET тому не исключение. Например, добавление Web-элемента управления, присоединение обработчиков событий и корректировка форматирования требует установки в разметке страницы ряда деталей. В Visual Studio такие детали устанавливаются автоматически;
* интуитивный стиль кодирования. По умолчанию Visual Studio форматирует код по мере его ввода, автоматически вставляя необходимые отступы и применяя цветовое кодирование для выделения элементов типа комментариев. Такие незначительные отличия делают код более удобным для чтения и менее подверженным ошибкам. Применяемые Visual Studio автоматически параметры форматирования можно даже настраивать, что очень удобно в случаях, когда разработчик предпочитает другой стиль размещения скобок;
* более высокая скорость разработки. Многие из функциональных возможностей Visual Studio направлены на то, чтобы помогать разработчику делать свою работу как можно быстрее. Удобные функции, вроде функции IntelliSense (которая умеет перехватывать ошибки и предлагать правильные варианты), функции поиска и замены (которая позволяет отыскивать ключевые слова как в одном файле, так и во всем проекте) и функции автоматического добавления и удаления комментариев (которая может временно скрывать блоки кода), позволяют разработчику работать быстро и эффективно;
* возможности отладки. Предлагаемые в Visual Studio инструменты отладки являются наилучшим средством для отслеживания загадочных ошибок и диагностирования странного поведения. Разработчик может выполнять свой код по строке за раз, устанавливать интеллектуальные точки прерывания, при желании сохраняя их для использования в будущем, и в любое время просматривать текущую информацию из памяти;
* Visual Studio также имеет и множество других функций: возможность управления проектом, встроенная функция управления исходным кодом, возможность рефакторизации кода, мощная модель расширяемости. Более того, в случае использования Visual Studio 2008 Team System разработчик получает расширенные возможности для модульного тестирования, совместной работы и управления версиями кода, что значительно больше того, что предлагается в более простых инструментах вроде Visual SourceSafe..

В качестве недостатка можно отметить невозможность отладчика Microsoft Visual Studio Debugger отслеживать в коде режима ядра. Отладка в Windows в режиме ядра в общем случае выполняется при использовании WinDbg, KD или SoftICE.

С# – один из наиболее быстро растущих, востребованных и при этом «удобных» языков программирования. Это модификация фундаментального языка С от компании Microsoft, призванная создать наиболее универсальное средство для разработки программного обеспечения для большого количества устройств и операционных систем.

Основные преимущества языка:

* С# популярен за счет своей «простоты». Простоты для современных программистов и больших команд разработчиков, чтобы те могли в сжатые сроки создавать функциональные и производительные приложения. Этому способствуют нетипичные конструкции языка и специфичный синтаксис, помогающий максимально органично реализовать намеченные функции;
* популярность языка – еще одно значимое преимущество. Большое количество поклонников C# способствуют его развитию. Также это благоприятно влияет на рост числа вакансий, связанных с разработкой на языке Microsoft. Программисты, хорошо знакомые с С#, востребованы в индустрии, несмотря на их большое и постоянно увеличивающееся количество;
* понятный синтаксис C# заметно упрощает не только разработку как таковую, но и другие важные аспекты совместной работы, например, чтение чужого кода. Это упрощает процесс рефакторинга и исправления ошибок при работе над приложениями в больших командах;

Недостатки:

* меньше библиотек и фреймворков по сравнению с другими языками, такими как Python или JavaScript;
* может быть сложным для начинающих из-за своей объектно-ориентированной природы и необходимости понимания принципов работы с памятью;
* менее гибкий и выразительный по сравнению с некоторыми другими языками, такими как Ruby или Python;
* меньше сообщества и поддержки по сравнению с более популярными языками;
* может быть менее производительным, чем некоторые другие языки, такие как C или C++.

Язык C# был выбран из-за своей универсальности языка для разработки приложений на разных платформах и операционных системах.

## 1.5 Детальное проектирование

Для создания информационной модели нужно определить сущности, которые будут удовлетворять всем требования к базе данных (Таблица 1).

Таблица 1. Список сущностей предметной области

|  |  |
| --- | --- |
| Наименование сущностей | Краткое описание |
| Страна | Страна пользователя |
| Валюты | Информация о валютах |
| Роль | Информация о ролях |
| Юзер | Пароль и логин пользователя |
| Кошелек | Описание информации о кошельке |
| Пол | Выбор пола |

Для каждой сущности определим атрибуты, которые будут хранится в базе данных:

Сущность «Страна»:

* Id\_Country (первичный ключ);
* Country;
* PhotoCountry

Сущность «Валюты»:

* Id\_Currency (первичный ключ);
* Currency
* PriceInRub
* CurrencyPhoto

Сущность «Роль»:

* Id\_role (первичный ключ);
* Наименование роли.

Сущность «Пользователь»:

* Id\_User (первичный ключ);
* Login;
* Password;
* NickName
* Email
* PhoneNumber
* Id\_Sex
* Id\_Country
* Id\_role (внешний ключ, связанный с таблицей «Роль»).

Сущность «Кошелек»:

* Id\_Wallet (первичный ключ);
* Id\_Currency
* Id\_User
* Balance

Сущность «Пол»:

* Id\_Sex (первичный ключ);
* Sex.

ER-диаграмма базы данных информационной системы для расчёта потребления газа (Рисунок 1) выглядит следующим образом:

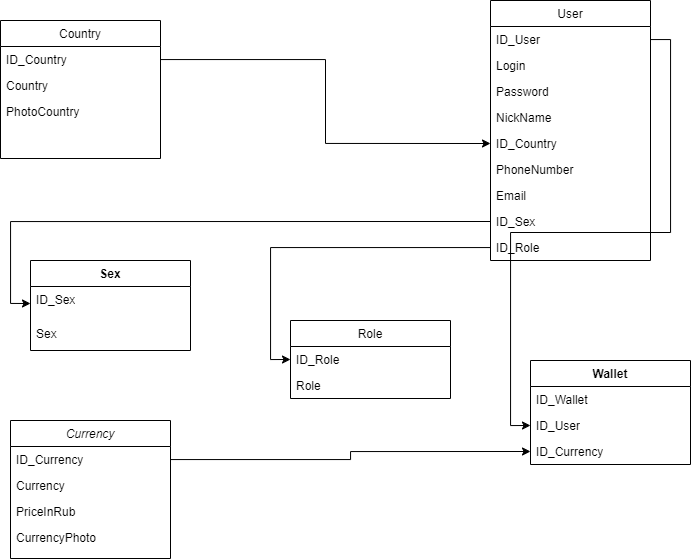


Рисунок 1 - ER-Диаграмма

Информационная модель базы данных (Рисунок 2) выглядит следующим образом:

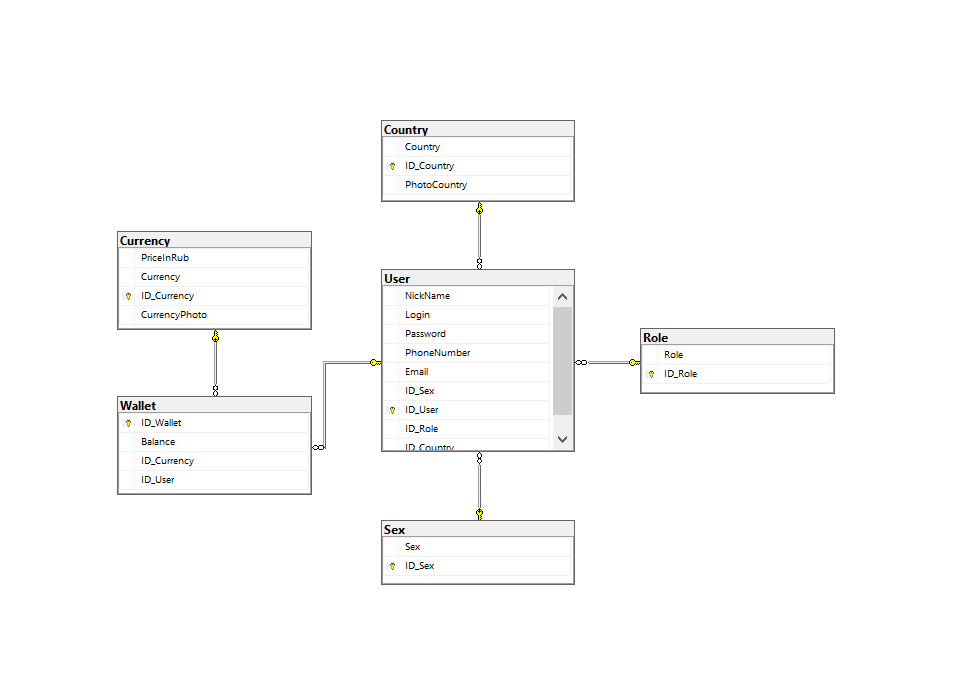


Рисунок 2 - Схема базы данных «Продажа компьютерной техники»

Составим проекты таблиц и заполним таблицы (Таблицы 2-9)

Таблица 2. Свойства таблицы «User»

|  |  |  |
| --- | --- | --- |
| Имя столбца | Тип данных | Разрешенить значение Null |
| Id\_User | Int | Unchecked |
| NickName | Nvarchar(50) | Unchecked |
| Login | Nvarchar(50) | Unchecked |
| Password | Nvarchar(50) | Сhecked |
| Email | Nvarchar(4) | Unchecked |
| PhoneNumber | Nvarchar(6) | Unchecked |
| Id\_Sex | Int | Unchecked |
| Id\_User | Int | Unchecked | |
| Id\_Role | int | Unchecked | |

Таблица 3. Свойство таблицы «Currency»

|  |  |  |
| --- | --- | --- |
| Имя столбца | Тип данных | Разрешить значения Null |
| Id\_Currency | Int | Unchecked |
| PriceInRub | Nvarchar(50) | Unchecked |
| Currency | Nvarchar(50) | Unchecked |
| CurrencyPhoto | Nvarchar(50) | Unchecked |

Таблица 4. Свойство таблицы «Role»

|  |  |  |
| --- | --- | --- |
| Имя столбца | Тип данных | Разрешить значения Null |
| Id\_Role | Int | Unchecked |
| Role | Nvarchar(50) | Checked |

Таблица 5. Свойство таблицы «Wallet»

|  |  |  |
| --- | --- | --- |
| Имя столбца | Тип данных | Разрешить значения Null |
| Id\_Wallet | Int | Unchecked |
| Id\_Currency | int | Unchecked |
| Balance | Nvarchar(50) | Unchecked |
| Id\_role | Int | Unchecked |

Таблица 6. Свойство таблицы «Sex»

|  |  |  |
| --- | --- | --- |
| Имя столбца | Тип данных | Разрешить значения Null |
| Id\_Sex | Int | Unchecked |
| Sex | Nvarchar(50) | Unchecked |

Таблица 7. Свойство таблицы «Country»

|  |  |  |
| --- | --- | --- |
| Имя столбца | Тип данных | Разрешить значение Null |
| Id\_Country | int | Unchecked |
| Country | Nvarchar(50) | Unchecked |

## 1.6 Контрольный пример

Контрольный пример играет ключевую роль в разработке программного обеспечения, поскольку он представляет собой основной инструмент для проверки корректности работы программы. Этот документ детально описывает ожидаемое поведение программы при обработке определенных входных данных. Обычно контрольные примеры разрабатываются на этапе проектирования и тесно связаны с требованиями к программе.

Один из важных аспектов контрольного примера заключается в том, чтобы он включал разнообразные сценарии использования программы, чтобы покрыть как можно больше возможных случаев. Это позволяет обнаружить и исправить ошибки и недочеты в программе до ее выпуска.

При создании контрольного примера необходимо учитывать не только правильность результатов, но и эффективность программы, ее скорость работы и обработку ошибок. Этот документ также может служить важным руководством для тестировщиков, помогая им оценить работоспособность программы в различных сценариях использования.

Более того, контрольный пример может быть использован как часть документации к программе, поскольку он предоставляет примеры работы и исходные данные, что помогает пользователям лучше понять, как использовать программу и какие результаты ожидать. В целом, контрольный пример является неотъемлемой частью процесса разработки программного обеспечения, обеспечивая его качество и надежность.

Входные данные для контрольного пример представлены в таблице 10:

Таблица 10. Входные данные контрольного примера «User»

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Id\_user | nickname | Login | Password | phonenumber | Balance | Currency | Id\_sex | email | Id\_role | Id\_cou |
| 1 | NeoThaGeek | Qwerty | Qwerty000 | 79991695388 | 50 | Рубль | 1 | [NeoThaGeek@gmail.com](mailto:NeoThaGeek@gmail.com) | 2 | 1 |
| 2 | Snoop Dogg | Smoke420 | 420Smoke | 79169161616 | 235 | Доллар | 1 | [Smoke420@icloud.com](mailto:Smoke420@icloud.com) | 1 | 2 |
| 3 | Nicki Minaj | CodeinCrazy | CrazzzyCodein | 79098080706 | 421 | Биткоин | 2 | [CodeinCrazy@yahoo.com](mailto:CodeinCrazy@yahoo.com) | 1 | 3 |
| 4 | Rico Nasty | NastyAss | NastyBaby999 | 79888088808 | 700 | Евро | 2 | [NastyAss@yandex.ru](mailto:NastyAss@yandex.ru) | 1 | 4 |
| 5 | Coi Leray | cCoi!L3RaY | c00!lerAAy | 79011019669 | 20 | Эфириум | 2 | [cCoi!L3RaY@mail.com](mailto:cCoi!L3RaY@mail.com) | 1 | 5 |

Продолжение таблицы 10

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Id\_user | nickname | Login | Password | phonenumber | Balance | Currency | Id\_SEx | email | Id\_role | Id\_coutry |
| 6 | Drake | DrakeOVO | Qwerty000 | 79991695388 | 50 | Рубль | 1 | [DrakeOVO@richpapi.com](mailto:DrakeOVO@richpapi.com) | 1 | 9 |
| 7 | Playboi Carti | F3!NF3!n | 420Smoke | 79169161616 | 235 | Доллар | 1 | [F3!NF3!n@swag.wrld](mailto:F3!NF3!n@swag.wrld) | 1 | 10 |
| 8 | Chief Keef | BigGSosaBaby | CrazzzyCodein | 79098080706 | 421 | Биткоин | 1 | [BigGSosaBaby@chiraq.usa](mailto:BigGSosaBaby@chiraq.usa) | 1 | 4 |
| 9 | Kid Cudi | MM00N | NastyBaby999 | 79888088808 | 700 | Евро | 1 | [MM00N@cudi.kid](mailto:MM00N@cudi.kid) | 1 | 6 |
| 10 | Rihanna | WorkWorkWork | c00!lerAAy | 79011019669 | 20 | Эфириум | 2 | [ririwork@riri.ri](mailto:ririwork@riri.ri) | 1 | 8 |

Таблица 11. Входные данные контрольного примера «Пол»

|  |  |
| --- | --- |
| Id\_Sex | Sex |
| 1 | Мужской |
| 2 | Женский |

Таблица 12. Входные данные контрольного примера «Роль»

|  |  |
| --- | --- |
| Id\_Role | Role |
| 1 | Админ |
| 2 | Клиент |

Таблица 13. Входные данные контрольного примера «Country»

|  |  |  |
| --- | --- | --- |
| Country | ID\_country | PhotoCountries |
| Belarussia | 1 | belarus.png |
| Kazakhstan | 2 | kazakhstan.png |
| USA | 3 | usa.png |
| United Kingdom | 4 | uk.png |
| Spain | 5 | spain.png |

Таблица 14. Входные данные контрольного примера «Wallet»

|  |  |  |  |
| --- | --- | --- | --- |
| id\_wallet | Id\_user | Id\_currency | Id\_country |
| 1 | 2 | 1 | 1 |
| 2 | 2 | 2 | 2 |
| 3 | 3 | 1 | 3 |
| 4 | 4 | 3 | 4 |
| 5 | 2 | 3 | 6 |
| 6 | 3 | 1 | 11 |
| 7 | 2 | 2 | 12 |
| 8 | 3 | 1 | 13 |
| 9 | 4 | 3 | 14 |
| 10 | 5 | 2 | 15 |
| 11 | 6 | 1 | 16 |
| 12 | 2 | 1 | 21 |
| 13 | 2 | 2 | 22 |
| 14 | 3 | 1 | 23 |
| 15 | 4 | 3 | 24 |
| 16 | 5 | 2 | 25 |
| 17 | 6 | 1 | 26 |
| 18 | 6 | 1 | 38 |

Таблица 16. Входные данные контрольного примера «Currency»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| PriceInRub | Currency | ID\_Currency | | CurrencyPhoto |
| 1 | Рубль | Рубль | | Рубль.png |
| 88 | Доллар | | Доллар | Доллар.png |
| 6150300 | Биткоин | | Биткоин | Биткоин.png |
| 96 | Евро | | Евро | Евро.png |
| 350000 | Эфириум | | Эфириум | Эфириум.png |

## 1.5 Компоненты, использованные в программе

Элементы управления представляют собой визуальные классы, которые получают введенные пользователем данные и могут инициировать разные события.

При создании информационной системы для магаизна сувениров были использованы следующие компоненты:

– TextBlock;

– Button;

– ComboBox;

– Frame;

– Page;

– PasswordBox;

– Grid;

– ListView;

– Windows;

– DataGrid;

– GridRowDefinitions;

– WrapPanel;

– TextBox.Background;

– ImageBrush;

– TextBox;

– StackPanel.

1. ComboBox - Он хранит коллекцию объектов ComboBoxItem, которые создаются явным или неявным образом. Как и ListBoxItem, ComboBoxItem является элементом управления содержимым, который может хранить любой вложенный элемент.
2. Grid - это контейнер, который позволяет размещать элементы управления в сетке с определенным количеством столбцов и строк.
3. GridRowDefinitions - это определение строки в Grid, которое позволяет задавать свойства строки, такие как высота и ширина.
4. Button - это базовый компонент пользовательского интерфейса, который может включать простое содержимое, например текст, а может — сложное, например изображения и элементы управления Panel.
5. TextBlock - Элемент управления обеспечивает гибкую поддержку текста для сценариев пользовательского интерфейса, требующих не более одного абзаца текста. Он поддерживает ряд свойств, которые обеспечивают точный контроль над отображением текста например: FontFamily, FontSize, FontWeight, TextEffects и TextWrapping.
6. ListView - это список, который отображает список элементов.
7. Window - это основное окно приложения, которое содержит все остальные элементы.
8. Frame - это компонент, который позволяет перемещаться между страницами, созданными в проекте. Он позволяет создать более сложные структуры окон.
9. Page – это компонент страница, в котором можно создавать другие компоненты.
10. PasswordBox - Элемент управления, которым можно использовать для сбора паролей или других конфиденциальных данных, например номеров карт социального страхования. Дополнительные сведения можно найти в статье об элементах управления текстом.
11. DataGrid – это элемент графического управления, который представляет табличное представление данных. Типичное представление сетки также поддерживает некоторые или все из следующего: Щелкните заголовок столбца, чтобы изменить порядок сортировки сетки.
12. TextBox - Предоставляет элемент управления "текстовое поле".
13. TextBox.Background – Устанавливает задний фон для элемента TextBox.
14. ImageBrush – Используется для закрашивания области с изображением.
15. StackPanel – Это элемент управления для упрощения размещения элементов на экране.

В данной информационной системе использовались стили для оформления приложения.

Стили WPF позволяют определить внешний вид и поведение элементов управления без необходимости изменения их исходного кода. Это дает возможность легко изменять внешний вид приложения без необходимости переписывать весь код.

Стили могут быть определены на уровне окна, контрола, шаблона или триггера. Они могут включать в себя такие свойства, как цвет, шрифт, размер, радиус углов, прозрачность и многое другое.

Существует несколько типов стилей: ключевые, триггерные, шаблоны данных и шаблоны элементов.

Ключевые стили определяются на уровне окна или контрола и применяются ко всем элементам этого типа. Триггерные стили используются для управления поведением элементов в зависимости от определенных условий. Шаблоны данных применяются к коллекциям данных и позволяют управлять отображением каждого элемента в коллекции. Шаблоны элементов используются для определения внешнего вида отдельного элемента управления.

Для каждого из этих компонентов были разработаны соответствующие пользовательские интерфейсы, которые обеспечивают удобство использования программы и интуитивно понятное взаимодействие с пользователем.

* 1. Описание пользовательского интерфейса

Интерфейс информационной системы построен из кнопок, панелей и таблиц в них выводиться (кроме кнопок) информация из базы данных.

Одним из главных преимуществ среды разработки Microsoft Visual Studio является простой и удобный интерфейс. Все элементы управления помещены слева в меню, с помощью поиска можно найти нужный компонент.

В данной информационной системе будут находиться элементы:

«Главное меню» - при помощи его пользователь выберет в какой радел ему нужно войти;

От оформления пользовательского интерфейса зависит, насколько легким будет процесс изучения и дальнейшей работы с информационной системой. Основные принципы оформления пользовательского интерфейса:

– простота;

– логичность;

– эстетичность.

Простота. Данный параметр имеет очень важное значение в создание информационной системы, поскольку излишняя сложность затрудняет изучение изучения приложения, что, следовательно, приводит к непроизводительности данного этапа.

Логичность. Оформление интерфейса пользователя для пользователя должен быть логичным. В результате непоследовательного оформления приложения будет выглядеть беспорядочно и хаотично, что приведет к нелогичности работы информационной системы.

Эстетичность. Параметр, который влияет на привлечение аудитории. Эстетичность нужна для нахождения человека готовый как минимум провести время в данной информационной системе и как максимум готовый купить ее. Эстетичность – наведения красоты приложения для массивности аудитории, которые используют приложение.

## 1.7 Описание программы

Программа Wallet.exe для сотрудников, занимающихся продажей или покупкой криптовалюты, написана на языке C# и взаимодействует с базой данных SQL Server.

Таблица 17 содержит подробное описание всех модулей системы. В этой таблице перечисляются все процедуры, которые входят в каждый модуль, а также даётся детальное описание всех действий, выполняемых в каждой из этих процедур. Каждый модуль рассматривается отдельно, с указанием всех его функциональных компонентов и шагов, предпринимаемых каждой процедурой. Таким образом, таблица 17 служит всеобъемлющим справочником, предоставляющим полное представление о внутренней структуре модулей, их процедурах и выполняемых ими действиях, что позволяет глубоко понять функционирование каждого модуля в системе.

Таблица 17. Описание процедур

|  |  |
| --- | --- |
| Процедуры | Описание процедуры |
| MainWindow.xaml | |
| Window | Основное окно программы. |
| Grid | Контейнер для размещения элементов управления. |
| Image | Изображение логотипа |
| Frame «mainFrame» | Фрейм для отображения содержимого страниц. |
| Button «Back» | Кнопка «Назад» для перехода на предыдущую страницу. |
| LoginPage.xaml «Авторизация» | |
| Page | Страница «Авторизация» |

Продолжение таблицы 17

|  |  |
| --- | --- |
| Grid | Контейнер для размещения элементов управления. |
| StackPanel | Панель для размещения элементов в горизонтальном положении в нижней части окна. |
| TextBlock «Логин» | Текстовое поле «Логин». |
| TextBlock «Пароль» | Текстовое поле «Пароль». |
| TextBox «LoginBox | Поле для ввода логина пользователя. |
| PasswordBox «PasswordBox» | Поле для ввода пароля пользователя. |
| Button «Войти» | Кнопка для выполнения входа в систему. |
| UsersPage.xaml «Users» |  |
| Page | Страница «Users» |
| Grid | Контейнер для размещения элементов управления. |
| Button «BtnAddSot» | Кнпока для перехода на страницу добавления клиентов. |
| Button «BtnDelSot» | Кнопка для удаления клиента. |
| Button «BtnCur» | Кнопка для перехода на страницу валют |
| ListBox «ListUser» | Отображает список пользователей. |
| StackPanel | Панель для размещения элементов в вертикальном положение. |
| Border | Создает границу вокруг другого элемента. |
| Textblock | Контейнер для ввода данных. |
| UserPage.xaml «Пользователи» |  |
| Page | Страница «Пользователи». |

Продолжение таблицы 17

|  |  |
| --- | --- |
| Grid | Контейнер для размещения элементов управления. |
| StackPanel | Панель для размещения элементов в вертикальном положения. |
| Button «BtnZakaz» | Кнопка для перехода на страницу Wallet. |
| ListView «ListMenu» | Отображает список валют. |
| WrapPanel | Располагает элементы последовательно, переходя на новую строку или столбец. |
| Border | Создает границу вокруг другого элемента. |
| Image | Изображение изображения стран. |
| ComboBox «Cmbkateg» | Поле со списком сортировки по категории. |
| TextBox «TxtSearch» | Текстовое окно для ввода в поиске. |
| WalletPage.xaml «Wallet» |  |
| Page | Страница «Wallet» |
| Grid | Контейнер для размещения элементов управления. |
| StackPanel | Панель для размещения элементов в горизонтальном положении. |
| ComboBox «CmbStatus» | Поле со списком сортировки по статусу. |
| Button «BtnAdd» | Кнопка для перехода на страницу добавления заказа. |
| ListBox «ListUK» | Отображает список всех заказов. |

Продолжение таблицы 17

|  |  |
| --- | --- |
| Border | Создает границу вокруг другого элемента управления. |
| TextBlock | Контейнер для вывода данных. |
| Button «BtnRed» | Кнопка для перехода на страницу редактирования заказа. |
| AddEditPage.xaml «Добавление клиента» |
| Page | Страница добавления сотрудника. |
| WrapPanel | Располагает элементы последовательно, переходя на новую строку или столбец. |
| Border | Создает границу вокруг другого элемента. |
| Image | Изображение изображения товаров. |
| ComboBox «Cmbkateg» | Поле со списком сортировки по категории. |
| TextBox «TxtSearch» | Текстовое окно для ввода в поиске. |
| ZakazPage.xaml «Заказы» |  |
| Page | Страница «Заказы» |
| Grid | Контейнер для размещения элементов управления. |
| StackPanel | Панель для размещения элементов в горизонтальном положении. |
| ComboBox «CmbStatus» | Поле со списком сортировки по статусу. |
| Button «BtnAdd» | Кнопка для перехода на страницу добавления заказа. |
| ListBox «ListUK» | Отображает список всех заказов. |

Продолжение таблицы 17

|  |  |
| --- | --- |
| TextBlock «Currency» | Текстовый блок с заголовком поля «Currency» |
| TextBlock «Country» | Текстовый блок с заголовком поля «Country» |
| TextBlock «Wallet» | Текстовый блок с заголовком поля «Wallet» |
| TextBlock «ID\_User» | Текстовый блок с заголовком поля «ID\_User» |
| TextBox «txtKod | Поле для ввода кода клиента. |
| TextBox txt1 | Поле для ввода Фамилии. |
| TextBox txt2 | Поле для ввода Имени. |
| TextBox txt3 | Поле для ввода Отчества. |
| TextBox txt4 | Поле для ввода серии паспорта. |
| TextBox txt5 | Поле для ввода номера паспорта. |
| TextBox txt7 | Поле для ввода улицы. |
| TextBox txt9 | Поле для ввода дома. |
| TextBox txt10 | Поле для ввода квартиры. |
| TextBox txtKodd | Поле для ввода кода юзера. |
| ImageBrush | Позволяет использовать изображение в качестве кисти для закрашивания. |
| ComboBox «CmbRole» | Поле со списком ролей. |
| Button «btnSave» | Кнопка сохранения новых данных и переход на страницу Users. |
| AddEditPage.xaml «Добавление заказов» |
| Page | Страница добавления заказов. |

Продолжение таблицы 17

|  |  |
| --- | --- |
| Grid | Контейнер для размещения элементов управления. |
| TextBlock «Добавление заказа» | Текстовый блок с заголовком поля «Добавление заказа» |
| Текстовый блок с заголовком поля «Код клиента» | Текстовый блок с заголовком поля «Код клиента» |
| Текстовый блок с заголовком поля «Клиент» | Текстовый блок с заголовком поля «Клиент» |
| Текстовый блок с заголовком поля «Статус» | Текстовый блок с заголовком поля «Статус» |
| Текстовый блок с заголовком поля «Wallet» | Текстовый блок с заголовком поля «Wallet» |
| TextBox | Поле для ввода кода заказа. |
| ComboBox «CmbSot» | Поле со списком сотрудников. |
| ComboBox «CmbStatus» | Поле со списком статусов. |
| ComboBox «CmbTovar» | Поле со списком товаров. |
| Button «BtnSave» | Кнопка для новых заказов и переход на страницу заказы. |

## 1.8 Запуск информационной системы

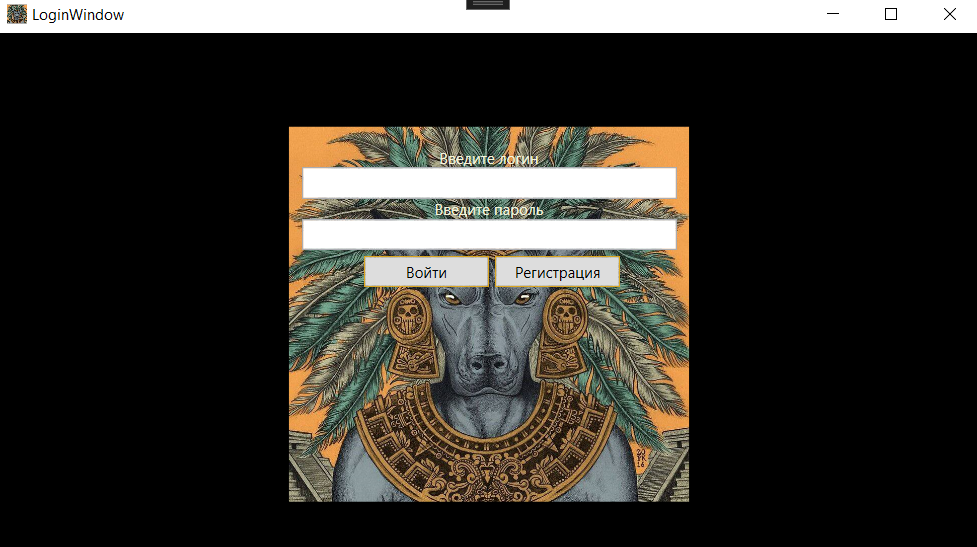
Для запуска и проведения работ потребуется компьютер или ноутбук с системными требованиями не ниже:

* Операционная система: Windows 7/10/11;
* HDD: 500 Gb;
* Процессор: intel core i3;
* Оперативная память (ОЗУ): 8 Gb;

Перед началом работы с системой следует:

* убедиться, что программно-аппаратное обеспечение соответствует требованиям;
* убедиться, что присутствует выход в интернет;
* запустить систему;
* пройти регистрацию в системе (при отсутствии логина и пароля);
* авторизоваться в системе.

После запуска приложения открывается главная страница авторизации пользователя (Рисунок 3)

 Рисунок 3 - Страница авторизации пользователя

В поля «Введите логин» и «Введите пароль» пользователь вводит свои данные для входа. (Рисунок 4)

C:\Users\dripper\Documents\Lightshot\Screenshot_19.png

Рисунок 4 - Кнопка «Войти»

Авторизация пользователя: Этот компонент позволяет пользователю войти в систему, используя свои учетные данные. Он обеспечивает безопасность данных пользователя и позволяет ему получить доступ к другим функциям программы.

C:\Users\dripper\Documents\Lightshot\Screenshot_24.png

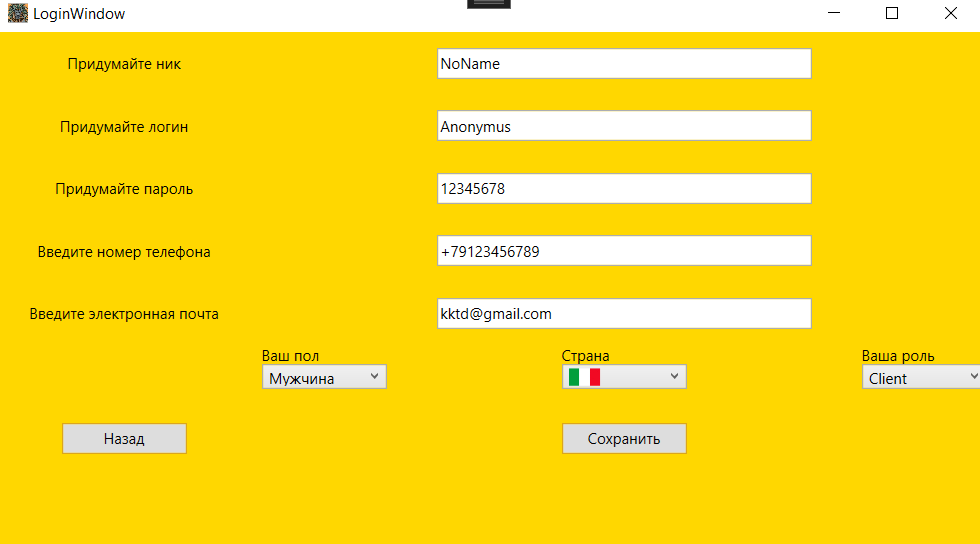
Рисунок 5. Кнопка «Регистрация»

Рисунок 7. Ввод регистрационных данных. После ввода нажимаем на кнопку

«Сохранить».

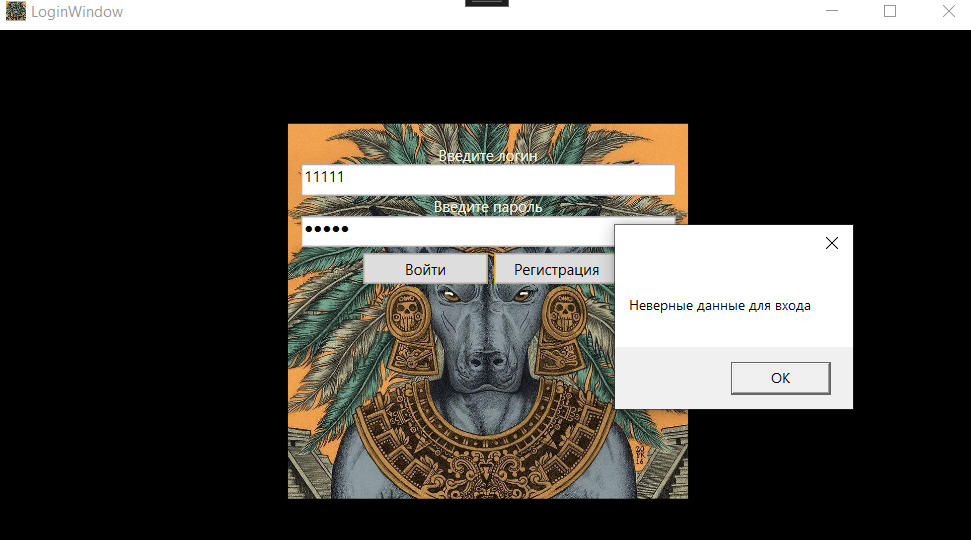


Рисунок 6. Попытка ввести пустые поля

Если данные введены не верно, появляется сообщение об ошибке (Рисунок 5).

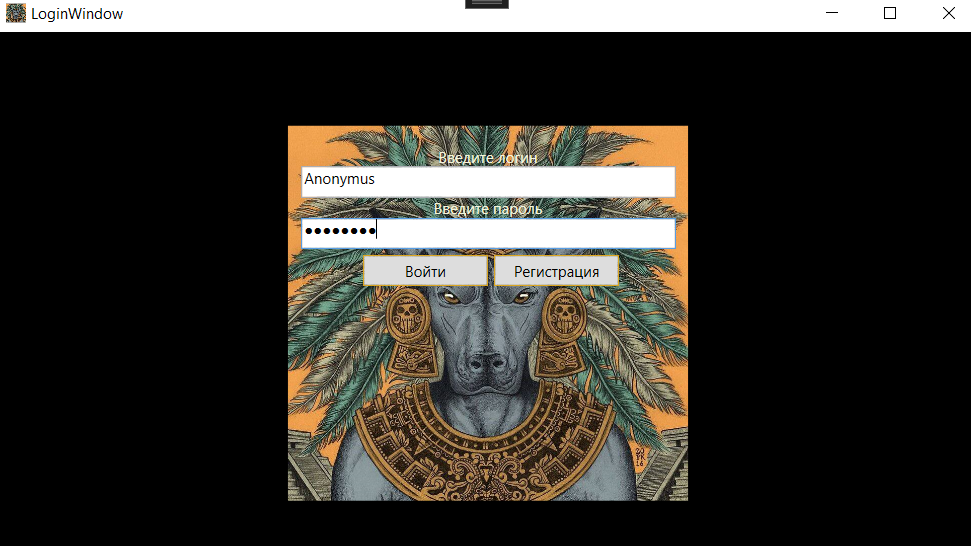


Рисунок 7. Вводим пароль и логин указанный при регистрации.

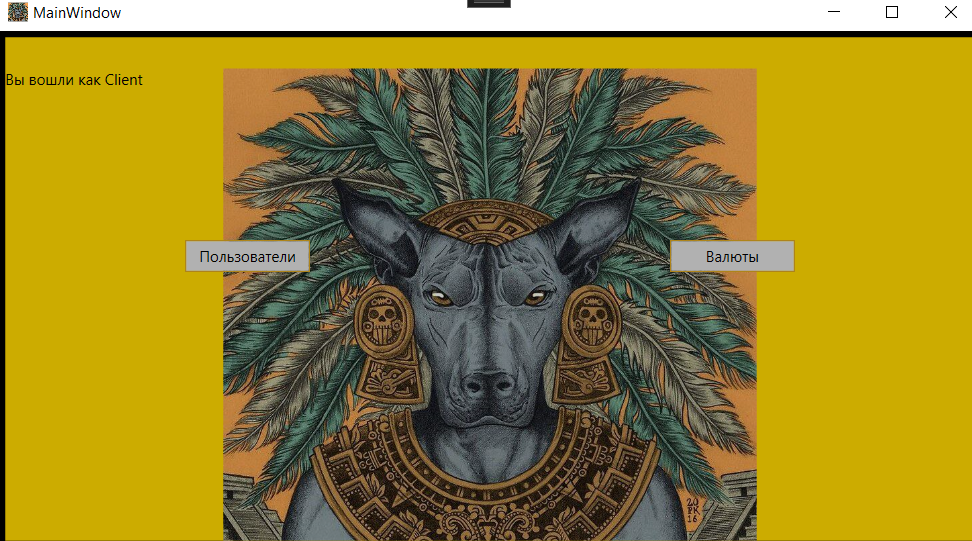


Рисунок 8. Открывается приложение, где представлен логотип и две кнопки.

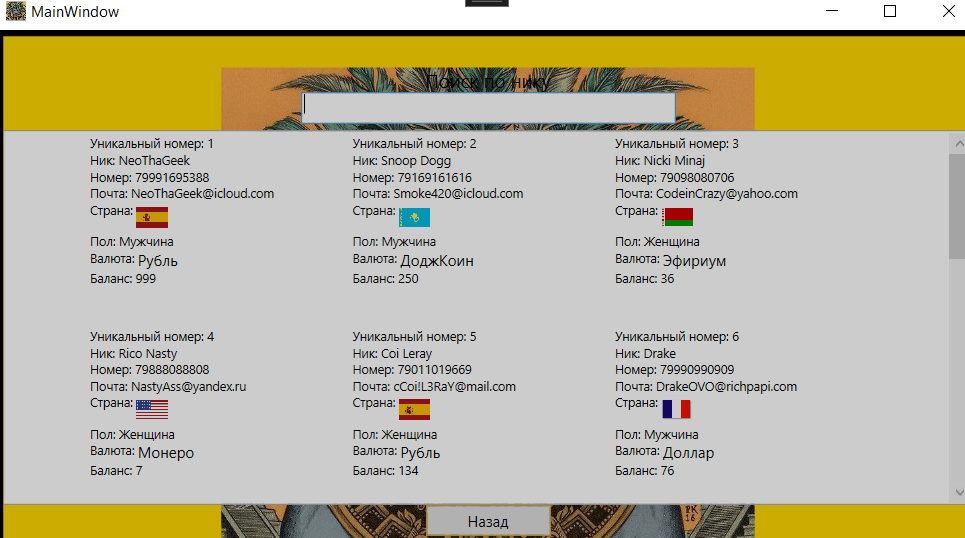
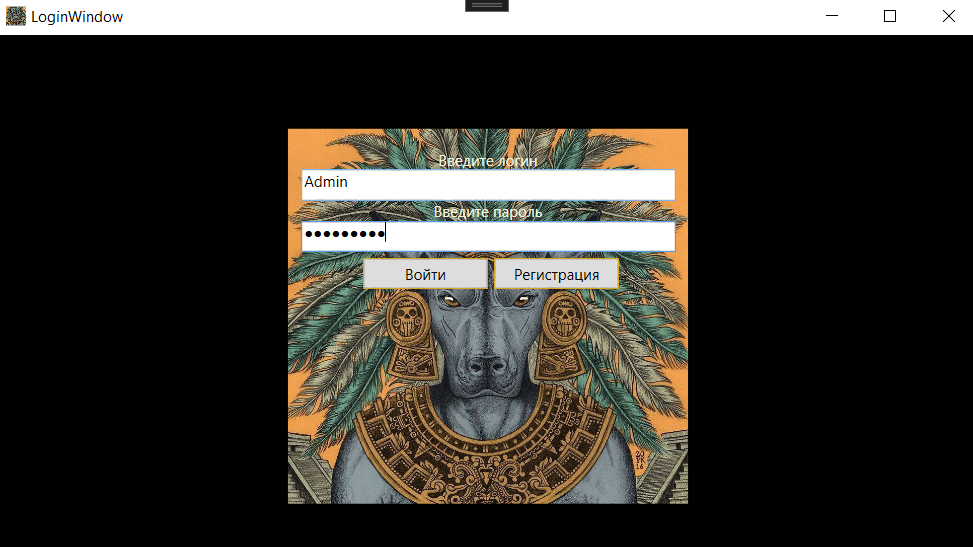


Рисунок 9. Результат нажатия на кнопку «Пользователи»

Здесь можно просмотреть зарегистрированных в системе пользователей.

C:\Users\dripper\Documents\Lightshot\Screenshot_23.png

Рисунок 10 - Кнопка назад



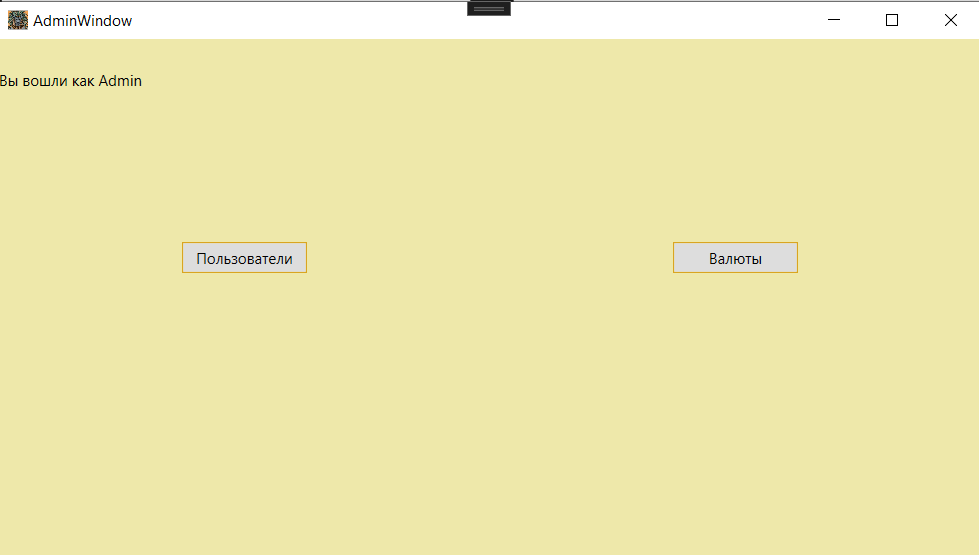
Рисунок 10. На странице авторизации в соответствующих полях вводим логин и пароль администратора, после чего нажимаем кнопку «Войти».

Рисунок 11. После входа в учётную запись администратора, показываются две кнопки. Нажимаем на кнопку «Пользователи»

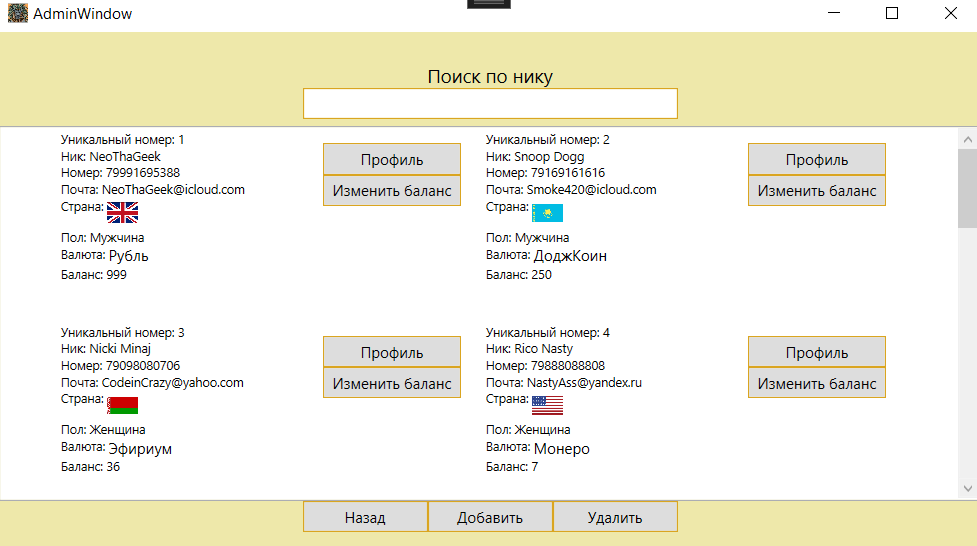


Рисунок 12. Здесь вашему вниманию представляются зарегистрированные и добавленные администратором пользователи.

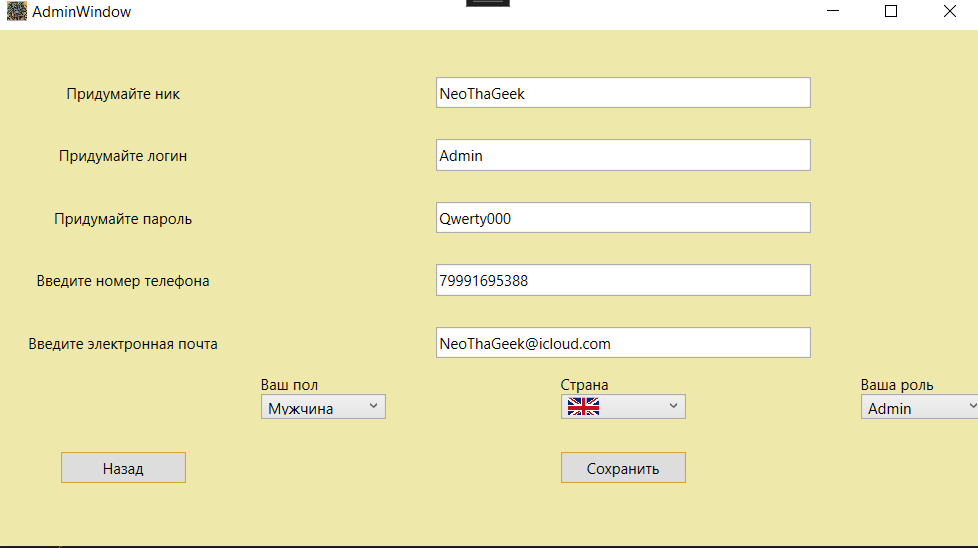


Рисунок 13. Результат нажатия на кнопку «Профиль»

Здесь администратор может редактировать данные пользователей.

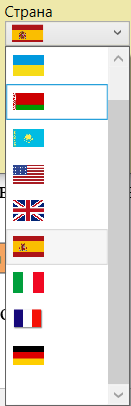


Рисунок 14. Выпадающий список со странами

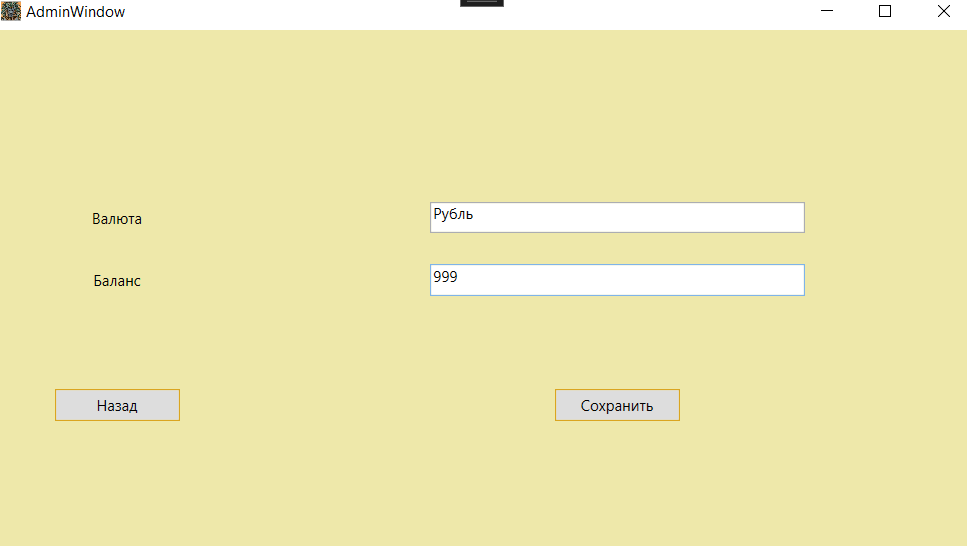


Рисунок 15. Результат нажатия на кнопку «Изменить баланс», здесь также можно редактировать валюту и баланс.

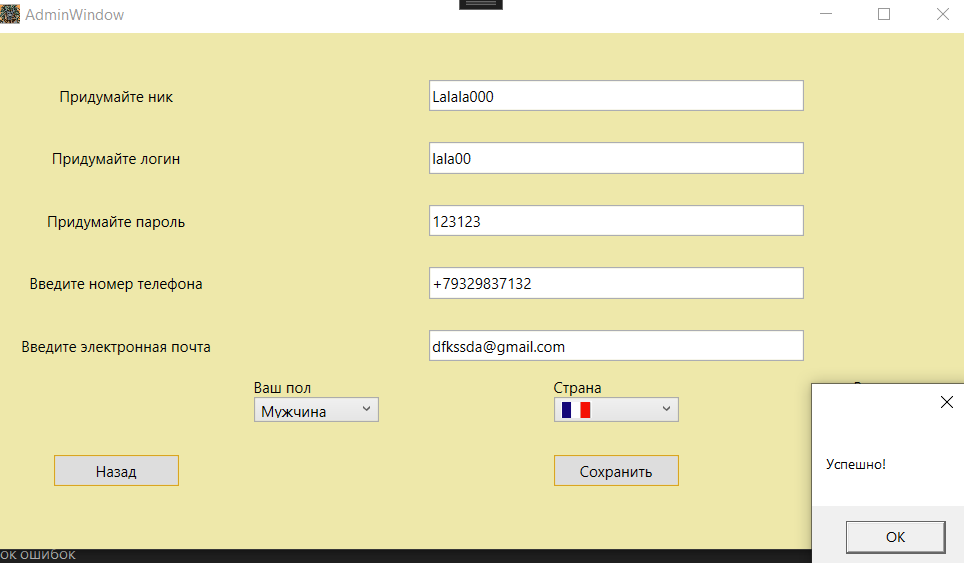


Рисунок 16. Добавление нового пользователя

C:\Users\dripper\Documents\Lightshot\Screenshot_22.png

Рисунок 17. Кнопка сохранить

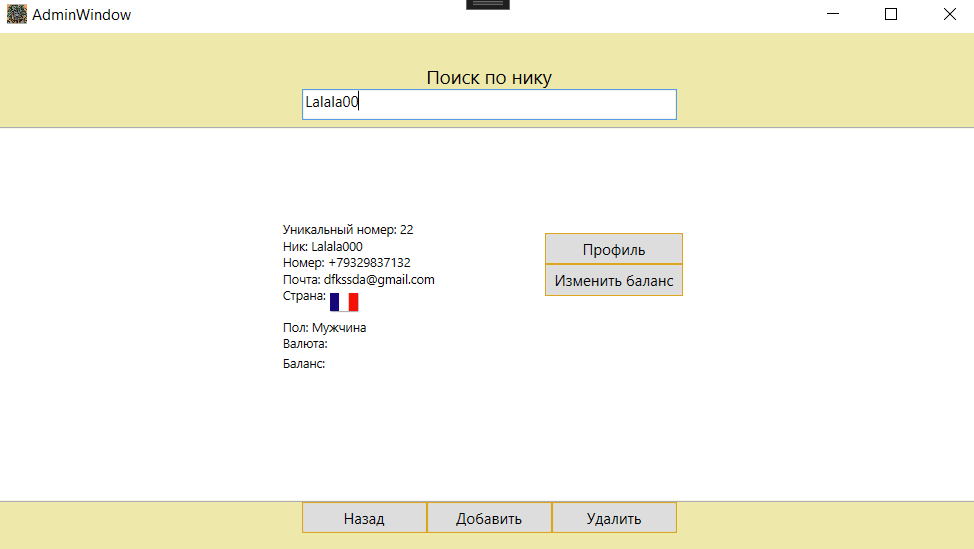


Рисунок 18. Поиск добавленного пользователя

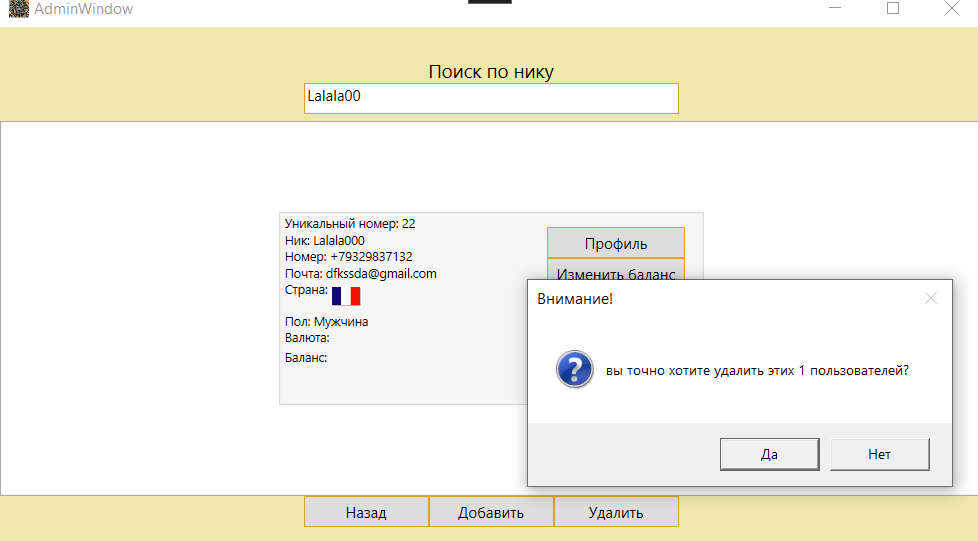


Рисунок 19. Удаление пользователя

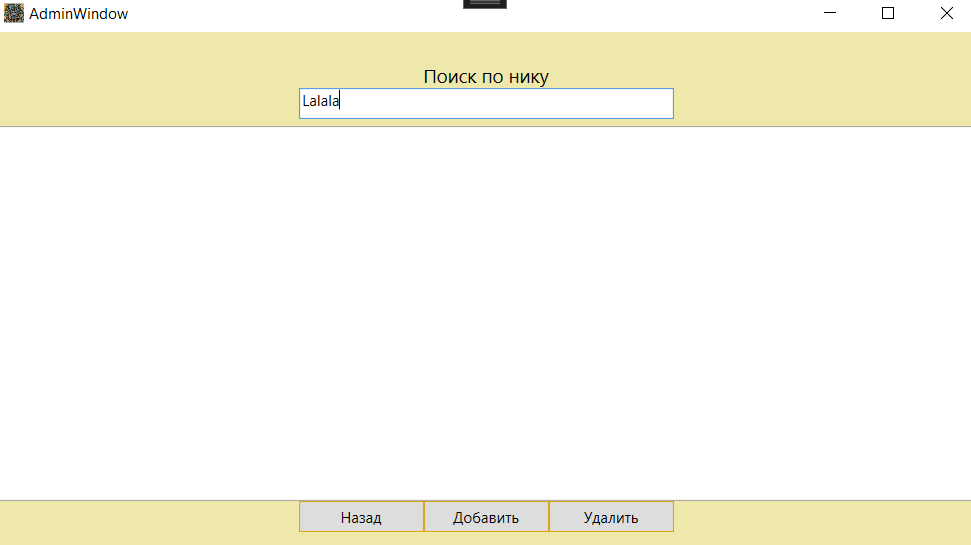


Рисунок 20. Поиск удаленного пользователя

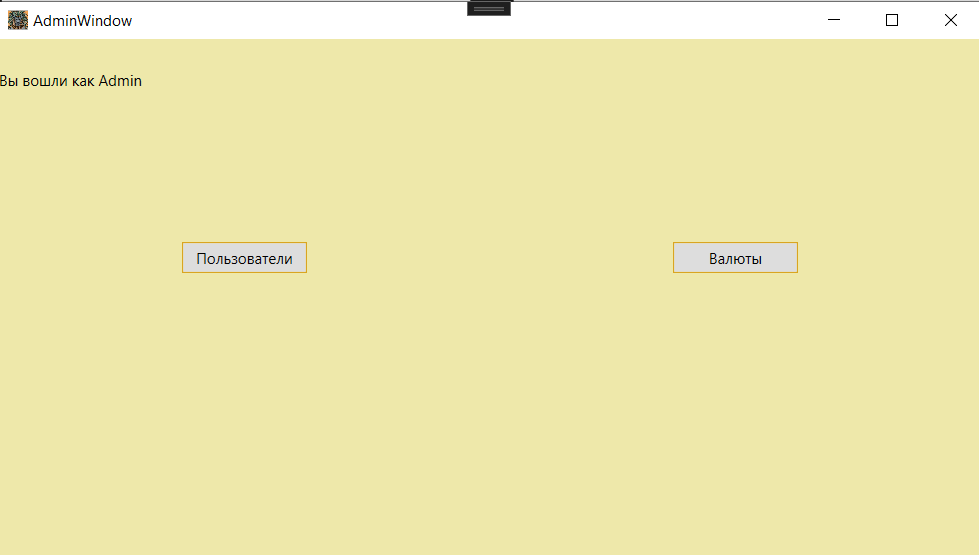


Рисунок 21. На главной странице есть кнопка «Валюты», при переходе можно ознакомиться с актуальными курсами валют.

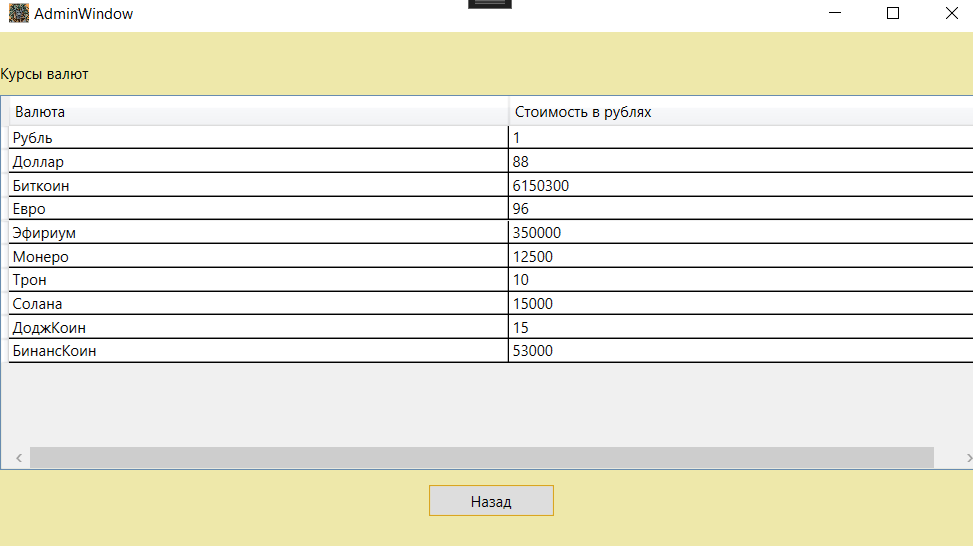


Рисунок 22. Результат нажатия кнопки «Валюты».

## Расчёт затрат на создание программного продукта

Расчет себестоимости машинного часа эксплуатации вычислительной и оргтехники (ВиОТ):

, (22)

где См.ч. – себестоимость машинного часа;

Зi – годовые затраты, связанные с эксплуатацией и обслуживанием ВиОТ,

Fп – годовой полезный фонд времени работы единицы оборудования

, (23)

где Fн = 1947 ч. – номинальный годовой фонд рабочего времени в 2024 г.,

αР = (0,05…0,2) – коэффициент, учитывающий время, затраченное на ремонт, настройки, обслуживание и пр.,

Кг = 0,95 – коэффициент готовности,

Fп = 1947 ∙ (1 – 0,07) = 1811 (ч.).

Расчет суммарных годовых затрат.

Для расчета годовых затрат, необходимо определить балансовую стоимость условного комплекта (таблица 18):

Таблица 18 – Состав условного комплекта

|  |  |  |  |
| --- | --- | --- | --- |
| № | Наименование | Кол-во, шт. | Цена, руб. |
| 1 | Монеро | 2 | 30000 |
| 2 | Эфириум | 1 | 10000 |
|  | Итого: | 70000 | |

, (24)

где РД – дополнительные расходы на доставку, установку, первоначальную наладку и пр.,

, (25)

РД = 0,1 ∙ (30000∙ 2 + 10000 ∙ 1) = 7000 (руб.),

СБАЛ = 30000∙2 + 10000 ∙1 + 7000 = 77000 (руб.).

Затраты на материалы:

, (26)

ЗМ = 0,02 ∙ 77000 =н1540 (руб.).

Основная и дополнительная заработная плата персонала, обслуживающего ВиОТ.

Предусмотрен 1 человек с окладом 47000 руб.

, (27)

где Оi – оклад i-го работника, обслуживающего ВиОТ,

Кд = 0,1 – коэффициент дополнительной заработной платы,

КР = 0,15 – районный коэффициент,

Начисления на заработную плату (таблица 19):

Таблица 19 – Состав единого страхового налога на 2024 г

|  |  |  |
| --- | --- | --- |
| № | Статья | % |
| 1 | Пенсионный фонд | 22 |
| 2 | Фонд социального страхования | 2,9 |
| 3 | Федеральный фонд обязательного медицинского страхования | 5,1 |
| 4 | Страховой налог от несчастного случая на производстве для служащих | 0,2 |
|  | Итого: | 30,2 |

Таким образом, начисление на заработную плату составляет 30,2%,

, (28)

НЗ/П = 0,302 ∙ 59455= 17955,41 (руб.).

Амортизационные отчисления ВиОТ.

, (29)

где  = 0,2 – норма амортизационных начислений.

АООБОР = 77000 ∙ 0,2 = 15400 (руб.).

Износ программных продуктов.

Условный комплект обладает необходимыми продуктами, представленными в (таблица 20):

Таблица 20 – Используемые программные средства

|  |  |
| --- | --- |
| Наименование | Цена, руб. |
| MS Windows 10 | 14000 |
| Visual Studio 2019 | 0 |
| Google Chrome | 0 |
| Microsoft SQL Server 2019 | 0 |
| Итого | 14000 |

Амортизационные отчисления программных продуктов – АОпп.

, (30)

где  = 0,5 – норма амортизационных начислений,

 = 14000 руб. – суммарная стоимость программных продуктов,

**** = 14000 ∙ 0,5 = 7000 (руб.).

Расходы на содержание и эксплуатацию оборудования.

Расходы на содержание и эксплуатацию оборудования состоят из:

* затраты на ремонт и специальное обслуживание

, (31)

ЗРЕМ = 0,03 ∙ 77000= 2310 (руб.),

* затраты на электрическую энергию

, (32)

где М = 1,055 кВт – суммарная мощность,

ЦкВт/ч = 3,68 руб. – стоимость кВт/ч,

КГ = 0,95 – коэффициент готовности,

ЗЭ = 1,055 ∙ 1947 ∙ 0,95 ∙ 3,68 = 7181,08 (руб.).

Расходы на содержание и эксплуатацию оборудования

 (33)

ЗРЭ = 2310 + 7181,08 = 9491,08 (руб.).

Прочие расходы.

Так как на объекте используется один тип техники и количество обслуживающего персонала не превышает трех человек, то

 , (34)

В нашем случае:

,

ЗПР = 0,5 ∙ 77000= 38500 (руб.).

Суммарные годовые затраты

, (35)

Зi = 1540 + 59455+ 17955,41 + 15400 + 7000 + 9491,08 + 38600=

=149441,49 (руб.).

Себестоимость машинного часа, из формулы (3.1.1), составляет:

(36)

#### Расчет цены предложения и минимального количества копий тиражирования

Фонд оплаты труда за время работы над программным продуктом.

 (37)

где О = 40000руб. – оклад, работающего над продуктом;

ТОБ = 3 месяца – общее время работы над программным продуктом;

Кд = 0,13 – коэффициент дополнительной заработной платы;

КР = 0,15 – районный коэффициент.

ФОТ = 40000 ∙ 3 ∙ (1 + 0,13) ∙ (1 + 0,15) = 155940 (руб.)

Начисления на ФОТ:

 (38)

НЗ/П = 155940∙ 0,302 = 47093,88 (руб.).

Затраты, связанные с эксплуатацией и обслуживанием ВиОТ:

, (39)

где ТМ = 3 месяца – машинное время работы над программным продуктом;

ЧРМ = 162 – число рабочих часов в месяце;

КИ = 0,9 – коэффициент использования оборудования;

N = 1 – количество условных комплектов.

ЗОВТ = 3 ∙ 162 ∙ 0,9 ∙ 1 ∙ 86,86 = 37992,56 (руб.).

Затраты на специальные программные продукты – ЗСПП.

Специальные программные продукты не использовались.

ЗСПП = 0 (руб.)

Затраты на хозяйственные операции и нужды – ЗХН (таблица 21):

Таблица 21 – Затраты на хозяйственные нужды

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование | Цена за единицу, руб. | Количество | Всего, руб. |
| Бумага | 500 | 2 | 1000 |
| Ручка | 20 | 4 | 80 |
| Картридж принтера | 1000 | 2 | 2000 |
| Итого: | | | 3080 |

ЗХН = 3080 (руб.).

Накладные расходы:

 (3.2.3)

РН = 0,4 ∙ 155940= 62376 (руб.).

Полные затраты на разработку программного продукта:

 (40)

ЗПОЛ = 155940+ 47093,88 + 37992,56 + 0 + 3080 + 62376 = 306412,44 (руб.).

Расчет установочной прибыли:

 (41)

где РУ = 20 % – установочная рентабельность.

ПРУ = 306412,44 ∙ 0,2 = 61282,49 (руб.).

Расчет величины налога на добавленную стоимость (НДС):

(42)

НДС = (306412,44 + 61282,49) ∙ 0,20 = 73538,99 (руб.).

Цена предложения разработанного программного продукта:

 (43)

ЦПР = 306412,44 + 61282,49 + 73538,99 = 441233,92 (руб.).

Расчет минимального количества копий, необходимых для реализации, для получения установочной прибыли:

, (45)

где ЦСПР = 12000 руб. – цена на рынке одной копии программного продукта;

ЗТИРАЖ – затраты на тиражирование одной копии программного продукта.

, (46)

где СCD = 200 руб. – цена диска носителя;

Ттираж = 0,15 часа – время копирования;

Зтираж = 200 + 0,15 ∙ 86,86 = 213,03 (руб.);

NMIN = ≈ 38 шт.

NMIN = 38копии.

# Глава 2. Мероприятия по охране труда и технике безопасности

## 2.1 Мероприятия по технике безопасности

Для информационной системы (ИС) продажи компьютерной техники применение мероприятий по технике безопасности, эргономическим требованиям, организации рабочего места, уходу за монитором, помещению и освещению, режиму труда и отдыха, а также мероприятиям по электробезопасности и противопожарной технике является крайне важным. Вот как эти аспекты могут быть применены в контексте ИС продажи компьютерной техники:

* 1. Мероприятия по технике безопасности:
* обучение сотрудников правилам безопасности при работе с компьютерами и другим оборудованием;
* установка антивирусного программного обеспечения для защиты от вредоносных программ;
* проведение регулярной проверки оборудования на предмет неисправностей и выявление потенциальных опасностей.
  1. Эргономические требования при работе на компьютере:
  + выбор удобной и эргономичной мебели и оборудования для рабочего места.
  + регулярные перерывы и упражнения для снятия напряжения и уменьшения усталости.
  1. Организация рабочего места за компьютером:
  + расположение монитора на уровне глаз для предотвращения напряжения в шее и спине;
  + правильная установка клавиатуры и мыши для минимизации напряжения в запястьях и руках;
  + организация рабочего пространства для снижения риска падения кабелей и других препятствий.
  1. Уход за монитором:
  + регулярная очистка монитора от пыли и загрязнений;
  + использование мягкой ткани для чистки монитора, чтобы избежать появления царапин.
  1. Помещение и освещение:
  + обеспечение достаточного естественного и искусственного освещения рабочего места;
  + предотвращение бликов на мониторе за счет правильного расположения светильников.
  1. Режим труда и отдыха:
  + разработка графика работы с перерывами для отдыха и разминки;
  + поощрение сотрудников к выполнению упражнений для глаз и рук во время перерывов.
  1. Мероприятия по электробезопасности и противопожарной технике:
  + проведение проверок электрооборудования на предмет неисправностей и перегрева;
  + обучение сотрудников действиям в случае возгорания и использование огнетушителей.

# Заключение

В современном информационном обществе автоматизация бизнес-процессов является ключевым фактором успеха любой компании. Развитие информационных технологий предоставляет уникальные возможности для оптимизации и повышения эффективности деятельности предприятия. В рамках данной дипломной работы была проведена разработка информационной системы (ИС) продажи криптовалюты, ориентированной на современные требования рынка и потребности бизнеса.

В процессе выполнения работы были рассмотрены ключевые аспекты, связанные с проектированием, разработкой и внедрением ИС продажи компьютерной техники. Начиная с анализа требований заказчика и определения функциональных характеристик системы, был проведен всесторонний анализ рынка и существующих решений. Это позволило определить основные направления развития проекта и обосновать его актуальность и значимость.

В результате разработки была создана ИС, включающая в себя следующие основные компоненты:

* модуль управления товарами и складом: позволяет вести учет товаров на складе, контролировать их движение, автоматизировать процессы приемки и отгрузки;
* модуль управления заказами и клиентами: обеспечивает возможность оформления заказов, ведения базы клиентов, анализа покупательского поведения и формирования отчетности;
* модуль аналитики и отчетности: предоставляет пользователю инструменты для анализа данных, формирования отчетов и принятия управленческих решений на основе имеющейся информации;
* модуль администрирования и безопасности: обеспечивает защиту данных, управление правами доступа пользователей и обеспечение безопасности информационной системы в целом.

В процессе тестирования и апробации разработанной ИС было выявлено ее высокая производительность, удобство использования и соответствие заявленным требованиям. Полученные результаты позволяют сделать вывод о том, что разработанная система является эффективным инструментом для автоматизации процессов продажи компьютерной техники и способствует повышению конкурентоспособности компании на рынке.

Однако следует отметить, что разработка информационной системы является лишь первым шагом на пути к ее успешному внедрению и использованию. Дальнейшее развитие проекта предполагает расширение функциональности системы, внедрение новых технологий и интеграцию с другими информационными системами предприятия.

Таким образом, результаты проведенной работы подтверждают актуальность и значимость созданной информационной системы для бизнеса, а также ее потенциал в обеспечении эффективного управления процессами продажи компьютерной техники.

# Библиографический список

1. C# 6.0 в двух словах: Окончательная ссылка Джозефа Альбахари и Бена Альбахари
2. C# 7 и .NET Core: Современная кроссплатформенная разработка, второе издание Марка Дж. Прайса
3. C# 7.0 в двух словах от Джозефа Альбахари и Бена Альбахари
4. C# 8.0 и .NET Core 3.0 – Современная кроссплатформенная разработка Марка Дж. Прайса
5. Essential C# 8.0 от Марка Михаэлиса и Эрика Липперта
6. Head First SQL: Ваш мозг на SQL, Линн Бейли и Майкл Моррисон
7. Microsoft Visual C# 2013 Шаг за шагом, Джон Шарп.
8. SQL All-in-One для чайников от Аллана Г. Тейлора
9. SQL для начинающих: Изучайте SQL с использованием MySQL и концепций проектирования баз данных от Шиванша Ананда
10. SQL за 10 минут, Sams Teach Yourself от Бена Форты
11. Антипаттерны SQL: как избежать ошибок программирования баз данных, Билл Карвин
12. Желтая книга по программированию на C# Роба Майлза
13. Изучение SQL: Освоите основы SQL от Алана Болье
14. Краткое руководство по SQL: Упрощенное руководство для начинающих по управлению, анализу и манипулированию данными с помощью SQL от Уолтера Шилдса
15. Начало работы с SQL Server 2008 для разработчиков: от новичка до профессионала, Робин Доусон.
16. Поваренная книга по SQL: Решения для запросов и методы для разработчиков баз данных, Энтони Молинаро
17. Полное руководство по SQLite Гранта Аллена и Майка Оуэнса
18. Про C# 7: С помощью .NET и .NET Core Эндрю Троельсена и Филипа Япиксе
19. Сначала изучите C#: Руководство для учащихся по программированию в реальном мире на C#, XAML и .NET, Дженнифер Грин и Эндрю Стеллман
20. Эксперт по C# 7.0 .NET Core 2.0 от Мухаммада Рахмана

# Код приложения

MainWindow.xaml

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="25"/>

<RowDefinition Height="400"/>

<RowDefinition Height="25"/>

</Grid.RowDefinitions>

<Image Source="\Resources\Xolo.jpg" Grid.Row="1"/>

<Frame Name="MainFrame" Grid.Row="1" NavigationUIVisibility="Hidden"/>

</Grid>

MainWindow.xaml.cs

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

MainFrame.Navigate(new Pages.MainPageClient());

}

}

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="0"/>

<RowDefinition Height="450"/>

<RowDefinition Height="0"/>

</Grid.RowDefinitions>

<Image Source="\Resources\Xolo.jpg" Height="300" Grid.Row="1"/>

<Frame Grid.Row="1" Name="LoginFrame" NavigationUIVisibility="Hidden"/>

</Grid>

public LoginWindow()

{

InitializeComponent();

LoginFrame.Navigate(new Pages.LoginPage());

}

<Grid.RowDefinitions>

<RowDefinition Height="25"/>

<RowDefinition Height="400"/>

<RowDefinition Height="25"/>

</Grid.RowDefinitions>

<Frame Name="AdminFrame" Grid.Row="1" NavigationUIVisibility="Hidden"/>

</Grid>

public AdminWindow()

{

InitializeComponent();

AdminFrame.Navigate(new Pages.MainPage());

}

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="50"/>

<RowDefinition Height="50"/>

<RowDefinition Height="50"/>

<RowDefinition Height="50"/>

<RowDefinition Height="50"/>

<RowDefinition Height="50"/>

<RowDefinition Height="50"/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="200"/>

<ColumnDefinition Width="600"/>

</Grid.ColumnDefinitions>

<TextBlock Text="Придумайте ник" VerticalAlignment="Center" HorizontalAlignment="Center" Grid.Row="0"/>

<TextBlock Text="Придумайте логин" VerticalAlignment="Center" HorizontalAlignment="Center" Grid.Row="1"/>

<TextBlock Text="Придумайте пароль" VerticalAlignment="Center" HorizontalAlignment="Center" Grid.Row="2"/>

<TextBlock Text="Введите номер телефона" VerticalAlignment="Center" HorizontalAlignment="Center" Grid.Row="3"/>

<TextBlock Text="Введите электронная почта" VerticalAlignment="Center" HorizontalAlignment="Center" Grid.Row="4"/>

<TextBox Grid.Column="1" Name="Nick" Height="25" Width="300" VerticalContentAlignment="Center" Text="{Binding NickName}"/>

<TextBox Grid.Column="1" Grid.Row="1" Name="TbLogin" Height="25" Width="300" VerticalContentAlignment="Center" Text="{Binding Login}"/>

<TextBox Grid.Column="1" Grid.Row="2" Name="PbPass" Height="25" Width="300" VerticalContentAlignment="Center" Text="{Binding Password}"/>

<TextBox Grid.Column="1" Grid.Row="3" Name="TbPhone" Height="25" Width="300" VerticalContentAlignment="Center" Text="{Binding PhoneNumber}"/>

<TextBox Grid.Column="1" Grid.Row="4" Name="TbEmail" Height="25" Width="300" VerticalContentAlignment="Center" Text="{Binding Email}"/>

<StackPanel Grid.Column="1" Grid.Row="5" VerticalAlignment="Top" HorizontalAlignment="Left" Width="100" Margin="10 0 0 0">

<TextBlock Text="Ваш пол"/>

<ComboBox Name="CbSex" Height="20" Width="100" ItemsSource="{Binding Sex}" SelectedItem="{Binding Sex}" DisplayMemberPath="Sex1"/>

</StackPanel>

<StackPanel Grid.Column="1" Grid.Row="5" Width="100" VerticalAlignment="Top" HorizontalAlignment="Center">

<TextBlock Text="Страна"/>

<ComboBox Name="CbCountry" ItemsSource="{Binding Country}" SelectedItem="{Binding Country}" Height="20" Width="100" VerticalContentAlignment="Center">

<ComboBox.ItemTemplate>

<DataTemplate>

<Image Source="{Binding GetPhoto}" Width="25" Height="25"/>

</DataTemplate>

</ComboBox.ItemTemplate>

</ComboBox>

</StackPanel>

<StackPanel Grid.Column="1" Grid.Row="5" VerticalAlignment="Top" HorizontalAlignment="Right" Width="100" Margin="0 0 10 0">

<TextBlock Text="Ваша роль" />

<ComboBox Name="CbRole" Height="20" Width="100" ItemsSource="{Binding Role}" SelectedItem="{Binding Role}" DisplayMemberPath="Role1"/>

</StackPanel>

<Button Grid.Column="1" Grid.Row="6" Name="Save" Click="Save\_Click" Content="Сохранить"/>

<Button Grid.Column="0" Grid.Row="6" Name="Back" Click="Back\_Click" Content="Назад"/>

</Grid>

public partial class AddEditPage : Page

{

private User \_currentUser = new User();

public AddEditPage(User selectedUser)

{

InitializeComponent();

if (selectedUser != null)

{

\_currentUser = selectedUser;

}

DataContext = \_currentUser;

CbRole.ItemsSource = CryptoWalletEntities.GetContext().Roles.ToList();

CbCountry.ItemsSource = CryptoWalletEntities.GetContext().Countries.ToList();

CbSex.ItemsSource = CryptoWalletEntities.GetContext().Sexes.ToList();

}

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

{

StringBuilder errors = new StringBuilder();

if (\_currentUser.NickName == null)

errors.AppendLine("Необходимо придумать ник");

if (\_currentUser.Login == null)

errors.AppendLine("Необходимо придумать логин");

if (\_currentUser.Password == null)

errors.AppendLine("Необходимо придумать пароль");

if (\_currentUser.PhoneNumber == null)

errors.AppendLine("Необходимо указать номер телефона");

if (\_currentUser.Email == null)

errors.AppendLine("Необходимо указать почту");

if (\_currentUser.Sex == null)

errors.AppendLine("Необходимо указать пол");

if (\_currentUser.Country == null)

errors.AppendLine("Необходимо указать страну");

if (\_currentUser.Role == null)

errors.AppendLine("Необходимо указать роль");

if(errors.Length > 0)

{

MessageBox.Show(errors.ToString());

return;

}

if (\_currentUser.ID\_User == 0)

{

CryptoWalletEntities.GetContext().Users.Add(\_currentUser);

}

try

{

CryptoWalletEntities.GetContext().SaveChanges();

MessageBox.Show("Успешно!");

NavigationService.GoBack();

}

catch (Exception ex)

{

MessageBox.Show(ex.ToString());

}

}

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

{

NavigationService.GoBack();

}

}

}

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="50"/>

<RowDefinition Height="50"/>

<RowDefinition Height="50"/>

<RowDefinition Height="50"/>

<RowDefinition Height="50"/>

<RowDefinition Height="50"/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="200"/>

<ColumnDefinition Width="600"/>

</Grid.ColumnDefinitions>

<TextBlock Grid.Row="3" Grid.Column="0" VerticalAlignment="Center" HorizontalAlignment="Center" Text="Баланс"/>

<TextBlock Grid.Row="2" Grid.Column="0" VerticalAlignment="Center" HorizontalAlignment="Center" Text="Валюта"/>

<TextBox Grid.Row="2" Name="TbCurrency" Text="{Binding Currency}" Grid.Column="1"/>

<TextBox Grid.Row="3" Name="TbBalance" Grid.Column="1" Text="{Binding Balance}"/>

<Button Grid.Row="5" Content="Назад" Name="Back" Click="Back\_Click"/>

<Button Grid.Column="1" Grid.Row="5" Content="Сохранить" Name="Save" Click="Save\_Click"/>

</Grid>

public partial class BalancePage : Page

{

private User \_currentUser = new User();

public BalancePage(User selectedUser)

{

InitializeComponent();

if (selectedUser != null)

{

\_currentUser = selectedUser;

}

DataContext = \_currentUser;

}

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

{

NavigationService.GoBack();

}

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

{

//StringBuilder errors = new StringBuilder();

//if (\_currentUser.Balance != null)

//errors.AppendLine("Введите баланс");

//if (\_currentUser.Currency != null)

//errors.AppendLine("Выберите валюту");

//if (errors.Length > 0)

//{

//MessageBox.Show(errors.ToString());

//return;

//}

if (\_currentUser.ID\_User == 0)

{

CryptoWalletEntities.GetContext().Users.Add(\_currentUser);

}

try

{

CryptoWalletEntities.GetContext().SaveChanges();

MessageBox.Show("Успешно!");

NavigationService.GoBack();

}

catch(Exception ex)

{

MessageBox.Show(ex.ToString());

}

}

}

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="25"/>

<RowDefinition Height="300"/>

<RowDefinition Height="50"/>

</Grid.RowDefinitions>

<DataGrid ItemsSource="{Binding Currency}" AutoGenerateColumns="False" Name="DGCurrency" Grid.Row="1" IsReadOnly="True">

<DataGrid.Columns>

<DataGridTextColumn Header="Валюта" Binding="{Binding Currency1}" Width="400"/>

<DataGridTextColumn Header="Стоимость в рублях" Binding="{Binding PriceInRub}" Width="400"/>

</DataGrid.Columns>

</DataGrid>

<TextBlock Text="Курсы валют"/>

<Button Grid.Row="2" Content="Назад" Name="Back" Click="Back\_Click"/>

</Grid>

public partial class CourseCurrencyPage : Page

{

public CourseCurrencyPage()

{

InitializeComponent();

DGCurrency.ItemsSource = CryptoWalletEntities.GetContext().Currencies.ToList();

}

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

{

NavigationService.GoBack();

}

}

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="0"/>

<RowDefinition Height="300"/>

<RowDefinition Height="0"/>

</Grid.RowDefinitions>

<StackPanel Grid.Row="1" HorizontalAlignment="Center" VerticalAlignment="Center">

<TextBlock Text="Введите логин" TextAlignment="Center" Foreground="Beige"/>

<TextBox Name="TbLogin"/>

<TextBlock Text="Введите пароль" TextAlignment="Center" Foreground="Beige"/>

<PasswordBox Name="PbPass"/>

<StackPanel Orientation="Horizontal" HorizontalAlignment="Center">

<Button Content="Войти" Name="SignIn" Click="SignIn\_Click" Margin="5"/>

<Button Content="Регистрация" Name="SignUp" Click="SignUp\_Click"/>

</StackPanel>

</StackPanel>

</Grid>

public partial class LoginPage : Page

{

public LoginPage()

{

InitializeComponent();

}

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

{

try

{

List<User>user = CryptoWalletEntities.GetContext().Users.ToList();

User u = user.FirstOrDefault(p => p.Password == PbPass.Password && p.Login == TbLogin.Text);

if (u != null)

{

switch (u.ID\_Role)

{

case 2: new MainWindow().ShowDialog();

break;

case 1:new Windows.AdminWindow().ShowDialog();

break;

}

}

else

{

MessageBox.Show("Неверные данные для входа");

}

}catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString());

}

}

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

{

NavigationService.Navigate(new Pages.SignUpPage(null));

}

}

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="100"/>

<RowDefinition Height="200"/>

<RowDefinition Height="100"/>

</Grid.RowDefinitions>

<TextBlock Text="Вы вошли как Admin"/>

<UniformGrid Grid.Row="1">

<Button Content="Пользователи" Name="UsersPage" Click="UsersPage\_Click"/>

<Button Content="Валюты" Name="CurrencyPage" Click="CurrencyPage\_Click"/>

</UniformGrid>

</Grid>

public partial class MainPage : Page

{

public MainPage()

{

InitializeComponent();

}

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

{

NavigationService.Navigate(new Pages.WalletPage());

}

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

{

NavigationService.Navigate(new Pages.CourseCurrencyPage());

}

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

{

NavigationService.Navigate(new Pages.AddEditPage((sender as Button).DataContext as User));

}

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

{

NavigationService.Navigate(new Pages.UserPage());

}

}

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="50"/>

<RowDefinition Height="50"/>

<RowDefinition Height="50"/>

<RowDefinition Height="50"/>

<RowDefinition Height="50"/>

<RowDefinition Height="50"/>

<RowDefinition Height="50"/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="200"/>

<ColumnDefinition Width="600"/>

</Grid.ColumnDefinitions>

<TextBlock Text="Придумайте ник" VerticalAlignment="Center" HorizontalAlignment="Center" Grid.Row="0"/>

<TextBlock Text="Придумайте логин" VerticalAlignment="Center" HorizontalAlignment="Center" Grid.Row="1"/>

<TextBlock Text="Придумайте пароль" VerticalAlignment="Center" HorizontalAlignment="Center" Grid.Row="2"/>

<TextBlock Text="Введите номер телефона" VerticalAlignment="Center" HorizontalAlignment="Center" Grid.Row="3"/>

<TextBlock Text="Введите электронная почта" VerticalAlignment="Center" HorizontalAlignment="Center" Grid.Row="4"/>

<TextBox Grid.Column="1" Name="Nick" Height="25" VerticalContentAlignment="Center" Text="{Binding NickName}"/>

<TextBox Grid.Column="1" Grid.Row="1" Name="TbLogin" VerticalContentAlignment="Center" Text="{Binding Login}"/>

<TextBox Grid.Column="1" Grid.Row="2" Name="PbPass" VerticalContentAlignment="Center" Text="{Binding Password}"/>

<TextBox Grid.Column="1" Grid.Row="3" Name="TbPhone" VerticalContentAlignment="Center" Text="{Binding PhoneNumber}"/>

<TextBox Grid.Column="1" Grid.Row="4" Name="TbEmail" VerticalContentAlignment="Center" Text="{Binding Email}"/>

<StackPanel Grid.Column="1" Grid.Row="5" VerticalAlignment="Top" HorizontalAlignment="Left" Width="100" Margin="10 0 0 0">

<TextBlock Text="Ваш пол"/>

<ComboBox Name="CbSex" Height="20" Width="100" ItemsSource="{Binding Sex}" SelectedItem="{Binding Sex}" DisplayMemberPath="Sex1"/>

</StackPanel>

<StackPanel Grid.Column="1" Grid.Row="5" Width="100" VerticalAlignment="Top" HorizontalAlignment="Center">

<TextBlock Text="Страна"/>

<ComboBox Name="CbCountry" ItemsSource="{Binding Country}" SelectedItem="{Binding Country}" Height="20" Width="100" VerticalContentAlignment="Center">

<ComboBox.ItemTemplate>

<DataTemplate>

<Image Source="{Binding GetPhoto}" Width="25" Height="25"/>

</DataTemplate>

</ComboBox.ItemTemplate>

</ComboBox>

</StackPanel>

<StackPanel Grid.Column="1" Grid.Row="5" VerticalAlignment="Top" HorizontalAlignment="Right" Width="100" Margin="0 0 10 0">

<TextBlock Text="Ваша роль" />

<ComboBox Name="CbRole" Height="20" Width="100" ItemsSource="{Binding Role}" SelectedItem="{Binding Role}" DisplayMemberPath="Role1"/>

</StackPanel>

<Button Grid.Column="1" Grid.Row="6" Name="Save" Click="Save\_Click" Content="Сохранить"/>

<Button Grid.Column="0" Grid.Row="6" Name="Back" Click="Back\_Click" Content="Назад"/>

</Grid>

private User \_currentUser = new User();

public SignUpPage(User selectedUser)

{

InitializeComponent();

if (selectedUser != null)

{

\_currentUser = selectedUser;

}

DataContext = \_currentUser;

CbCountry.ItemsSource = CryptoWalletEntities.GetContext().Countries.ToList();

CbSex.ItemsSource = CryptoWalletEntities.GetContext().Sexes.ToList();

CbRole.ItemsSource = CryptoWalletEntities.GetContext().Roles.ToList();

}

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

{

StringBuilder errors = new StringBuilder();

if (\_currentUser.NickName == null)

errors.AppendLine("Необходимо придумать ник");

if (\_currentUser.Login == null)

errors.AppendLine("Необходимо придумать логин");

if (\_currentUser.Password == null)

errors.AppendLine("Необходимо придумать пароль");

if (\_currentUser.PhoneNumber == null)

errors.AppendLine("Необходимо указать номер телефона");

if (\_currentUser.Email == null)

errors.AppendLine("Необходимо указать почту");

if (\_currentUser.Sex == null)

errors.AppendLine("Необходимо указать пол");

if (\_currentUser.Country == null)

errors.AppendLine("Необходимо указать страну");

if (\_currentUser.Role == null)

errors.AppendLine("Необходимо указать роль");

if (errors.Length > 0)

{

MessageBox.Show(errors.ToString());

return;

}

if (\_currentUser.ID\_User == 0)

{

CryptoWalletEntities.GetContext().Users.Add(\_currentUser);

}

try

{

CryptoWalletEntities.GetContext().SaveChanges();

MessageBox.Show("Успешно!");

NavigationService.GoBack();

}

catch (Exception ex)

{

MessageBox.Show(ex.ToString());

}

}

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

{

NavigationService.GoBack();

}

}

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="50"/>

<RowDefinition Height="300"/>

<RowDefinition Height="50"/>

</Grid.RowDefinitions>

<StackPanel Grid.Row="0" >

<TextBlock Text="Поиск по нику" HorizontalAlignment="Center" FontSize="15"/>

<TextBox Name="TbSearchNick" TextChanged="TbSearchNick\_TextChanged" HorizontalAlignment="Center" Width="300" BorderBrush="Goldenrod"/>

</StackPanel>

<ListView Grid.Row="1" ItemsSource="{Binding Path=User}" Name="LWUsers"

ScrollViewer.HorizontalScrollBarVisibility="Disabled" >

<ListView.ItemsPanel>

<ItemsPanelTemplate>

<WrapPanel Orientation="Horizontal" HorizontalAlignment="Center"

VerticalAlignment="Center"/>

</ItemsPanelTemplate>

</ListView.ItemsPanel>

<ListView.ItemTemplate>

<DataTemplate>

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="200"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<StackPanel Grid.Column="0" Orientation="Vertical"

VerticalAlignment="Center"

HorizontalAlignment="Center" Height="150">

<StackPanel Orientation="Horizontal">

<TextBlock Text="Уникальный номер: " FontSize="10"></TextBlock>

<TextBlock Text="{Binding ID\_User}" FontSize="10"></TextBlock>

</StackPanel>

<StackPanel Orientation="Horizontal">

<TextBlock Text="Ник: " FontSize="10"></TextBlock>

<TextBlock Text="{Binding NickName}" FontSize="10"></TextBlock>

</StackPanel>

<StackPanel Orientation="Horizontal">

<TextBlock Text="Номер: " FontSize="10"></TextBlock>

<TextBlock Text="{Binding PhoneNumber}" FontSize="10"></TextBlock>

</StackPanel>

<StackPanel Orientation="Horizontal" Width="400">

<TextBlock Text="Почта: " FontSize="10"></TextBlock>

<TextBlock Text="{Binding Email}" FontSize="10"></TextBlock>

</StackPanel>

<StackPanel Orientation="Horizontal">

<TextBlock Text="Страна: " FontSize="10"></TextBlock>

<Image Source="{Binding Country.GetPhoto}" Width="25" Height="25"></Image>

</StackPanel>

<StackPanel Orientation="Horizontal">

<TextBlock Text="Пол: " FontSize="10"></TextBlock>

<TextBlock Text="{Binding Sex.Sex1}" FontSize="10"></TextBlock>

</StackPanel>

<StackPanel Orientation="Horizontal">

<TextBlock Text="Валюта: " FontSize="10"></TextBlock>

<TextBlock Text="{Binding Currency}" FontSize="12"></TextBlock>

</StackPanel>

<StackPanel Orientation="Horizontal">

<TextBlock Text="Баланс: " FontSize="10"></TextBlock>

<TextBlock Text="{Binding Balance}" FontSize="10"></TextBlock>

</StackPanel>

</StackPanel>

<StackPanel Margin="10" Grid.Column="1">

<Button Name="Edit" Width="110" Content="Профиль" Click="Edit\_Click"/>

<Button Content="Изменить баланс" Width="110" Name="Balance" Click="Balance\_Click"/>

</StackPanel>

</Grid>

</DataTemplate>

</ListView.ItemTemplate>

</ListView>

<StackPanel Orientation="Horizontal" HorizontalAlignment="Center" Grid.Row="2" VerticalAlignment="Top">

<Button Content="Назад" Name="Back" Click="Back\_Click"/>

<Button Content="Добавить" Name="Add" Click="Add\_Click"/>

<Button Content="Удалить" Name="Del" Click="Del\_Click"/>

</StackPanel>

</Grid>

public partial class UserPage : Page

{

public UserPage()

{

InitializeComponent();

LWUsers.ItemsSource = CryptoWalletEntities.GetContext().Users.ToList();

}

private void Update()

{

var currentUsers = CryptoWalletEntities.GetContext().Users.OrderBy(p => p.ID\_User).ToList();

currentUsers = currentUsers.Where(p => p.NickName.ToLower().Contains(TbSearchNick.Text.ToLower())).ToList();

//currentUsers = currentUsers.Where(p => p.Surname.ToLower().Contains(TbSearchSurname.Text.ToLower())).ToList();

LWUsers.ItemsSource = currentUsers;

}

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

{

NavigationService.Navigate(new Pages.AddEditPage((sender as Button).DataContext as User));

}

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

{

NavigationService.GoBack();

}

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

{

NavigationService.Navigate(new Pages.AddEditPage(null));

}

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

{

var usersForRemoving = LWUsers.SelectedItems.Cast<User>().ToList();

if (MessageBox.Show($"вы точно хотите удалить этих {usersForRemoving.Count()} пользователей?", "Внимание!",

MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)

{

try

{

CryptoWalletEntities.GetContext().Users.RemoveRange(usersForRemoving);

CryptoWalletEntities.GetContext().SaveChanges();

MessageBox.Show("Пользователи удалены");

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString());

}

}

}

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

{

Update();

}

private void Page\_IsVisibleChanged(object sender, DependencyPropertyChangedEventArgs e)

{

if (Visibility == Visibility.Visible)

LWUsers.ItemsSource = null;

// загрузка обновленных данных

CryptoWalletEntities.GetContext().ChangeTracker.Entries().ToList().ForEach(p => p.Reload());

List<User> users = CryptoWalletEntities.GetContext().Users.OrderBy(p => p.ID\_User).ToList();

LWUsers.ItemsSource = users;

}

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

{

NavigationService.Navigate(new Pages.BalancePage((sender as Button).DataContext as User));

}

}