Минобрнауки России

Юго-Западный государственный университет

Кафедра программной инженерии

ОТЧЕТ

о преддипломной (производственной) практике

наименование вида и типа практики

на (в) ООО «МЦОБ. Онлайн-сервисы»

наименование предприятия, организации, учреждения

Студента 4 курса, группы ПО-11б

курса, группы

Газинского Антона Игоревича

фамилия, имя, отчество

Руководитель практики от предприятия, организации, учреждения

директор

должность, звание, степень

Куркина А. В.

фамилия и. о.

Оценка _____

_____

подпись, дата

Руководитель практики от университета

к.т.н. доцент

должность, звание, степень

Чаплыгин А. А.

фамилия и. о.

Оценка _____

_____

подпись, дата

Члены комиссии

_____ _____

подпись, дата фамилия и. о.

_____ _____

подпись, дата фамилия и. о.

_____ _____

подпись, дата фамилия и. о.

Курск 2025 г.

СОДЕРЖАНИЕ

ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

БД – база данных.

ИС – информационная система.

РП – рабочий проект.

СУБД – система управления базами данных.

ТЗ – техническое задание.

ТП – технический проект.

АН- агентство недвижимости.

1 Анализ предметной области

1.1 Описание предметной области

Агентство недвижимости (АН) — это организация, предоставляющая посреднические услуги при совершении сделок с недвижимостью. Деятельность АН охватывает широкий спектр операций, включая:

• Покупка и продажа недвижимости: поиск объектов, соответствующих требованиям клиентов, организация просмотров, переговоры о цене, оформление договоров купли-продажи.

• Аренда (долгосрочная и краткосрочная): подбор объектов для арендаторов, поиск арендаторов для собственников, составление договоров аренды.

• Управление недвижимостью (Property Management): обслуживание объектов недвижимости, контроль платежей, ремонт и техническое обслуживание.

• Консультационные услуги: оценка стоимости недвижимости, юридическое сопровождение сделок, помощь в получении ипотеки и т.д.

Эффективная работа АН предполагает обработку и анализ большого объёма информации, относящейся к различным категориям:

• Объекты недвижимости: квартиры, дома, земельные участки, коммерческая недвижимость (офисы, магазины, склады) и т.д. Информация об объектах включает адрес, характеристики (площадь, количество комнат, материалы стен, год постройки и т.д.), фотографии, цену, описание, юридические документы.

• Клиенты: потенциальные покупатели, продавцы, арендаторы и арендодатели. Информация о клиентах включает контактные данные, предпочтения, требования к недвижимости, историю сделок.

• Сотрудники: риелторы, менеджеры, юристы, оценщики и другие специалисты. Информация о сотрудниках включает контактные данные, специализацию, историю работы, комиссионные.

• Сделки: договоры купли-продажи, аренды, оказания услуг. Информация о сделках включает сведения об объекте, клиентах, сотрудниках, дате заключения, цене, условиях оплаты, комиссии АН.

• Рекламные кампании и источники: информация о размещенных объявлениях, каналах привлечения клиентов, результатах рекламных кампаний.

1.2 Цели и задачи агентства недвижимости

Основными целями АН являются:

• Максимизация прибыли: за счет успешных сделок и оказания качественных услуг.

• Удовлетворение потребностей клиентов: обеспечение оптимального подбора недвижимости и сопровождение сделок.

• Повышение эффективности работы сотрудников: оптимизация рабочих процессов, сокращение временных затрат и повышение производительности.

• Расширение клиентской базы: привлечение новых клиентов и удержание существующих.

• Укрепление репутации: Предоставление надежных и профессиональных услуг.

Для достижения этих целей АН выполняет следующие задачи:

• Поиск и привлечение клиентов: Использование различных каналов рекламы и маркетинга.

• Поиск и оценка объектов недвижимости: анализ рынка, поиск подходящих объектов, оценка их стоимости.

• Организация просмотров и переговоров: встречи с клиентами, организация показов объектов, ведение переговоров о цене и условиях сделки.

• Юридическое сопровождение сделок: проверка юридической чистоты объектов, подготовка и оформление договоров, регистрация сделок.

• Финансовый контроль: контроль платежей, ведение бухгалтерского учета, расчет комиссионных.

• Анализ рынка: сбор и анализ данных о рынке недвижимости, прогнозирование тенденций.

1.3 Особенности предметной области, влияющие на проектирование БД

При проектировании базы данных для АН необходимо учитывать следующие особенности:

• Большой объем данных: база данных должна быть способна обрабатывать большие объемы данных о недвижимости, клиентах и сделках.

• Необходимость поиска и фильтрации данных: пользователям нужна возможность быстрого поиска объектов по различным критериям (цена, площадь, местоположение, количество комнат и т. д.), а также фильтрации данных по различным параметрам.

• Необходимость формирования отчетов: база данных должна обеспечивать возможность формирования различных отчетов, необходимых для анализа деятельности АН (отчеты о сделках, комиссионных, продажах и т. д.).

• Безопасность данных: необходимо обеспечить защиту конфиденциальной информации о клиентах и сотрудниках.

• Масштабируемость: база данных должна быть масштабируемой, чтобы учитывать рост агентства и увеличение объема обрабатываемой информации.

• Интеграция с другими системами (возможно): интеграция с сайтом АН, CRM-системами, системами учета и отчетности.

1.4 Анализ бизнес-процессов

На основе анализа неформального описания предметной области были сформулированы бизнес-правила:

• У каждого объекта недвижимости должен быть владелец

• У каждого владельца должен быть телефон для связи

• Каждая сделка имеет определенный тип

• При регистрации каждой сделки данные о продавце и покупателе обязательны

• Каждый объект недвижимости, сотрудник, владелец, покупатель, сделка должны иметь уникальный код (ID)

• Один владелец может иметь несколько квартир в собственности

• Необходимо корректно вводить данные во всех полях

Ограничения целостности для таблицы ОБЪЕКТ НЕДВИЖИМОСТИ

• Код объекта недвижимости является уникальным для каждого объекта недвижимости, разрешены только цифры

• Количество комнат, цена- данные строки могут содержать только значения в виде цифр

• Недопустимы пустые значения во всех полях, кроме срока аренды

Ограничения целостности для таблицы ПОКУПАТЕЛЬ/АРЕНДАТОР

• Код покупателя/арендатора является уникальным для каждого покупателя/арендатора, разрешены только цифры

• Паспортные данные, контактные данные- данные строки могут содержать только значения в виде цифр

• Фамилия, имя, отчество – строка символов, длиной до 50 символов. Может содержать только буквы русского алфавита

• Недопустимы пустые значения во всех полях, кроме отчества клиента

Ограничения целостности для таблицы ПРОДАВЕЦ/АРЕНДОДАТЕЛЬ

Правила для контроля уникальности в ключевом поле и требования к типам данных и ограничения на допустимые значения данных во всех полях разрабатываются по аналогии с приведенными для таблицы ПОКУПАТЕЛЬ-/АРЕНДОДАТЕЛЬ.

Ограничения целостности для таблицы ДОГОВОР АРЕНДЫ

• Код договора аренды является уникальным для каждого договора, разрешены только цифры

• Дата заключения договора- календарная дата

• При заключении договора обязательно должны быть заполнены данные о арендодателе, арендаторе, сотруднике агентства, объекте недвижимости

• Недопустимы пустые значения во всех полях

Ограничения целостности для таблицы ДОГОВОР ПРОДАЖИ

• Правила для контроля уникальности в ключевом поле и требования к типам данных и ограничения на допустимые значения данных во всех полях разрабатываются по аналогии с приведенными для таблицы ДОГОВОР АРЕНДЫ

2 Техническое задание

2.1 Основание для разработки

Полное наименование системы: «База данных агентства недвижимости». Основанием для разработки программы является приказ ректора ЮЗГУ от «17» апреля 2025 г. №1828-с «О направлении (допуске) на практику».

2.2 Цель и назначение разработки

Программно-информационная система предназначена для создания договоров аренды и договоров купли-продажи в агентстве недвижимости. С системой должны работать следующие группы пользователей:

- сотрудник агентства недвижимости;
- руководство агентства недвижимости.

Сотрудник должен иметь возможность добавлять, удалять и редак- тировать клиентов и объекты недвижимости. Руководство агентства недвижимости должно иметь те же возможности, что и сотрудник, а также добавлять и удалять данные сотрудников.

Данная разработка направлена на оптимизацию деятельности агентства недвижимости.

В рамках этой разработки предусмотрены следующие задачи:

- создание структуры базы данных;
- разработка дизайна пользовательского интерфейса;
- разработка методов отображения данных из базы данных;
- разработка инструментов для администрирования базы данных, чтобы поддерживать актуальность информации.

2.3 Требования к программной системе

2.3.1 Требования к данным

Входными данными для системы являются:

- информация о сотруднике, предоставляемая им в процессе реги- страции в системе;

- информация о покупателе, предоставляемая им в процессе оформ- ления сделки;

- информация о владельце объекта недвижимости, предоставляемая им в процессе регистрации в системе;

- информация об объекте недвижимости;

Выходными данными для системы являются:

- список сотрудников;

- список клиентов;

- список объектов недвижимости;

- созданный договор аренды ;

- созданный договор купли-продажи;

- сообщения об ошибках.

2.3.2 Функциональные требования

Приложение имеет две группы пользователей с разными правами: руководство и сотрудники.

Сотрудникам должны быть доступны следующие функции:

- добавление покупателей;

- добавление владельцев недвижимости;

- добавление объектов недвижимости;

- просмотр информации о сотрудниках;

- просмотр информации о покупателях;

- просмотр информации о владельцах объектов недвижимости;

- просмотр информации о объектах недвижимости;

- удаление различной информации;

- создание договоров аренды;

- создание договоров купли-продажи;

- удаление договоров.

На рисунке 2.1изображены прецеденты для сотрудника агентства.

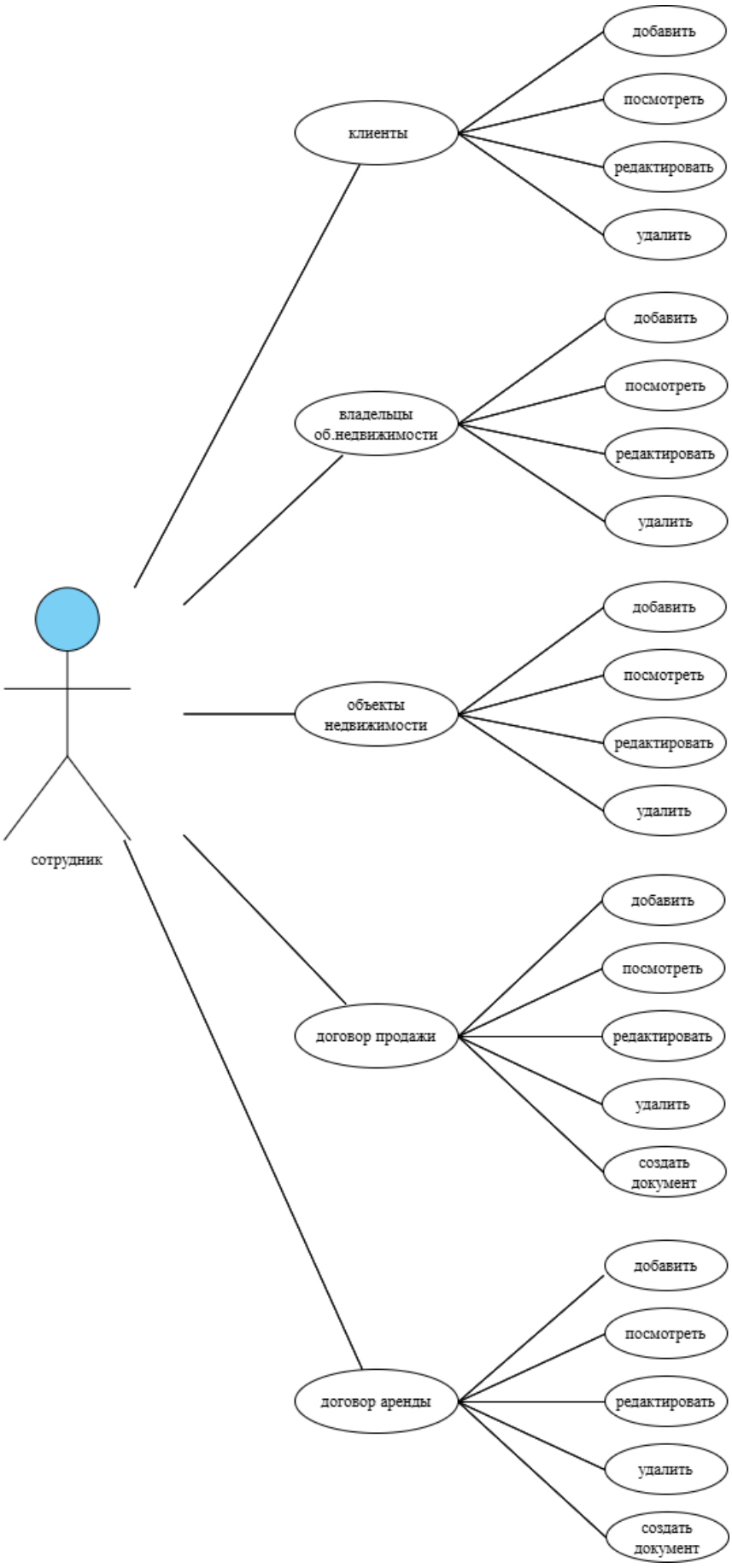

Рисунок 2.1 – Прецеденты для сотрудника

Руководству должны быть доступны следующие функции: - добавление сотрудников;

- удаление сотрудников;
- добавление покупателей;
- добавление владельцев недвижимости;
- добавление объектов недвижимости;
- просмотр информации о сотрудниках;
- просмотр информации о покупателях;
- просмотр информации о владельцах объектов недвижимости;
- просмотр информации о объектах недвижимости;
- удаление различной информации;
- создание договоров аренды;
- создание договоров купли-продажи;
- удаление договоров.

2.3.2.1 Сценарий прецедента сотрудника «добавление информации о покупателе/арендаторе»

Основной успешный сценарий для прецедента «добавление информации о покупателе/арендаторе».

1. Открыть окно «добавить покупателя/арендатора».
2. Заполнить данные покупателя/арендатора.
3. Нажать кнопку «добавить»

2.3.2.2 Сценарий прецедента сотрудника «просмотр информации о покупателе/арендаторе»

Основной успешный сценарий для прецедента «просмотр информации о покупателе/арендаторе».

1. Открыть окно «посмотреть покупателей/арендаторов».

2.3.2.3 Сценарий прецедента сотрудника «добавление информации о продавце/арендодателе»

Основной успешный сценарий для прецедента «добавление информации о продавце/арендодателе».

1. Открыть окно «добавить продавца/арендодателя».
2. Заполнить данные продавца/арендодателя.
3. Нажать кнопку «добавить».

2.3.2.4 Сценарий прецедента сотрудника «просмотр информации о продавце/арендодателе»

Основной успешный сценарий для прецедента «просмотр информации о продавце/арендодателе».

1.Открыть окно «посмотреть продавца/арендодателя».

2.3.2.5 Сценарий прецедента сотрудника «добавление объекта недвижимости»

Основной успешный сценарий для прецедента «добавление объекта недвижимости».

1. Открыть окно «добавить объект недвижимости».
2. Заполнить информацию об объекте недвижимости.
3. Нажать кнопку «добавить»

2.3.2.6 Сценарий прецедента сотрудника «просмотр объектов недвижимости»

Основной успешный сценарий для прецедента «просмотр объектов недвижимости».

1. Открыть окно «просмотреть объекты недвижимости».

2.3.2.7 Сценарий прецедента сотрудника «добавить договор купли/продажи»

Основной успешный сценарий для прецедента «добавить договор купли/продажи»

1. Открыть окно «добавить договор купли/продажи»
2. Заполнить все данные.
3. Нажать кнопку «добавить».

2.3.2.8 Сценарий прецедента сотрудника «посмотреть договора купли/продажи»

Основной успешный сценарий для прецедента «посмотреть договор купли/продажи».

1. Открыть окно «посмотреть договор купли/продажи».
2. Выбрать договор
3. Нажать кнопку «создать договор».
4. Сохранить договор на устройстве.

2.3.2.9 Сценарий прецедента сотрудника «добавить договор аренды»

Основной успешный сценарий для прецедента «добавить договор аренды»

1. Открыть окно «добавить договор аренды»
2. Заполнить все данные.
3. Нажать кнопку «добавить».

2.3.2.10 Сценарий прецедента сотрудника «посмотреть договор аренды»

Основной успешный сценарий для прецедента «просмотреть договора аренды».

1. Открыть окно «посмотреть договор аренды»

2. Выбрать договор

3. Нажать кнопку «создать договор»

4. Сохранить договор на устройстве.

2.3.2.11 Сценарий прецедента руководителя «Добавление сотрудника»

Основной успешный сценарий для прецедента «Добавление сотрудника»

1. Открыть окно «добавить сотрудника»

2. Заполнить данные сотрудника.

3. Нажать кнопку «добавить».

2.3.2.12 Сценарий прецедента руководителя «Просмотр сотрудников»

Основной успешный сценарий для прецедента «Просмотр сотрудников».

1. Открыть окно «просмотреть сотрудников агентства».

2.3.3 Требования пользователя к интерфейсу приложения

Приложение должно иметь следующие окна:

- Главное окно;
- Окно с добавлением покупателя/арендатора;
- Окно с просмотром покупателей/арендаторов;
- Окно с добавлением продавцов/арендодателей;
- Окно с просмотром продавцов/арендодателей;
- Окно с добавлением объектов недвижимости;
- Окно с просмотром объектов недвижимости;
- Окно с добавлением сотрудников;
- Окно с просмотром сотрудников;
- Окно с добавлением договоров продажи;
- Окно с просмотром договоров продажи;

- Окно с добавлением договоров аренды;
- Окно с просмотром договоров аренды.

На рисунке 2.2 представлен интерфейс приложения.

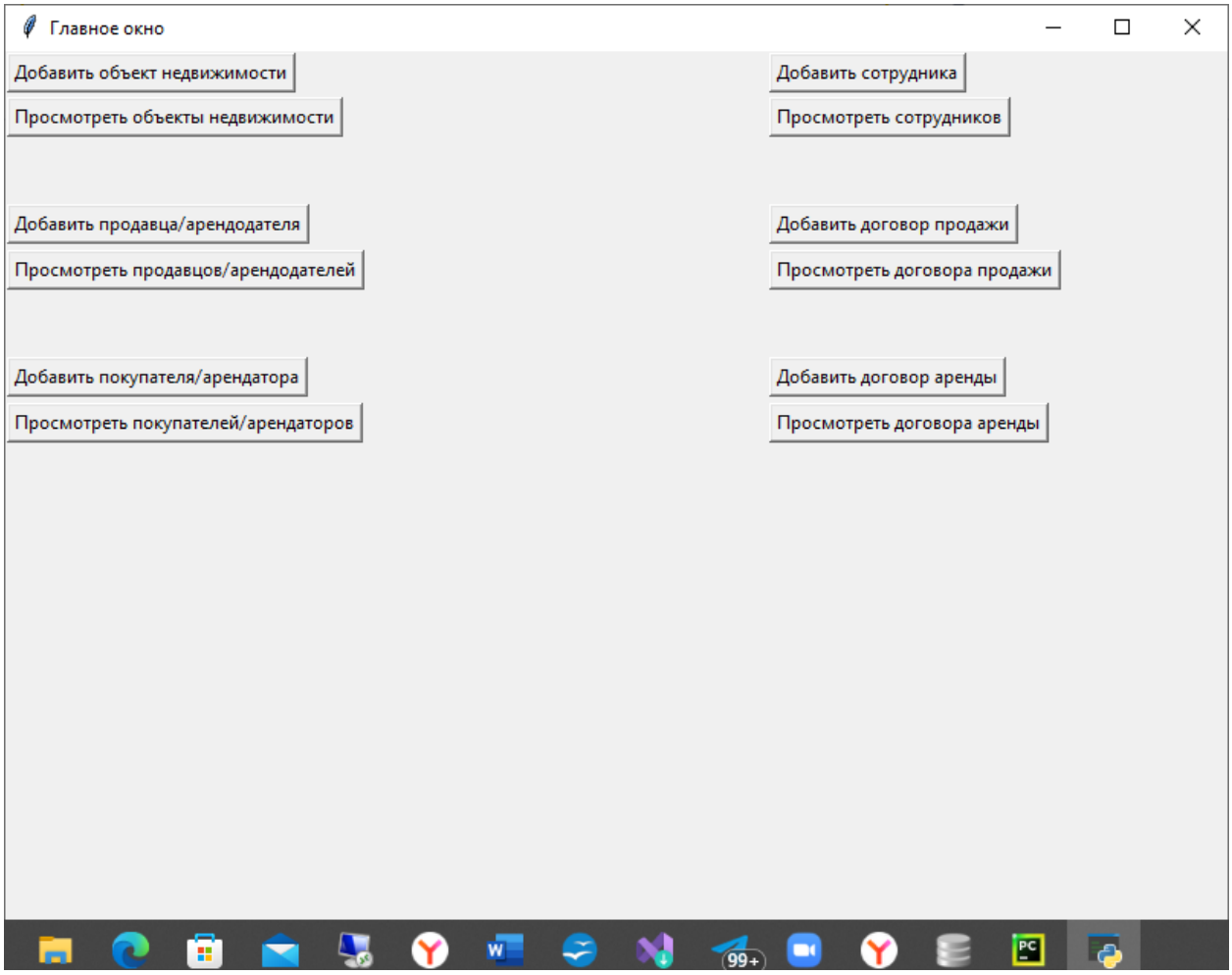

Рисунок 2.2 – Интерфейс приложения

2.3.4 Нефункциональные требования

2.3.4.1 Требования к безопасности

Необходимо устранить уязвимости, возможные для приложений.

- Роли пользователей: Установить разные роли (администратор, агент, клиент) с соответствующими правами доступа.

- Защита от атак с перебором паролей: Внедрить блокировку учетной записи после нескольких неудачных попыток входа.

- Шифрование данных: Использовать алгоритмы шифрования для хранения конфиденциальных данных, таких как пароли и личная информация

- Защита передачи данных: Все данные, передаваемые по сети, должны быть защищены с помощью HTTPS.

- Защита от SQL-инъекций: Использовать технологии ORM (Object-Relational Mapping) и параметризованные запросы для защиты от SQL-инъекций.

- Ведение журналов: Реализовать механизмы ведения журналов действий пользователей и администраторов для последующего анализа.

- Управление доступом к данным: Ограничить доступ пользователей к конфиденциальной информации (например, информации о других пользователях, сделках).

- Регулярные обновления: Обеспечить своевременное обновление всех библиотек и зависимостей для защиты от известных уязвимостей.

- Безопасность сервера: Хранить серверы в защищенных помещениях с ограниченным доступом.

- Резервное копирование: Регулярно делать резервные копии базы данных и важных данных.

- Обучение сотрудников: Проводить тренинги по вопросам безопасности для пользователей и сотрудников агентства.

- Соблюдение законодательства: Соблюдать местное и международное законодательство по защите персональных данных (такие как GDPR).

2.3.4.2 Требования к программному обеспечению

Для создания программного решения потребуется подготовить ряд ключевых элементов:

- фреймворк Laravel, обеспечивающий структуру и упрощающий разработку;

- среду исполнения Python, необходимую для запуска кода;

- систему управления базами данных;.

Laravel демонстрирует широкую совместимость, работая на актуальных операционных системах, таких как Windows, macOS и Linux.

2.3.4.3 Требования к аппаратному обеспечению

Для работы приложения требуется дисковое пространство не менее 1 Гб. Рекомендуется использовать процессор с 2 или более ядрами и частотой 2 ГГц или выше.

2.4 Требования к оформлению документации

Стадии разработки программного обеспечения и требования к программной документации для вычислительной техники, комплексов и систем любого назначения и области применения регламентируются ГОСТ 19.102–77. В состав программной документации входят:

- анализ предметной области;
- техническое задание;
- технический проект;
- рабочий проект.

3 Технический проект

3.1 Выбор технологии проектирования

3.1.1 Паттери MVC

MVC (Model-View-Controller) — это широко распространённый архитектурный шаблон, используемый при разработке графических пользовательских интерфейсов (GUI) и веб-приложений. Он разделяет приложение на три взаимосвязанных компонента: модель, представление и контроллер. Такое разделение позволяет упорядочить код, упростить разработку, тестирование и поддержку, а также повысить эффективность повторного использования компонентов. Рассмотрим MVC как архитектурный шаблон для разработки, выделив его сильные и слабые стороны, а также области применения.

3.2 Выбор средств разработки

3.2.1 Python

Python – это высокоуровневый, интерпретируемый, объектно-ориентированный язык программирования, который выделяется своей читаемостью и простотой. Его универсальность и богатая экосистема делают его отличным выбором для широкого спектра задач разработки. Рассмотрим Python с точки зрения выбора средств разработки, выделив его сильные и слабые стороны, а также области применения.

Преимущества Python для разработки:

• Читаемость и простота синтаксиса: Python разработан с акцентом на читаемость кода. Его синтаксис интуитивно понятен, что упрощает обучение и понимание кода, особенно для начинающих разработчиков. Меньше времени тратится на разбор сложных конструкций, позволяя сосредоточиться на логике программы. Это снижает порог вхождения и ускоряет процесс разработки.

• Большая и активная экосистема библиотек и фреймворков: Python обладает огромным количеством библиотек и фреймворков для решения практически любой задачи. Это позволяет разработчикам использовать готовые решения вместо написания кода с нуля, что значительно экономит время и ресурсы. Примеры:

• NumPy, SciPy, Pandas: для научных вычислений и анализа данных.

• Django, Flask: для веб-разработки.

• TensorFlow, PyTorch: для машинного обучения и искусственного интеллекта.

• Requests: для работы с HTTP-запросами.

• Beautiful Soup, Scrapy: для парсинга веб-страниц.

• Кроссплатформенность: Python работает на различных операционных системах (Windows, macOS, Linux и др.), что обеспечивает гибкость при разработке и развертывании приложений. Код, написанный на Python, может быть легко перенесен на другую платформу без значительных изменений.

• Поддержка различных парадигм программирования: Python поддерживает объектно-ориентированное, императивное и функциональное программирование, что позволяет разработчикам выбирать наиболее подходящий стиль для решения конкретной задачи.

• Интерпретируемость: Python – интерпретируемый язык, что означает, что код выполняется построчно без предварительной компиляции. Это упрощает отладку и позволяет быстро вносить изменения в код.

• Большое сообщество и широкая поддержка: Python имеет огромное и активное сообщество разработчиков, которые предоставляют помощь, создают библиотеки и фреймворки, а также активно участвуют в развитии языка. Это обеспечивает широкую поддержку и доступность ресурсов для решения возникающих проблем.

• Быстрая разработка (Rapid Prototyping): благодаря простоте синтаксиса и доступности библиотек Python идеально подходит для быстрой разработки прототипов и MVP (минимально жизнеспособного продукта). Недостатки Python для разработки:

• Производительность: Python, как интерпретируемый язык, часто уступает в производительности компилируемым языкам, таким как C++ или Java. Это может быть критично для задач, требующих высокой вычислительной мощности или низкой задержки. Однако для многих задач разница в производительности не является решающим фактором, и преимущества Python в скорости разработки перевешивают этот недостаток. Кроме того, можно использовать библиотеки, написанные на C/C++, для оптимизации критических участков кода.

• Глобальная блокировка интерпретатора (GIL): GIL ограничивает возможность параллельного выполнения кода Python в многопоточных приложениях, что может снижать производительность на многоядерных процессорах. Однако для задач, связанных с вводом-выводом (I/O), GIL не является существенным ограничением, и для достижения параллелизма можно использовать многопроцессорность или асинхронное программирование.

• Веб-разработка: создание веб-приложений и API с использованием фреймворков, таких как Django и Flask.

• Анализ данных и машинное обучение: обработка и анализ больших объемов данных, построение моделей машинного обучения с использованием библиотек, таких как NumPy, SciPy, Pandas, Scikit-learn, TensorFlow и PyTorch.

• Автоматизация и скрипты: написание скриптов для автоматизации рутинных задач, администрирования систем и сетей.

• Тестирование: автоматизация тестирования программного обеспечения с использованием фреймворков, таких как pytest и unittest.

• Разработка игр: создание игр с использованием библиотек, таких как Pygame.

• Научные вычисления: моделирование физических процессов, обработка изображений и сигналов с использованием библиотек, таких как NumPy, SciPy и Matplotlib.

• DevOps: Автоматизация процессов развертывания, мониторинга и управления инфраструктурой. Python в сравнении с другими языками:

• Python против Java: Python обычно проще в изучении и использовании, чем Java, но Java может быть более производительной для некоторых задач. Java также имеет более развитую систему статической типизации, которая помогает обнаруживать ошибки на этапе компиляции.

• Python против C++: C++ обеспечивает более высокую производительность, чем Python, но разработка на C++ требует больше времени и усилий. Python часто используется для создания прототипов и быстрого решения задач, а C++ — для задач, требующих максимальной производительности.

• Python против JavaScript: JavaScript является основным языком для разработки веб-интерфейсов, а Python чаще используется для серверной части веб-приложений. Однако с появлением Node.js JavaScript также можно использовать для разработки серверной части.

Python — это мощный и универсальный язык программирования, который отлично подходит для широкого спектра задач разработки. Его читаемость, простота и богатая экосистема позволяют учитывать специфику проекта и требования к производительности, но в большинстве случаев Python будет отличным выбором.

3.2.2 pgAdmin4

pgAdmin 4 — это современный и многофункциональный инструмент администрирования и разработки для СУБД PostgreSQL. Это один из самых популярных и широко используемых графических пользовательских интерфейсов (GUI) для работы с PostgreSQL, предоставляющий удобный и интуитивно понятный интерфейс для выполнения различных задач, от администрирования и мониторинга до разработки и отладки. Рассмотрим pgAdmin 4 с точки зрения выбора средств разработки, подчеркнув его сильные и слабые стороны, а также целевую аудиторию.

Преимущества pgAdmin 4 для разработки:

• Кроссплатформенность: pgAdmin 4 можно запускать как веб-приложение в браузере, что обеспечивает кроссплатформенность и позволяет использовать его в различных операционных системах (Windows, macOS,

Linux и др.) без необходимости установки дополнительных компонентов. Также существует настольная версия.

• Удобный и интуитивно понятный интерфейс: интерфейс pgAdmin 4 разработан с учетом удобства пользователя. Он предоставляет визуальные инструменты для управления серверами, базами данных, таблицами, представлениями, функциями и другими объектами PostgreSQL. Навигация по интерфейсу интуитивно понятна, что облегчает выполнение различных задач.

• Мощный редактор SQL: pgAdmin 4 оснащен мощным редактором SQL с подсветкой синтаксиса, автодополнением кода и возможностью выполнения нескольких запросов одновременно. Редактор также предоставляет инструменты для отладки SQL-кода и анализа планов запросов.

• Визуальные инструменты для управления базами данных: pgAdmin 4 предоставляет визуальные инструменты для создания, изменения и удаления баз данных, таблиц, представлений, функций и других объектов PostgreSQL. Это упрощает процесс проектирования и разработки баз данных, особенно для начинающих разработчиков.

• Поддержка расширений PostgreSQL: pgAdmin 4 поддерживает расширения PostgreSQL, что позволяет разработчикам использовать дополнительные функциональные возможности СУБД.

• Мониторинг производительности: pgAdmin 4 предоставляет инструменты для мониторинга производительности PostgreSQL, что позволяет выявлять узкие места и оптимизировать работу базы данных.

• Управление пользователями и ролями: pgAdmin 4 упрощает управление пользователями и ролями PostgreSQL, что позволяет контролировать доступ к данным и обеспечивать безопасность базы данных.

• Резервное копирование и восстановление данных: pgAdmin 4 предоставляет инструменты для резервного копирования и восстановления данных PostgreSQL, что обеспечивает защиту данных от потери.

• Бесплатность и открытый исходный код: pgAdmin 4 — это бесплатный инструмент с открытым исходным кодом, что позволяет использовать его без ограничений и модифицировать в соответствии с потребностями.

pgAdmin 4 — отличный выбор для разработки и администрирования баз данных PostgreSQL. Он предоставляет удобный и интуитивно понятный интерфейс с широким набором инструментов для выполнения различных задач. pgAdmin 4 особенно полезен для разработчиков, администраторов баз данных и аналитиков данных, работающих с PostgreSQL. Хотя существуют и другие альтернативные инструменты, pgAdmin 4 остается одним из самых популярных и востребованных клиентов для PostgreSQL благодаря своей функциональности, кроссплатформенности и бесплатному доступу. При выборе инструмента следует учитывать требования к ресурсам, уровень знаний и специфические потребности проекта.

3.2.3 Фреймворк Laravel

Laravel – это бесплатный, с открытым исходным кодом PHP-фреймворк, предназначенный для разработки современных веб-приложений, следующих архитектурному шаблону MVC (Model-View-Controller). Известный своей элегантностью, выразительностью и богатым набором встроенных инструментов, Laravel значительно упрощает и ускоряет процесс разработки, делая его популярным выбором среди PHP-разработчиков. Рассмотрим Laravel с точки зрения выбора средств разработки, выделив его сильные и слабые стороны, а также целевую аудиторию.

Преимущества Laravel для разработки:

• Элегантный синтаксис и выразительность: Laravel известен своим чистым и выразительным синтаксисом, который делает код легко читаемым и поддерживаемым. Это снижает когнитивную нагрузку на разработчиков и позволяет им быстрее понимать и изменять код.

• MVC-архитектура: Laravel использует архитектурный шаблон MVC, который разделяет приложение на три основных компонента: модель (данные), представление (интерфейс пользователя) и контроллер (логика прило-

жения). Это облегчает организацию кода, повторное использование компонентов и тестирование.

• Встроенная система маршрутизации: Laravel предоставляет мощную и гибкую систему маршрутизации, которая позволяет легко определять правила обработки HTTP-запросов и связывать их с соответствующими контроллерами.

• Шаблонизатор Blade: Blade – это простой, но мощный шаблонизатор в Laravel, который позволяет создавать динамические веб-страницы с использованием PHP-кода и специальных директив. Blade обеспечивает наследование шаблонов, компоненты и другие полезные функции.

• Миграции баз данных: Laravel предоставляет систему миграций баз данных, которая позволяет легко создавать и изменять структуру базы данных, а также отслеживать изменения в ее истории. Миграции упрощают развертывание приложения на различных средах.

• Artisan Console: Artisan – это консольная утилита Laravel, которая предоставляет множество полезных команд для автоматизации рутинных задач, таких как создание контроллеров, моделей, миграций, генерация кода и многое другое.

• Тестирование: Laravel разработан с учетом тестирования и предоставляет встроенные инструменты для написания модульных и интеграционных тестов.

• Безопасность: Laravel предоставляет встроенные инструменты для защиты от распространенных веб-угроз, таких как CSRF (Cross-Site Request Forgery), XSS (Cross-Site Scripting) и SQL-инъекции.

• Авторизация и аутентификация: Laravel упрощает реализацию систем авторизации и аутентификации пользователей, предоставляя готовые компоненты для регистрации, входа в систему, сброса пароля и т.д.

• Очереди: Laravel предоставляет систему очередей для обработки задач в фоновом режиме, что позволяет разгрузить основной поток приложения и улучшить его отзывчивость.

• Активное сообщество и документация: Laravel имеет большое и активное сообщество разработчиков, которые создают пакеты, делятся знаниями и оказывают помощь. Официальная документация Laravel хорошо написана и содержит множество примеров.

• Пакеты (Composer): расширяемость за счет использования Composer и обширной экосистемы пакетов.

Недостатки Laravel для разработки:

• Изучение: хотя Laravel имеет элегантный синтаксис, изучение фреймворка может потребовать времени, особенно для начинающих PHP-разработчиков. Необходимо освоить концепции MVC, ORM, шаблонизатора и другие компоненты Laravel.

• Производительность: Laravel, как и другие PHP-фреймворки, может уступать в производительности компилируемым языкам, таким как Java. Однако для большинства веб-приложений разница в производительности не является критической, и можно принять меры для оптимизации приложений на Laravel.

• Размер: Laravel — довольно большой фреймворк, что может привести к увеличению размера приложения и времени загрузки.

Laravel — отличный выбор для PHP-разработчиков, которые хотят создавать современные, элегантные и масштабируемые веб-приложения. Его выразительный синтаксис, богатый набор встроенных инструментов и активное сообщество делают его одним из самых популярных PHP-фреймворков в мире. При выборе Laravel стоит учитывать требования к производительности, сложность проекта и опыт команды разработчиков. Для проектов, требующих высокой производительности или глубокой интеграции с существующей инфраструктурой, могут подойти другие фреймворки. Но в целом Laravel обеспечивает отличный баланс между функциональностью, простотой использования и производительностью.

3.3 Архитектура программной системы

Система состоит из следующих основных компонентов:

3.3.1 Клиент-Приложение

Описание: Этот компонент объединяет уровень представления (пользовательский интерфейс) и уровень приложений (бизнес-логику). Он отвечает за взаимодействие с пользователем, обработку запросов и передачу данных в базу данных.

ТЕХНОЛОГИИ:

Python: Основной язык программирования для реализации приложения.

psycopg2: Библиотека Python для подключения и взаимодействия с сервером PostgreSQL.

Tkinter: для создания простого графического интерфейса (GUI) или текстового интерфейса.

ИЕТЕРФЕЙС ПОЛЬЗОВАТЕЛЯ:

Отображение данных о недвижимости, клиентах и сделках в виде таблицы или списка.

Предоставление форм для добавления, редактирования и удаления данных.

Реализация функциональности поиска и фильтрации данных по различным критериям.

Навигация по данным.

БИЗНЕС ЛОГИКА:

Получение данных от пользователя.

Проверка введенных данных (например, проверка формата даты, типа данных и т. д.).

Формирование SQL-запросов для взаимодействия с базой данных.

Обработка результатов запросов, полученных от базы данных.

Вывод результатов на экран.

3.3.2 Уровень данных

Описание: Отвечает за хранение и управление данными системы.

Технологии:

PostgreSQL: выбрана в качестве СУБД. PostgreSQL — мощная объектно-реляционная СУБД с открытым исходным кодом, отличающаяся высокой надежностью, производительностью и поддержкой стандартов SQL.

ФУНКЦИОНАЛЬНОСТЬ:

Хранение данных о недвижимости, клиентах и сделках в структурированном виде.

Обеспечение целостности данных с использованием ограничений, ключей и транзакций.

Предоставление доступа к данным через язык SQL.

Индексирование данных для ускорения поиска.

3.3.3 Технологии и инструменты разработки

Язык программирования: Python СУБД: PostgreSQL

Python-библиотека для PostgreSQL: psycopg2

GUI-библиотека (опционально): Tkinter

Инструменты разработки: IDE (PyCharm), система контроля версий (Git, GitHub)

3.3.4 Функциональность

Операции CRUD: реализация функций для создания, чтения, обновления и удаления данных в таблицах realty, client и owner.

Поиск и фильтрация: реализация возможности поиска объектов недвижимости по различным критериям (например, по типу, адресу, цене).

Вывод данных: отображение данных в удобном формате (таблица, список).

Добавление сделок: реализация возможности добавления информации о сделках в таблицу.

3.3.5 Перспективы развития

Реализация более продвинутых функций поиска и фильтрации данных.

Интеграция с внешними сервисами (например, картографическими сервисами).

3.4 Структура базы данных

На основе анализа неформального описания предметной области были определены наборы объектов и связей.

ОПРЕДЕЛЕНИЯ ОБЪЕКТОВ

Потенциальные объекты и атрибуты:

Каждый объект недвижимости характеризуется следующими параметрами:

• Код объекта недвижимости

• Тип сделки

• Регион

• Город

• Улица

• Номер дома

• Номер квартиры (если есть)

• Площадь, кв.м

• Кол-во комнат

• Срок сдачи (если сдается)

• Цена

Информация о владельце объекта недвижимости включает следующее:

• ФИО

• Паспортные данные

• Контактные данные

• ID недвижимости

Информация о покупателе/арендаторе включает следующее:

• ФИО

• Паспортные данные

• Контактные данные

Информация о сотруднике агентства включает следующее:

• ФИО

• Паспортные данные

• Контактные данные

• Дата устройства на работу в агентство

Каждая сделка по покупке/аренде характеризуется следующими параметрами:

• Номер сделки сделки

• Код объекта недвижимости

• Данные владельца

• Данные покупателя/арендатора

• Данные сотрудника агентства

• Тип сделки

• Срок аренды (если есть)

• Цена

• Дата сделки

Сущности и отношения между ними отображены на ER-диаграмме.

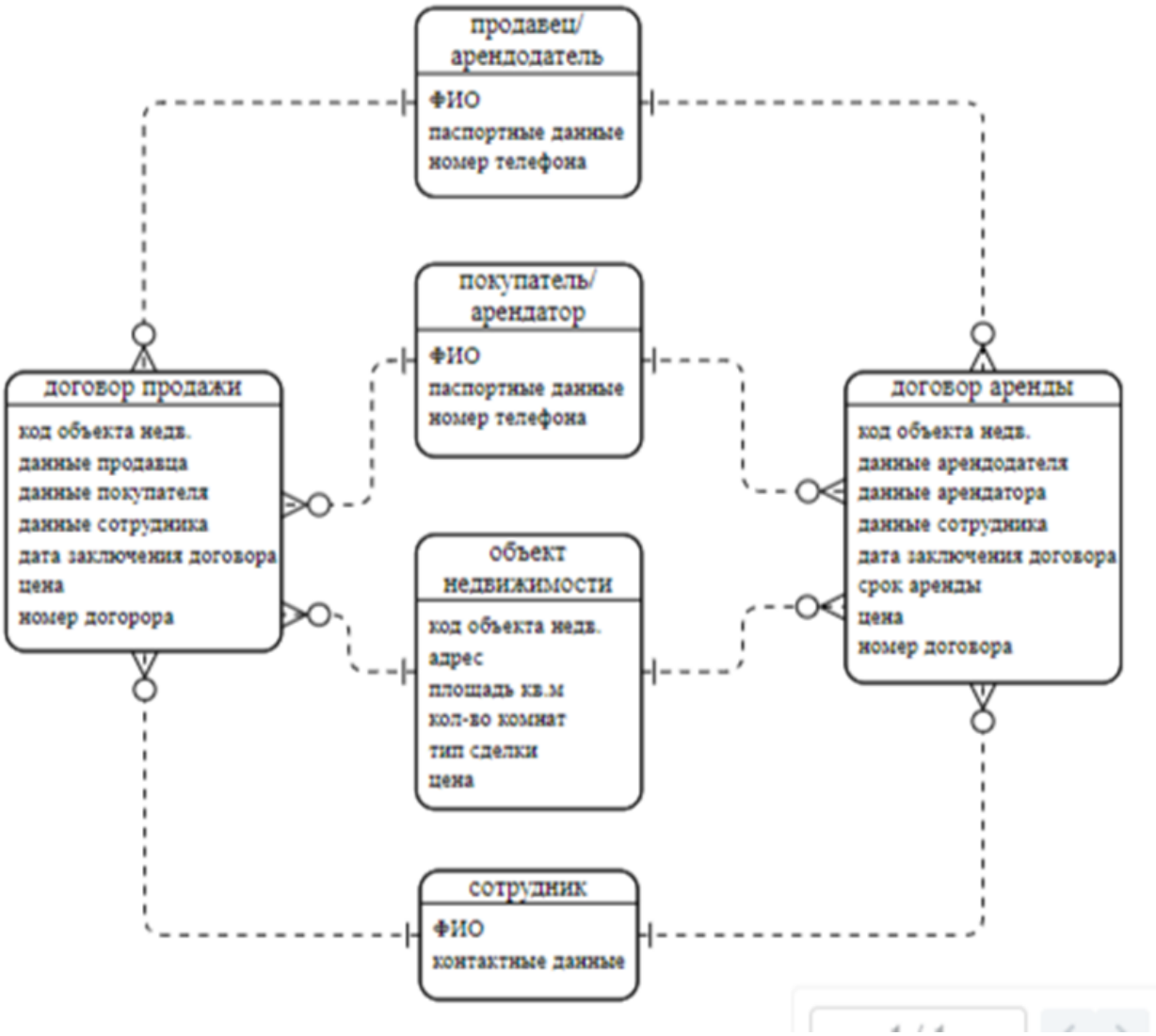

Рисунок 3.1 – ER-диаграмма

3.5 Логическое проектирование базы данных

3.5.1 Нормализация ER-модели данных

Первая нормальная форма: каждый атрибут данной модели имеет только одно значение.

Вторая форма: каждый атрибут зависит только от полного UID объекта.

Третья форма: каждый атрибут зависит только от UID своего объекта.

Объект «Продавец/арендодатель» соответствует 1NF, так как каждый атрибут имеет только одно значение.

Объект «Продавец/арендодатель» соответствует 2NF, так как каждый атрибут зависит от полного UID данного объекта.

Объект «Продавец/арендодатель» соответствует 3NF, так как каждый атрибут зависит только от UID данного объекта.

Объект «Покупатель/арендатор» соответствует 1NF, так как каждый атрибут имеет только одно значение.

Объект «Покупатель/арендатор» соответствует 2NF, так как каждый атрибут зависит от полного UID данного объекта.

Объект «Покупатель/арендатор» соответствует 3NF, так как каждый атрибут зависит только от UID данного объекта.

Объект «Объект недвижимости» соответствует 1NF, так как каждый атрибут имеет только одно значение.

Объект «Объект недвижимости» соответствует 2NF, так как каждый атрибут зависит от полного UID данного объекта.

Объект «Объект недвижимости» соответствует 3NF, так как каждый атрибут зависит только от UID данного объекта.

Объект «Сотрудник» соответствует 1NF, так как каждый атрибут имеет только одно значение.

Объект «Сотрудник» соответствует 2NF, так как каждый атрибут зависит от полного UID данного объекта.

Объект «Сотрудник» соответствует 3NF, так как каждый атрибут зависит только от UID данного объекта.

Объект «Договор продажи» соответствует 1NF, так как каждый атрибут имеет только одно значение.

Объект «Договор продажи» соответствует 2NF, так как каждый атрибут зависит от полного UID данного объекта.

Объект «Договор продажи» соответствует 3NF, так как каждый атрибут зависит только от UID данного объекта.

Объект «Договор аренды» соответствует 1NF, так как каждый атрибут имеет только одно значение.

На основе ER-модели данных в онлайн-сервисе Lucidcharts построена реляционная модель данных, показанная на рисунке 3.2.

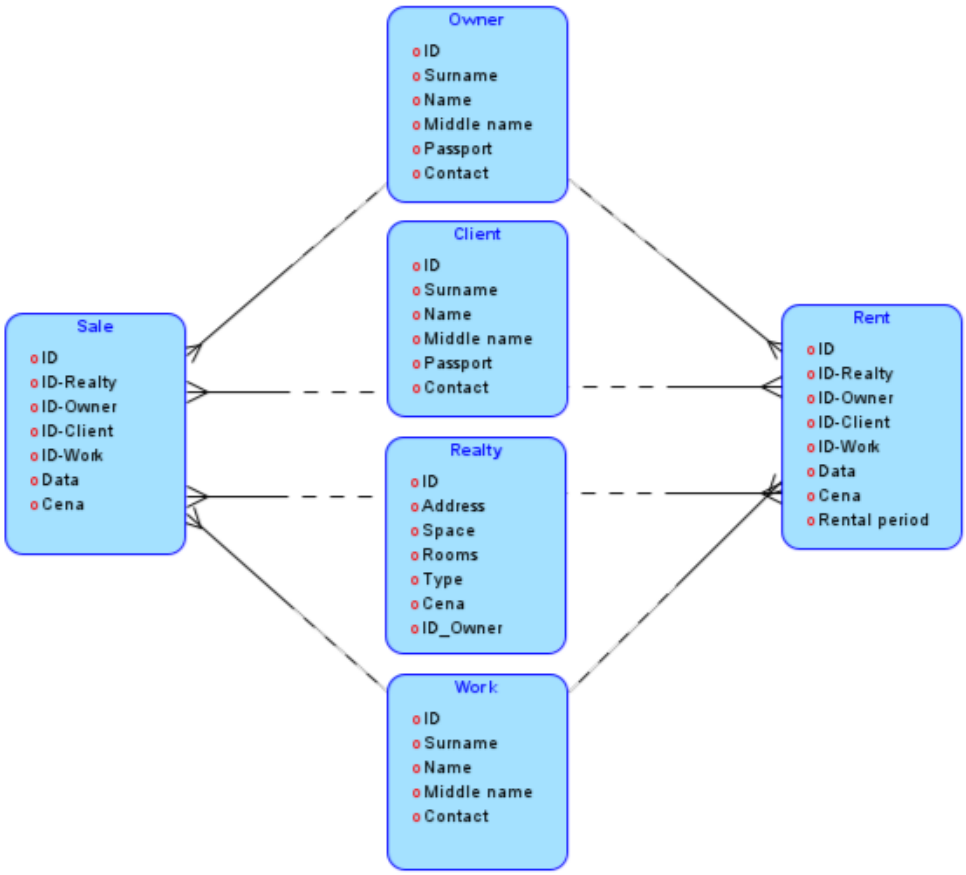

Рисунок 3.2 – Реляционная модель данных

Названия таблиц и названия столбцов каждой таблицы приведены в таблицах 3.1 -3.6.

Таблица 3.1 – Владелец объекта недвижимости

Имя таблицы	Краткое имя таблицы		
Owner	Own		
Key Type	Optionality	Column name	Data type
pk	*	Surname	TEXT
	*	Name	TEXT
	*	Middle name	TEXT
	*	Passport	INTEGER
	*	Contact	INTEGER

Таблица 3.2 – Клиент

Имя таблицы	Краткое имя таблицы		
Client	Client		
Key Type	Optionality	Column name	Data type
pk	*	Surname	TEXT
	*	Name	TEXT
	*	Middle name	TEXT
	*	Passport	INTEGER
	*	Contact	INTEGER

Таблица 3.3 – Объект недвижимости

Имя таблицы	Краткое имя таблицы		
Realty	Real		
Key Type	Optionality	Column name	Data type
pk	*	Address	TEXT
	*	Space	TEXT
	*	Rooms	INTEGER
	*	Type	TEXT
	*	Cena	INTEGER
	*	ID_owner	INTEGER

Таблица 3.4 – Сотрудник агентства

Имя таблицы	Краткое имя таблицы		
Work	Work		
Key Type	Optionality	Column name	Data type
pk	*	Surname	TEXT
	*	Name	TEXT
	*	Middle name	TEXT
	*	Contact	INTEGER

Таблица 3.5 – Договор продажи

Имя таблицы	Краткое имя таблицы		
sale	PTP		
Key Type	Optionality	Column name	Data type
pk	*	ID_Realty	INTEGER
	*	ID_Owner	INTEGER
	*	ID_Client	INTEGER
	*	ID_Work	INTEGER
	*	Data	DATE
	*	Cena	INTEGER

Таблица 3.6 – Договор аренды

Имя таблицы	Краткое имя таблицы		
Rent	Rent		
Key Type	Optionality	Column name	Data type
pk	*	ID_Realty	INTEGER
	*	ID_Owner	INTEGER
	*	ID_Client	INTEGER
	*	ID_Work	INTEGER
	*	Data	DATE
	*	Cena	INTEGER
	*	Rental period	TEXT

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Кузнецов С.Д. Базы данных: учебник для студ. учреждений высш. проф. образования / С.Д. Кузнецов. — 2-е изд., испр. — М.: Издательский центр «Академия», 2010. — 496 с.

2. Дейт, К. Дж. Введение в системы баз данных = An Introduction to Database Systems / К. Дж. Дейт. — 8-е изд. — М.: Вильямс, 2006. — 1328 с. (Общая теория баз данных)

3. Коннолли, Т., Бегг, К. Базы данных. Проектирование, реализация и сопровождение. Теория и практика. 3-е изд. / Т. Коннолли, К. Бегг. — М.: Вильямс, 2003. — 816 с. (Общая теория баз данных)

4. Рамкришнан Р., Гехре Д. Управление базами данных = Database Management Systems / Р. Рамкришнан, Д. Гехре. — 3-е изд. — М.: Вильямс, 2003. — 1056 с. (Общая теория баз данных)

5. Васильев А.Н. Python на примерах. Практический курс программирования. — СПб.: Наука и техника, 2017. — 432 с. (Основы языка Python)

6. Бизли Д.М. Справочник по Python (4-е издание). — Addison-Wesley Professional, 2009. — 704 страницы. (Справочник по Python)

7. Расперри П.Л., Расин Б. Сборник рецептов по анализу данных на Python — O’Reilly Media, 2015. — 514 страниц. (Рецепты по анализу данных на Python)

8. Петров А.А. Анализ предметной области «Агентство недвижимости» и разработка концептуальной модели базы данных // Вестник [Название университета/организации]. — 2020. — № 3. — С. 45-52. (Анализ предметной области)

9. Сидорова Е.В. Проектирование реляционной базы данных для управления недвижимостью // Информационные технологии. — 2019. — № 12. — С. 67-74. (Проектирование реляционных БД)

10. Иванов И.И., Козлов П.С. Методы оптимизации запросов в базах данных агентств недвижимости // Системы управления базами данных. — 2021. — № 2. — С. 102-110. (Оптимизация запросов)

11. Смирнов П.А. Применение ER-диаграмм для моделирования баз данных агентств недвижимости // Современные научные исследования и инновации. – 2018. – № 5. (ER-диаграммы)

12. Васильев С.В. Особенности построения баз данных для информационных систем агентств недвижимости // Наука и образование: сохраняя прошлое, создаём будущее. – 2022. – С. 88-90. (Особенности БД для агентств)

13. Борисова О.И. Проблемы внедрения и использования современных баз данных в агентствах недвижимости // Экономика и управление в XXI веке. – 2019. – С. 123-126. (Проблемы внедрения)

14. Официальная документация Python. — URL: https://docs.python.org/3/ (дата обращения: 15.05.2024). (Официальная документация)

15. Официальная документация SQLite. — URL: https://www.sqlite.org/docs.html (дата обращения: 15.05.2024). (Если вы используете SQLite)

16. psycopg.org — адаптер PostgreSQL для Python. — URL: https://www.psycopg.org/ (дата обращения: 15.05.2024). (Если вы используете PostgreSQL)

17. Учебник по SQL [Электронный ресурс] // W3Schools. — URL: https://www.w3schools.com/sql/ (дата обращения: 15.05.2024). (Основы SQL)

18. Настоящие уроки Python — URL: https://realpython.com/ (дата обращения: 15.05.2024) (Уроки Python)

19. pandas.pydata.org — библиотека Python для анализа данных. — URL: https://pandas.pydata.org/ (дата обращения: 15.05.2024) (Если вы используете Pandas)