Содержание

1. Бизнес-анализ 3

1.1. Интервьюирование заказчика и пользователей 3

1.2. Описание текущей ситуации 5

1.3. Описание будущей ситуации 7

1.4. Описание заинтересованных лиц и конечных пользователей 9

1.5. Бизнес-требования 11

2. Анализ существующих решений 13

2.1. Обзор существующих решений 13

2.1.1. Решение 1 «Intrum CRM» 13

2.1.2. Решение 2 «1С:Риэлтор» 13

2.1.3. Решение 3 «Риелтор Про Web» 13

2.2. Сравнительные анализ решений 14

2.3. Выводы по анализу 15

3. Концепция решения 17

3.1. Техническое задание 17

3.1.1. Общие сведения 17

3.1.2. Назначение и цели создания системы 17

3.1.2.1. Назначение системы 17

3.1.2.2. Цели системы 17

3.1.3. Требования к системе 18

3.1.3.1. Требования к структуре и функционированию системы. 18

3.1.3.2. Требования к численности и квалификации пользователей 18

3.1.3.3. Требования к составу и параметрам технических средств 19

3.1.3.4. Требования к информационной и программной совместимости 19

Требования перечислены в Таблице 3.1. 19

3.1.3.5. Требования к защите, безопасности и сохранности данных. 19

3.1.4. Состав и содержание работ по созданию системы 19

3.2. Модель прецедентов 20

3.2.1. Акторы 20

3.2.2. Диаграмма прецедентов 20

3.2.3. Сценарии использования 21

3.3. Customer Journey Mapping 23

3.4. User Story Mapping 23

3.5. Анализ факторов риска проекта 24

4. Проектирование решения 27

4.1. Описание архитектуры системы 27

4.2. Описание возможных технических решений 27

4.3. Проектирование интерфейса пользователя 31

4.4. Проектирование базы данных 31

5. Программная реализация системы 33

6. Функциональное тестирование системы 34

7. Список литературы 37

Приложение 1 – Листинг базы данных 38

Приложение 2 – Листинг программной реализации системы 49

1. Бизнес-анализ
   1. Интервьюирование заказчика и пользователей

Для успешного проектирования и разработки системы автоматизации взаимодействия риэлторов и клиентов было проведено интервьюирование заказчика. Целью интервью было выявление ключевых потребностей, проблем и ожиданий, связанных с процессом хранения информации о клиентах и объектах недвижимости, формирования договоров на продажу и покупку, а также анализ текущих процессов и их недостатков.

Вопросы и ответы заказчика представлены в Таблице 1.1.

Таблица 1.1 – Вопросы и ответы заказчика.

|  |  |
| --- | --- |
| Вопрос | Ответ |
| Какие основные проблемы вы видите в текущем процессе взаимодействия риэлторов и клиентов? | Основные проблемы включают сложность хранения данных клиентов и объектов, отсутствие автоматизации в процессе формирования договоров и ручное ведение учета сделок, что затрудняет принятие управленческих решений. |
| Какие функции вы считаете обязательными для новой системы? | Обязательными функциями являются: управление записями клиентов и объектов, автоматизация формирования договоров, возможность хранения различных документов. |
| Какие требования к производительности системы вы предъявляете? | Система должна работать стабильно даже в пиковые периоды, обеспечивать минимальное время отклика (не более 5 секунд) и поддерживать одновременную работу нескольких пользователей. |

Продолжение Таблицы 1.1

|  |  |
| --- | --- |
| Вопрос | Ответ |
| Какие данные должны быть доступны в системе для анализа и отчетности? | Необходимы данные о сделках и договорах. |

Вопросы и ответы пользователей представлены в Таблице 1.2.

Таблица 1.2 – Вопросы и ответы пользователей

|  |  |
| --- | --- |
| Вопрос | Ответ |
| Какие функции новой системы облегчили бы вашу работу? | Удобный интерфейс для работы с базой клиентов и объектов, также автоматизация составления договора. |
| Какие данные вам необходимы для эффективной работы | Информация о доступных объектах, включая различные обязательные документы для сделок, также данные о клиентах (например, их предпочтения). |

Выводы по результатам интервью:

На основе проведенного интервьюирования были выявлены ключевые проблемы и потребности заказчика и пользователей. Основные требования к системе включают:

Упрощение процесса взаимодействия риэлторов и клиентов.

* автоматизация формирования договоров и их печать;
* возможность хранения документов для объектов в одном месте;
* просмотр статистики по оформленным договорам;
* удобный интерфейс для сотрудников.

Эти данные будут использованы для проектирования функциональных и нефункциональных требований к системе, а также для разработки архитектуры модуля взаимодействия риэлторов и клиентов.

* 1. Описание текущей ситуации

Для более детального анализа текущей ситуации были описаны основные бизнес-процессы в нотации IDEF0 (Рисунок 1.1 – 1.2.).

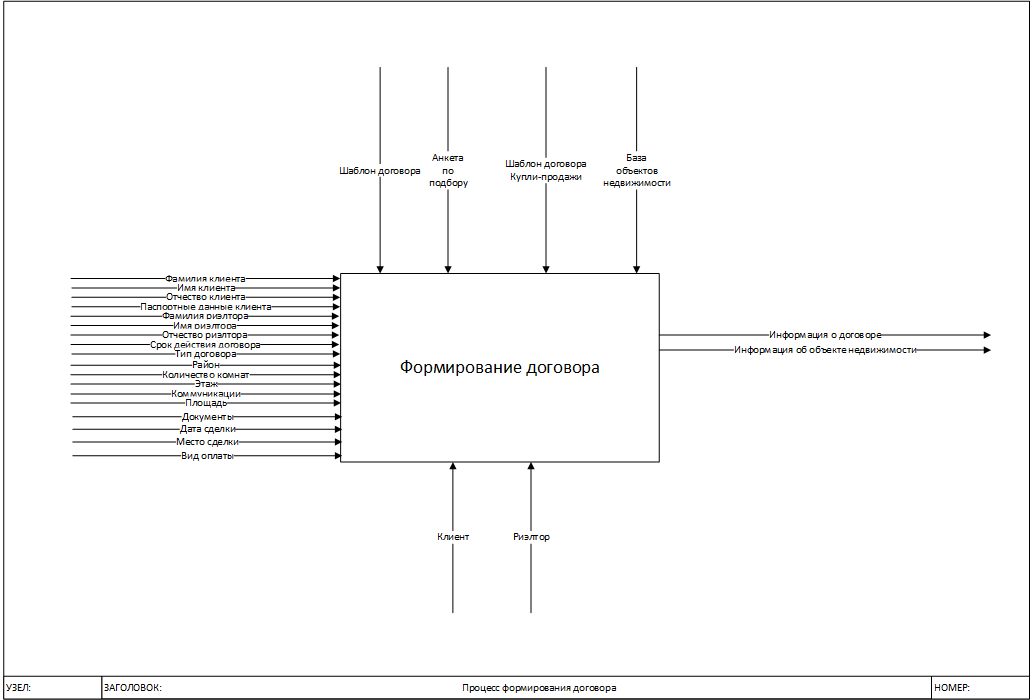


Рисунок 1.1 – IDEF0 до внедрения системы.

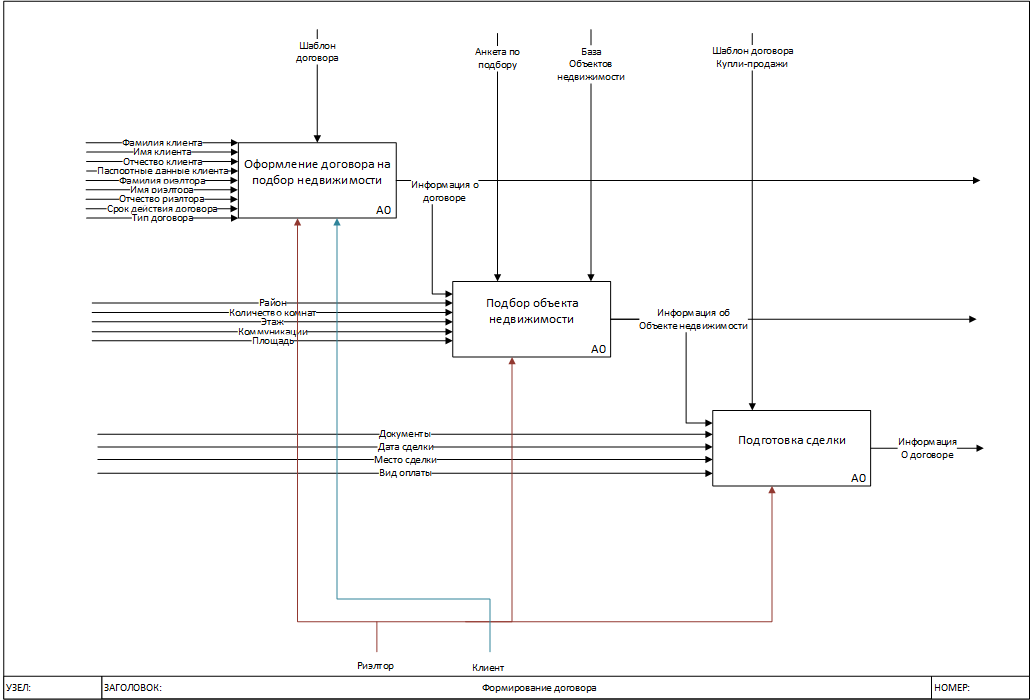


Рисунок 1.2 – IDEF0 до внедрения системы.

Для работы агентства и удовлетворенность клиентов на основе проведенного интервьюирования и анализа бизнес-процессов были выявлены ключевые трудности и их последствия, представленные в Таблице 1.3.

Таблица 1.3 – Ключевые трудности и последствия.

|  |  |  |
| --- | --- | --- |
| Проблема | Бумаги теряются и портятся, что доставляет неудобство, а также могут быть не в том месте, где они требуются. | Много времени уходит на регулярный ручной ввод имеющихся данных. |
| Воздействует на | Риэлторов | Риэлторов |
| Результатом чего является | Утрата важных документов, потеря эффективности и времени на восстановление. | Необходимость ручного ввода данных, ошибки при оформлении записей, сложность генерации отчетов. |
| Сейчас решается | Копии файлов хранят в мессенджерах и на почте. | Ручной перенос данных между системами. |

Продолжение Таблицы 1.3

|  |  |  |
| --- | --- | --- |
| Имеющиеся решения не | Обеспечивают стабильность и надежность работы | Автоматизируют процесс и не обеспечивают точность данных. |

Текущая ситуация демонстрирует необходимость внедрения информационной системы, которая устранит существующие проблемы и повысит эффективность процессов.

* 1. Описание будущей ситуации

После успешного внедрения модуля информационной системы автоматизации взаимодействия риэлторов и клиентов ожидается повышение эффективности сотрудников. Будущая ситуация будет характеризоваться изменениями, представленными в Таблице 1.4.

Таблица 1.4 – Основные улучшения после внедрения системы

|  |  |  |
| --- | --- | --- |
| Улучшение | Как будет | Результат |
| Повышение стабильности и надежности | Система будет разработана с учетом высокой нагрузки и пиковых периодов, что обеспечит ее стабильную работу без сбоев. | Снижение количества простоев, повышение доверия пользователей к системе. |
| Удобный интерфейс | Система будет оснащена интуитивно понятным интерфейсом, который упростит работу и позволит сотрудникам быстро находить нужную информацию. | Снижение времени обучения сотрудников. |

Продолжение Таблицы 1.4

|  |  |  |
| --- | --- | --- |
| Улучшение | Как будет | Результат |
| Повышение эффективности | Система будет оснащена интуитивно понятным интерфейсом, который упростит работу и позволит сотрудникам быстро находить нужную информацию. | Снижение времени обучения сотрудников. |

Для более детального анализа будущей ситуации были описаны основные бизнес-процессы в нотации IDEF0 (Рисунок 1.3 – 1.4).

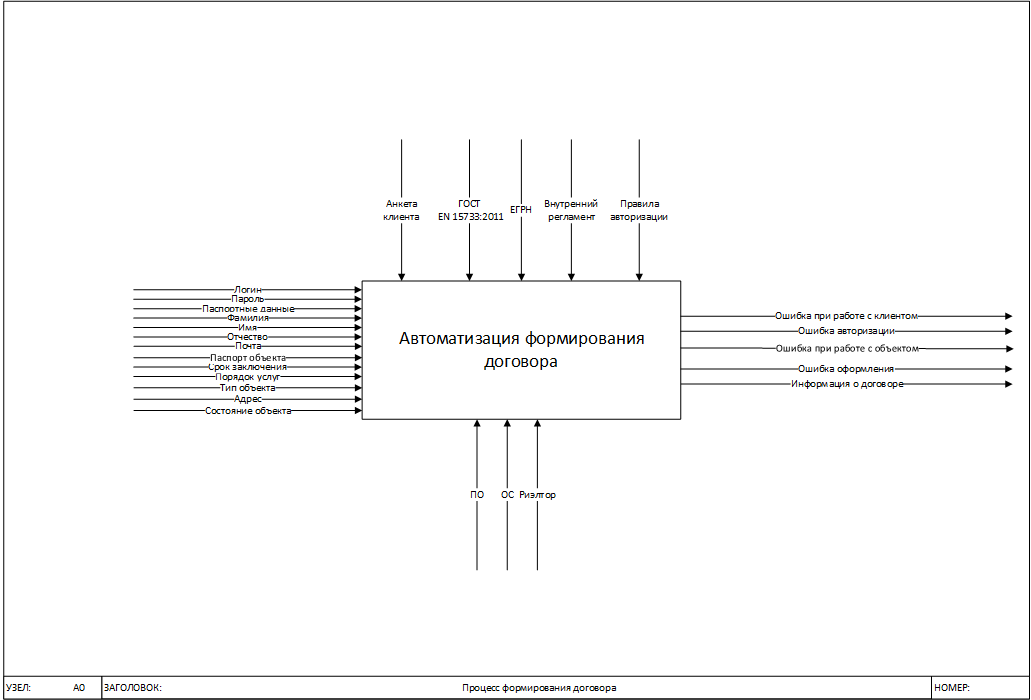


Рисунок 1.3. – IDEF0 после внедрения системы.

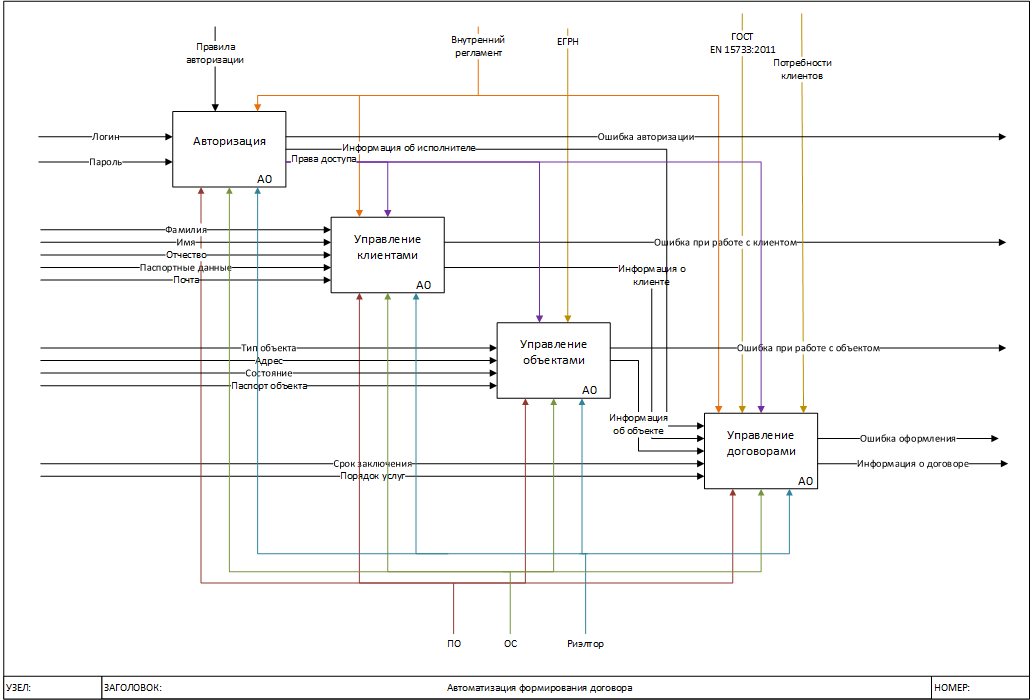


Рисунок 1.4. – IDEF0 после внедрения системы.

* 1. Описание заинтересованных лиц и конечных пользователей

Для успешного проектирования и разработки модуля информационной системы автоматизации взаимодействия риэлторов и клиентов необходимо определить всех заинтересованных лиц и конечных пользователей системы. В таблице ниже представлены ключевые заинтересованные лица, их роли, ответственность и вовлеченность в проект.

Таблица 1.5 – Описание заинтересованных лиц и конечных пользователей.

|  |  |  |
| --- | --- | --- |
| Представитель | Администрация агентства | Риэлтор |
| Описание | |  |  | | --- | --- | | Администратор агентства недвижимости, отвечающий за внедрение системы и безопасность данных. |  | | Сотрудники, непосредственно работающие с системой. |
| Тип | Эксперт | Профессионал |
| Ответственность | Определение требований к системе, контроль за внедрением и эксплуатацией системы. | Использование системы для управления объектами, клиентами, договорами, документами. |
| Критерий успеха | Повышение эффективности работы сотрудников. | Удобство работы с системой, минимизация ошибок, сокращение времени на оформление. |
| Вовлеченность | Рецензирование требований, защита данных, утверждение архитектурных решений, контроль за внедрением. | Тестирование системы, предоставление обратной связи по удобству интерфейса. |
| Поставляемые артефакты | Аналитика по эффективности работы системы. | Удобный интерфейс для работы, быстрый доступ к данным. |
| Комментарии / Проблемы | Недостаточная техническая подготовка для оценки сложных технических решений. | Необходимость обучения работе с новой системой. |

Определение заинтересованных лиц и их ролей в проекте позволяет четко понять, кто будет использовать систему, какие требования они предъявляют и как их вовлечь в процесс разработки. Основные заинтересованные лица включают администрацию агентства и риэлторов. Каждая группа имеет свои критерии успеха и вовлеченность в проект, что необходимо учитывать при проектировании и разработке системы.

* 1. Бизнес-требования

Бизнес-требования к модулю информационной системы автоматизации процесса взаимодействия риэлторов и клиентов формируются на основе ключевых бизнес-целей заказчика и ожиданий других заинтересованных лиц. Эти требования отражают основные направления развития проекта и определяют, какие функции системы будут реализованы для достижения поставленных целей. Ключевые бизнес-цели представлены в Таблице 1.6.

Таблица 1.6 – Ключевые бизнес-цели

|  |  |  |
| --- | --- | --- |
| Цель | Бизнес-требование | Ключевые функции |
| Увеличение эффективности за счет автоматизации и упрощение поиска данных. | Система должна обеспечивать быстрый и простой ввод и вывод данных с минимальным временем отклика. | Быстрый поиск, генерация документов. |
| Минимизация простоев и сбоев в работе системы. | Система должна быть устойчива к высоким нагрузкам. | Поддержка работы в пиковые периоды, автоматическое резервирование данных для предотвращения потери информации. |

Для наглядного представления связи бизнес-целей, требований и ключевых функций может быть использована карта воздействий (Рисунок 1.5).



Рисунок 1.5. – Impact Mapping.

1. Анализ существующих решений

Для решения проблемы взаимодействия риэлторов и клиентов было проведено исследование существующих приложений, предназначенных для этих целей. В результате были выделены их достоинства и недостатки.

* 1. Обзор существующих решений
     1. Решение 1 «Intrum CRM»

Онлайн CRM система. База объектов и клиентов, прием входящих заявок и автоматический подбор для них объектов, ведение сделки по стадиям и формы для быстрого создания документов, выгрузка объектов недвижимости на сайты объявлений, возможность sms, email рассылок, звонки непосредственно из браузера через встроенную voip телефонию, интеграция с социальными сетями. Готовое отраслевое решение для агентств недвижимости. Ссылка - <https://www.intrumnet.com/> .

* + 1. Решение 2 «1С:Риэлтор»

Конфигурация 1С для управления продажами недвижимости. Продукт предназначен для компаний, занимающихся сделками по купле-продаже объектов недвижимости как на первичном, так и на вторичном рынках. Решение позволяет повысить эффективность подготовки и проведения сделок с недвижимостью в строительных и девелоперских компаниях, в риэлторских компаниях и агентствах недвижимости. Ссылка - <https://www.intrumnet.com/> .

* + 1. Решение 3 «Риелтор Про Web»

CRM для агентств недвижимости и частных риелторов. Сайт, парсер, база клиентов, выгрузки на площадки и в соц. сети, телефония, ежедневник, органайзер, напоминания, аналитика, поддержка. Ссылка - <https://realtorproweb.ru/> .

* 1. Сравнительные анализ решений

Для сравнения рассмотренных решений были выделены следующие критерии, которые отражают ключевые бизнес-требования заказчика, описанные в разделе 1, а также наиболее приоритетные детализированные требования и ограничения:

* интуитивно понятный интерфейс — важный критерий для обеспечения удобства использования;
* стоимость лицензии — критичный фактор для мелких и новых компаний, которые стремятся минимизировать затраты;
* формирование договоров — функция, которая позволяет автоматизировать формирование документов;
* легкость интеграции — критерий, который определяет, насколько быстро и просто можно внедрить решение на предприятии;
* ведение аналитической отчетности — функция, которая позволяет анализировать эффективность работы риэлторов.

На основе выделенных критериев было проведено сравнение рассмотренных решений. Результаты представлены в Таблице 2.1.

Таблица 2.1 – Сравнение решений

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Критерии сравнения | Intrum CRM | 1С:Риэлтор | Риелтор Про Web | Разрабатываемая программа |
| Наличие интуитивно понятного интерфейса | + | + | + | + |
| Высокая стоимость лицензии | + | + | - | - |

Продолжение Таблицы 2.1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Критерии сравнения | Intrum CRM | 1С:Риэлтор | Риелтор Про Web | Разрабатываемая программа |
| Формирование договоров | + | + | - | + |
| Легкость интеграции на предприятие | - | - | + | + |
| Ведение аналитической отчетности | + | + | + | + |

* 1. Выводы по анализу

Проведенный анализ существующих решений позволил выявить их ключевые достоинства и недостатки, а также определить направления для разработки собственного решения, которое полностью удовлетворит требования заказчика. Было принято решение разработать собственное решение «с нуля». Разрабатываемая программа будет включать следующие ключевые особенности:

* интуитивно понятный интерфейс;
* возможность формирования договоров;
* расширенные функции аналитической отчетности для анализа работы;
* доступная стоимость;
* функция хранения документов.

Таким образом, разрабатываемое решение будет сочетать в себе лучшие черты существующих аналогов, устраняя их недостатки и предлагая уникальные функции, которые полностью удовлетворят потребности заказчика. Это позволит создать конкурентоспособный продукт, оптимизирующий процессы хранения данных о клиентах, договорах и объектах в риэлторском бизнесе.

1. Концепция решения
   1. Техническое задание
      1. Общие сведения

Полное наименование системы – Риэлторская система.

Плановые сроки работы по созданию системы: начало – 25.12.2024, окончание – 01.02.2025.

* + 1. Назначение и цели создания системы
       1. Назначение системы

Система предназначена для автоматизации взаимодействия риэлторов и клиентов. Она позволит хранить и управлять клиентской и объектной базой, автоматизировать формирование договоров и просматривать статистику в виде диаграммы по подписанным договорам.

* + - 1. Цели системы

Основной целью создания системы является повышение эффективности работы риэлторов путем:

* ведения базы клиентов;
* ведения базы объектов;
* автоматизации формирования договоров;
* повышения качества выполняемой работы;
* формирования наглядной статистики для анализа и принятия решений.

Система позволит риэлторскому агентству повысить эффективность и прибыльность своей деятельность, предоставляя централизованный контроль над всеми аспектами.

* + 1. Требования к системе
       1. Требования к структуре и функционированию системы.

Программа должна обеспечивать возможность выполнения следующих функций:

* регистрация пользователей (высокий уровень);
* авторизация пользователей (высокий уровень);
* редактирование профиля (низкий уровень);
* просмотр клиентов с возможностью поиска (высокий уровень);
* детальный просмотр клиентов (высокий уровень);
* добавление и редактирование клиентов (высокий уровень);
* просмотр объектов с возможностью поиска (высокий уровень);
* детальный просмотр объектов (средний уровень);
* добавление и редактирование объектов (высокий уровень);
* просмотр, добавление, удаление документов (средний уровень);
* формирование договора (высокий уровень);
* просмотр статистики (низкий уровень).
  + - 1. Требования к численности и квалификации пользователей

Минимальное количество персонала, требуемого для работы программы – 2 человека: системный администратор, риэлтор.

Системный администратор должен иметь среднее специальное техническое образование. В перечень задач должны входить:

* поддержание работоспособности технических средств;
* задача установки программы;
* задача создания резервных копий базы данных.

Риэлтор должен уметь работать с компьютером на уровне пользователя.

* + - 1. Требования к составу и параметрам технических средств

В состав технических средств должен входить IBM-совместный персональный компьютер (ПЭВМ), выполняющий роль сервера, включающий в себя:

* процессор с частотой не менее 2.0 hz;
* оперативная память не менее 4Гб;
* жесткий диск объемом не менее 256Гб.
  + - 1. Требования к информационной и программной совместимости

Требования перечислены в Таблице 3.1.

Таблица 3.1 – Требования к совместимости

|  |  |
| --- | --- |
| База данных | PostgreSQL (17 версия) |
| Язык программирования | C# (12 версия) |
| SDK | Net core 8.0 |
| IDE | Rider |
| ОС | Windows (7,8,10,11) |

* + - 1. Требования к защите, безопасности и сохранности данных.

Программа должна обеспечивать защиту персональных данных пользователей и клиентов, обрабатываемых в системе, в соответствии с требованиями действующего законодательства Российской Федерации.

* + 1. Состав и содержание работ по созданию системы

Состав и содержание работ представлены в Таблице 3.2.

Таблица 3.2 – Состав работ

|  |  |  |
| --- | --- | --- |
| Стадия | Сроки | Исполнитель |
| Анализ | 4 дня | Аналитик |
| Планирование | 3 дня | Аналитик |
| Проектирование и дизайн | 3 дня | Аналитик, программист |
| Разработка | 18 дней | Программист |
| Тестирование | 6 дней | Тестировщик, программист |
| Развертывание | 1 день | Программист |

* 1. Модель прецедентов
     1. Акторы

Для проектирования и разработки модуля автоматизации взаимодействия риэлторов и клиентов необходимо определить акторов, которые будут взаимодействовать с системой. Акторы — это пользователи, внешние системы или объекты, которые инициируют выполнение сценариев или играют пассивную роль в системе. В Таблице 3.3 представлены ключевые акторы, их типы и описательные характеристики.

Таблица 3.3 – Акторы системы

|  |  |  |
| --- | --- | --- |
| Наименование актора | Тип актора | Описательная характеристика |
| Риэлтор | Первичный (пользователь) | Основной пользователь системы, который имеет возможность управлять клиентами, объектами и договорами, с возможностью прикрепления документов и редактирования профиля. |

* + 1. Диаграмма прецедентов

Диаграмма прецедентов представлена на Рисунке 3.1.

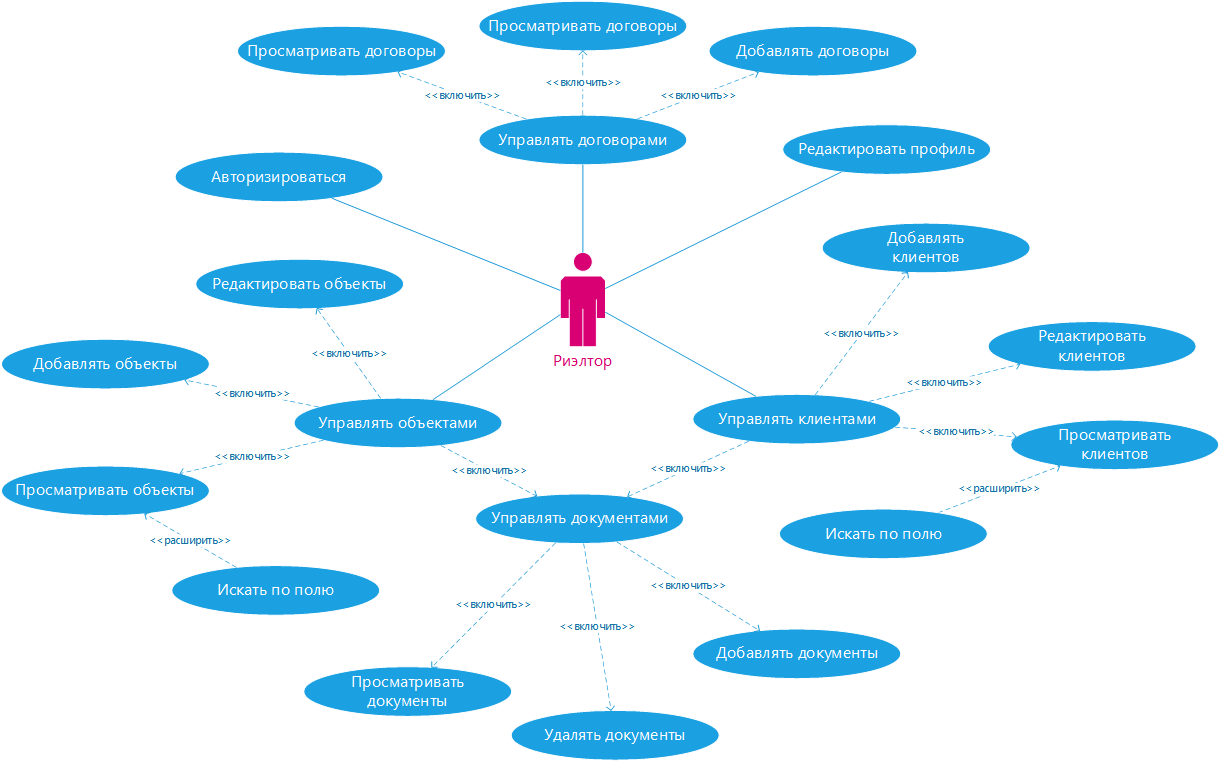


Рисунок 3.1 – UseCase диаграмма.

* + 1. Сценарии использования

Таблица 3.4 – Сценарий 1. «Авторизация»

|  |  |
| --- | --- |
| Описание | Авторизация пользователя |
| Идентификатор прецедента | UC 1.1 |
| Список акторов | Риэлтор. |
| Предусловия | Пользователь имеет учетную запись. |
| Действия пользователя | Реакция системы |
| Ввод данных | Отображение данных в полях. |
| Нажатие кнопки «Войти» | Поиск пользователя в базе. |
| Постусловие | Пользователь найден. |
| Включение | Отсутствует |
| Расширение | Отсутствует |
| Обобщение | Отсутствует |

Таблица 3.5 – Сценарий 2. «Просмотр клиентов»

|  |  |
| --- | --- |
| Описание | Просмотр клиентов |
| Идентификатор прецедента | UC 1.2 |
| Список акторов | Риэлтор. |
| Предусловия | Пользователь авторизирован. |

Продолжение Таблицы 3.5

|  |  |
| --- | --- |
| Действия пользователя | Реакция системы |
| Переход на вкладку «Клиенты» | Отображение таблицы с данными. |
| Постусловие | Отсутствует |
| Включение | Отсутствует |
| Расширение | Искать по полю |
| Обобщение | Отсутствует |

Таблица 3.6 – Сценарий 3. «Просмотр объектов»

|  |  |
| --- | --- |
| Описание | Просмотр объектов |
| Идентификатор прецедента | UC 1.3 |
| Список акторов | Риэлтор. |
| Предусловия | Пользователь авторизирован. |
| Действия пользователя | Реакция системы |
| Переход на вкладку «Объекты» | Отображение таблицы с данными. |
| Постусловие | Отсутствует |
| Включение | Отсутствует |
| Расширение | Искать по полю |
| Обобщение | Отсутствует |

Таблица 3.7 – Сценарий 4. «Редактирование профиля»

|  |  |
| --- | --- |
| Описание | Редактирование профиля |
| Идентификатор прецедента | UC 1.4 |
| Список акторов | Риэлтор. |
| Предусловия | Пользователь авторизирован. |
| Действия пользователя | Реакция системы |
| Переход на вкладку «Профиль». | Отображение заполненной формы. |
| Изменение параметра в поле. | Отображение введенного параметра. |
| Нажатие кнопки «Сохранить» | Сохранение данных в базу. |
| Постусловие | Отсутствует |
| Включение | Отсутствует |
| Расширение | Отсутствует |
| Обобщение | Отсутствует |

* 1. Customer Journey Mapping

Customer Journey Mapping представлен на Рисунке 3.2.



Рисунок 3.2 – Customer Journey Mapping.

* 1. User Story Mapping

User Story Mapping представлен на Рисунке 3.3.

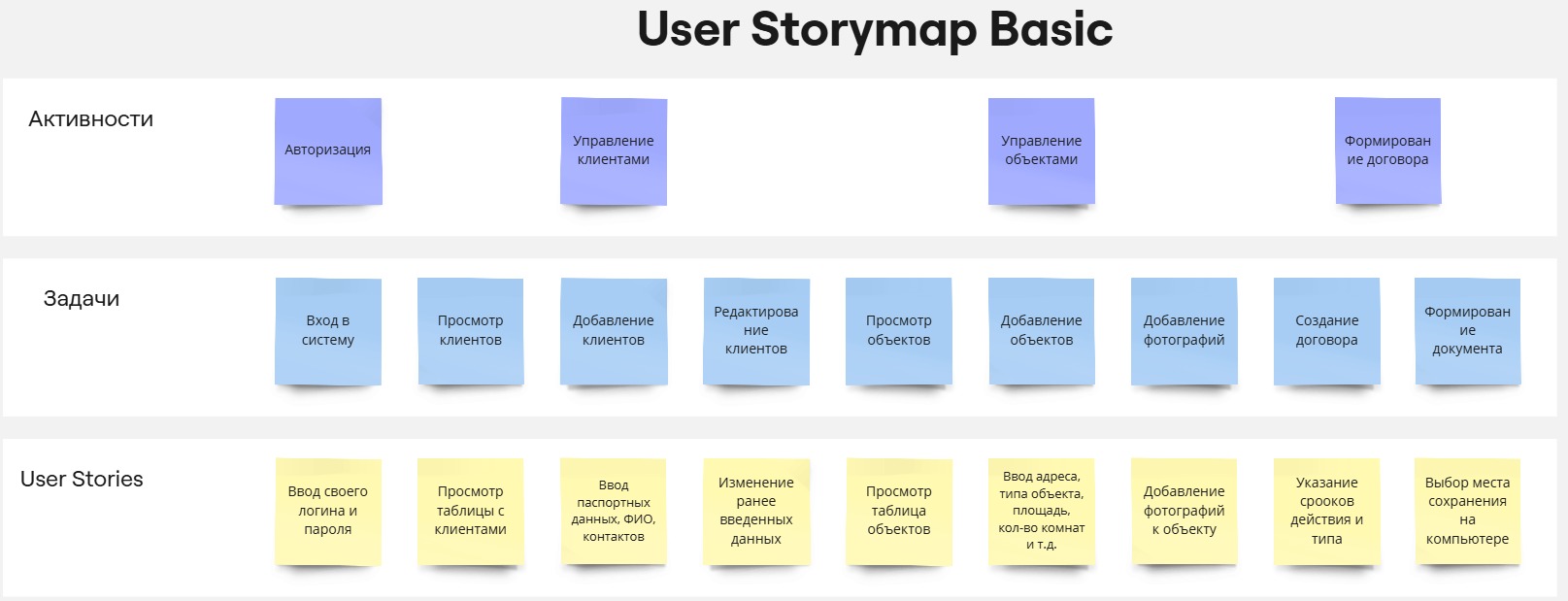


Рисунок 3.3 – User Story Mapping

* 1. Анализ факторов риска проекта

Анализ факторов риска проекта представлен в Таблице 3.8.

Таблица 3.8 – Анализ факторов риска

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Тип | Риск | Последствия | Меры по предотвращению | Меры по минимизации последствий | Вероятность свершения риска [P] | Степень ущерба [L] | Влияние  [R] = [P] x [L] |
| Технический | Сбои в работе системы из-за недостаточной производительности. | Задержки в работе системы, недовольство клиентов и сотрудников. | Проведение нагрузочного тестирования на этапе разработки. | Оптимизация кода и использование масштабируемых технологий. | 6 | 9 | 54 |
| Организационный | Недостаточная квалификация сотрудников для работы с новой системой. | Ошибки при использовании системы, снижение эффективности работы. | Проведение обучения сотрудников до внедрения системы. | Создание подробной документации и поддержка пользователей на этапе внедрения. | 7 | 5 | 35 |
| Организационный | Недостаточная вовлеченность заказчика в процесс разработки. | Несоответствие системы ожиданиям заказчика, задержки в согласовании решений. | Регулярное проведение встреч с заказчиком и предоставление промежуточных результатов. | Назначение ответственного представителя заказчика для взаимодействия. | 4 | 7 | 28 |

Продолжение Таблицы 3.8

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Тип | Риск | Последствия | Меры по предотвращению | Меры по минимизации последствий | Вероятность свершения риска [P] | Степень ущерба [L] | Влияние  [R] = [P] x [L] |
| Коммерческий | Превышение бюджета проекта из-за неправильной оценки трудозатрат. | Финансовые потери, задержка сроков реализации. | Проведение детальной оценки трудозатрат и создание резервного бюджета. | Регулярный мониторинг расходов и корректировка плана работ. | 5 | 7 | 35 |
| Проектный | Неполное или некорректное понимание требований заказчика. | Разработка системы, не соответствующей ожиданиям заказчика. | Проведение детальных интервью с заказчиком и пользователями, создание прототипов. | Регулярное согласование требований и внесение изменений на ранних этапах разработки. | 5 | 8 | 40 |

1. Проектирование решения
   1. Описание архитектуры системы

Для наглядного представления архитектуры системы используется диаграмма развертывания (Рисунок 4.1). На диаграмме отображены основные компоненты системы и их взаимосвязи.

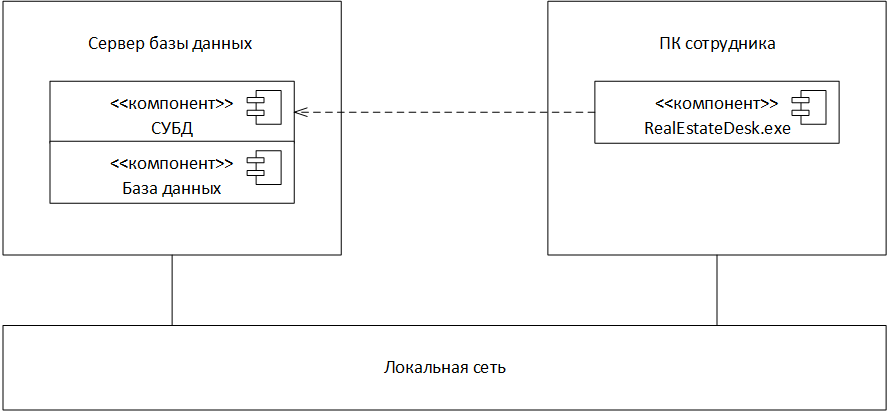


Рисунок 4.1 – Диаграмма развертывания.

* 1. Описание возможных технических решений

В качестве инструментальных средств разработки была выбрана IDE JB Rider, СУБД PostgreSQL ввиду своей современности, надежности и удобства использования. В Таблицах 4.1 – 4.2 представлены обоснования выбора.

Таблица 4.1 – Обоснование выбора средств разработки

|  |  |  |
| --- | --- | --- |
| Критерии сравнения | Visual Studio | Rider |
| Удобство | Интуитивно понятный пользовательский интерфейс, который позволяет легко создавать и отлаживать приложения.  Встроенная поддержка отладки и профилирования для упрощения разработки приложений. | Понятный пользовательский интерфейс, позволяющий создавать, оперировать и отслеживать приложения. Встроенный анализ кода для всех поддерживаемых языков с отображением ошибок и проблем с качеством кода. |
| Языковая поддержка | Широкий спектр поддерживаемых языков программирования, включая C++, C#, F#, Visual Basic и другие. | Отличное решение для разработки на ASP.NET, XAML, JavaScript, TypeScript, SASS, LESS, HTML, SQL, XSLT, YAML и другие. |
| Производительность | Имеет высокую производительность благодаря множеству оптимизаций и улучшений, которые Microsoft регулярно внедряет в свою среду разработки. | Высокая производительность и поддержка современных технологий, таких как Unreal Engine, Mercurial, SVN, и многих других. |
| Наличие  инструментов  разработки | Интеграция с множеством инструментов для управления проектами, контроля версий и развертывания приложений. | Мощные инструменты для автоматической проверки кода и обеспечения высокого качества разработки, встроенные шаблоны проектов и возможность рефакторинга проекта. |

Выбор среды разработки зависит от ваших индивидуальных предпочтений и потребностей. Visual Studio часто используется для работы с технологиями Microsoft, такими как .NET и C#, в то время как Rider который является альтернативой используется в большем спектре языков. Обе среды разработки обладают мощными инструментами, интегрированной средой разработки и возможностью расширения функциональности через плагины, но в отличии от Rider Visual Studio хоть и имеет меньше инструментов разработки, но обладает более простым интерфейсом с встроенным отображением ошибок.

В качестве средств разработки клиентской части была выбрана JB Rider так как Visual Studio не имеет кроссплатформенности, что делает невозможным разработку на ОС Linux.

Таблица 4.2 – Обоснование выбора СУБД

|  |  |  |
| --- | --- | --- |
| Критерии сравнения | СУБД MS SQL Server | СУБД PostgreSQL |
| Удобство | Высокая производительность и масштабируемость.  Обширные инструменты для администрирования и мониторинга баз данных. | Бесплатное и открытое программное обеспечение с активным сообществом разработчиков.  Поддержка распределенных транзакций и многопоточности. |
| Совместимость | Широкие возможности для разработки приложений и интеграции с другими технологиями Microsoft. | Обладает отличной совместимостью с различными приложениями и инструментами, а также поддерживает стандарт SQL (Structured Query Language). |

Продолжение Таблицы 4.2

|  |  |  |
| --- | --- | --- |
| Критерии сравнения | СУБД MS SQL Server | СУБД PostgreSQL |
| Поддержка | Официальные каналы поддержки Microsoft, включая документацию, форумы сообщества, онлайн-ресурсы и техническую поддержку. Кроме того, существует множество сторонних поставщиков услуг поддержки, которые могут предоставить консультации, обучение и техническую поддержку для MS SQL Server. | PostgreSQL осуществляется сообществом разработчиков, а также коммерческими компаниями, которые предоставляют услуги поддержки, консультирования, обучения и разработки для этой системы управления базами данных. |
| Надежность | Считается надежной и стабильной. Она имеет встроенные механизмы высокой доступности, отказоустойчивости и аварийного восстановления, такие как Always On Availability Groups, механизмы резервного копирования и восстановления, и многие другие. | Обладает высокой надежностью и отказоустойчивостью. Она широко используется в различных критически важных сферах, таких как финансы, здравоохранение, государственное управление и многие другие. |

Выбор между MS SQL Server и PostgreSQL зависит от потребностей и требований конкретного проекта. Обе СУБД имеют свои преимущества и недостатки. В качестве системы управления базами данных была выбрана PostgreSQL так как этот продукт больше зарекомендовал себя на рынке.

* 1. Проектирование интерфейса пользователя

Результат проектирования интерфейса представлен на Рисунке 4.2.

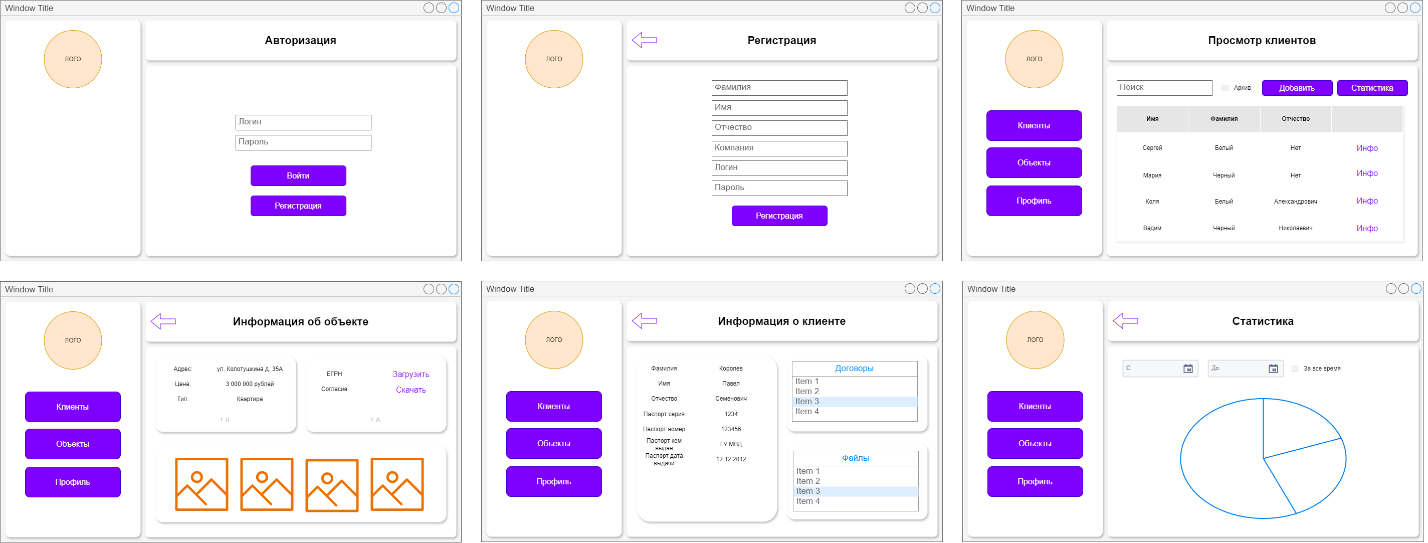


Рисунок 4.2 – Интерфейс приложения

* 1. Проектирование базы данных

Диаграмма базы данных представлена на Рисунке 4.3. Листинг базы данных представлен в Приложении 1.

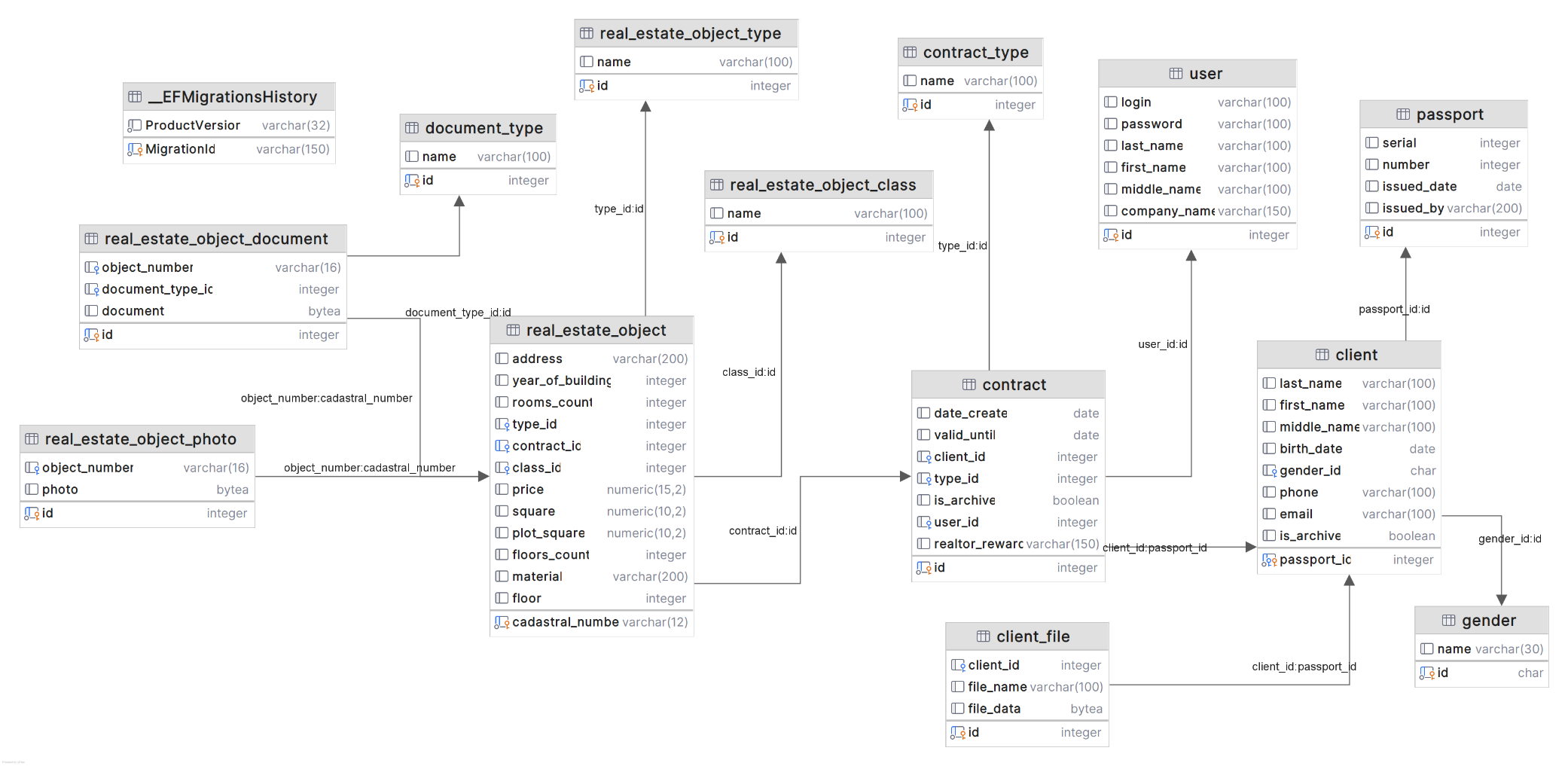


Рисунок 4.3 – Диаграмма базы данных

1. Программная реализация системы

Программная реализация программы представлена в листингах в Приложении 2.

1. Функциональное тестирование системы

Тест-кейс 1 – Авторизация (положительный)

|  |  |
| --- | --- |
| Идентификатор тест-варианта | TC-Auth-1 |
| Набор входных данных | Логин: 1  Пароль: 1 |
| Ожидаемые результаты | Пользователь авторизуется в системе и попадет на главное окно. |
| Выполняемые действия | 1. Открыть приложение; 2. Ввести логин; 3. Ввести пароль; 4. Нажать кнопку «Войти»   Пользователь авторизируется, открывает главное окно. |

Тест-кейс 2 – Регистрация (негативный)

|  |  |
| --- | --- |
| Идентификатор тест-варианта | TC-Reg-1 |
| Набор входных данных | Логин: 1  Пароль: 123  Повторный пароль: 12344  Фамилия: Иванов  Имя: Иван  Компания: Мир Недвижимости |
| Ожидаемые результаты | Пароль и Повторный пароль не совпадают, выводится ошибка |
| Выполняемые действия | 1. Открыть приложение; 2. Нажать кнопку «Регистрация»; 3. Ввести данные; 4. Нажать кнопку «Регистрация»   Система отображает ошибку «Пароли не совпадают». |

Тест-кейс 3 – Регистрация (положительный)

|  |  |
| --- | --- |
| Идентификатор тест-варианта | TC-Reg-2 |
| Набор входных данных | Логин: 1  Пароль: 123  Повторный пароль: 123  Фамилия: Иванов  Имя: Иван  Компания: Мир Недвижимости |
| Ожидаемые результаты | Пользователь зарегистрируется в системе. Система переведет пользователя на страницу авторизации. |
| Выполняемые действия | 1. Открыть приложение; 2. Нажать кнопку «Регистрация»; 3. Ввести данные; 4. Нажать кнопку «Регистрация»   Пользователь зарегистрировался, система перевела пользователя на страницу авторизации. |

Тест-кейс 4 – Добавление клиента (положительный)

|  |  |
| --- | --- |
| Идентификатор тест-варианта | TC-AddClient-1 |
| Набор входных данных | Фамилия: Иванов  Имя: Иван  Отчество: Петрович  Паспорт серия: 1234  Паспорт номер: 123456  Выдача паспорта: 12.12.2000  Выдан: ГУ МВД РОССИИ |
| Ожидаемые результаты | Пользователь добавляет в систему нового клиента, система уведомляет пользователя |

Продолжение Тест-кейса 4

|  |  |
| --- | --- |
| Выполняемые действия | 1. Авторизироваться в системе; 2. Перейти в блок Клиенты; 3. Нажать кнопку «Добавить»; 4. Ввести данные 5. Нажать кнопку «Сохранить»   Клиент добавляется в базу данных, система оповещает пользователя. |

1. Список литературы
2. Основы UML – диаграммы использования (use-case).- URL: https://pro-prof.com/archives/2594
3. Фаулер М., Скотт К UML. Основы СПб.: Символ, 2006, 184 с.
4. Вирт, Н. Алгоритмы и структуры данных / Н. Вирт. - М.: Мир, 2016.
5. Л.Г. Гагарина “Разработка и эксплуатация автоматизированных информационных систем”: Учебное пособие / Л.Г. Гагарина. – Москва: ИД “ФОРУМ”: ИНФРА – М, 2021. – 384 с.;
6. Дону; Таганрог: Издательство Южного федерального университета, 2019. – 96 с.;
7. C# – URL: https://dotnet.microsoft.com/en-us/languages/csharp (11.12.2022)

Приложение 1 – Листинг базы данных

--

-- PostgreSQL database dump

--

-- Dumped from database version 17.0

-- Dumped by pg\_dump version 17.0

SET statement\_timeout = 0;

SET lock\_timeout = 0;

SET idle\_in\_transaction\_session\_timeout = 0;

SET transaction\_timeout = 0;

SET client\_encoding = 'UTF8';

SET standard\_conforming\_strings = on;

SELECT pg\_catalog.set\_config('search\_path', '', false);

SET check\_function\_bodies = false;

SET xmloption = content;

SET client\_min\_messages = warning;

SET row\_security = off;

--

-- Name: realtor\_system\_db; Type: DATABASE; Schema: -; Owner: postgres

--

CREATE DATABASE realtor\_system\_db WITH TEMPLATE = template0 ENCODING = 'UTF8' LOCALE\_PROVIDER = libc LOCALE = 'Russian\_Russia.1251';

ALTER DATABASE realtor\_system\_db OWNER TO postgres;

\connect realtor\_system\_db

SET statement\_timeout = 0;

SET lock\_timeout = 0;

SET idle\_in\_transaction\_session\_timeout = 0;

SET transaction\_timeout = 0;

SET client\_encoding = 'UTF8';

SET standard\_conforming\_strings = on;

SELECT pg\_catalog.set\_config('search\_path', '', false);

SET check\_function\_bodies = false;

SET xmloption = content;

SET client\_min\_messages = warning;

SET row\_security = off;

SET default\_tablespace = '';

SET default\_table\_access\_method = heap;

--

-- Name: \_\_EFMigrationsHistory; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public."\_\_EFMigrationsHistory" (

"MigrationId" character varying(150) NOT NULL,

"ProductVersion" character varying(32) NOT NULL

);

Рисунок 1.1.1 –Листинг базы

Продолжение приложения 1

ALTER TABLE public."\_\_EFMigrationsHistory" OWNER TO postgres;

--

-- Name: client; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public.client (

passport\_id integer NOT NULL,

last\_name character varying(100),

first\_name character varying(100),

middle\_name character varying(100),

birth\_date date,

gender\_id character(1),

phone character varying(100),

email character varying(100),

is\_archive boolean DEFAULT false

);

ALTER TABLE public.client OWNER TO postgres;

--

-- Name: client\_file; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public.client\_file (

id integer NOT NULL,

client\_id integer,

file\_name character varying(100),

file\_data bytea

);

ALTER TABLE public.client\_file OWNER TO postgres;

--

-- Name: client\_file\_id\_seq; Type: SEQUENCE; Schema: public; Owner: postgres

--

CREATE SEQUENCE public.client\_file\_id\_seq

AS integer

START WITH 1

INCREMENT BY 1

NO MINVALUE

NO MAXVALUE

CACHE 1;

ALTER SEQUENCE public.client\_file\_id\_seq OWNER TO postgres;

--

-- Name: client\_file\_id\_seq; Type: SEQUENCE OWNED BY; Schema: public; Owner: postgres

--

Рисунок 1.1.2 – Продолжение листинга базы

Продолжение приложения 1

ALTER SEQUENCE public.client\_file\_id\_seq OWNED BY public.client\_file.id;

--

-- Name: contract; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public.contract (

id integer NOT NULL,

date\_create date,

valid\_until date,

client\_id integer,

type\_id integer,

is\_archive boolean,

user\_id integer,

realtor\_reward character varying(150)

);

ALTER TABLE public.contract OWNER TO postgres;

--

-- Name: contract\_id\_seq; Type: SEQUENCE; Schema: public; Owner: postgres

--

CREATE SEQUENCE public.contract\_id\_seq

AS integer

START WITH 1

INCREMENT BY 1

NO MINVALUE

NO MAXVALUE

CACHE 1;

ALTER SEQUENCE public.contract\_id\_seq OWNER TO postgres;

--

-- Name: contract\_id\_seq; Type: SEQUENCE OWNED BY; Schema: public; Owner: postgres

--

ALTER SEQUENCE public.contract\_id\_seq OWNED BY public.contract.id;

--

-- Name: contract\_type; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public.contract\_type (

id integer NOT NULL,

name character varying(100)

);

ALTER TABLE public.contract\_type OWNER TO postgres;

Рисунок 1.1.3 – Продолжение листинга базы

Продолжение приложения 1

--

-- Name: contract\_type\_id\_seq; Type: SEQUENCE; Schema: public; Owner: postgres

--

CREATE SEQUENCE public.contract\_type\_id\_seq

AS integer

START WITH 1

INCREMENT BY 1

NO MINVALUE

NO MAXVALUE

CACHE 1;

ALTER SEQUENCE public.contract\_type\_id\_seq OWNER TO postgres;

--

-- Name: contract\_type\_id\_seq; Type: SEQUENCE OWNED BY; Schema: public; Owner: postgres

--

ALTER SEQUENCE public.contract\_type\_id\_seq OWNED BY public.contract\_type.id;

--

-- Name: document\_type; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public.document\_type (

id integer NOT NULL,

name character varying(100)

);

ALTER TABLE public.document\_type OWNER TO postgres;

--

-- Name: document\_type\_id\_seq; Type: SEQUENCE; Schema: public; Owner: postgres

--

CREATE SEQUENCE public.document\_type\_id\_seq

AS integer

START WITH 1

INCREMENT BY 1

NO MINVALUE

NO MAXVALUE

CACHE 1;

ALTER SEQUENCE public.document\_type\_id\_seq OWNER TO postgres;

--

-- Name: document\_type\_id\_seq; Type: SEQUENCE OWNED BY; Schema: public; Owner: postgres

--

Рисунок 1.1.4 – Продолжение листинга базы

Продолжение приложения 1

ALTER SEQUENCE public.document\_type\_id\_seq OWNED BY public.document\_type.id;

--

-- Name: gender; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public.gender (

id character(1) NOT NULL,

name character varying(30)

);

ALTER TABLE public.gender OWNER TO postgres;

--

-- Name: passport; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public.passport (

id integer NOT NULL,

serial integer,

number integer,

issued\_date date,

issued\_by character varying(200)

);

ALTER TABLE public.passport OWNER TO postgres;

--

-- Name: passport\_id\_seq; Type: SEQUENCE; Schema: public; Owner: postgres

--

CREATE SEQUENCE public.passport\_id\_seq

AS integer

START WITH 1

INCREMENT BY 1

NO MINVALUE

NO MAXVALUE

CACHE 1;

ALTER SEQUENCE public.passport\_id\_seq OWNER TO postgres;

--

-- Name: passport\_id\_seq; Type: SEQUENCE OWNED BY; Schema: public; Owner: postgres

--

ALTER SEQUENCE public.passport\_id\_seq OWNED BY public.passport.id;

--

Рисунок 1.1.5 – Продолжение листинга базы

Продолжение приложения 1

-- Name: real\_estate\_object; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public.real\_estate\_object (

cadastral\_number character varying(12) NOT NULL,

address character varying(200),

year\_of\_building integer,

rooms\_count integer,

type\_id integer,

contract\_id integer,

class\_id integer,

price numeric(15,2),

square numeric(10,2),

plot\_square numeric(10,2),

floors\_count integer,

material character varying(200),

floor integer

);

ALTER TABLE public.real\_estate\_object OWNER TO postgres;

--

-- Name: real\_estate\_object\_class; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public.real\_estate\_object\_class (

id integer NOT NULL,

name character varying(100)

);

ALTER TABLE public.real\_estate\_object\_class OWNER TO postgres;

--

-- Name: real\_estate\_object\_class\_id\_seq; Type: SEQUENCE; Schema: public; Owner: postgres

--

CREATE SEQUENCE public.real\_estate\_object\_class\_id\_seq

AS integer

START WITH 1

INCREMENT BY 1

NO MINVALUE

NO MAXVALUE

CACHE 1;

ALTER SEQUENCE public.real\_estate\_object\_class\_id\_seq OWNER TO postgres;

--

-- Name: real\_estate\_object\_class\_id\_seq; Type: SEQUENCE OWNED BY; Schema: public; Owner: postgres

--

Рисунок 1.1.6 – Продолжение листинга базы

Продолжение приложения 1

ALTER SEQUENCE public.real\_estate\_object\_class\_id\_seq OWNED BY public.real\_estate\_object\_class.id;

--

-- Name: real\_estate\_object\_document; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public.real\_estate\_object\_document (

id integer NOT NULL,

object\_number character varying(16),

document\_type\_id integer,

document bytea

);

ALTER TABLE public.real\_estate\_object\_document OWNER TO postgres;

--

-- Name: real\_estate\_object\_document\_id\_seq; Type: SEQUENCE; Schema: public; Owner: postgres

--

CREATE SEQUENCE public.real\_estate\_object\_document\_id\_seq

AS integer

START WITH 1

INCREMENT BY 1

NO MINVALUE

NO MAXVALUE

CACHE 1;

ALTER SEQUENCE public.real\_estate\_object\_document\_id\_seq OWNER TO postgres;

--

-- Name: real\_estate\_object\_document\_id\_seq; Type: SEQUENCE OWNED BY; Schema: public; Owner: postgres

--

ALTER SEQUENCE public.real\_estate\_object\_document\_id\_seq OWNED BY public.real\_estate\_object\_document.id;

--

-- Name: real\_estate\_object\_photo; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public.real\_estate\_object\_photo (

id integer NOT NULL,

object\_number character varying(16),

photo bytea

);

Рисунок 1.1.7 – Продолжение листинга базы

Продолжение приложения 1

ALTER TABLE public.real\_estate\_object\_photo OWNER TO postgres;

--

-- Name: real\_estate\_object\_photo\_id\_seq; Type: SEQUENCE; Schema: public; Owner: postgres

--

CREATE SEQUENCE public.real\_estate\_object\_photo\_id\_seq

AS integer

START WITH 1

INCREMENT BY 1

NO MINVALUE

NO MAXVALUE

CACHE 1;

ALTER SEQUENCE public.real\_estate\_object\_photo\_id\_seq OWNER TO postgres;

--

-- Name: real\_estate\_object\_photo\_id\_seq; Type: SEQUENCE OWNED BY; Schema: public; Owner: postgres

--

ALTER SEQUENCE public.real\_estate\_object\_photo\_id\_seq OWNED BY public.real\_estate\_object\_photo.id;

--

-- Name: real\_estate\_object\_type; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public.real\_estate\_object\_type (

id integer NOT NULL,

name character varying(100)

);

ALTER TABLE public.real\_estate\_object\_type OWNER TO postgres;

--

-- Name: real\_estate\_object\_type\_id\_seq; Type: SEQUENCE; Schema: public; Owner: postgres

--

CREATE SEQUENCE public.real\_estate\_object\_type\_id\_seq

AS integer

START WITH 1

INCREMENT BY 1

NO MINVALUE

NO MAXVALUE

CACHE 1;

ALTER SEQUENCE public.real\_estate\_object\_type\_id\_seq OWNER TO postgres;

Рисунок 1.1.8 – Продолжение листинга базы

Продолжение приложения 1

--

-- Name: real\_estate\_object\_type\_id\_seq; Type: SEQUENCE OWNED BY; Schema: public; Owner: postgres

--

ALTER SEQUENCE public.real\_estate\_object\_type\_id\_seq OWNED BY public.real\_estate\_object\_type.id;

--

-- Name: user; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public."user" (

id integer NOT NULL,

login character varying(100),

password character varying(100),

last\_name character varying(100),

first\_name character varying(100),

middle\_name character varying(100),

company\_name character varying(150)

);

ALTER TABLE public."user" OWNER TO postgres;

--

-- Name: user\_id\_seq; Type: SEQUENCE; Schema: public; Owner: postgres

--

CREATE SEQUENCE public.user\_id\_seq

AS integer

START WITH 1

INCREMENT BY 1

NO MINVALUE

NO MAXVALUE

CACHE 1;

ALTER SEQUENCE public.user\_id\_seq OWNER TO postgres;

--

-- Name: user\_id\_seq; Type: SEQUENCE OWNED BY; Schema: public; Owner: postgres

--

ALTER SEQUENCE public.user\_id\_seq OWNED BY public."user".id;

--

-- Name: client\_file id; Type: DEFAULT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public.client\_file ALTER COLUMN id SET DEFAULT nextval('public.client\_file\_id\_seq'::regclass);

Рисунок 1.1.9 – Продолжение листинга базы

Продолжение приложения 1

--

-- Name: contract id; Type: DEFAULT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public.contract ALTER COLUMN id SET DEFAULT nextval('public.contract\_id\_seq'::regclass);

--

-- Name: contract\_type id; Type: DEFAULT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public.contract\_type ALTER COLUMN id SET DEFAULT nextval('public.contract\_type\_id\_seq'::regclass);

--

-- Name: document\_type id; Type: DEFAULT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public.document\_type ALTER COLUMN id SET DEFAULT nextval('public.document\_type\_id\_seq'::regclass);

--

-- Name: passport id; Type: DEFAULT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public.passport ALTER COLUMN id SET DEFAULT nextval('public.passport\_id\_seq'::regclass);

--

-- Name: real\_estate\_object\_class id; Type: DEFAULT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public.real\_estate\_object\_class ALTER COLUMN id SET DEFAULT nextval('public.real\_estate\_object\_class\_id\_seq'::regclass);

--

-- Name: real\_estate\_object\_document id; Type: DEFAULT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public.real\_estate\_object\_document ALTER COLUMN id SET DEFAULT nextval('public.real\_estate\_object\_document\_id\_seq'::regclass);

--

-- Name: real\_estate\_object\_photo id; Type: DEFAULT; Schema: public; Owner: postgres

--

Рисунок 1.1.10 – Продолжение листинга базы

Продолжение приложения 1

ALTER TABLE ONLY public.real\_estate\_object\_photo ALTER COLUMN id SET DEFAULT nextval('public.real\_estate\_object\_photo\_id\_seq'::regclass);

--

-- Name: real\_estate\_object\_type id; Type: DEFAULT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public.real\_estate\_object\_type ALTER COLUMN id SET DEFAULT nextval('public.real\_estate\_object\_type\_id\_seq'::regclass);

--

-- Name: user id; Type: DEFAULT; Schema: public; Owner: postgres

--

Рисунок 1.1.11 – Продолжение листинга базы

Приложение 2 – Листинг программной реализации системы

using RealtorSystemDesk.Database;

namespace RealtorSystemDesk.Models;

public class NewContractModel

{

public int ClientId { get; set; }

public ContractType Type { get; set; }

private int \_days;

public int Days

{

get => \_days;

set => \_days = value > 0 ? value : 1;

}

public DateTime DateCreate { get; set; }

}

Рисунок 2.1.1 – Листинг NewContractModel.cs

<Page x:Class="RealtorSystemDesk.Pages.ClientManagePages.AddContractPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:RealtorSystemDesk.Pages.ClientManagePages"

mc:Ignorable="d"

xmlns:design="http://materialdesigninxaml.net/winfx/xaml/themes"

Loaded="AddContractPage\_OnLoaded"

Title="Добавление договора"

d:DesignHeight="650"

d:DesignWidth="800">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*" />

<ColumnDefinition Width="\*" />

</Grid.ColumnDefinitions>

<design:Card Margin="5"

Padding="5">

<StackPanel Orientation="Vertical"

Margin="7">

<TextBlock Style="{DynamicResource MaterialDesignTitleSmallTextBlock}"

HorizontalAlignment="Center"

Text="Информация о договоре" />

<DockPanel>

<TextBlock HorizontalAlignment="Left"

Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

Рисунок 2.1.2 – Листинг AddContractPage.xaml

Продолжение приложения 2

VerticalAlignment="Center"

Text="Дата заключения:" />

<DatePicker HorizontalAlignment="Right"

Width="200"

Margin="7"

SelectedDate="{Binding DateCreate}" />

</DockPanel>

<DockPanel>

<TextBlock HorizontalAlignment="Left"

Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="Срок действия (в днях):" />

<TextBox HorizontalAlignment="Right"

Width="50"

MaxLength="3"

Margin="7"

Text="{Binding Days}" />

</DockPanel>

<DockPanel>

<TextBlock HorizontalAlignment="Left"

Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

TextWrapping="Wrap"

Text="Вознаграждение:" />

<TextBox HorizontalAlignment="Right"

Width="200"

MaxLength="150"

Margin="7"

Text="{Binding RealtorReward}" />

</DockPanel>

<DockPanel>

<TextBlock HorizontalAlignment="Left"

Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="Тип договора:" />

<ComboBox HorizontalAlignment="Right"

x:Name="TypeComboBox"

Margin="7"

Width="200"

SelectionChanged="TypeComboBox\_OnSelectionChanged"

SelectedItem="{Binding Type}"

DisplayMemberPath="Name" />

</DockPanel>

<Button x:Name="SaveButton"

Margin="10"

Click="SaveButton\_OnClick"

Content="Сохранить" />

</StackPanel>

</design:Card>

Рисунок 2.1.3 – Продолжение листинга AddContractPage.xaml

Продолжение приложения 2

<design:Card Grid.Column="1"

Margin="5"

Padding="5">

<StackPanel x:Name="ObjectPanel"

Orientation="Vertical"

Visibility="Collapsed">

<TextBlock Style="{DynamicResource MaterialDesignTitleSmallTextBlock}"

HorizontalAlignment="Center"

Text="Информация об объекте" />

<DockPanel>

<TextBlock HorizontalAlignment="Left"

Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="Кадастровый номер:" />

<TextBox HorizontalAlignment="Right"

Width="100"

Margin="7"

Text="{Binding CadastralNumber}"

MaxLength="12" />

</DockPanel>

<DockPanel>

<TextBlock HorizontalAlignment="Left"

Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="Адрес:" />

<TextBox HorizontalAlignment="Right"

Width="200"

Margin="7"

MaxLength="200"

Text="{Binding Address}" />

</DockPanel>

<DockPanel>

<TextBlock HorizontalAlignment="Left"

Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="Площадь:" />

<TextBox HorizontalAlignment="Right"

Margin="7"

Width="100"

MaxLength="40"

Text="{Binding Square}" />

</DockPanel>

<DockPanel>

<TextBlock HorizontalAlignment="Left"

Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="Количество комнат:" />

<TextBox HorizontalAlignment="Right"

Width="50"

Рисунок 2.1.4 – Продолжение листинга AddContractPage.xaml

Продолжение приложения 2

MaxLength="2"

Margin="7"

Text="{Binding RoomsCount}" />

</DockPanel>

<DockPanel>

<TextBlock HorizontalAlignment="Left"

Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="Год постройки здания:" />

<TextBox HorizontalAlignment="Right"

Width="70"

MaxLength="4"

Margin="7"

Text="{Binding BuildingYear}" />

</DockPanel>

<DockPanel>

<TextBlock HorizontalAlignment="Left"

Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="Цена:" />

<TextBox HorizontalAlignment="Right"

Width="100"

Margin="7"

MaxLength="40"

Text="{Binding Price}" />

</DockPanel>

<DockPanel>

<TextBlock HorizontalAlignment="Left"

Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="Тип объекта:" />

<ComboBox HorizontalAlignment="Right"

x:Name="ObjectTypeComboBox"

Margin="7"

SelectionChanged="ObjectTypeComboBox\_OnSelectionChanged"

SelectedItem="{Binding Type}"

Width="200"

DisplayMemberPath="Name" />

</DockPanel>

<StackPanel x:Name="ApartmentPanel"

Visibility="Collapsed"

Orientation="Vertical">

<DockPanel>

<TextBlock HorizontalAlignment="Left"

Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="Класс квартиры:" />

<ComboBox HorizontalAlignment="Right"

x:Name="ClassComboBox"

Рисунок 2.1.5 – Продолжение листинга AddContractPage.xaml

Продолжение приложения 2

Margin="7"

Width="200"

SelectedItem="{Binding Class}"

DisplayMemberPath="Name" />

</DockPanel>

<DockPanel>

<TextBlock HorizontalAlignment="Left"

Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="Этаж:" />

<TextBox HorizontalAlignment="Right"

Width="50"

MaxLength="2"

Margin="7"

Text="{Binding Floor}" />

</DockPanel>

</StackPanel>

<StackPanel x:Name="HousePanel"

Visibility="Collapsed"

Orientation="Vertical">

<DockPanel>

<TextBlock HorizontalAlignment="Left"

Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="Количество этажей в здании:" />

<TextBox HorizontalAlignment="Right"

Width="50"

MaxLength="2"

Margin="7"

Text="{Binding FloorsCount}" />

</DockPanel>

<DockPanel>

<TextBlock HorizontalAlignment="Left"

Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="Площадь участка:" />

<TextBox HorizontalAlignment="Right"

Width="100"

MaxLength="40"

Margin="7"

Text="{Binding PlotSquare}" />

</DockPanel>

<DockPanel>

<TextBlock HorizontalAlignment="Left"

Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="Материал здания:" />

<TextBox HorizontalAlignment="Right"

Width="200"

MaxLength="200"

Margin="7"

Рисунок 2.1.6 – Продолжение листинга AddContractPage.xaml

Продолжение приложения 2

Text="{Binding Material}" />

</DockPanel>

</StackPanel>

</StackPanel>

</design:Card>

</Grid>

</Page>

Рисунок 2.1.7 – Продолжение листинга AddContractPage.xaml

using System.Windows;

using System.Windows.Controls;

using Microsoft.EntityFrameworkCore;

using Microsoft.VisualBasic.CompilerServices;

using RealtorSystemDesk.Database;

using RealtorSystemDesk.Models;

using RealtorSystemDesk.Services;

namespace RealtorSystemDesk.Pages.ClientManagePages;

public partial class AddContractPage : Page

{

private NewContractModel \_model;

private RealEstateObject \_estateObject;

public AddContractPage(int clientId)

{

\_model = new() { ClientId = clientId, DateCreate = DateTime.Today };

\_estateObject = new();

InitializeComponent();

DataContext = \_model;

ObjectPanel.DataContext = \_estateObject;

}

private void SaveButton\_OnClick(object sender, RoutedEventArgs e)

{

try

{

if (\_model.DateCreate != null && \_model.Days > 0 && \_model.Type != null)

{

Contract contract = new()

{

ClientId = \_model.ClientId,

UserId = App.AuthorizedUser.Id,

Type = \_model.Type,

DateCreate = DateOnly.FromDateTime(\_model.DateCreate),

ValidUntil = DateOnly.FromDateTime(DateTime.Today.AddDays(\_model.Days)),

};

\_estateObject.Contract = contract;

if (TypeComboBox.SelectedIndex == 1 && AddObject())

Рисунок 2.1.8 – Листинг AddContractPage.xaml.cs

Продолжение приложения 2

{

List<RealEstateObjectDocument> documents = new();

for (int i = 1; i < 7; i++)

documents.Add(new() { ObjectNumber = \_estateObject.CadastralNumber, DocumentTypeId = i });

Db.Context.RealEstateObjectDocuments.AddRange(documents);

Db.Context.Contracts.Add(contract);

DatabaseSaveService.SaveWithMessage("Данные добавлены");

NavigationService.GoBack();

}

}

else

MessageService.ShowInfo("Заполните все поля!");

}

catch (Exception exception)

{

Console.WriteLine(exception);

MessageService.ShowError(exception);

}

}

private bool AddObject()

{

try

{

if (!String.IsNullOrEmpty(\_estateObject.Address) &&

!String.IsNullOrEmpty(\_estateObject.CadastralNumber) &&

\_estateObject is { Square: > 0, Price: > 0, RoomsCount: > 0, Type: not null })

{

if (Db.Context.RealEstateObjects.Find(\_estateObject.CadastralNumber) != null)

{

MessageService.ShowError(new Exception("Объект с таким кадастровым номером уже существует!"));

return false;

}

switch (\_estateObject.Type.Id)

{

case 1:

if (\_estateObject is { FloorsCount: > 0, PlotSquare: > 0 } &&

!String.IsNullOrEmpty(\_estateObject.Material))

{

Db.Context.RealEstateObjects.Add(\_estateObject);

return true;

}

break;

case 2:

if (\_estateObject is { Floor: > 0, Class: not null })

{

Рисунок 2.1.9 – Продолжение листинга AddContractPage.xaml.cs

Продолжение приложения 2

Db.Context.RealEstateObjects.Add(\_estateObject);

return true;

}

break;

}

}

MessageService.ShowInfo("Заполните все поля");

}

catch (Exception e)

{

Console.WriteLine(e);

MessageService.ShowError(e);

}

return false;

}

private async void AddContractPage\_OnLoaded(object sender, RoutedEventArgs e)

{

try

{

TypeComboBox.ItemsSource = await Db.Context.ContractTypes.ToListAsync();

ObjectTypeComboBox.ItemsSource = await Db.Context.RealEstateObjectTypes.ToListAsync();

ClassComboBox.ItemsSource = Db.Context.RealEstateObjectClasses.ToList();

}

catch (Exception exception)

{

Console.WriteLine(exception);

MessageService.ShowError(exception);

}

}

private void TypeComboBox\_OnSelectionChanged(object sender, SelectionChangedEventArgs e)

{

try

{

ContractType type = (ContractType)TypeComboBox.SelectedItem;

ObjectPanel.Visibility = type.Id == 2 ? Visibility.Visible : Visibility.Collapsed;

}

catch (Exception exception)

{

Console.WriteLine(exception);

MessageService.ShowError(exception);

}

}

Рисунок 2.1.10 – Продолжение листинга AddContractPage.xaml.cs

Продолжение приложения 2

private void ObjectTypeComboBox\_OnSelectionChanged(object sender, SelectionChangedEventArgs e)

{

RealEstateObjectType type = (RealEstateObjectType)ObjectTypeComboBox.SelectedItem;

if (type.Id == 1)

{

HousePanel.Visibility = Visibility.Visible;

ApartmentPanel.Visibility = Visibility.Collapsed;

}

else

{

HousePanel.Visibility = Visibility.Collapsed;

ApartmentPanel.Visibility = Visibility.Visible;

}

}

}

Рисунок 2.1.11 – Продолжение листинга AddContractPage.xaml.cs

<Page x:Class="RealtorSystemDesk.Pages.ClientManagePages.ClientInfoPagePage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:design="http://materialdesigninxaml.net/winfx/xaml/themes"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:RealtorSystemDesk.Pages.ClientManagePages"

mc:Ignorable="d"

Loaded="ClientInfoPagePage\_OnLoaded"

Title="Информация о клиенте"

d:DesignHeight="650"

d:DesignWidth="800">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*" />

<ColumnDefinition Width="1.3\*" />

</Grid.ColumnDefinitions>

<design:Card Grid.Column="0"

Margin="5"

Padding="5">

<StackPanel Orientation="Vertical">

<TextBlock Style="{DynamicResource MaterialDesignTitleSmallTextBlock}"

VerticalAlignment="Center"

HorizontalAlignment="Center"

Text="Информация о клиенте" />

<DockPanel>

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Margin="7"

FontWeight="Medium"

Text="Фамилия:" />

Рисунок 2.1.12 – Листинг ClientInfoPage.xaml

Продолжение приложения 2

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="{Binding LastName}" />

</DockPanel>

<DockPanel>

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Margin="7"

FontWeight="Medium"

Text="Имя:" />

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="{Binding FirstName}" />

</DockPanel>

<DockPanel>

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Margin="7"

FontWeight="Medium"

Text="Отчество:" />

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="{Binding MiddleName}" />

</DockPanel>

<DockPanel>

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Margin="7"

FontWeight="Medium"

Text="Дата рождения:" />

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="{Binding BirthDate}" />

</DockPanel>

<DockPanel>

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Margin="7"

FontWeight="Medium"

Text="Телефон:" />

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Margin="7"

Text="{Binding Phone}" />

</DockPanel>

<DockPanel>

Рисунок 2.1.13 – Продолжение листинга ClientInfoPage.xaml

Продолжение приложения 2

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Margin="7"

FontWeight="Medium"

Text="Почта:" />

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Margin="7"

Text="{Binding Email}" />

</DockPanel>

<DockPanel>

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Margin="7"

FontWeight="Medium"

Text="Пол:" />

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Margin="7"

Text="{Binding Gender.Name}" />

</DockPanel>

<DockPanel>

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Margin="7"

FontWeight="Medium"

Text="Серия паспорта:" />

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="{Binding Passport.Serial}" />

</DockPanel>

<DockPanel>

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Margin="7"

FontWeight="Medium"

Text="Номер паспорта:" />

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="{Binding Passport.Number}" />

</DockPanel>

<DockPanel>

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Margin="7"

FontWeight="Medium"

Рисунок 2.1.14 – Продолжение листинга ClientInfoPage.xaml

Продолжение приложения 2

Text="Дата выдачи паспорта:" />

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="{Binding Passport.IssuedDate}" />

</DockPanel>

<DockPanel>

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Margin="7"

FontWeight="Medium"

Text="Кем выдан паспорт:" />

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Margin="7"

Text="{Binding Passport.IssuedBy}" />

</DockPanel>

<Button x:Name="EditButton"

Margin="5"

Content="Редактировать данные"

Click="EditButton\_OnClick" />

<Button x:Name="ArchiveButton"

Margin="5"

Content="Архивировать"

Click="ArchiveClientButton\_OnClick" />

</StackPanel>

</design:Card>

<Grid Grid.Column="1">

<Grid.RowDefinitions>

<RowDefinition Height="\*" />

<RowDefinition Height="\*" />

</Grid.RowDefinitions>

<design:Card Grid.Column="0"

Grid.Row="0"

Margin="5"

Padding="5">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="Auto" />

<RowDefinition Height="\*" />

</Grid.RowDefinitions>

<StackPanel Grid.Row="0">

<TextBlock Style="{DynamicResource MaterialDesignTitleSmallTextBlock}"

VerticalAlignment="Center"

HorizontalAlignment="Center"

Text="Договоры" />

<Button x:Name="AddContractButton"

HorizontalAlignment="Left"

Style="{DynamicResource MaterialDesignFlatButton}"

Рисунок 2.1.15 – Продолжение листинга ClientInfoPage.xaml

Продолжение приложения 2

FontSize="17"

Content="+"

Click="AddContractButton\_OnClick" />

</StackPanel>

<ListView x:Name="ContractListView"

Grid.Row="1">

<ListView.ItemTemplate>

<DataTemplate>

<StackPanel Orientation="Vertical">

<DockPanel>

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Margin="7"

FontWeight="Medium"

Text="Дата составления:" />

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Margin="7,0"

Text="{Binding DateCreate}" />

</DockPanel>

<DockPanel>

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Margin="7"

FontWeight="Medium"

Text="Действителен до:" />

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Margin="7,0"

Text="{Binding ValidUntil}" />

</DockPanel>

<DockPanel>

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Margin="7"

FontWeight="Medium"

Text="Тип:" />

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Margin="7,0"

Text="{Binding Type.Name}" />

</DockPanel>

<StackPanel Orientation="Horizontal"

HorizontalAlignment="Center">

Рисунок 2.1.16 – Продолжение листинга ClientInfoPage.xaml

Продолжение приложения 2

<Button Content="Объект"

Style="{DynamicResource MaterialDesignFlatButton}"

Visibility="{Binding ObjectButtonVisibility}"

Margin="5"

Click="NavigateToObject\_OnClick" />

<Button Style="{DynamicResource MaterialDesignFlatButton}"

Margin="5"

Content="Скачать"

Click="PrintButton\_OnClick" />

</StackPanel>

</StackPanel>

</DataTemplate>

</ListView.ItemTemplate>

</ListView>

</Grid>

</design:Card>

<design:Card Grid.Column="0"

Grid.Row="1"

Margin="5"

Padding="5">

<StackPanel Orientation="Vertical">

<TextBlock Style="{DynamicResource MaterialDesignTitleSmallTextBlock}"

VerticalAlignment="Center"

HorizontalAlignment="Center"

Text="Документы" />

<Button x:Name="AddDocumentButton"

Content="+"

FontSize="17"

HorizontalAlignment="Left"

Style="{DynamicResource MaterialDesignFlatButton}"

Click="DocumentAddButton\_OnClick" />

<DataGrid x:Name="FileDataGrid"

IsReadOnly="True"

AutoGenerateColumns="False">

<DataGrid.Columns>

<DataGridTextColumn Width="2\*"

Header="Название файла:"

Binding="{Binding FileName}" />

<DataGridTemplateColumn Width="\*">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<Button Content="Скачать"

Style="{DynamicResource MaterialDesignFlatButton}"

Рисунок 2.1.17 – Продолжение листинга ClientInfoPage.xaml

Продолжение приложения 2

Click="DocumentLoadButton\_OnClick" />

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

<DataGridTemplateColumn Width="\*">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<Button Content="Удалить"

Style="{DynamicResource MaterialDesignFlatButton}"

Click="DocumentDeleteButton\_OnClick" />

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

</DataGrid.Columns>

</DataGrid>

</StackPanel>

</design:Card>

</Grid>

</Grid>

</Page>

Рисунок 2.1.18 – Продолжение листинга ClientInfoPage.xaml

using System.IO;

using System.Windows;

using System.Windows.Controls;

using Microsoft.EntityFrameworkCore;

using Microsoft.Win32;

using RealtorSystemDesk.Database;

using RealtorSystemDesk.Pages.ObjectManagePages;

using RealtorSystemDesk.Services;

using Xceed.Document.NET;

using Xceed.Words.NET;

namespace RealtorSystemDesk.Pages.ClientManagePages;

public partial class ClientInfoPagePage : Page

{

private Client \_client;

private int \_id;

public ClientInfoPagePage(int id)

{

\_id = id;

InitializeComponent();

}

private async void ClientInfoPagePage\_OnLoaded(object sender, RoutedEventArgs e)

{

try

{

Рисунок 2.1.19 –Листинг ClientInfoPage.xaml.cs

Продолжение приложения 2

\_client = await Db

.Context.Clients

.Include(c => c.Passport)

.Include(c => c.Gender)

.FirstAsync(c => c.PassportId == \_id);

DataContext = \_client;

if (\_client.IsArchive.Value)

{

EditButton.Visibility = Visibility.Collapsed;

AddContractButton.Visibility = Visibility.Collapsed;

AddDocumentButton.Visibility = Visibility.Collapsed;

ArchiveButton.Visibility = Visibility.Collapsed;

}

LoadData();

}

catch (Exception exception)

{

Console.WriteLine(exception);

MessageService.ShowError(exception);

}

}

private async void LoadData()

{

try

{

ContractListView.ItemsSource = null;

ContractListView.ItemsSource =

await Db

.Context.Contracts

.Include(c => c.Type)

.Where(c => c.ClientId == \_client.PassportId && c.UserId == App.AuthorizedUser.Id)

.ToListAsync();

FileDataGrid.ItemsSource = null;

FileDataGrid.ItemsSource =

await Db

.Context.ClientFiles

.Where(c => c.ClientId == \_client.PassportId)

.ToListAsync();

}

catch (Exception e)

{

Console.WriteLine(e);

MessageService.ShowError(e);

}

}

private void EditButton\_OnClick(object sender, RoutedEventArgs e) =>

NavigationService.Navigate(new EditClientPage(\_client));

private void PrintButton\_OnClick(object sender, RoutedEventArgs e)

Рисунок 2.1.20 – Продолжение листинга ClientInfoPage.xaml.cs

Продолжение приложения 2

{

try

{

Contract contract = ((Button)sender).DataContext as Contract;

if (contract.TypeId == 2)

contract = Db.Context.Contracts.Include(c => c.Client)

.Include(c => c.RealEstateObject)

.Include(c => c.RealEstateObject.Type)

.Include(c => c.Client.Passport)

.First(c => c.Id == contract.Id);

else

contract = Db.Context.Contracts.Include(c => c.Client)

.Include(c => c.Client.Passport)

.First(c => c.Id == contract.Id);

SaveFileDialog dialog = new()

{

FileName =

$"договор\_{contract.Client.LastName}\_{contract.DateCreate.Value.ToDateTime(new()).ToString("dd-mm-yyyy")}",

AddExtension = true,

DefaultExt = "docx"

};

if (dialog.ShowDialog() == true)

{

var document = contract.TypeId == 2

? DocX.Load(@"../../../resources/sell\_template.docx")

: DocX.Load(@"../../../resources/buy\_template.docx");

List<Bookmark> bookmarks = document.GetBookmarks().ToList();

bookmarks.Find(c => c.Name == "CONTRACT\_ID").SetText(contract.Id.ToString());

bookmarks.Find(c => c.Name == "DATE\_DAY").SetText(contract.DateCreate.Value.Day.ToString());

bookmarks.Find(c => c.Name == "DATE\_MONTH").SetText(GetMonthName(contract.DateCreate.Value.Month));

bookmarks.Find(c => c.Name == "DATE\_YEAR").SetText(contract.DateCreate.Value.Year.ToString());

bookmarks.Find(c => c.Name == "ORG\_NAME").SetText(App.AuthorizedUser.LastName);

bookmarks.Find(c => c.Name == "CLIENT\_NAME").SetText(contract.Client.FullName);

bookmarks.Find(c => c.Name == "CLIENT\_PASS\_S").SetText(contract.Client.Passport.Serial.ToString());

bookmarks.Find(c => c.Name == "CLIENT\_PASS\_N").SetText(contract.Client.Passport.Number.ToString());

bookmarks.Find(c => c.Name == "CLIENT\_PASS\_BY").SetText(contract.Client.Passport.IssuedBy);

bookmarks.Find(c => c.Name == "CLIENT\_PASS\_DATE")

.SetText(contract.Client.Passport.IssuedDate.Value.ToString());

bookmarks.Find(c => c.Name == "CLIENT\_PHONE").SetText(contract.Client.Phone);

bookmarks.Find(c => c.Name == "USER\_NAME").SetText(

Рисунок 2.1.21 – Продолжение листинга ClientInfoPage.xaml.cs

Продолжение приложения 2

$"{App.AuthorizedUser.LastName} {App.AuthorizedUser.FirstName} {App.AuthorizedUser.MiddleName}");

bookmarks.Find(c => c.Name == "CONTRACT\_DATE\_VALID")

.SetText(contract.ValidUntil.Value.ToDateTime(new()).ToString("dd.MM.yy"));

bookmarks.Find(c => c.Name == "REWARD").SetText(contract.RealtorReward);

if (contract.TypeId == 2)

{

bookmarks.Find(c => c.Name == "OBJECT\_TYPE")

.SetText(contract.RealEstateObject.Type.Name);

bookmarks.Find(c => c.Name == "OBJECT\_ADDRESS").SetText(contract.RealEstateObject.Address);

bookmarks.Find(c => c.Name == "OBJECT\_PRICE").SetText(contract.RealEstateObject.Price.ToString());

}

document.SaveAs(dialog.FileName);

}

}

catch (Exception exception)

{

Console.WriteLine(exception);

MessageService.ShowError(exception);

}

}

private string GetMonthName(int month)

{

try

{

switch (month)

{

case 1:

return "Января";

case 2:

return "Февраля";

case 3:

return "Марта";

case 4:

return "Апреля";

case 5:

return "Мая";

case 6:

return "Июня";

case 7:

return "Июля";

case 8:

return "Августа";

case 9:

return "Сентября";

case 10:

return "Октября";

case 11:

Рисунок 2.1.22 – Продолжение листинга ClientInfoPage.xaml.cs

Продолжение приложения 2

return "Ноября";

case 12:

return "Декабря";

}

}

catch (Exception e)

{

Console.WriteLine(e);

MessageService.ShowError(e);

}

return "";

}

private void AddContractButton\_OnClick(object sender, RoutedEventArgs e) =>

NavigationService.Navigate(new AddContractPage(\_client.PassportId));

private void DocumentLoadButton\_OnClick(object sender, RoutedEventArgs e)

{

try

{

ClientFile file = (((Button)sender).DataContext as ClientFile)!;

SaveFileDialog dialog = new SaveFileDialog()

{ DefaultExt = file.FileName.Split('.').Last(), FileName = file.FileName };

if (dialog.ShowDialog() == true)

{

File.WriteAllBytes(dialog.FileName, file.FileData);

Clipboard.SetText(dialog.FileName);

MessageService.ShowOk("Файл сохранен! Путь к файлу скопирован в буфер обмена.");

}

}

catch (Exception exception)

{

Console.WriteLine(exception);

MessageService.ShowError(exception);

}

}

private void DocumentDeleteButton\_OnClick(object sender, RoutedEventArgs e)

{

try

{

ClientFile file = (ClientFile)((Button)sender).DataContext;

Db.Context.ClientFiles.Remove(file);

DatabaseSaveService.SaveWithMessage();

LoadData();

}

catch (Exception exception)

{

Console.WriteLine(exception);

Рисунок 2.1.23 – Продолжение листинга ClientInfoPage.xaml.cs

Продолжение приложения 2

MessageService.ShowError(exception);

}

}

private async void DocumentAddButton\_OnClick(object sender, RoutedEventArgs e)

{

try

{

OpenFileDialog dialog = new() { Filter = "documents (.docx, .doc, .pdf)|\*.docx; \*.doc; \*.pdf; " };

if (dialog.ShowDialog() == true)

{

ClientFile file = new()

{

ClientId = \_client.PassportId, FileData = File.ReadAllBytes(dialog.FileName),

FileName = dialog.SafeFileName

};

Db.Context.ClientFiles.Add(file);

DatabaseSaveService.SaveWithMessage("Документ загружен!");

LoadData();

}

}

catch (Exception exception)

{

Console.WriteLine(exception);

MessageService.ShowError(exception);

}

}

private void ArchiveClientButton\_OnClick(object sender, RoutedEventArgs e)

{

try

{

\_client.IsArchive = true;

DatabaseSaveService.SaveWithMessage("Клиент добавлен в архив");

NavigationService.GoBack();

}

catch (Exception exception)

{

Console.WriteLine(exception);

MessageService.ShowError(exception);

}

}

private void NavigateToObject\_OnClick(object sender, RoutedEventArgs e) =>

NavigationService.Navigate(new ObjectInfoPage(((Contract)((Button)sender).DataContext).Id));

}

Рисунок 2.1.24 – Продолжение листинга ClientInfoPage.xaml.cs

Продолжение приложения 2

<Page x:Class="RealtorSystemDesk.Pages.ClientManagePages.EditClientPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:design="http://materialdesigninxaml.net/winfx/xaml/themes"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:RealtorSystemDesk.Pages.ClientManagePages"

mc:Ignorable="d"

Title="Управление клиентом"

Loaded="EditClientPage\_OnLoaded"

d:DesignHeight="450"

d:DesignWidth="800">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*" />

<ColumnDefinition Width="\*" />

</Grid.ColumnDefinitions>

<design:Card Grid.Column="0"

Margin="5"

Padding="5">

<StackPanel Orientation="Vertical">

<DockPanel>

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="Фамилия:" />

<TextBox HorizontalAlignment="Right"

Margin="7"

MaxLength="100"

Width="200"

Text="{Binding LastName}" />

</DockPanel>

<DockPanel>

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="Имя:" />

<TextBox HorizontalAlignment="Right"

Margin="7"

MaxLength="100"

Width="200"

Text="{Binding FirstName}" />

</DockPanel>

<DockPanel>

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="Отчество:" />

<TextBox HorizontalAlignment="Right"

Margin="7"

MaxLength="100"

Width="200"

Text="{Binding MiddleName}" />

</DockPanel>

Рисунок 2.1.25 – Продолжение листинга EditClientPage.xaml

<DockPanel>

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="Дата рождения:" />

<DatePicker HorizontalAlignment="Right"

Margin="7"

Width="200"

SelectedDate="{Binding BirthDateTime}" />

</DockPanel>

<DockPanel>

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="Телефон:" />

<TextBox HorizontalAlignment="Right"

Margin="7"

MaxLength="17"

Width="200"

Text="{Binding Phone}" />

</DockPanel>

<DockPanel>

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="Почта:" />

<TextBox HorizontalAlignment="Right"

Margin="7"

MaxLength="100"

Width="200"

Text="{Binding Email}" />

</DockPanel>

<DockPanel>

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="Пол:" />

<ComboBox HorizontalAlignment="Right"

Margin="7"

Width="200"

x:Name="GenderComboBox"

DisplayMemberPath="Name"

SelectedItem="{Binding Gender}" />

</DockPanel>

<Button x:Name="SaveButton"

Click="SaveButton\_OnClick"

Content="Сохранить" />

</StackPanel>

</design:Card>

<design:Card Grid.Column="1"

Margin="5"

Padding="5">

<StackPanel Orientation="Vertical">

<DockPanel>

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="Серия паспорта:" />

<TextBox HorizontalAlignment="Right"

Margin="7"

Width="200"

MaxLength="4"

Text="{Binding Passport.Serial}" />

</DockPanel>

<DockPanel>

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="Номер паспорта:" />

<TextBox HorizontalAlignment="Right"

Margin="7"

Width="200"

MaxLength="6"

Text="{Binding Passport.Number}" />

</DockPanel>

<DockPanel>

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="Дата выдачи паспорта:" />

<DatePicker HorizontalAlignment="Right"

Margin="7"

Width="200"

SelectedDate="{Binding Passport.IssuedDateTime}" />

</DockPanel>

<DockPanel>

<TextBlock Style="{DynamicResource MaterialDesignBodyMediumTextBlock}"

VerticalAlignment="Center"

Text="Кем выдан паспорт:" />

<TextBox HorizontalAlignment="Right"

Margin="7"

MaxLength="200"

Width="200"

Text="{Binding Passport.IssuedBy}" />

</DockPanel>

</StackPanel>

</design:Card>

</Grid>

</Page>

Рисунок 2.1.26 – Продолжение листинга EditClientPage.xaml