**sДЕПАРТАМЕНТ ОБРАЗОВАНИЯ ЯРОСЛАВСКОЙ ОБЛАСТИ**

государственное профессиональное образовательное учреждение   
Ярославской области

Рыбинский полиграфический колледж

**КУРСОВОЙ ПРОЕКТ**

|  |  |
| --- | --- |
| Клиент-серверное приложение агентство недвижимости | |
|  | |
| по дисциплине | Технология разработки и защит баз данных |
|  | |

Пояснительная записка

|  |
| --- |
| КП.0902.04.000000.00 ПЗ |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Студент группы | 4-ИС-2 |  |  |  | А. М. Дубровский |
|  | *(Код учебной группы)* |  | *(Подпись, дата)* |  | *(И.О.Фамилия)* |
| Руководитель | преподаватель |  |  |  | Лобанова Е А |
|  | *(Должность, звание)* |  | *(Подпись, дата)* |  | *( И.О.Фамилия)* |
| Нормоконтроль | преподаватель |  |  |  | Лобанова Е А |
|  | *(Должность, звание)* |  | *(Подпись, дата)* |  | *(И.О.Фамилия)* |

г. Рыбинск

2022

**СОДЕРЖАНИЕ**

[Введение 3](#_Toc121223211)

[1 Исследовательский раздел 4](#_Toc121223212)

[2 Конструкторский раздел 8](#_Toc121223213)

[2.1 Проектирование информационной модели данных 8](#_Toc121223214)

[2.2 Проектирование серверной части приложения 9](#_Toc121223215)

[2.2.1 Разработка схемы базы данных 9](#_Toc121223216)

[2.2.2 Разработка сущностей базы данных 12](#_Toc121223217)

[2.3 Проектирование клиентской части приложения 13](#_Toc121223218)

[2.3.1 Разработка модулей схемы 13](#_Toc121223219)

[2.3.2 Разработка пользовательского интерфейса 15](#_Toc121223220)

[2.3.3 Организация доступа к объектам базы данных 19](#_Toc121223221)

[2.3.4 Разработка блох-схем алгоритмов процедур и функций 20](#_Toc121223222)

[2.4 Обеспечение коллективного доступа. Защита информации 22](#_Toc121223223)

[3 Технологическая часть 25](#_Toc121223224)

[3.1 Тестирование и отладка приложения 25](#_Toc121223225)

[3.2 Инструкция администратора базы данных 29](#_Toc121223226)

[3.3 Инструкция по эксплуатации приложения 39](#_Toc121223227)

[4 Раздел охраны труда 47](#_Toc121223228)

[Заключение 49](#_Toc121223229)

[Список используемых источников 50](#_Toc121223230)

[Приложение А 51](#_Toc121223231)

Введение

В рамках данного курсового проекта планируется разработка приложения, где будет реализована покупка и продажа недвижимости клиентом. Такое приложение наиболее разумнее реализовывать с помощью вычислительной техники, так как это позволит максимально расширить функционал и эффективность использования, за счет относительной простоты реализации. Распространение также будет эффективнее и целесообразнее в цифровом формате нежели в каком-либо физическом представлении. Для любого бизнеса необходимо иметь свое представление в цифровом виде в 21 веке, так как клиентам будет проще взаимодействовать с ним. Возможность через приложение покупать и продовать недвижимость станет приятным плюсом.

Приложение упростить взаимодействие пользователя с документами, необходимыми для совершения сделки. Оно позволит покупать и продавать недвижимость, а регистрация также упростит дальнейшее использование приложения для пользователя. Для клиентов это шанс без каких-либо проблем взаимодействовать со своей собственностью.

1 Исследовательский раздел

Процесс разработки программного обеспечения – набор правил, согласно которым построена разработка программного обеспечения. Приложение можно назвать клиент серверным если оно включает в себя клиент-серверную архитектуру. Разработку клиент-серверного приложения необходимо начинать с выбора архитектуры клиент-сервера.

Для разработки клиент/серверных систем имеется два подхода. Первый подход построение систем на основе двухзвенной архитектуры. Состоит из клиентской и серверной части. Как правило, серверная часть представляет собой сервер БД, на котором расположены общие данные. А клиентская часть представляет приложение, которое связывается с сервером БД, осуществляет к нему запросы и получает ответы. Такие системы используются в локальных сетях, т.к. нет затруднений с установкой клиентской части. Также системы с такой архитектурой более безопасны, т.к. могут использовать собственные протоколы передачи данных, не известные злоумышленникам. Поэтому многие крупные компании, которые располагаются не в едином месте и для соединения подразделений используют глобальную сеть Интернет, выбирают именно такую архитектуру построения клиент/серверных систем.

При разработке информационных систем, рассчитанных на широкую аудиторию, возникают проблемы с использованием двухзвенной архитектуры. Во-первых, пользователю необходимо иметь в наличии клиентскую часть, а, во-вторых, у неопытного пользователя, могут возникнуть проблемы с конфигурированием такой системы. Поэтому в последнее время, более часто разрабатывают приложения на базе трехзвенной архитектуры.

Второй подход построение систем на основе трехзвенной архитектуры. состоит из двух частей: клиента и сервера. Но серверная часть в этой архитектуре представляет собой сервер приложений и сервер БД. А в качестве клиента выступает web-браузер. Такая система очень проста для пользователя. Ему необходимо знать только адрес сервера приложения и наличие web-браузера на рабочем компьютере. Все данные представляются в виде html-разметки, с использование графики (jpeg, gif, flash) и JavaScript. Передача запросов от клиента к серверу приложений происходит по средствам CGI-интерфейса. Сервер приложений общается с сервером БД, используя другой интерфейс, зависящий от того, на основе каких средств строится конкретная информационная система. Недостатками такой архитектуры является использование общеизвестных протоколов и интерфейсов передачи данных. Злоумышленник может осуществить взлом системы, если она будет недостаточно хорошо проверять поступившие запросы от клиента.

При разработке клиент/серверных приложений необходимо учитывать:

- на каких пользователей будет рассчитана данная информационная система;

- какие требования предъявляются к безопасности.

Если информационная система должна быть общедоступной и рассчитана на широкую аудиторию, то необходимо использовать трехзвенную архитектуру.

Если информационная система используется внутри предприятия, доступ имеют к ней ограниченные пользователи и требуется создать максимально безопасную и защищенную систему, то следует отдать предпочтение двухзвенной архитектуре. [1]

В рамках курсового проекта был выбран первый способ для разработки клиент/серверной системы на основе двухзвенной архитектуры.

При выборе WPF можно выделить такие преимущества как аппаратное ускорение через DirectX, что сильно влияет на производительность. Также можно отметить веб-подобную модель компоновки. Вместо того чтобы фиксировать элементы управления на месте с определенными координатами, WPF поддерживает гибкий поток, размещающий элементы управления на основе их содержимого. В результате получается пользовательский интерфейс, который может быть адаптирован для отображения высоко динамичного содержимого или к разным языкам.

Преимуществами WPF являются:

* **веб-подобная модель компоновки;**
* **богатая модель рисования;**
* **развитая текстовая модель;**
* **анимация;**
* **поддержка аудио и видео;**
* **стили и шаблоны;**
* **команды;**
* **декларативный пользовательский интерфейс;**
* **приложения на основе страниц.**

При выборе сред разработки были рассмотрены Visual Studio и Project Rider. Visual Studio – это удобная интегрированная среда разработки (IDE) от Microsoft, позволяющая быстро и эффективно создавать, и разрабатывать проект, выбрав для этого все необходимое. Среда использует платформы разработки программного обеспечения Microsoft: Windows API, Windows Forms, Windows Presentation Foundation, Windows Store и Microsoft Silverlight. Так же она принимает плагины, которые расширяют функциональные возможности практически на каждом уровне, включая добавление поддержки систем управления исходным кодом (таких как Subversion) и обеспечивает стандартный для Windows вид окон приложения. Единственным минусом можно считать сложность освоения данной среды разработки из-за её большого количества различных функций, спрятанных в подразделах меню. Информация из работы [2].

Project Rider – это среда от JetBrains для работы с платформой .NET. Она обладает поддержкой полного цикла. Фирменная черта продуктов JetBrains, воплощенная и в Project Rider. С Project Rider появиться возможность организовать весь цикл создания программного обеспечения: от идеи до поддержки. Функциональность Project Rider позволяет подключить MSBuild и XBuild, работать с CLI-проектами и организовать отладку приложений .NET and Mono. Множество опций для быстрого создания кода улучшает производительность. Кроссплатформенность Project Rider работает с Windows, Linux и MacOS. Из минусов можно выделить её молодость. Часть функциональности еще в разработке, не все стартовые ошибки исправлены. Так же можно отметить её стоимость. Самая дешевая версия Project Rider обойдется в 139 долларов за первый год использования. Но есть триал-версия и специальные предложения для студентов и непрофильных организаций. Информация из работы [3].

Из этих двух сред разработки был выбран Visual Studio, так как он обладает всем необходимым функционалом для реализации проекта, а также она является бесплатной и дольше находится на рынке труда.

В интернете существует аналог разрабатываемого приложения [4], у которого есть ряд ключевых отличий. У аналога нет возможности мгновенной купли-продажи недвижимости, из-за чего процесс покупки или продажи может затянуться на несколько дней. Можно отметить хороший функционал у аналога, например, в отличии от разрабатываемого приложения у аналога есть возможность получить консультацию по интересующей недвижимости.

Приложение должно предоставить возможность купить или продать пользователю интересующую для него недвижимость. Пользователь может указать характеристики имущества при продаже, а так же отсортировать интересующие его предложения по покупке. Данные будут автоматически браться из базы данных. В приложении будет присутствовать отдельная админ панель, которая будет доступна для пользователей с определенным уровнем доступа. В админ панели, администратор сможет редактировать базу данных.

Основным процессом будет выступать процесс занесения сведений о купли-продажи недвижимости. Этот функционал будет необходим для отслеживания действий пользователя при покупке недвижимости.

2 Конструкторский раздел

2.1 Проектирование информационной модели данных

Черная сфера представляет собой систему, внутреннее устройство которой не важно. В эту систему подаются входные данные, а на выходе из системы поступают выходные данные. Черная сфера представлена на рисунке 2.1.

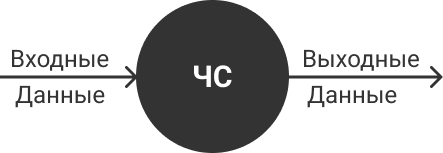


Рисунок 2.1 – Модель «Черная сфера»

Представим наш класс в виде черной сферы. В приложении будут присутствовать такие выходные данные, как данные об пользователе, размер окон и управляющее воздействие пользователя. На выходе из приложения будет поступать изменения в базе данных и изображения. Данные об пользователе – это те данные, которые пользователь вводит при регистрации и в своем личном кабинете. Размеры будут поступать после изменения размеров окон с помощью манипулятора типа «мышь». Управляющее воздействие будет происходить с помощью клавиатуры и манипулятора типа «мышь». Изменения в базе данных – это те изменения, которые происходят после определенных взаимодействий пользователя в интерфейсе приложения. Изображение – интерфейс который будет показываться пользователю для взаимодействия с программой. Черная сфера с перечисленными входными и выходными параметрами представлена на рисунке 2.2.

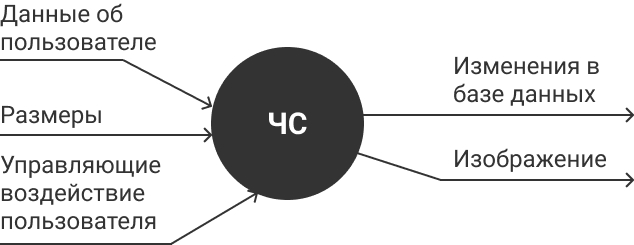


Рисунок 2.2 – Черная сфера с параметрами

В приложении основной процесс будет отвечать за действие пользователя. Окно будет хранит в себе объект, представляющий из себя сущность с определенными свойствами необходимыми для их визуального отображения.

2.2 Проектирование серверной части приложения

2.2.1 Разработка схемы базы данных

Для выявления всех возможных сущностей будущей базы и получения концептуальной модели данных будет проведено несколько серий нормализации.

На первом этапе нормализации можно представить модель как связь между клиентом и агенством недвижимости. Первый этап нормализации представлен на рисунке 2.3.

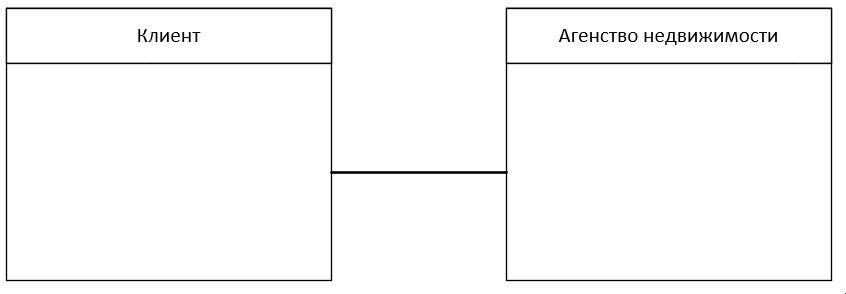


Рисунок 2.3 – Первый этап нормализации

Во втором этапе нормализации разобьём сущность агенства недвижимости на 3 сущности: недвижимость, тип недвижимости и статус недвижимости. Сущность недвижимостей будет содержать в себе информацию о владельце, тип недвижимости и статус недвижимости. Второй этап нормализации на рисунке 2.4.

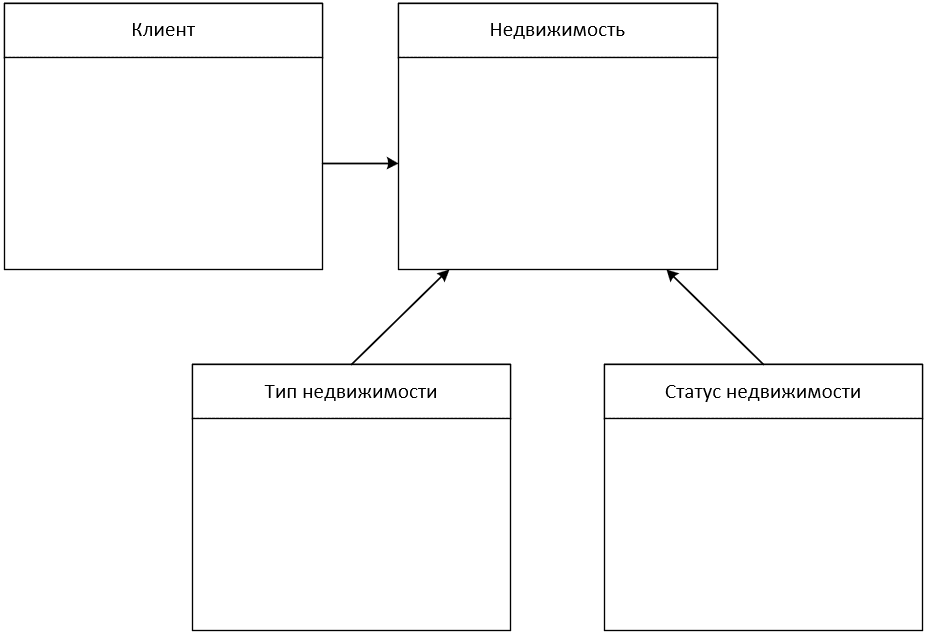


Рисунок 2.4 – Второй этап нормализации

В третьем заключительном этапе нормализации получим полную концептуальную схему вынеся информацию о транзакциях с недвижимостью. Третий этап нормализации представлена на рисунке 2.5.

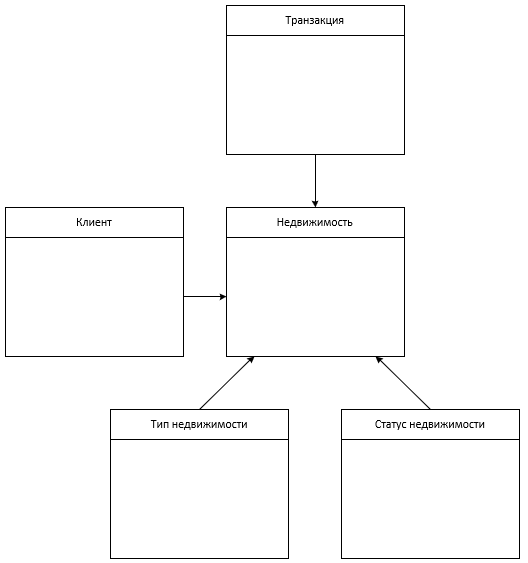


Рисунок 2.5 – Третий этап нормализации

Получим логическую модель данных с содержанием всех сущностей, связей и атрибутов данных. Логическая модель данных представлена на рисунке 2.6.

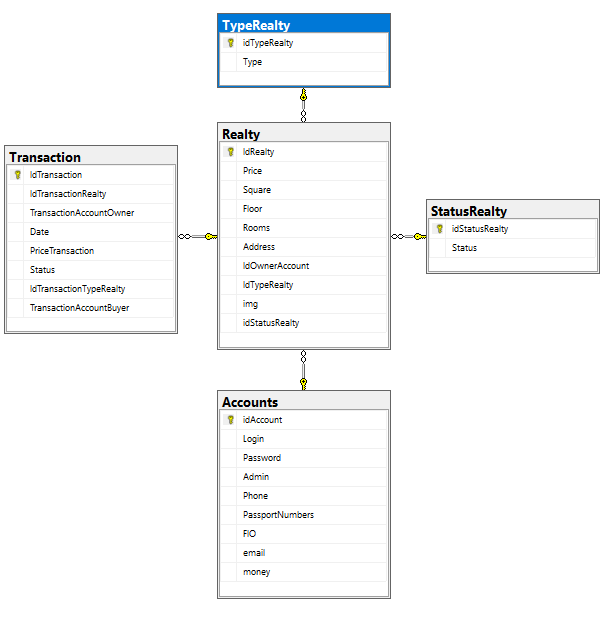


Рисунок 2.6 – Логическая модель данных

Получим физическую модель данных, включающая ассоциативные таблицы, которые иллюстрируют отношения между сущностями, а также первичные и внешние ключи для связи данных. Физическая модель данных представлена на рисунке 2.7.

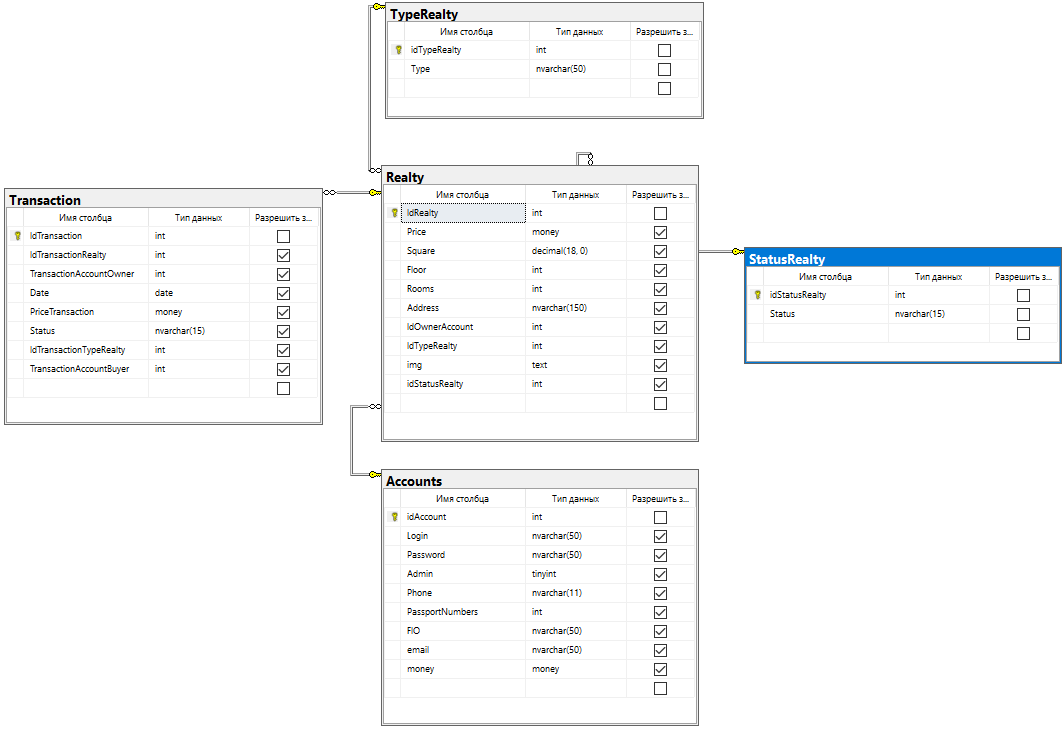


Рисунок 2.7 – Физическая модель данных

2.2.2 Разработка сущностей базы данных

В результатах раздела «Разработка схемы базы данных» получена схема базы данных, из которой следует необходимость присутствия определенных сущностей необходимых для полноценной работы приложения. Для удобства все сущности сведены в табличном виде. Сущности схемы базы данных представлены в таблице 2.1.

Таблица 2.1 – Сущности схемы базы данных

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя сущности | Назначение сущности | Типы данных | Перечисление наименований сущностей, которые подчиняются текущей сущности | Перечисление наименований сущностей, которым подчиняется текущая сущность |
| Accounts | Содержит данные об пользователя приложения | int, nvarchar, money, tinyint | - | Realty |
| Realty | Содержит информацию о недвижимости | int, nvarchar, money, text, decimal | TypeRealty, StatusRealty,  Accounts | Transaction |
| Transaction | Содержит информацию о действиях с недвижимостью | int, nvarchar, money, date | Realty | - |
| StatusRealty | Содержит информацию о статусе недвижимости | Int, nvarchar | - | Realty |
| TypeRealty | Содержит информацию о типе недвижимости | int, nvarchar | - | Realty |

2.3 Проектирование клиентской части приложения

2.3.1 Разработка модулей схемы

WPF предоставляет комплексный набор функций разработки приложений, которые включают в себя язык XAML, элементы управления, привязку к данным, макет, двумерную и трехмерную графику, анимацию, стили, шаблоны, документы, мультимедиа, текст и типографические функции. WPF является частью .NET, поэтому вы можете создавать приложения, включающие другие элементы .NET API. [5].

Представим клиентскую часть приложения в виде модульной схемы показывающая связь между окнами, классами и страницами при организации клиентской части приложения. Модульная схема клиентской части приложения представлена на рисунке 2.8.

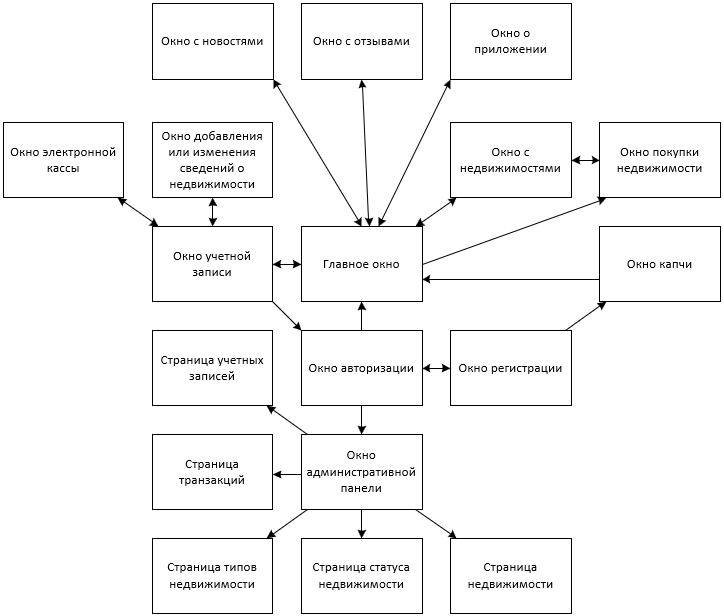


Рисунок 2.7 – Модульная схема клиентской части приложения

В составе модульной схемы присутствуют следующие элементы:

* Окно учетной записи, для редактирования информации о себе, переход в окно изменение или добавления сведений о недвижимости, переход в окно электронной кассы, переход в окно авторизации
* Окно электронной кассы, для пополнения счета пользователя и переход в личный кабинет
* Окно добавления или изменения сведений о недвижимости, для добавления или изменения информации о недвижимости, переход в личный кабинет
* Окно с новостями, для информации о последних событиях проекта и переход в главное окно
* Окно с отзывами, для просмотра отзывов пользователей и переход в главное окно
* Окно о приложении, для просмотра информации о приложении и переход в главное окно
* Окно с недвижимостью, для просмотра интересующих предложений о продаже недвижимости, переход в главное окно или переход на окно объявления о продаже
* Окно покупки недвижимости, для покупки недвижимости, переход на окно с недвижимостью
* Главное окно, для перехода в личный кабинет или покупки недвижимости
* Окно регистрации, для регистрации пользователей
* Окно авторизации, для авторизации пользователей
* Окно административной панели, для взаимодействия с базой данных
* Окно капчи, для различия компьютеров и людей
* Страница учетных записей, для взаимодействия с таблицей с учетными записями
* Страница транзакций, для взаимодействия с таблицей с транзакциями
* Страница недвижимости, для взаимодействия с таблицей с недвижимостью
* Страница с типом недвижимости, для взаимодействия с таблицей с типом недвижимости
* Страница со статусом недвижимости, для взаимодействия с таблицей со статусом недвижимости

2.3.2 Разработка пользовательского интерфейса

Пользовательский интерфейс – это совокупность информационной модели проблемной области, средств и способов взаимодействия пользователя с информационной моделью, а также компонентов, обеспечивающих формирование информационной модели в процессе работы программной системы.

**Графический пользовательский интерфейс (Graphical User Interface или GUI).** Самый популярный тип UI. Представляет собой окошко с различными элементами управления. Пользователи взаимодействуют с ними с помощью клавиатуры, мыши и голосовых команд: жмут на кнопки, тыкают мышкой, смахивают пальцем.

**XAML (Extensible Application Markup Language — расширяемый язык разметки приложений)** представляет собой язык разметки, используемый для создания экземпляров объектов .NET. Хотя язык XAML — это технология, которая может быть применима ко многим различным предметным областям, его главное назначение — конструирование пользовательских интерфейсов WPF. Другими словами, документы XAML определяют расположение панелей, кнопок и прочих элементов управления, составляющих окна в приложении WPF.

Состав блоков модульной схемы: что тут писать? Все страницы с окнами описывать?

2.3.3 Организация доступа к объектам базы данных

В WPF привязка (binding) является мощным инструментом программирования, без которого не обходится ни одно серьезное приложение.

Привязка подразумевает взаимодействие двух объектов: источника и приемника. Объект-приемник создает привязку к определенному свойству объекта-источника. В случае модификации объекта-источника, объект-приемник также будет модифицирован. [6]

Возьмем для примера страницу с недвижимостью, где участвует DataGrid. DataGrid страницы с фильмами представлен на рисунке 2.17.



Рисунок 2.17 – DataGrid страница с недвижимостью

У Binding в свойство Path мы записываем свойство объекта источника. Привязка объекта с данными к DataGrid представлена на рисунке 2.18.



Рисунок 2.18 – Привязка объекта с данными к DataGrid

Беря данные из базы данных, мы привязываем их с DataGrid. Сам DataGrid верстается под конкретно содержимое.

2.3.4 Разработка блох-схем алгоритмов процедур и функций

Основной функциональной задачей приложения является покупка недвижимости и занесение данных о транзакции в таблицу. Для этого необходимо выбрать предложение о продаже недвижимости.

Метод buyButton\_Click предназначен для добавления новых записей в базу данных о транзакции. Изменение записей о владельце недвижимости и изменение записей о сумме счета владельца и покупателя. У метода нет входных и выходных данных. Блок-схема метода buyButton\_Click представлена на рисунке 2.19.

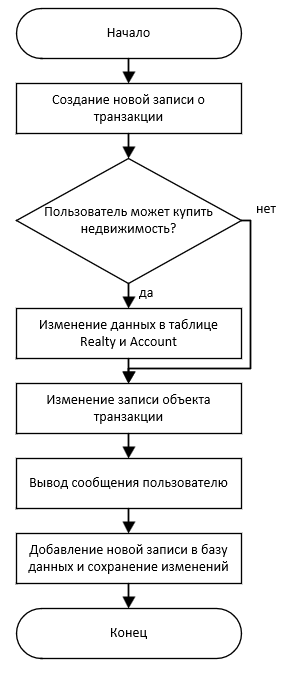


Рисунок 2.19 – Блок-схема метода buyButton\_Click

2.4 Обеспечение коллективного доступа. Защита информации

Основная идея ролевой модели контроля за доступом (Role-Based Access Control — RBAC) основана на максимальном приближении логики работы системы к реальному разделению функций персонала в организации.

Ролевой метод управления доступом контролирует доступ пользователей к информации на основе типов их активностей в системе. Применение данного метода подразумевает определение ролей в системе. Понятие роль можно определить, как совокупность действий и обязанностей, связанных с определенным видом деятельности. Таким образом, вместо того, чтобы указывать все типы доступа для каждого пользователя к каждому объекту, достаточно указать тип доступа к объектам для роли. А пользователям, в свою очередь, указать их роли. Пользователь, «выполняющий» роль, имеет доступ, определенный для роли [7].

Для авторизации пользователю необходимо ввести свой логин и пароль. В случае если пользователь не зарегистрирован, он сможет перейти на страницу регистрации из окна авторизации. Окно авторизации представлено на рисунке 2.21.

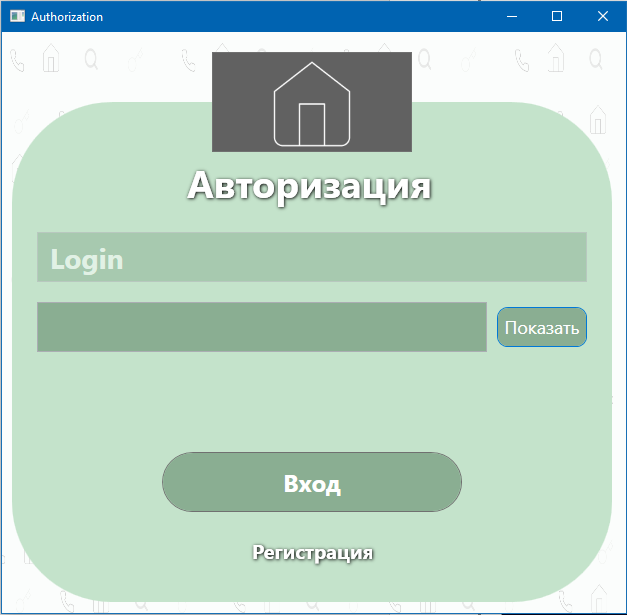


Рисунок 2.21 – Окно авторизации

Для регистрации пользователю необходимо ввести логин и пароль. Все поля обладают своей валидацией и в случае некорректного ввода данных, появится окно с предупреждением. При желании пользователь может вернуться на окно с авторизацией. Окно регистрации представлено на рисунке 2.22.

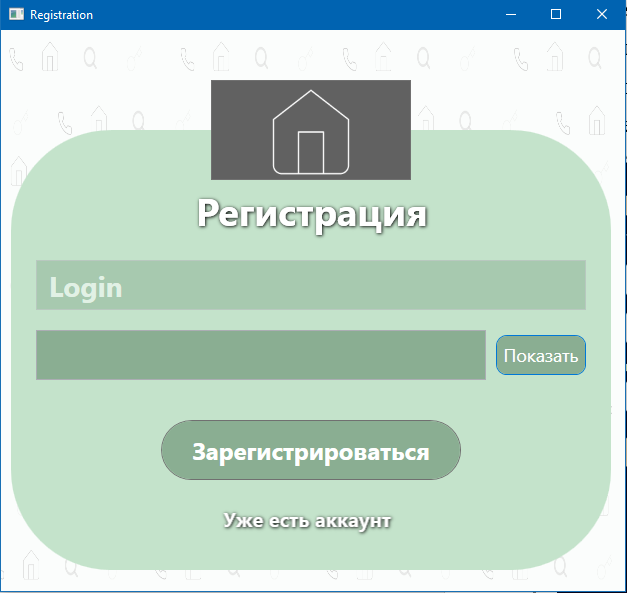


Рисунок 2.22 – Окно регистрации

После ввода всех данных, пользователю необходимо ввести капчу для сохранения нового пользователя в базе данных Окно капчи представлено на рисунке 2.23.

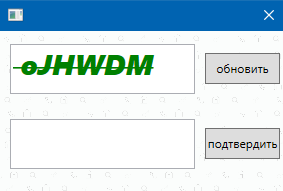


Рисунок 2.23 – Окно капчи

3 Технологическая часть

3.1 Тестирование и отладка приложения

Отладка — этап разработки компьютерной программы, на котором обнаруживают, локализуют и устраняют ошибки, информация из работы [8]. В связи с тем, что почти невозможно составить реальную программу без ошибок, и почти невозможно для достаточно сложной программы быстро найти и устранить все имеющиеся в ней ошибки. Разумно уже при разработке программы на этапах алгоритмизации и программирования готовиться к обнаружению ошибок на стадии отладки принимать профилактические меры по их предупреждению, информация из работы [9].

Тестирование будет происходить через тест кейсы. Тест кейс – это артефакт, описывающий совокупность шагов, конкретных условий и параметров, необходимых для проверки реализации тестируемой функции или её части. При передаче тестировщику тест-кейсов, он должен пройтись по всем его пунктам и выполнить описанные действия, которые должны привести к определенным результатам. Информация из работы [10]. Тест кейс для функций представлен в таблице 3.1.

Таблица 3.1 – Тест-кейс для методов

|  |  |  |
| --- | --- | --- |
| Имя метода | Управляющее воздействие | Результат воздействия |
|  |  |  |

3.2 Инструкция администратора базы данных

Перед началом работы с приложением необходимо установить и настроить SQL Server 2019. MS SQL Server это лидирующая РСУБД (Реляционная система управления базами данных) а также главный конкурент Oracle Database в корпоративном сегменте. В СНГ MSSQL чаще всего применяется для собственных разработок прикладного ПО и для 1С.

Для установки переходим на официальный сайт Microsoft и скачиваем бесплатную версию SQL Server 2019 для тестирования и разработки (Developer). Далее запускаем установщик и выбираем тип установки «Пользовательский». Как на рисунке 3.1

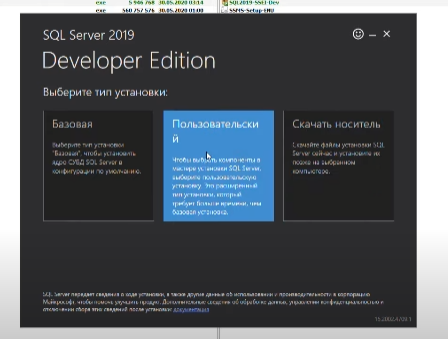


Рисунок 3.1 – «Пользовательский» тип установки

После выбора типа установки открывается следующее окно где предлагается выбрать язык и место расположения носителя, можно выбрать стандартные настройки и нажать на кнопку «Установить». После чего начнется процесс загрузки. Изображение окна представлено на рисунке 3.2.

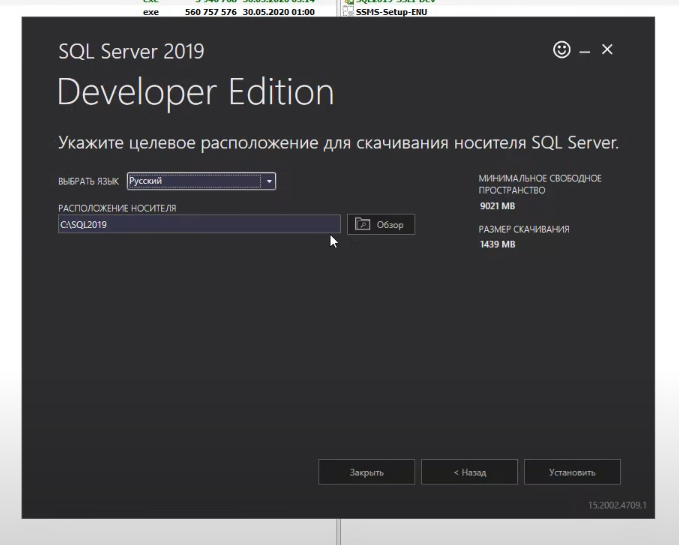


Рисунок 3.2 – Изображение окна

После установки откроется центр установки SQL server, где мы переходим в раздел установки «Новая установке изолированного экземпляра SQL Server или добавление компонентов к существующей установке», как показано на рисунке 3.3.

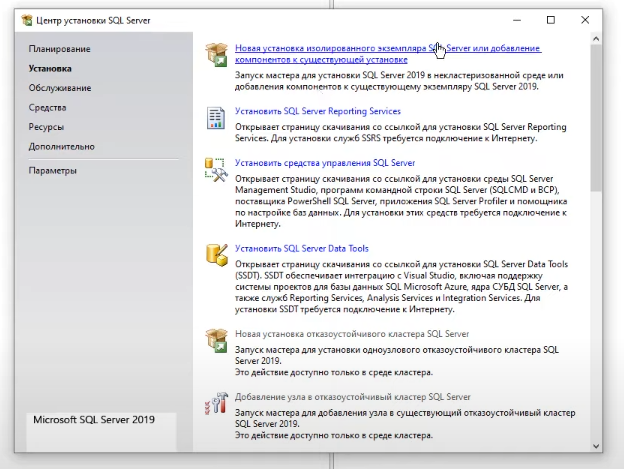


Рисунок 3.3 – Новая установке изолированного экземпляра SQL Server

После установки произойдет обновление продукта. Рисунок 3.4.

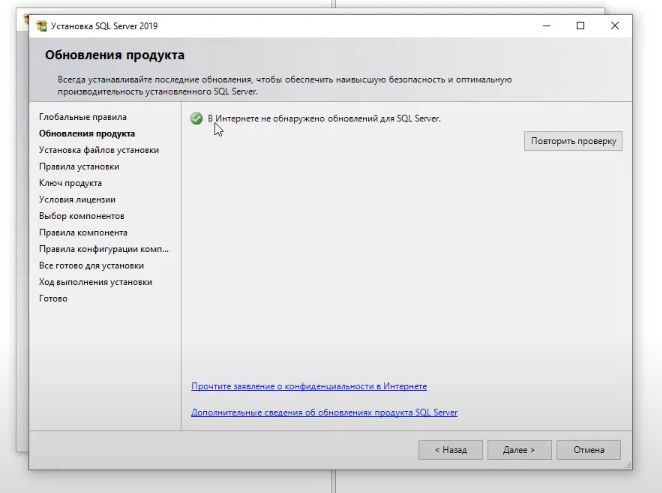


Рисунок 3.4 – Обновление продукта

После обновления пропускаем все пункты до ключа продукта и выбираем версию «Developer» как на рисунке 3.5.

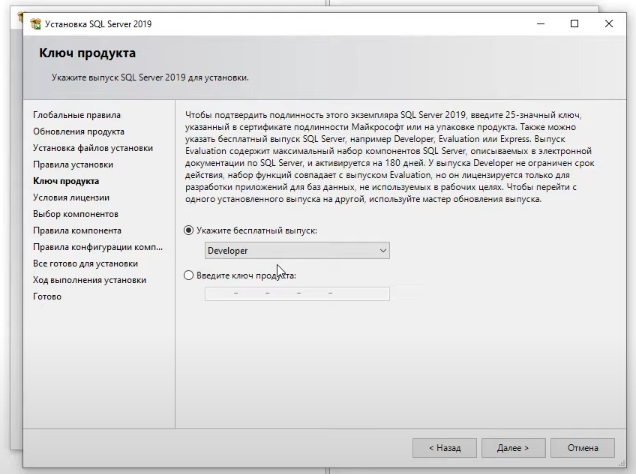


Рисунок 3.4 – Версия «Developer»

В пункте «Условия лицензии» принимаем условия и переходим в раздел «Выбор компонентов», где установим базовый набор компонентов: «Служба ядра СУБД» и «Полнотекстовой и семантический поиск». Как представлено на рисунке 3.5.

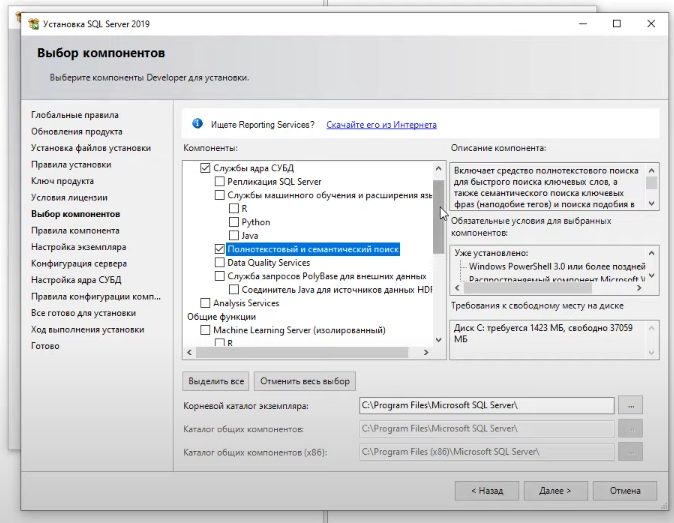


Рисунок 3.4 – Выбор компонентов

В разделе правила компонента всё оставляем по умолчанию и переходим в раздел «Конфигурация сервера» где можно настроить работу служб SQL Server. Задать тип запуска какой-либо службы. Поставить ее на автозапуск, вручную, или отключить. Так же можем зайти в меню "Параметры сортировки" — это настройки таблицы кодировок. Выполнять сортировку, как учитывать верхний и нижний регистр, как реагировать на символы, и т.п. Настройки можно оставить по умолчанию как на рисунке 3.5.

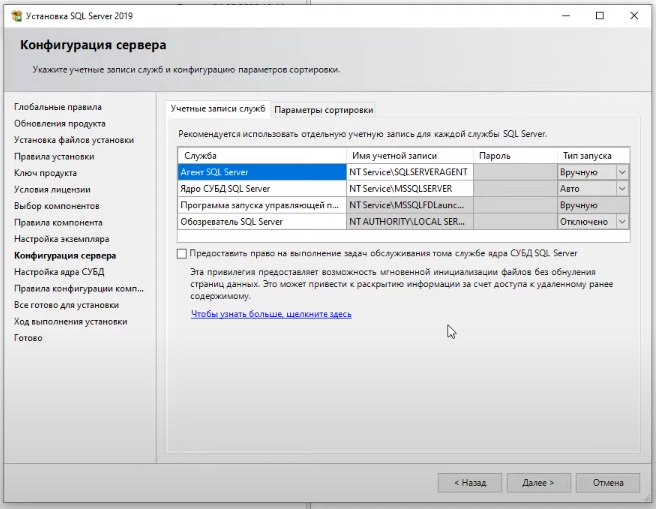


Рисунок 3.5 – Конфигурация сервера

В разделе «Настройка ядра СУБД» нам предлагают выбор режим входа под учетными записями Windows, либо смешанный режим, т.е. возможность входа под учетной записью Windows и под учетной записью SQL Server, если выбрать смешанную, то вам предложат создать учетную запись SQL Server. Оставляем режим аутентификации Windows и выбираем пользователя как показано на рисунке 3.6.

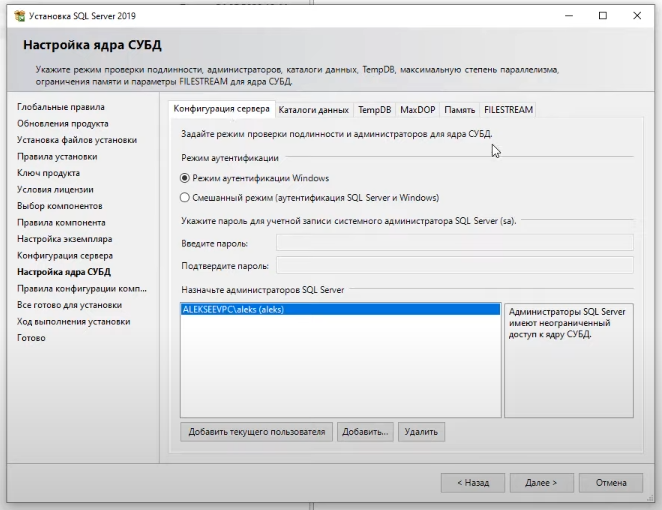


Рисунок 3.6 – Настройка ядра СУБД

В разделе «Все готово для установки» можно сверится с выбранными настройками» и начать установку. Окно с разделом представлено на рисунке 3.7.

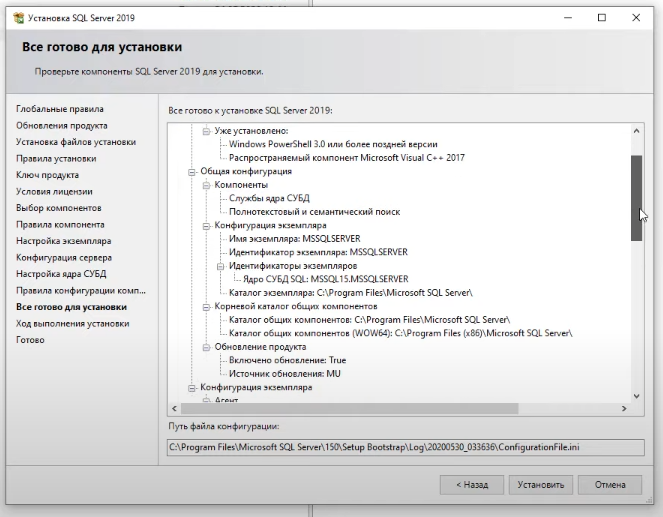


Рисунок 3.7 – Все готово для установки

Установка завершена и можно закрыть окно. Финальный экран представлен на рисунке 3.8.

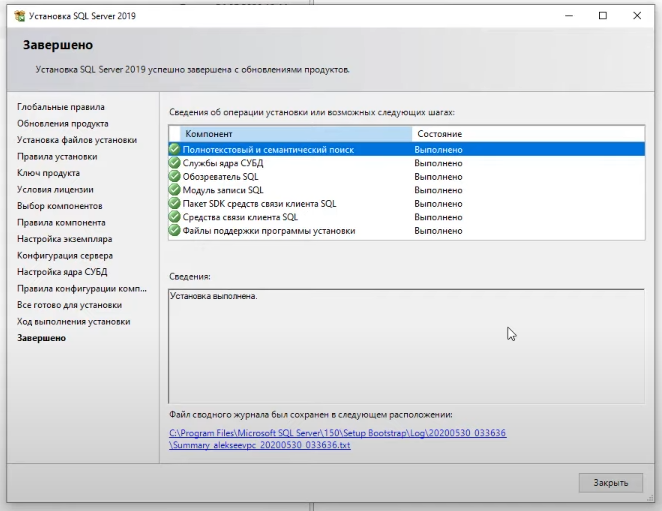


Рисунок 3.8 – Финальный экран

Если все прошло успешно, закрываем окно. После того, как установка SQL Server 2019 завершена, нам нужно установить приложение, с помощью которого мы будем подключаться к серверу баз данных. Это приложение SQL Server Management Studio (SSMS).

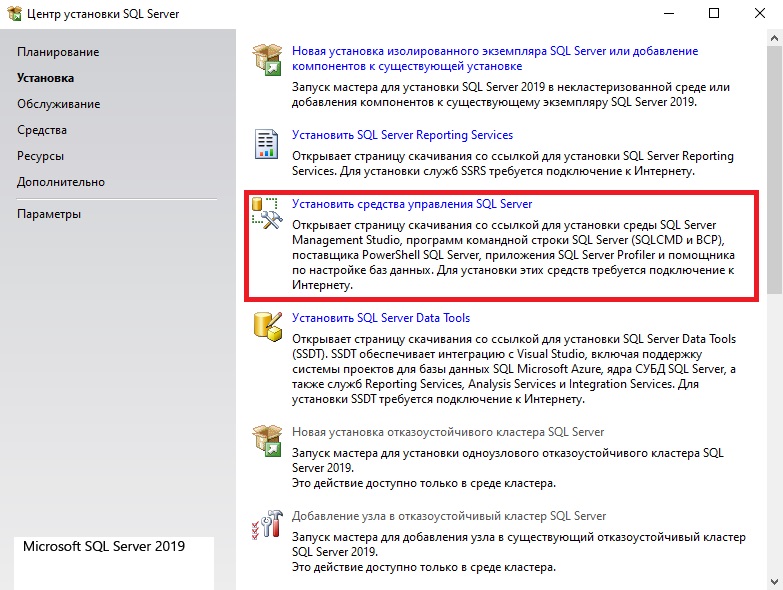
Заходим снова в центр установки SQL Server и наживаем "Установить средства управления SQL Server", как показано на рисунке 3.9.

Рисунок 3.9 – Установка SQL Server Management Studio

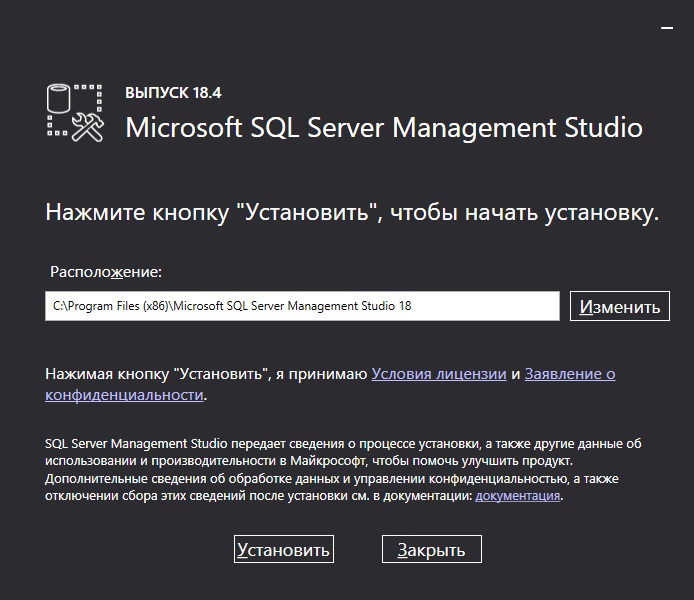
При нажатии у нас откроется [сайт Microsoft](https://docs.microsoft.com/ru-ru/sql/ssms/download-sql-server-management-studio-ssms?redirectedfrom=MSDN&view=sql-server-ver15) и нам нужно будет скачать SSMS. Нажимаем "Установить", как показано на рисунке 3.10.

Рисунок 3.10 – Установка SQL Server Management Studio

Реляционная схема базы данных представлена на рисунке 3.11.

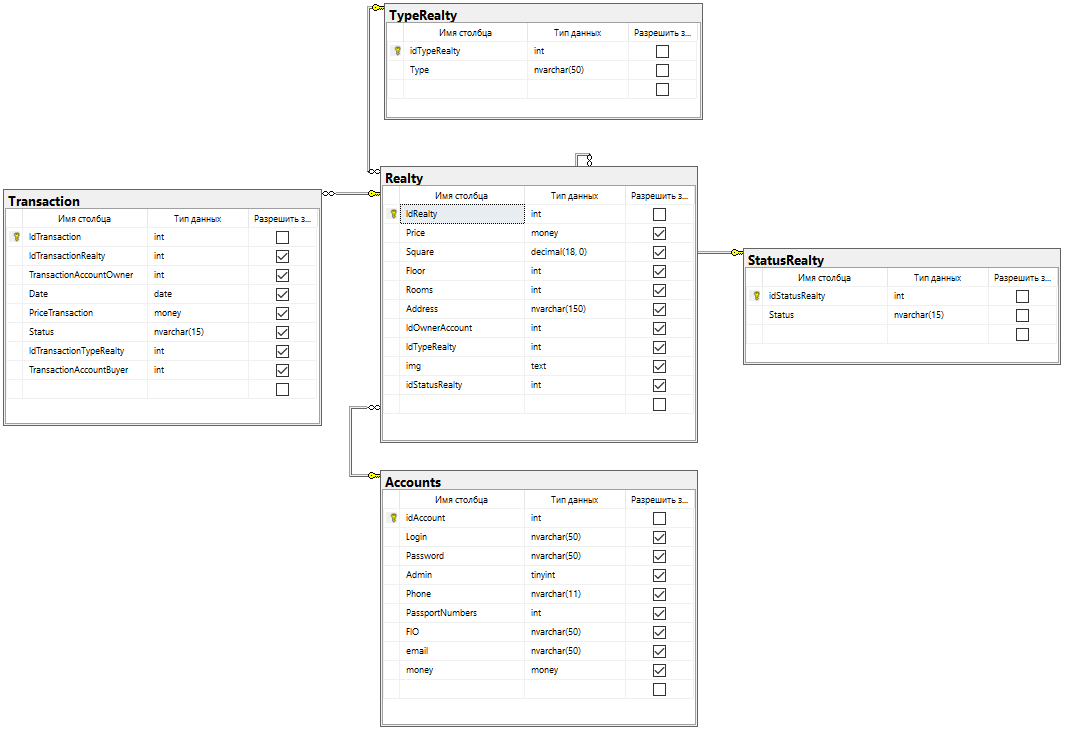


Рисунок 3.11 – Реляционная схема базы данных

Схема базы данных состоит из шести, описывающих сущности, отношений:

* Accounts – Аккаунты;
* TypeRealty – Тип недвижимости;
* StatusRealty – Статус недвижимости;
* Realty – Недвижимость;
* Transaction – Транзакция;

3.3 Инструкция по эксплуатации приложения

Приложение предназначено для удобной покупоки и продажи недвижимости. Пользователь может установить приложения, после чего пройти регистрацию и в любой момент воспользоваться им, просматривая различные варианты покупки, выбирая понравившиеся недвижимость, после чего купить её для себя или своих близких. Так же пользователь может продать свою недвижимость.

Чтобы начать работать с приложение достаточно его запустить. После запуска, приложение готово к работе. Для доступа к справочникам должен войти пользователь со статусом администратора. Тогда ему будет административная панель, позволяющая начать работать со справочниками. Окно с авторизованным пользователем с правами администратора представлено на рисунке 3.12.

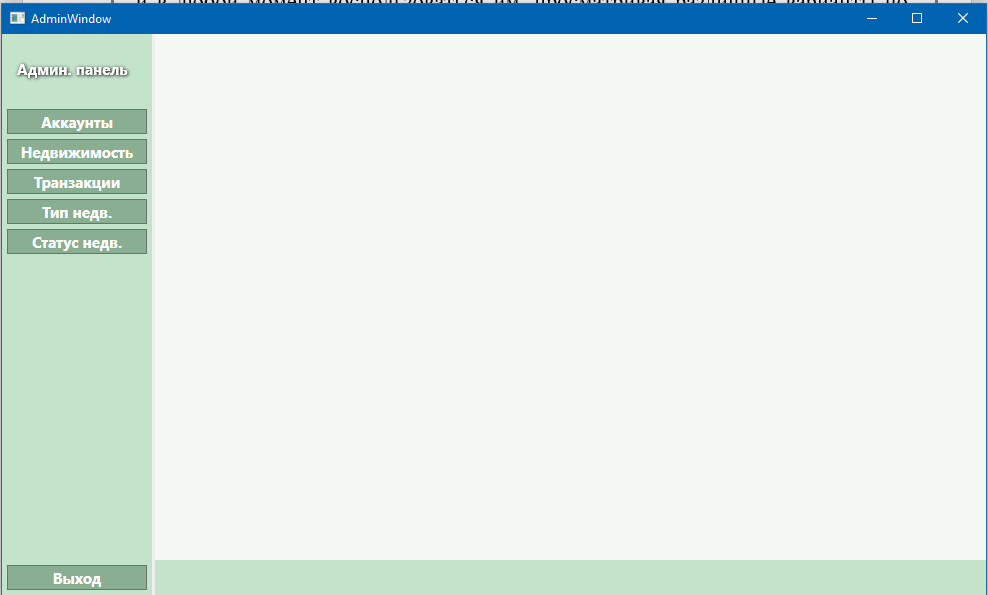


Рисунок 3.12 – Окно с авторизованным пользователем с правами администратора

Для взаимодействия со справочником необходимо выбрать нужный из списка, представленного в верхнем левом углу. Список состоит из кнопок:

* Аккаунты – справочник с учетными записями;
* Недвижимость – справочник с недвижимостью;
* Транзакции – справочник с транзакциями;
* Тип недвижимости – справочник с типом недвижимости;
* Статус недвижимости – справочник со статусом недвижимости;
* Выход – кнопка возврата в окно авторизации;

При выборе откроется справочник как на рисунке 3.14.

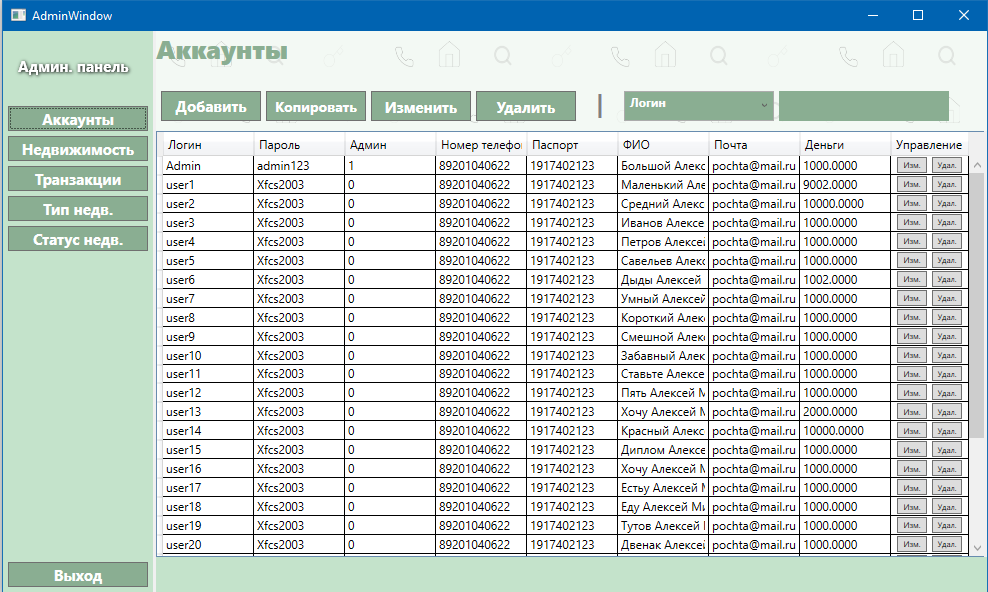


Рисунок 3.14 – Справочник с учетными записями

Справочник представляет собой окно с расположенной в нём таблицей, заполненной данными, меню с операциями управления данными и фильтрацией данных. Для работы с ними предоставляется четыре действия:

* Добавление;
* Копирование;
* Изменение;
* Удаление;

Подобным образом выглядят справочники с сеансами и пользователями. При выборе действия добавления, копирования или удаления появится колонка, позволяющая ввести данные. Окно с колонкой представлено в окне 3.15.

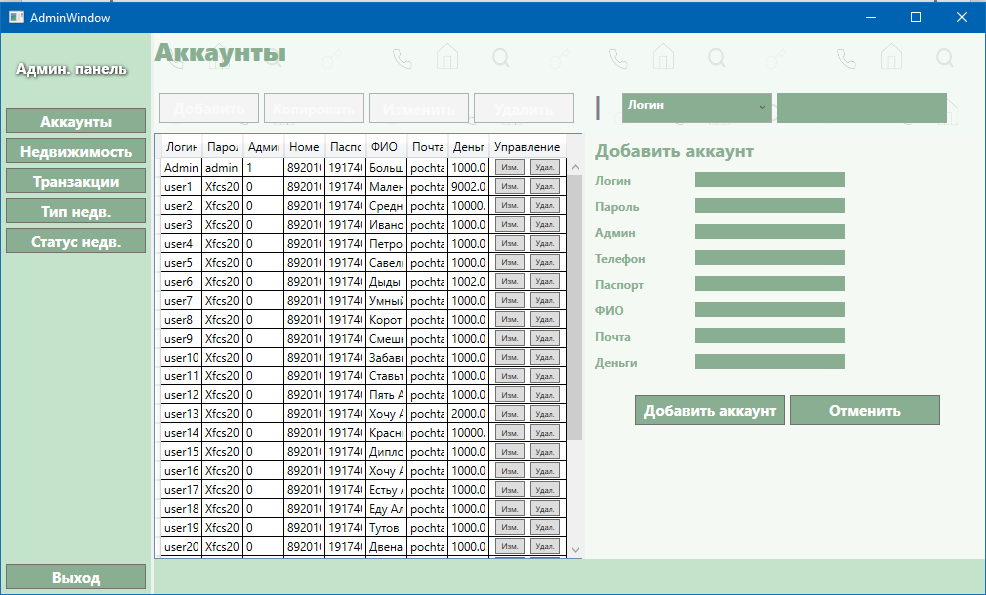


Рисунок 3.15 – Окно с колонкой

Для выбора недвижимости, пользователю необходимо нажать на объявление продажи недвижимости. Пример представлен на рисунке 3.18.

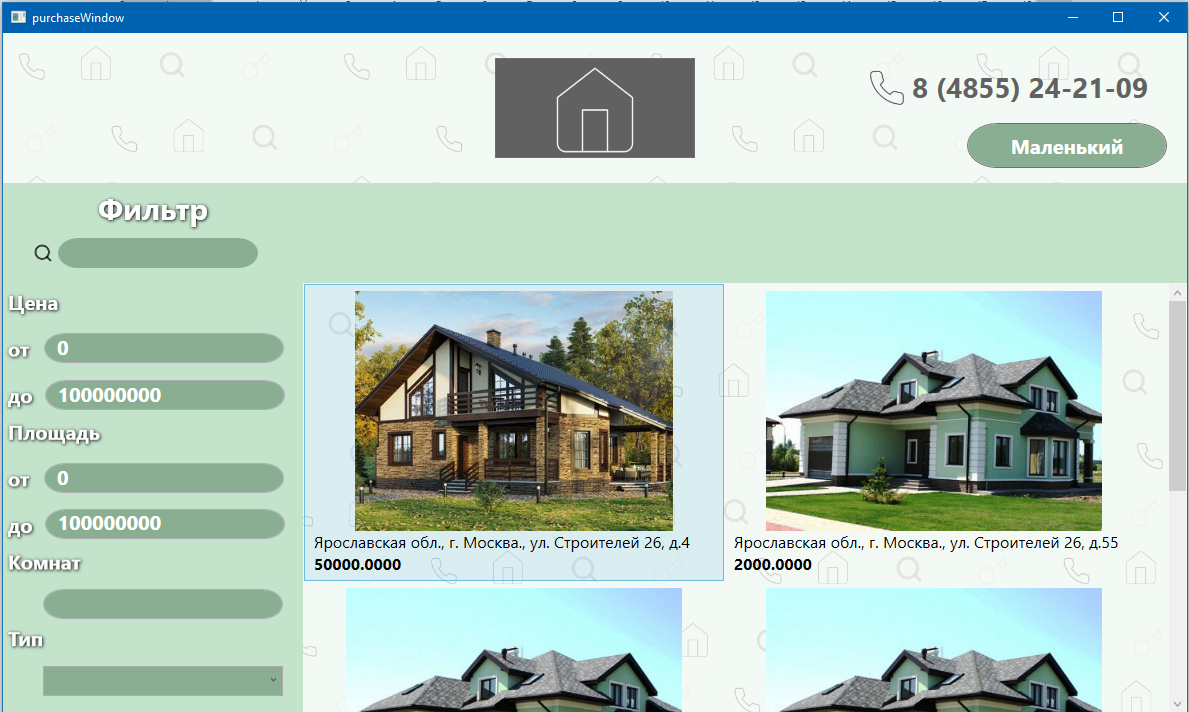


Рисунок 3.18 – Пример выбора недвижимости

После чего можно ознакомиться с информацией о недвижимости. После ознакомления пользователь может купить недвижимость. Вид окна представлен на рисунке 3.19.



Рисунок 3.19 – Вид окна недвижимости

В личной кабинете вы можете добавить свою недвижимость. Вид окна представлен на рисунке 3.20.

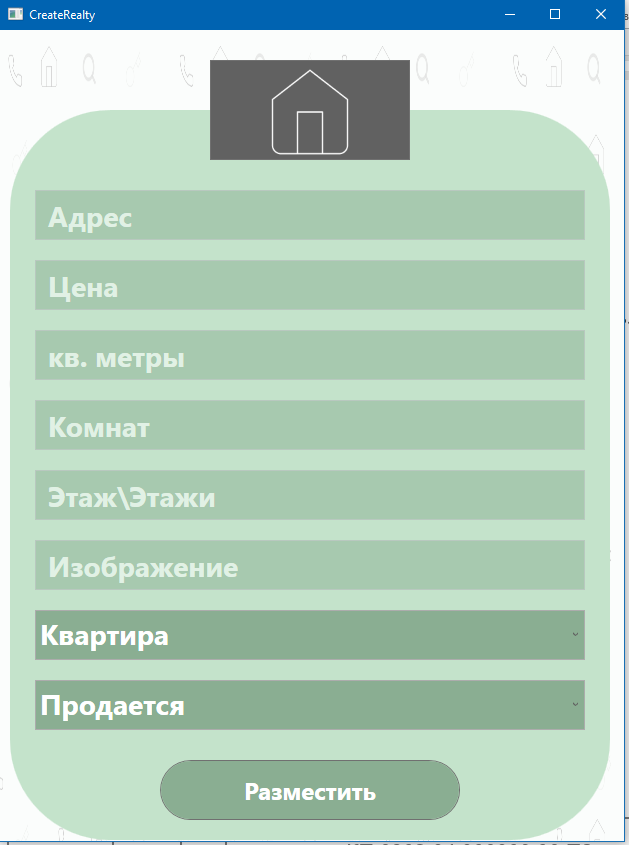


Рисунок 3.20 – Вид окна добавления недвижимости

После добавления недвижимости можно её удалить в личной кабинете правой клавишой мыши, изменить данные о недвижимости двойным нажатием левой клавиши мыши. Вид окна личный кабинет представлен на рисунке 3.21.

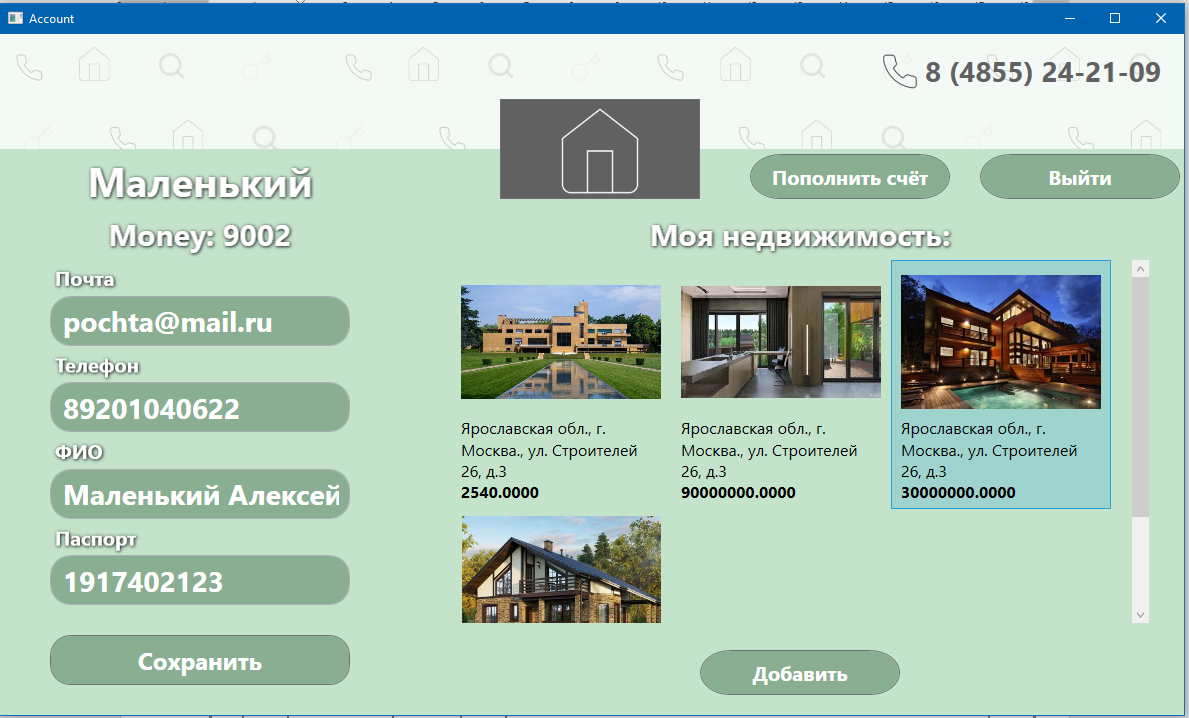


Рисунок 3.21 – Вид окна личный кабинет

Так же можно пополнить счет своей учетной записи на кнопку пополнить счет. Откроется окно пополнения счета, где нужно ввести сумму, которую хотите добавить к своему кошельку. Вид окна пополнения счета на рисунке 3.21.

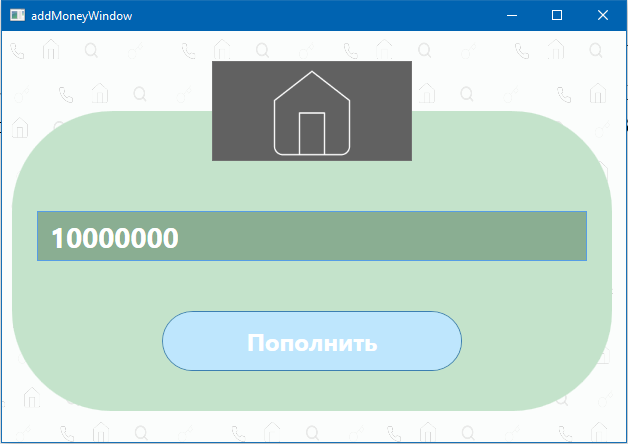


Рисунок 3.21 – Вид окна пополнения счета

Для просмотра новостей, отзывов и информации о компанни пользователю нужно нажать на соответствующую надпись, откроется окно с соотвествующим текстом. Вид окна с новостями на рисунке 3.21.

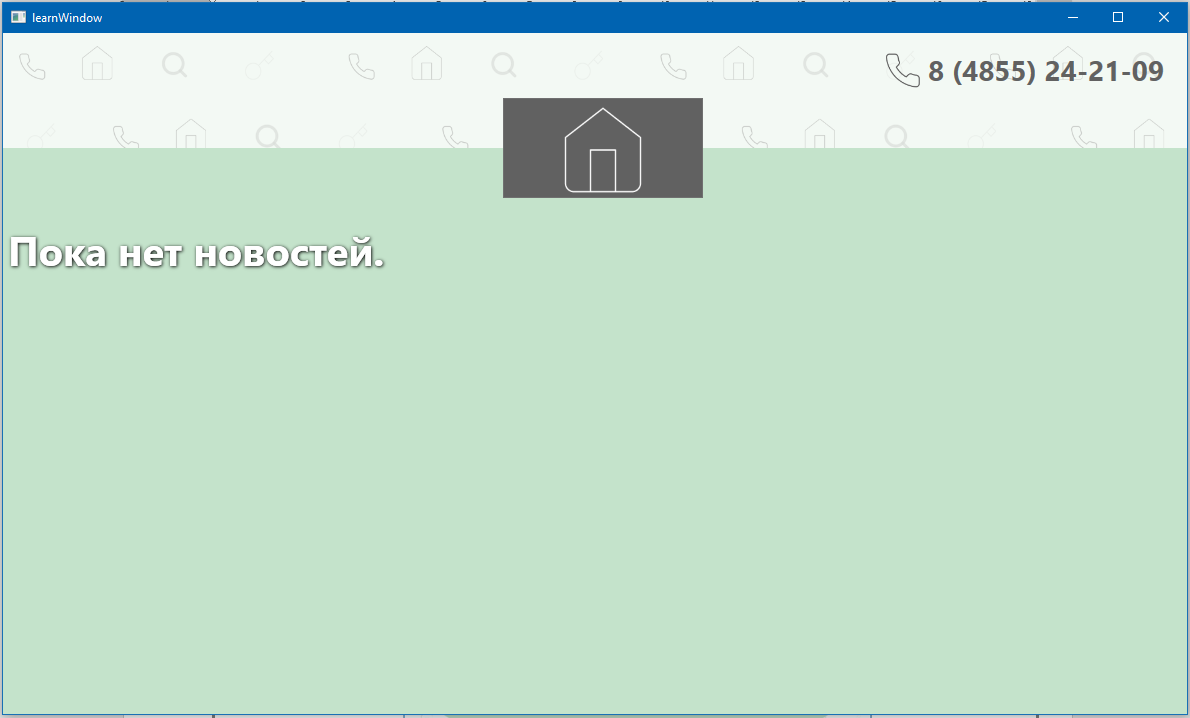


Рисунок 3.21 – Вид окна с новостями

4 Раздел охраны труда

Охрана труда – это целая система законодательных и нормативно-правовых актов, технических, гигиенических, лечебно-профилактических мероприятий и средств, которые обеспечивают безопасность, сохранение здоровья и работоспособности человека в процессе труда. В наши дни труд стал более интенсивным и требует огромных затрат умственной, эмоциональной и физической нагрузок.

На рабочем месте программист осуществляет трудовую деятельность и проводит большую часть рабочего времени. Правильная организация рабочего места программиста повышает производительность труда от 8 до 20%. Следуя рекомендациям ГОСТ 12.2.032-78, необходимо организовать рабочее место таким образом, чтобы взаимное расположение всех его элементов соответствовало физическим и психологическим требованиям. Главные элементы рабочего места программиста – это письменный стол и кресло. Рабочее место организуется в соответствии с ГОСТ 12.2.032-78, информация из работы [11].

Площадь рабочего места с компьютером с жидкокристаллическим или плазменным экраном должна быть не менее 4,5 кв. м, а расстояние между столами с мониторами (от тыла одного монитора до экрана другого) не менее 2 м. Монитор должен располагаться на расстоянии 50-70 см от глаз программиста. Параметры рабочего стола сотрудника: возможность регулировки высоты рабочего стола, или точная высота — 72,5 см, ширина — 80, 100, 120 или 140 см, глубина рабочего стола 80 или 100 см, высота и ширина пространства под столешницей (для ног) – не менее 50 см, глубина на уровне колен не менее 45 см, а на уровне вытянутых ног не менее 65 см.

Правильное освещение рабочего места – это очень важный момент в трудовой деятельности человека, влияющий на эффективность труда, при этом такой момент предупреждает травматизм и профессиональные заболевания. При недостаточном освещении приходится напрягать зрение, при этом ослабляется внимание и это приводит к наступлению преждевременной утомленности. Слишком яркое освещение тоже плохо, так как оно вызывает ослепление, раздражение и резь в глазах. При искусственном освещении, источниками света служат два вида ламп: лампы накаливания и люминесцентные.

Известно, что шум ухудшает условия труда и оказывает вредное воздействие на организм человека. Согласно ГОСТ 12.1.003-88 «Шум для помещений расчетчиков и программистов, уровни шума не должны превышать соответственно: 71, 61, 54, 49, 45, 42, 40, 38 дБ», информация из работы [12].

При работе компьютерной техники выделяется много тепла, что может привести к пожароопасной ситуации. Источниками зажигания так же могут служить приборы, применяемые для технического обслуживания, устройства электропитания, кондиционеры воздуха. Серьёзную опасность представляют различные электроизоляционные материалы, используемые для защиты от механических воздействий отдельных радиодеталей. В связи с этим, участки, на которых используется компьютерная техника, по пожарной опасности относятся к категории пожароопасных “В”. При пожаре люди должны покинуть помещение в течение минимального времени. В помещениях с компьютерной техникой, недопустимо применение воды и пены ввиду опасности повреждения или полного выхода из строя дорогостоящего электронного оборудования. Для тушения пожаров необходимо применять углекислотные и порошковые огнетушители, которые обладают высокой скоростью тушения, большим временем действия, возможностью тушения электроустановок, высокой эффективностью борьбы с огнем. Воду разрешено применять только во вспомогательных помещениях, информация из работы [13].

Заключение

По итогу работы было разработано приложение, позволяющее пользователю ознакомиться с рынком недвижимости и выбрать недвижимость для покупки или продажи. Для удобства пользователь может отсортировать недвижимость в приложения для того, чтобы выбрать нужный ему вариант. Для пользователей с правами администратора есть доступ к редактированию таблиц из базы данных.

Главным достоинством можно выделить приятный интерфейс и простоту в использовании приложения. Все дейсвтия выполняются на интуитовно понятном уровне.

Приложение можно использовать для любого агенства с недвижимостью.

Разработанное приложение можно доработать, добавив возможность аренды недвижимости. А также добавив электронную кассу для легализации денежных средств, договора купли-продажи и проверки документов кадастрового учета и регистрации прав.

Список используемых источников

1. <https://kazedu.com/referat/133091/5>
2. <https://ruprogi.ru/software/visual-studio>
3. <https://gb.ru/posts/c_sharp_ides>
4. <https://kosmos-kino.ru/>
5. <https://learn.microsoft.com/ru-ru/dotnet/desktop/wpf/overview/?view=netdesktop-6.0>
6. <https://metanit.com/sharp/wpf/11.php>
7. <https://steptosleep.ru/ролевая-модель/#:~:text=Основная%20идея%20ролевой%20модели%20контроля,типов%20их%20активностей%20в%20системе>
8. <https://studopedia.ru/22_29871_neobhodimost-otladki-programmnogo-produkta.html>
9. <https://infopedia.su/4x1ec5.html>
10. <https://sergeygavaga.gitbooks.io/kurs-lektsii-testirovanie-programnogo-obespecheni/content/lektsiya-4-ch3.html>
11. <https://www.retail.ru/rbc/pressreleases/tsentr-povysheniya-kvalifikatsii-lider-organizatsiya-rabochego-mesta-ofisnogo-rabotnika/>
12. <https://xn--d1aux.xn--p1ai/opisanie-rabochego-mesta-programmista-na-predpriyatii/>
13. <https://studopedia.ru/8_107307_osveshchenie-pomeshcheniy-vichislitelnih-tsentrov.html>

Приложение А