

Geekbrains

**ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ ДЛЯ ЗАПИСИ В ГРУМИНГ-САЛОН**

Аналитик больших данных Мастер

Разработчик – Data Engineer

Пьянкова Дарья Максимовна

Екатеринбург

2025

|  |
| --- |
| Оглавление  [ЗАДАНИЕ 3](#_Toc185185413)  [**АННОТАЦИЯ** 5](#_Toc185185414)  [**ВВЕДЕНИЕ** 8](#_Toc185185415)  [**1.ХАРАКТЕРИСТИКА ПРОБЛЕМНОЙ СИТУАЦИИ И ПОСТАНОВКА ЗАДАЧ** 12](#_Toc185185416)  [**1.1 Характеристика проблемной области** 12](#_Toc185185417)  [**1.2 Аналитический обзор решений** 12](#_Toc185185418)  [**2. СИСТЕМНЫЙ АНАЛИЗ ОБЪЕКТА УПРАВЛЕНИЯ** 19](#_Toc185185419)  [**2.1 Концептуальное моделирование** 19](#_Toc185185420)  [**2.1.1 Модель вариантов использования (прецедентов).** 26](#_Toc185185421)  [**2.2 Функциональное моделирование** 51](#_Toc185185422)  [**2.3 Формирование требований к информационной системе** 54](#_Toc185185423)  [**3. ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ** 58](#_Toc185185424)  [**3.1 Проектирование архитектуры информационной системы** 58](#_Toc185185425)  [**3.2 Инфологическое проектирование** 68](#_Toc185185426)  [**3.3** **Проектирование интерфейса** 79](#_Toc185185427)  [**3.4 Программная реализация** 82](#_Toc185185428)  [**3.4.1 Выбор используемых технологий** 82](#_Toc185185429)  [**3.4.2** **Скриншоты экранов приложения** 83](#_Toc185185430)  [**3.4.3 Структура разработанного проекта** 88](#_Toc185185431)  [**3.4.4 Листинги** 89](#_Toc185185432)  [**Заключение** 96](#_Toc185185433)  [**Список используемых источников:** 99](#_Toc185185434) |

# **ВВЕДЕНИЕ**

Современный рынок услуг по уходу за домашними животными демонстрирует устойчивый рост, что связано с повышением уровня жизни и растущей заботой владельцев о своих питомцах. В условиях конкурентной борьбы за клиента, наличие удобной и эффективной системы записи является важным аспектом для повышения качества обслуживания и привлечения новых клиентов. Многие груминг-салоны, включая "Домовенок", не имеющие собственной информационной системы для записи на услуги, сталкиваются с проблемами, связанными с управлением записями, что может приводить к неэффективному использованию рабочего времени, путанице в графиках и, как следствие, снижению уровня удовлетворенности клиентов. Более того, отсутствие собственной веб-платформы ограничивает возможности взаимодействия с клиентами, что негативно сказывается на репутации и финансовых показателях заведения. Внедрение информационной системы для записи позволит автоматизировать процессы, минимизировать вероятность ошибок и упростить управление расписанием. Это также обеспечит доступность информации об услугах, мастерах, отзывах, истории и наградах салона, что будет способствовать увеличению числа клиентов и укреплению позиций салона на рынке.

**Объектом исследования** является создание сервиса для записи в груминг-салон

**Целью** данной выпускной работы является создание веб-платформы для груминг-салона «Домовенок», которая бы позволила быстрее и удобнее пользователям записывать своих питомцев в груминг-салон, сократить временные затраты администратора салона на регулирование расписания, предоставить клиентам подробную информацию о салоне и его мастерах, реализовать возможность оставлять и просматривать отзывы об услугах, тем самым повысить число довольных услугами клиентов и, в последствии, увеличить прибыль груминг-салона.

**Задачи** выпускной квалификационной работы, связанные с проектировкой и реализацией цифровой платформы:

1. **Проектирование и реализация самостоятельной записи в салон**

* Создание пользовательского интерфейса для записи в груминг-салон: разработка удобного для пользователя интерфейса для самостоятельной записи в груминг-салон.
* Создание базы данных с информацией о клиентах и их записях, мастерах, отзывах и пользователях веб-платформы: создание базы данных с информацией о клиентах, их записях, о мастерах, отзывах, и пользователях веб-платформы.
* Создание функция отображения клиентам только доступных для записи таймслотов: реализация функции для отображения пользователю при самостоятельной записи только доступные дату и время мастера.

1. **Проектирование и реализация отправки заявки на обратный звонок**

* Создание интерфейса для отправки заявки на обратный звонок: разработка удобного для пользователей интерфейса для отправки заявки на обратный звонок администратора без авторизации в системе.
* Создание функционала по управлению заявками на обратный звонок: реализация функционала по отображению информации администратору о поступлении заявки от клиента на обратный звонок и с контактными данными клиента для последующей связи менеджера с клиентом.

1. **Проектирование и реализация публикации отзывов**

* Создание пользовательского интерфейса для публикации отзывов о салоне: разработка удобного для пользователя интерфейса для размещения и просмотра отзывов о салоне.
* Предоставление администратору прав для модерации отзывов: реализация функционала по регулированию опубликованных на платформе отзывов администратором.
* Ограничение прав доступа на отправку отзывов: предоставление возможности оставлять отзывы только авторизованным пользователям.

1. **Проектирование и реализация личного кабинета для администратора**

* Создание личного кабинета администратора web-платформы: разработка личного кабинета для администратора, с удобным интерфейсом, с отображением информации из базы данных (пользователи, мастера, клиенты, записи, отзывы, заявки на обратный звонок) и возможностью ее редактирования.
* Реализация функции по сбросу пароля администратору: добавление администратору возможности сброса пароля пользователей платформы.

1. **Проектирование и реализация личного кабинета пользователя.**

* Создание пользовательского интерфейса для авторизации/регистрации в системе: разработка удобного, интуитивного интерфейса для регистрации/входа в систему.
* Реализация базы данных для хранения информации о пользователях: создание базы данных с информацией о пользователях в т.ч. их данных для доступа (логин, пароль), личных данных.
* Реализация функционала для регистрации пользователя в системе: создание функционала для создания пользователем своего аккаунта.
* Реализация функционала для авторизации пользователя в системе: создание функционала для входа пользователя в личный кабинет по эл. почте и паролю
* Добавление возможности пользователю просматривать и редактировать свои данные в личном кабинете: реализация функционала для корректировки пользователем личных данных в системе либо удаления личного кабинета.
* Отображение в личном кабинете информации о записях, отзывах пользователя: реализация раздела в личном кабинете, в котором будет отображаться информация о личных данных пользователя, его записях и оставленных отзывах, а также об актуальных статусах записей (подтвержден/не подтвержден/ожидает подтверждения/отменен).
* Предоставление пользователю возможности отменять запись: реализация функционала по удалению записи пользователем в личном кабинете.

1. **Проектирование и реализация разных разделов на веб-платформе.**

* Разработка разделов с услугами для питомцев: реализация отображения данных об услугах салона в разделах «Кошки»/ «Собаки» с подразделами для разных типов (мелкие/средние/крупные -для собак и короткошерстные/средне шёрстные/длинношерстные – для кошек) и пород животных для удобной навигации и поиска информации для всех пользователей (в т.ч. для неавторизованных).
* Разработка раздела «Мастера»: реализация отображения информации о мастерах, их опыте, специализации и образовании в разделе «Мастера».
* Разработка раздела «Отзывы»: реализация отображения отзывов о салоне в разделе «Отзывы».
* Разработка раздела «О нас»: реализация описания истории салона, его наградах и лицензиях в разделе «О нас».
* Разработка раздела «Контакты»: реализация отображения контактных данных салона (телефон, эл. почта, адреса).

**Назначение системы.** Решением проблемы отсутствия собственной информационной системы для записи клиентов в груминг-салон «Домовенок» может стать создание Web-платформы, позволяющей клиентам легко и удобно записывать своих питомцев на подходящие дату и время, а администратору более эффективно регулировать расписание салона. Новая web-платформа позволит увеличить количество довольных сервисом клиентов и, соответственно, увеличит прибыль груминг-салона.

**Инструменты:** Python, Django, MySQL, PyCharm, Python Virtualenv, Git, HTML, CSS, Java Script.

Состав команды: Пьянкова Дарья Максимовна (Backend-, frontend-разработчик, аналитик)

**1.ХАРАКТЕРИСТИКА ПРОБЛЕМНОЙ СИТУАЦИИ И ПОСТАНОВКА ЗАДАЧ**

**1.1 Характеристика проблемной области**

В современном мире, где услуги по уходу за домашними животными становятся все более популярными, создание удобных и эффективных решений для записи на такие услуги является важной задачей. Груминг-салон "Домовенок" сталкивается с необходимостью оптимизации процесса записи клиентов. Отсутствие собственной платформы для записи приводит к неэффективному управлению временем, увеличению нагрузки на сотрудников и снижению уровня обслуживания клиентов. В связи с этим проектирование и разработка информационной системы для записи в груминг-салон "Домовенок" представляется актуальным и востребованным решением.

**Проблемы, существующие в груминг-салоне "Домовенок":**

* **Отсутствие автоматизации процессов:**

- В настоящее время в "Домовенке" отсутствует информационная система, что приводит к ручному ведению записей клиентов. Это увеличивает вероятность ошибок, задержек в обслуживании и, как следствие, неудовлетворенность клиентов.

* **Сложности в управлении расписанием:**

- Ручной процесс записи клиентов не только занимает много времени, но и затрудняет управление расписанием. Это может привести к перекрытиям записей или невыполненным записям, что негативно сказывается на репутации салона.

* **Необходимость повышения качества обслуживания:**

- Конкуренция в сфере услуг по уходу за животными возрастает. Для того чтобы выделиться, "Домовенку" необходимо обеспечить высокий уровень обслуживания и удобство для клиентов, что невозможно без внедрения технологий.

**1.2 Аналитический обзор решений**

Одним из решений на рынке является сайт https://paradogs.ru/product/strizhka-sobak/ для записи в груминг-салон «Paradogs».

На рисунке 1 изображена часть интерфейса веб-сайта «Paradogs»

**Paradogs**

Сайт: https://paradogs.ru/

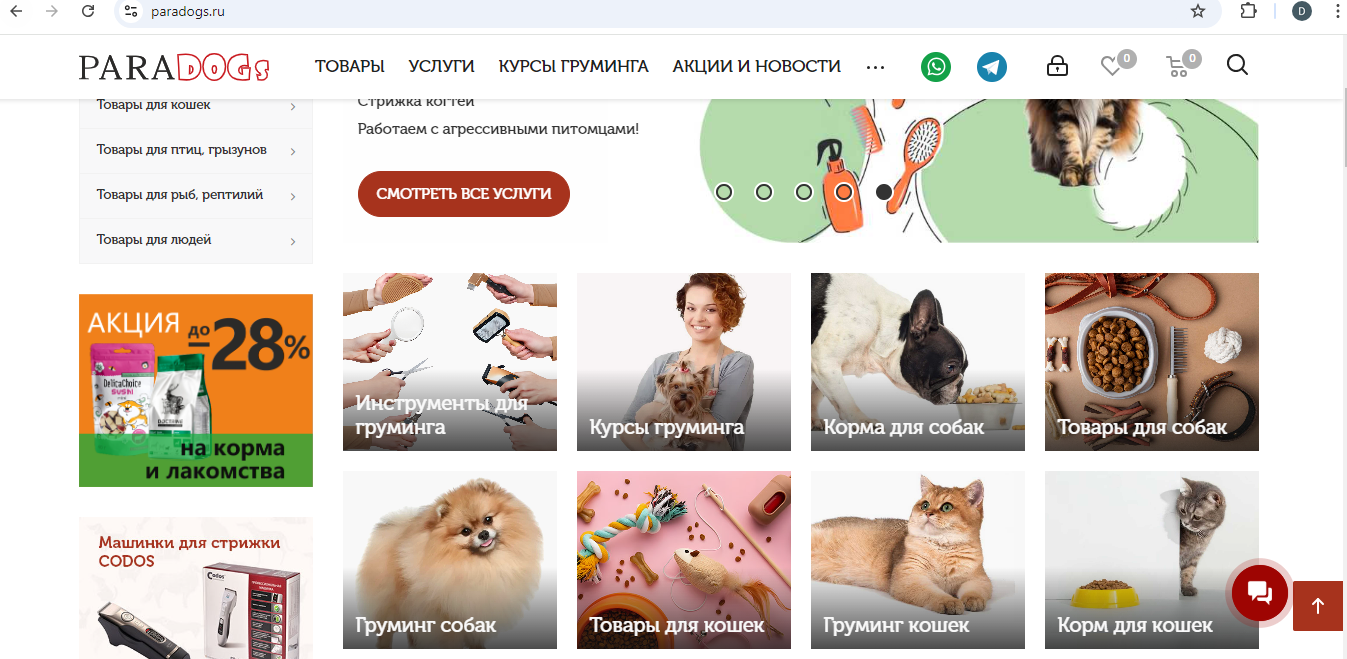


Рисунок 1 – Часть интерфейса web-сайта «Paradogs»

Основное назначение сайта paradogs.ru – предоставление информации о салоне, его товарах и услугах, также на нем есть возможность записи в салон или приобретения товаров для животных онлайн. Если говорить о возможностях записи в салон, то их несколько:

* Отправка заявки на обратный звонок администратору
* Связь с менеджером через контактный телефон и мессенджеры (what’s app, telegram)

В данном сервисе отправка заявки на обратный звонок позволяет клиентам быстро и без регистрации заполнить и отправить заявку в салон и ожидать обратного звонка от администратора, не тратя время на ожидание его ответа на линии. Такой способ записи является довольно распространенным, несложным для реализации и экономит время клиента на ожидание ответа оператора.

Связь с менеджером через контактный телефон или мессенджеры дает возможность клиенту лично позвонить или написать в салон и оставить заявку на запись. Такой метод записи является традиционным и подходит для пользователей, не привыкших взаимодействовать с компьютером и самостоятельно записываться на услуги, а предпочитают лично общаться с менеджером и договариваться о записи. Также этот метод позволяет менеджеру установить личный контакт с клиентом, услышать и принять все его пожелания к записи.

Однако, стоит отметить, что эти методы имеют свои ограничения. Отправка заявки на обратный звонок является не самым быстрым способом записи, клиенту приходится ожидать обратного звонка, также этот способ требует времени у администратора на осуществление ручной записи клиента в салон. Более того, клиенту может не всегда быть удобно ответить на обратный звонок менеджера, поэтому контакт может не состояться. Запись через чат в мессенджере или соцсети может быть также затратной по времени как для клиента, которому приходится ожидать ответа администратора салона в чате, так и для менеджера, которому надо вручную обрабатывать все заявки на запись и регулировать расписание. Более того, клиенту необходимо дождаться рабочего времени салона для получения ответа от менеджера в мессенджере, что также не очень удобно.

В целом, информационная система для записи в груминг-салон «Paradogs» предоставляет пользователям возможность совершать запись в груминг-салон распространёнными и достаточно удобными способами для клиентов. Однако, в контексте развития новых технологий и возможностей, будущее решение для записи в салон может быть более автоматизированным, удобным и быстрым как для клиентов салона, так и для менеджеров.

Еще одним решением на рынке является сайт <https://zapovednik96.ru/grooming/> для записи в салон «Заповедник».

На рисунках 2 и 3 изображены части интерфейса веб-сайта «Заповедник»

**Заповедник**

Сайт: <https://zapovednik96.ru/grooming/>

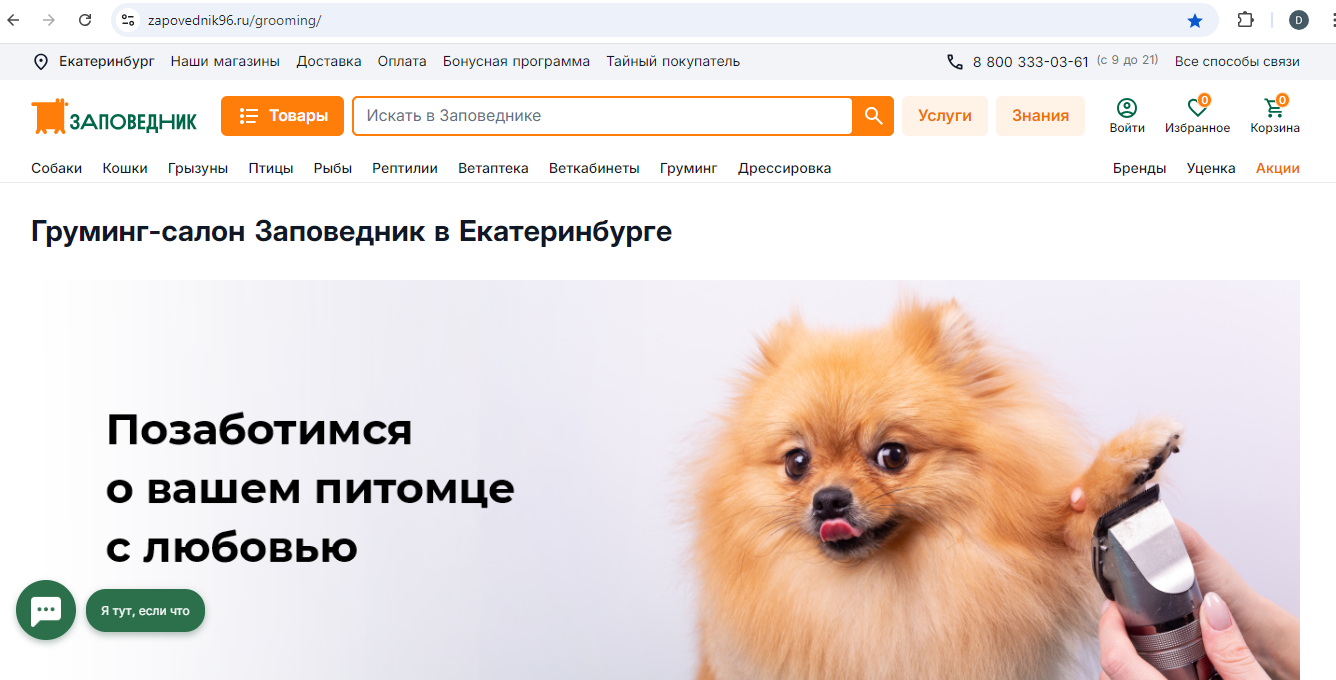


Рисунок 2 – Часть интерфейса web-сайта «Заповедник»

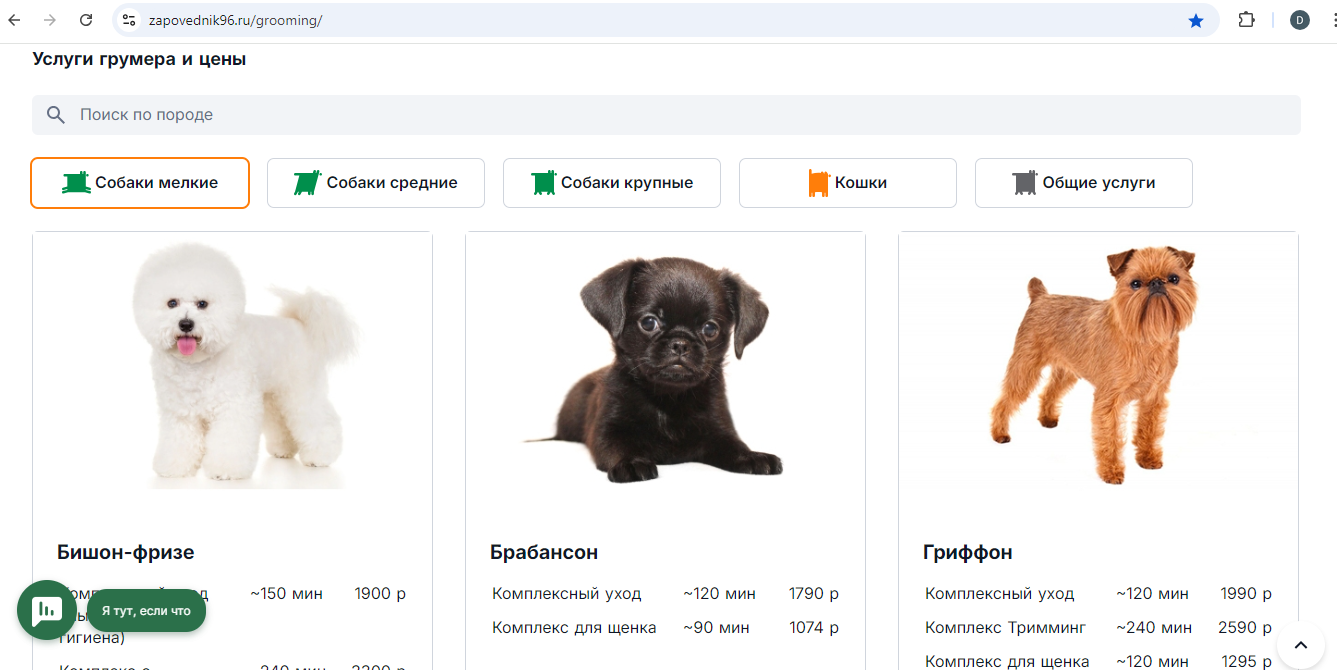


Рисунок 3 – Часть интерфейса web-сайта «Заповедник»

Основное назначение сайта zapovednik96.ru – предоставление пользователю возможности ознакомиться с товарами и услугами для животных компании «Заповедник», также на сайте есть возможность заказать товары и записаться на услуги (груминг, дрессировка). Функционал по записи в салон представлен в нескольких вариантах:

* Пользователь может позвонить в салон
* Пользователь может записаться онлайн на сайте самостоятельно

Звонок в салон позволяет пользователю в удобное время связаться лично с менеджером и оставить заявку на запись с конкретными пожеланиями к записи.

Запись онлайн – удобный способ, позволяющий клиентам в подходящее для них время самостоятельно произвести онлайн - запись в салон, выбрав нужное время оказания услуги и адрес.

Но данные способы записи имеют свои недочеты. Звонок в салон - не самый удобный метод, так как пользователю не всегда комфортно звонить лично в салон, а также это требует много времени, особенно если на линии оператора очередь. Также, совершить звонок в салон можно только в рабочее время салона, что может быть не подходящим для клиента. От сотрудника также требуется много времени и усилий, чтобы совершить запись вручную по звонку клиента. Запись онлайн на web-платформе <https://zapovednik96.ru/grooming/> не предоставляет клиенту возможности выбора подходящего мастера салона и оставления комментария к записи, если клиенту потребуется, что также может быть очень неудобно для пользователей.

В целом, web-платформа <https://zapovednik96.ru/grooming/> для записи клиентов в груминг-салон «Заповедник» дает пользователям возможность совершить запись по звонку в салон или самостоятельно онлайн, но функционал для самостоятельной записи в салон может быть расширен с целью предоставления пользователю более комфортного и качественного сервиса.

Другим примером решения является web-платформа для записи в салон Lasti Grooming.

На рисунке 4 изображена часть интерфейса веб-сайта «Lasti Grooming»

**Lasti Grooming**

Сайт: https://lastigrum.ru/

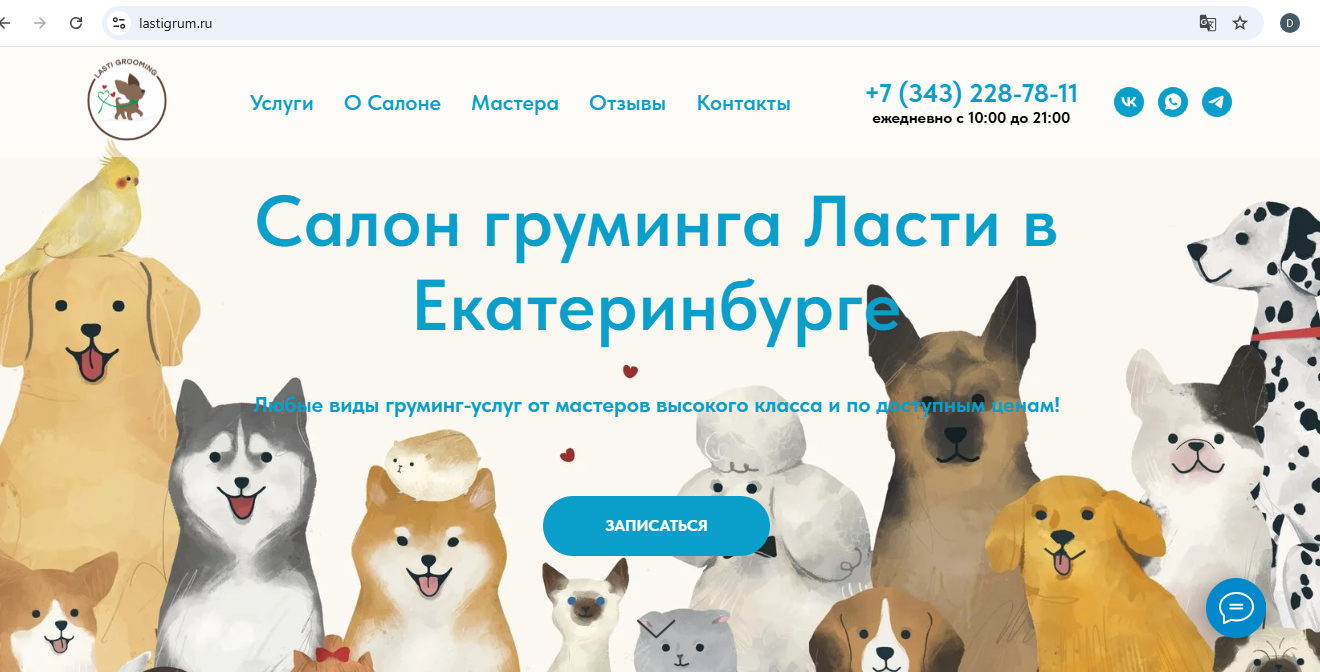


Рисунок 4 – Часть интерфейса web-сайта «Lasti grooming»

Основное назначение web-платформы – предоставление информации клиентам о салоне и его услугах, а также возможности записи на услуги. Из способов записи в данном салоне имеются следующие:

* Связаться с менеджером по контактному телефону или в мессенджере (vk, telegram, what’s app)
* Записаться онлайн самостоятельно (через интегрированную CRM – платформу «ПрофСалон»)

Запись через звонок менеджеру или сообщение в мессенджерах, как уже ранее упоминалось, дает возможность клиентам в удобное для них время написать/позвонить в салон и записаться, а менеджеру установить связь с будущим клиентом и учесть его пожелания по записи.

Интеграция CRM-платформы в сайт груминг-салона позволяет клиенту быстро, удобно и эффективно совершать запись, выбрав услугу, дату, время и мастера, а салону автоматизировать процесс записи клиентов на услуги, что снижает вероятность ошибок и экономит время как для сотрудников, так и для клиентов.

Однако, такие способы записи имеют свои минусы. Звонок по телефону - не самый быстрый и удобный метод для клиентов, так как ограничивается режимом работы салона, также требует трудозатрат со стороны администратора для того, чтобы записать клиента по звонку/сообщению в салон вручную. Интеграция CRM-системы может быть технически сложным процессом, требовать времени и ресурсов. Более того, такой метод может потребовать дополнительных финансовых вложений как на подписку, так и на обслуживание, что может быть критичным для небольших салонов таких как «Домовенок». Также интеграция CRM-приложения делает салон зависимым от внешних технологий, и, в случае технических сбоев или проблем с интернетом, доступ к системе может быть ограничен, что негативно скажется на работе салона.

В целом, описанные методы записи клиентов на веб-сайте «lastigroom.ru» позволяют клиентам легко и удобно записываться в салон. Однако, метод записи по звонку или сообщению клиента может быть неудобным как для клиента, так и для администратора, а интеграция CRM-системы может иметь массу недостатков для небольших салонов, таких как «Домовенок», из-за финансовых и временных затрат на процесс интеграции и сопровождения CRM-приложения.

**1.3 Постановка задач, решаемых в проекте**

На основе проблем и стратегии развития сервиса были сформированы цель и задачи для нового сервиса, который будет включать в себя возможность клиентам быстро и удобно записываться в груминг-салон:

**Цель** – создание универсальной веб платформы для удобной и самостоятельной записи клиентов в груминг-салон, а также интегрированного личного кабинета администратора для управления записями, получения заявок на обратный звонок от клиентов и модерации отзывов.

**Задачи** в рамках написания выпускной квалификационной работы:

1. Сформировать концепт будущей системы.
2. Провести функциональное моделирование с применением методологии IDEF0 и диаграммы вариантов использования USE CASE UML.
3. Сформировать требования к системе по классификации FURPS + [1].
4. Подобрать наиболее подходящую под концепт системы архитектуру и архитектурный паттерн.
5. Спроектировать и реализовать базу данных.
6. Спроектировать и реализовать интерфейс системы.
7. Реализовать web платформу и ее функционал.

**Задачи** в рамках реализации проекта в целом:

* Добавление возможности самостоятельной записи в груминг-салон: у пользователя должна быть возможность самостоятельно производить запись в салон в удобное для него время, к нужному мастеру и в нужный филиал, а также с возможностью ввода комментария к записи.
* Реализация функционала для «Быстрой записи» в виде отправки пользователем заявки на обратный звонок администратора без необходимости авторизации пользователя.
* Добавление функционала обратной связи об указанных услугах: В новой системе должна быть возможность авторизованным пользователям оставлять отзывы об указанных услугах для возможности улучшения качества оказываемых услуг салона, а также обмена опытом и мнениями о салоне его клиентов.
* Удобная навигация на сайте: не менее важным является ориентированный на пользователя интуитивный интерфейс информационной системы. Клиенту должно быть удобно и легко пользоваться сайтом и находить нужную для себя информацию на нем.
* Предоставление детальной информации об указываемых услугах, мастерах и салоне в целом: необходимо детально описать услуги, предоставляемые салоном, опыт работы и образование мастеров и предоставить основную информацию о салоне (историю основания, награды, лицензии, адреса филиалов, контактную информацию).
* Реализация функции регистрации и авторизации пользователей в системе: создание личного кабинета для веб-сайта поможет клиентам сэкономить время на заполнение контактных данных при записи (так как они будут автоматически заполнены данными из личного кабинета), предоставит возможность оставлять авторизованные отзывы, что уменьшит количество спама, клиенты смогут видеть историю своих записей, что поможет им отслеживать посещения и планировать будущие визиты, улучшит клиентский опыт, так как наличие личного кабинета повышает уровень сервиса и удовлетворенности клиентов.

**Выводы по 1 главе**

В первой главе была проведена характеристика проблемной ситуации, связанной с отсутствием у груминг салона "Домовенок" собственной веб-платформы для записи клиентов. В условиях современного рынка услуг, где онлайн-сервисы становятся неотъемлемой частью клиентского опыта, а спрос на груминг-услуги и конкуренция на рынке груминг-салонов значительно выросли, отсутствие такой платформы ограничивает возможности салона по привлечению новых клиентов и улучшению качества обслуживания.

Анализ текущих методов записи клиентов показал их недостатки, такие как высокая вероятность ошибок, недостаточная информативность и неэффективность в управлении расписанием. Это создаёт не только неудобства для клиентов, но и приводит к потерям для бизнеса. В связи с этим была поставлена задача проектирования и разработки информационной системы, которая обеспечит удобный и эффективный процесс записи, управления клиентской базой и предоставления актуальной информации о предоставляемых услугах.

Таким образом, данная глава обосновала необходимость создания веб-платформы для салона "Домовенок", что позволит значительно улучшить качество обслуживания, повысить уровень клиентской удовлетворенности и, в конечном итоге, увеличить прибыльность бизнеса. В следующих главах будет подробно рассмотрен процесс проектирования и реализации предложенной информационной системы, а также ее интеграция в бизнес-процессы салона.

**2. СИСТЕМНЫЙ АНАЛИЗ ОБЪЕКТА УПРАВЛЕНИЯ**

**2.1 Концептуальное моделирование**

Перед тем, как приступить к созданию веб-платформы для записи в груминг-салон, необходимо сформировать понятия о событиях, фактах и предметах, с которыми будет оперировать разрабатываемая система. Для того чтобы привести эти понятия к определенной модели данных, необходимо заменить их соответствующими информационными представлениями. В данном случае, наиболее удобным инструментом для такого представления данных является модель "сущность-связь" (Entity-Relationship Model, ER-модель), которая предназначена для логического представления данных, независимо от используемого программного обеспечения. Эта модель позволяет описать структуру и взаимосвязи сущностей, таких как клиенты, мастера, отзывы, записи и другие в системе.

Была создана концептуальная ER-диаграмма Питера Чена[2], содержащая в себе сущности Client, User, Review, Master, MasterSchedule, CatBreedService, DogBreedService, CatService, DogService, CatBreed, DogBreed, Appointment, Salon, CallbackRequest. Изображение ER-диаграммы Питера Чена представлены на Рисунке 19. Сущности по отдельности и отношения между ними представлены на рисунках 5-18.

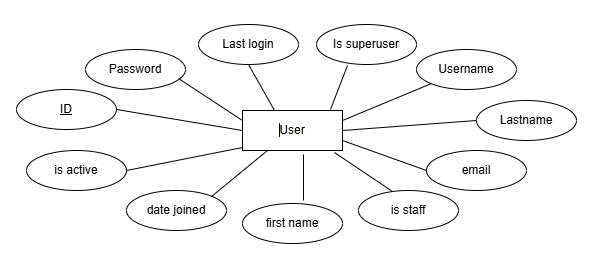


Рисунок 5 – Сущность «User»

На рисунке 5 представлена сущность «User», которая представляет собой пользователя системы, обладающего правами в зависимости от уровня прав доступа. User имеет атрибуты:

* ID: уникальный идентификатор пользователя
* Password: пароль для входа в систему
* Last login: дата и время последней авторизации в системе
* Is superuser: флаг наличия прав администратора системы
* Username: имя пользователя в системе
* Lastname: фамилия пользователя
* Email: эл. Адрес
* Is staff: флаг сотрудника салона
* First name: имя пользователя
* Date joined: дата регистрации в системе
* Is active: флаг нахождения пользователя в системе

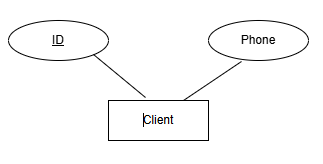


Рисунок 6 – Сущность «Client»

На рисунке 6 представлена сущность «Client», которая представляет собой клиента салона, обладающего правами регистрации и авторизации в системе, записи на услуги, отмены записи, отправки заявки на обратный звонок, написания и просмотра отзывов, редактирования своих данных, удаления аккаунта, просмотра информации о салоне, его мастерах и услугах. Сущность Client имеет следующие атрибуты:

* ClientID: уникальный идентификатор клиента
* Phone: контактный номер телефона клиента

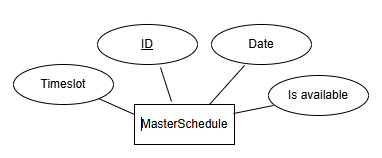


Рисунок 7 – Сущность «MasterSchedule»

На рисунке 7 представлена сущность «Masterschedule», которая является расписанием работы определенного мастера. «MasterSchedule» имеет атрибуты:

* ID: Уникальный идентификатор расписания мастера
* Timeslot: время, в которое работает мастер
* Date: дата, когда работает мастер
* Is available: флаг доступности мастера для записи

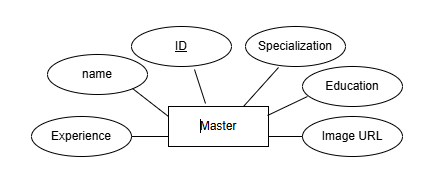


Рисунок 8 – Сущность «Master»

На рисунке 8 представлена сущность «Master», которая представляет собой сотрудника салона-специалиста по грумингу. Ей принадлежат атрибуты:

* MasterID: уникальный идентификатор мастера
* Education: образование мастера
* Name: имя мастера
* Specialization: специализация мастера
* Experience: опыт работы мастера
* Image URL: URL-ссылка на фото мастера

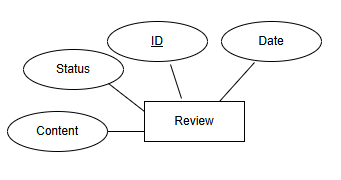


Рисунок 9 – Сущность «Review»

На рисунке 9 представлена сущность «Review», являющаяся отзывом, который оставляют клиенты-авторизованные пользователи на сайте. Ей принадлежат следующие атрибуты:

* ID: уникальный идентификатор отзыва
* Date: дата оставления отзыва
* Content: текст отзыва
* Status: статус отзыва (approved-подтвержден, disapproved-не подтвержден). На сайте отображаются только отзывы со статусом approved, статус выставляется пользователем с правами администратора.

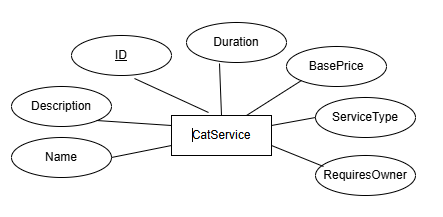


Рисунок 10 – Сущность «CatService»

На рисунке 10 представлена сущность «CatService» – услуги для кошек, предоставляемые салоном. Она состоит из атрибутов:

* ID: уникальный идентификатор услуги
* Duration: длительность услуги
* BasePrice: цена услуги
* Description: описание услуги
* Name: наименование услуги
* ServiceType: тип сервиса
* RequiresOwner: флаг необходимости присутствия хозяина питомца во время выполнения услуги

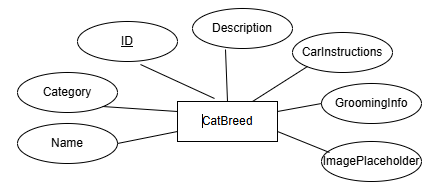


Рисунок 11 – Сущность «CatBreed»

На рисунке 11 представлена сущность Сущность «CatBreed» – породы кошек. Состоит из атрибутов:

* ID: уникальный идентификатор породы
* Name: наименование породы
* Category: категория породы (короткошерстные, средне шёрстные, длинношерстные)
* Description: описание породы
* CaringInstructions: рекомендации по уходу
* GroomingInfo: информация по грумингу
* ImagePlaceholder: изображение-заполнитель

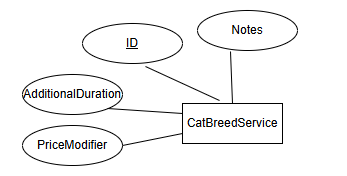


Рисунок 12 – Сущность «CatBreedService»

На рисунке 12 представлена сущность «CatBreedService», которая является услугами салона для определенной породы кошек. Она имеет атрибуты:

* ID – уникальный идентификатор сервиса для определенной породы кошек
* PriceModifier – цена услуги для определенной породы кошек
* Additional Duration – дополнительное время на выполнение услуги для определенной породы
* Notes - пометки

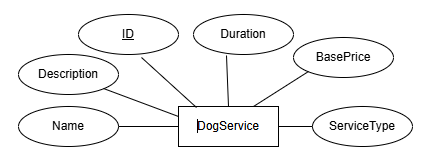


Рисунок 13 – Сущность «DogService»

На рисунке 13 представлена сущность «DogService» – услуги для собак. Состоит из атрибутов:

* ID: уникальный идентификатор услуги для собак
* Name: наименование услуги
* Description: описание услуги
* Duration: длительность услуги
* BasePrice: цена услуги
* Service Type: тип услуги

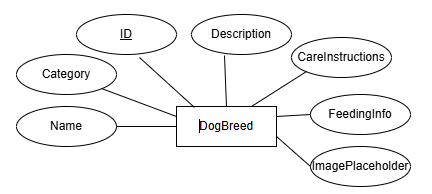


Рисунок 14 – Сущность «DogBreed»

На рисунке 14 представлена «DogBreed» - сущность породы собак. Состоит из атрибутов:

* ID: уникальный идентификатор породы
* Name: наименование породы
* Category: категория проды (мелкие, средние, крупные)
* Description: описание породы
* CareInstructions: рекомендации по уходу
* FeedingInfo: рекомендации по кормлению
* ImagePlaceholder: изображение-заполнитель

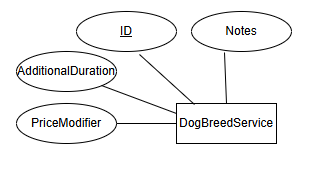


Рисунок 15 – Сущность «DogBreedService»

На рисунке 15 представлена сущность «DogBreedService» - услуги для определённой породы собак. Имеет атрибуты:

* ID: уникальный идентификатор услуги для определеной породы собак
* PriceModifier: коэффициент изменения стоимости услуги в зависимости от породы собаки
* Additional duration: дополнительная длительность услуги в зависимости от породы собаки
* Notes: заметки

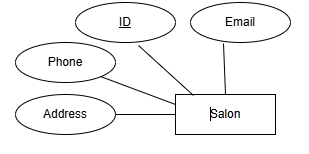


Рисунок 16 – Сущность «Salon»

На рисунке 16 представлена сущность «Salon» – филиал груминг-салона. У нее есть несколько атрибутов:

* ID: уникальный идентификатор салона
* Address: адрес салона
* Phone: контактный номер телефона салона
* Email: эл. Адрес салона

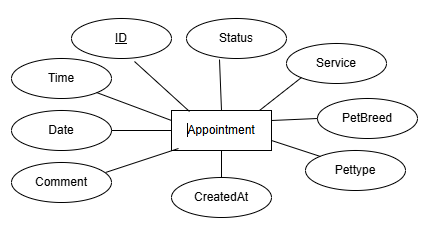


Рисунок 17 – Сущность «Appointment»

На рисунке 17 представлена сущность «Appointment» - запись в салон. Она состоит из:

* ID: уникальный идентификатор записи
* Date: дата, на которую записан клиент
* Time: время записи
* Status: статус записи (подтверждена/отменена/ ожидает подтверждения)
* Service: услуга, на которую записан клиент
* PetBreed: порода питомца
* PetType: тип питомца (собака/кошка)
* CreatedAt: дата и время создания записи
* Comment: комментарий к записи



Рисунок 18 – Сущность «CallbackRequest»

На рисунке 18 представлена сущность «CallbackRequest» - заявка на обратный звонок. Она состоит из:

* ID: уникальный идентификатор заявки
* Created at: дата и время создания
* Status: статус заявки (ожидает/обработан/отменен)
* Name: имя клиента
* Phone: контактный телефон клиента
* PetBreed: порода питомца

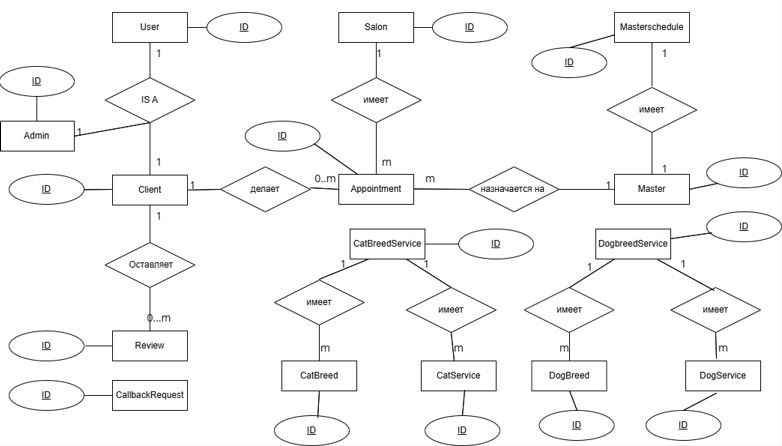


Рисунок 19 – Концептуальная ER-диаграмма Питера Чена

Рисунок 19 представляет концептуальную ER-диаграмму, разработанную с использованием модели Питера Чена. Эта диаграмма представляет все сущности, их первичные ключи и отражает отношения между ними.

**2.1.1 Модель вариантов использования (прецедентов).**

Разрабатываемое приложение для записи в груминг-салон предоставит возможность пользователю удобно и быстро подобрать услугу и записаться на нее, а админисратору легко управлять записями. Через диаграмму вариантов исользования UML Use Case [3] будут представлены подробные диаграммы функционала, доступного клиентам (роль «Пользователь») (Рисунок 20) и администраторам (роль «Администратор») (Рисунок 21), чтобы обеспечить понимание и удобство использования приложения.

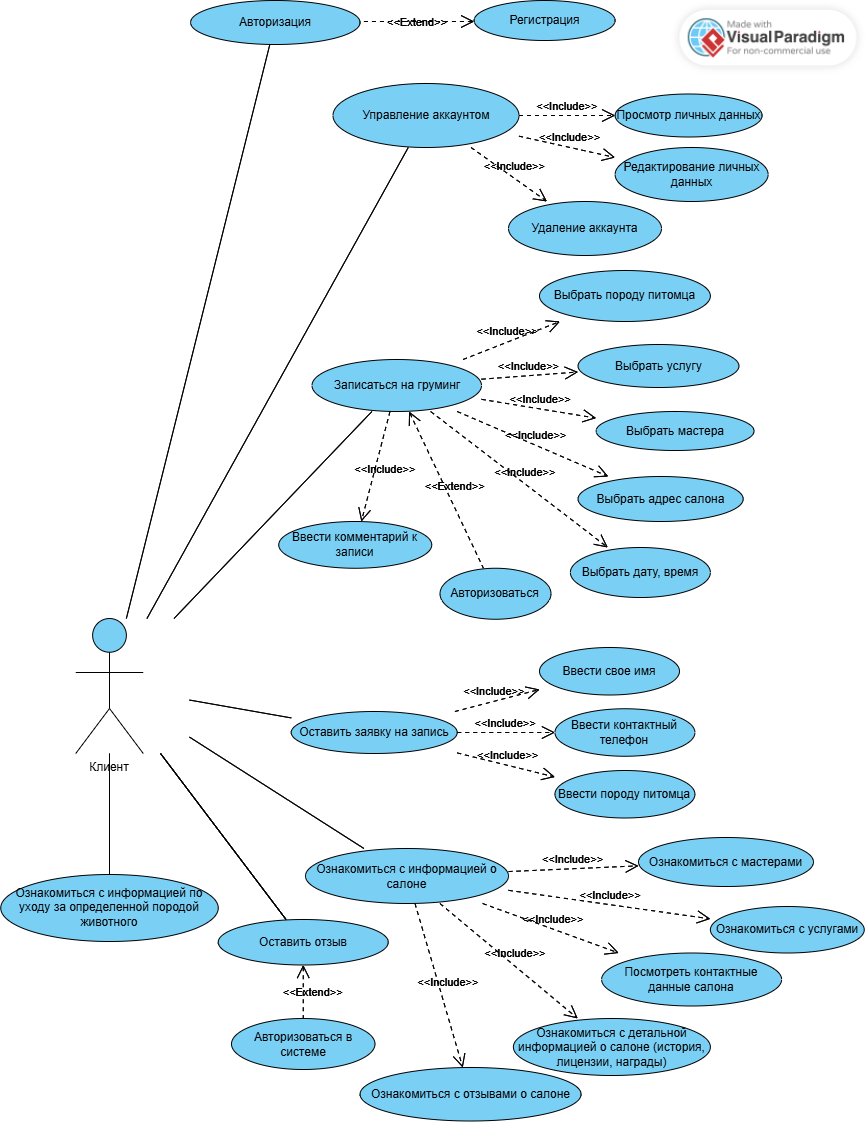
****

Рисунок 20 – UML Use Case диаграмма пользователя

Согласно диаграмме вариантов использования, для пользователя системы или клиента определены следующие прецеденты [4]: авторизация, просмотр персональных данных в личном кабинете, изменение данных пользователя в личном кабинете, запись на услугу, удаление аккаунта, отправка заявки пользователем на обратной звонок, оставить отзыв, ознакомиться с информацией обо всех услугах для собак, ознакомиться с информацией об услугах для определённой породы собак, ознакомиться с информацией о всех услугах для кошек, ознакомиться с информацией об услугах для определенного вида кошек, посмотреть контакты салона, посмотреть подробную информацию о салоне (историю, награды, лицензии), ознакомиться с информацией о мастерах, ознакомиться с отзывами о салоне, ознакомиться с информацией по уходу за определенной породой собак. Подробное описание данных сценариев представлено ниже в таблицах 1-15:

Таблица 1 – Авторизация

| **Основной сценарий** |
| --- |
| 1. Пользователь нажимает на кнопку «Войти» |
| 1. Система переводит пользователя на страницу авторизации |
| 1. Пользователь вводит учётные данные |
| 1. Пользователь нажимает на «Войти» |
| 1. Система переводит пользователя на главную страницу личного кабинета |
| **Расширение** |
| 1а. Пользователь не зарегистрирован в системе  .1: Пользователь нажимает на кнопку «Регистрация»  .2: Пользователь вводит свои данные  .3: Пользователь нажимает на «Зарегистрироваться»  .4: Система сохраняет данные пользователя  .5: Система переводит пользователя на страницу входа |

Окончание Таблицы 1

|  |
| --- |
| 3а. Пользователь ввел некорректные данные  .1: Поля ввода подсвечиваются.  .2: Система выводит сообщение “Неверный email или пароль” |
| 3б. Пользователь не зарегистрирован в системе  .1: Пользователь нажимает на кнопку «Регистрация»  .2: Пользователь вводит свои данные  .3: Пользователь нажимает на «Зарегистрироваться»  .4: Система сохраняет данные пользователя  .5: Система переводит пользователя на страницу входа |

Таблица 2 – Просмотр персональных данных в личном кабинете

|  |
| --- |
| **Основной сценарий** |
| 1. В личном кабинете пользователь нажимает на «Мой профиль» |
| 1. На экране отображаются личные данные пользователя |

Таблица 3 – Изменение данных пользователя в личном кабинете

| **Основной сценарий:** |
| --- |
| 1. В личном кабинете пользователь нажимает на кнопку «Мой профиль» |
| 1. Пользователь изменяет свои личные данные |
| 1. Пользователь нажимает на кнопку «Сохранить» |
| 1. Система сохраняет измененные данные |
| 1. Отображаются измененные данные |

Окончание Таблицы 3

|  |
| --- |
| **Расширенный сценарий:** |
| 3а. Пользователь не нажал на «Сохранить».  .1: Пользователь Закрывает страницу  .2: Данные не изменились |

Таблица 4 – Удаление аккаунта

| **Основной сценарий:** |
| --- |
| 1. Авторизованный пользователь нажимает на кнопку «Мой профиль» |
| 1. Пользователь нажимает на «Удалить аккаунт» |
| 1. Возникает уведомление: «Вы уверены, что хотите удалить аккаунт? Это действие нельзя отменить.» |
| 1. Пользователь нажимает «Подтвердить удаление» |
| 1. Система удаляет пользователя из базы |
| 1. Система переводит пользователя на главную страницу |
| 1. Система отображает уведомление «Ваш аккаунт успешно удален!» |
| **Расширенный сценарий:** |
| 4а. Пользователь нажал на «Отмена».  .1: Система не удаляет аккаунт |

Таблица 5 – Запись на услугу

| **Основной сценарий:** |
| --- |
| 1. Пользователь нажимает на «Записаться на груминг» |
| 1. Пользователь выбирает «Записаться онлайн» |
| 1. Пользователь заполняет данные для записи |
| 1. Пользователь нажимает на «Записаться» |
| 1. Система сохраняет данные о записи |
| 1. Система переводит пользователя на главный экран |
| 1. Система отображает уведомление «Запись успешно создана!» |
| **Расширение** |
| 2а. Пользователь не авторизован  .1: открывается страница авторизации  .2: пользователь вводит свои данные  .3: пользователь нажимает на «Войти»  .4: система переводит пользователя на главный экран личного кабинета  .5: пользователь нажимает на «Записаться на груминг»  .6: пользователь нажимает на «Записаться онлайн»  .7: пользователь заполняет данные о заявке  .8: пользователь нажимает на «Записаться»  .9: система сохраняет данные о записи  .10: система переводит пользователя на главный экран  .11: система отображает уведомление «Запись успешно создана!» |

Окончание таблицы 5

| 2б. Пользователь не зарегистрирован в системе  .1: Пользователь нажимает на кнопку «Регистрация»  .2: Пользователь вводит свои данные  .3: Пользователь нажимает на «Зарегистрироваться»  .4: Система сохраняет данные пользователя  .5: Система переводит пользователя на страницу входа  .6: Пользователь вводит свои учетные данные  .7: пользователь нажимает на «Войти»  .8: система переводит пользователя на главный экран личного кабинета  .5: пользователь нажимает на «Записаться на груминг»  .6: пользователь нажимает на «Записаться онлайн»  .7: пользователь заполняет данные о заявке  .8: пользователь нажимает на «Записаться»  .9: система сохраняет данные о записи  .10: система переводит пользователя на главный экран  .11: система отображает уведомление «Запись успешно создана!» |
| --- |
| 3а. Пользователь некорректно заполняет данные  .1: пользователь нажимает на «Записаться»  .2: Система выдает ошибку, что данные некорректно заполнены |
| 3б. Пользователь не заполнил все поля  .1: пользователь нажимает на «Записаться»  .2: Система выдает ошибку, что не все поля заполнены |

Таблица 6 – Отправка заявки пользователем на обратной звонок администратора

| **Основной сценарий:** |
| --- |
| 1. Пользователь нажимает на «Записаться на груминг»/ «Записаться» |
| 1. Пользователь заполняет форму для записи |

Окончание Таблицы 6

|  |
| --- |
| 1. Пользователь нажимает на «Оставить заявку» |

Таблица 7 – Оставить отзыв

| **Основной сценарий:** |
| --- |
| 1. Пользователь выбирает раздел меню «Отзывы» |
| 1. Пользователь вводит текст отзыва в поле «Комментарий» |
| 1. Пользователь нажимает на «Оставить отзыв» |
| 1. На экране появляется уведомление «Спасибо за Ваш отзыв» |
| **Расширение** |
| 1а. Пользователь не авторизован  .1: пользователь нажимает на «Войти»  .2: открывается страница авторизации  .3: пользователь вводит свои данные  .4: пользователь нажимает на «Войти»  .5: система переводит пользователя на главный экран личного кабинета  .6: пользователь выбирает раздел меню «Отзывы»  .7: пользователь вводит текст отзыва в поле «Комментарий»  .8: пользователь нажимает на «Оставить отзыв»  .9: на экране появляется уведомление «Спасибо за Ваш отзыв» |

Окончание таблицы 7

| 4а. Пользователь не авторизован  .1: открывается страница авторизации  .2: пользователь вводит свои данные  .3: пользователь нажимает на «Войти»  .4: система переводит пользователя на главный экран личного кабинета  .5: пользователь выбирает раздел меню «Отзывы»  .6: пользователь вводит текст отзыва в поле «Комментарий»  .7: пользователь нажимает на «Оставить отзыв»  .8: на экране появляется уведомление «Спасибо за Ваш отзыв» |
| --- |

Таблица 8 – Ознакомиться с информацией обо всех услугах для собак

| **Основной сценарий:** |
| --- |
| 1. Пользователь выбирает раздел «Собаки» |
| 1. Система открывает страницу раздела «Собаки» |
| 1. Пользователь пролистывает ниже, где представлены все услуги салона для собак |
| 1. Пользователь знакомится с информацией по всем услугам солона для собак |

Таблица 9– Ознакомиться с информацией об услугах для определённой породы собак

| **Основной сценарий:** |
| --- |
| 1. Пользователь выбирает раздел «Собаки» |
| 1. Система открывает страницу раздела «Собаки» |
| 1. Пользователь нажимает на нужный тип собаки: «Мелкие»/ «Средние»/ «Крупные» |

Окончание Таблицы 9

|  |
| --- |
| 1. Пользователь нажимает на нужную пароду собак |
| 1. Система отображает на экране все услуги для выбранной породы собак |

Таблица 10– Ознакомиться с информацией о всех услугах для кошек

| **Основной сценарий:** |
| --- |
| 1. Пользователь выбирает раздел «Кошки» |
| 1. Система открывает страницу раздела «Кошки» |
| 1. Пользователь пролистывает ниже до заголовка «Услуги» |
| 1. Система отображает на экране все услуги для кошек |

Таблица 11 – Ознакомиться с информацией об услугах для определенного вида кошек

| **Основной сценарий:** |
| --- |
| 1. Пользователь выбирает раздел «Кошки» |
| 1. Система открывает страницу раздела «Кошки» |
| 1. Пользователь нажимает на определенный вид кошек |
| 1. Система отображает на экране все услуги для выбранного вида кошек |

Таблица 12 – Посмотреть контакты салона

| **Основной сценарий:** |
| --- |
| 1. Пользователь нажимает на раздел «Контакты» |
| 1. Система открывает страницу раздела «Контакты» с контактными данными салона |

Таблица 13 – Посмотреть подробную информацию о салоне (историю, награды, лицензии)

| **Основной сценарий:** |
| --- |
| 1. Пользователь нажимает на раздел «О нас» |
| 1. Система открывает страницу раздела «О нас» с подробной информацией о салоне |

Таблица 14 – Ознакомиться с информацией о мастерах

| **Основной сценарий:** |
| --- |
| 1. Пользователь выбирает раздел «мастера» |
| 1. Система открывает раздел «Мастера», где представлен перечень мастеров с фото, специальностью и опытом работы |
| 1. Пользователь нажимает под определенным мастером на «Подробнее» |
| 1. Система открывает страницу с фото выбранного мастера, специализацией, опытом работы и уровнем образования |

Таблица 15 – ознакомиться с отзывами о салоне

| **Основной сценарий:** |
| --- |
| 1. Пользователь выбирает раздел «Отзывы» |
| 1. Система отображает отзывы о салоне |

Таблица 16 – Ознакомиться с информацией по уходу за определенной породой собак

| **Основной сценарий:** |
| --- |
| 1. Пользователь выбирает раздел «Собаки» |
| 1. Система открывает страницу раздела «Собаки» |
| 1. Пользователь нажимает на нужный тип собаки: «Мелкие»/ «Средние»/ «Крупные» |
| 1. Пользователь нажимает на нужную пароду собак |
| 1. Пользователь пролистывает ниже до заголовка «Об уходе за…» |
| 1. Пользователь читает рекомендации по уходу и кормлению для выбранной породы |

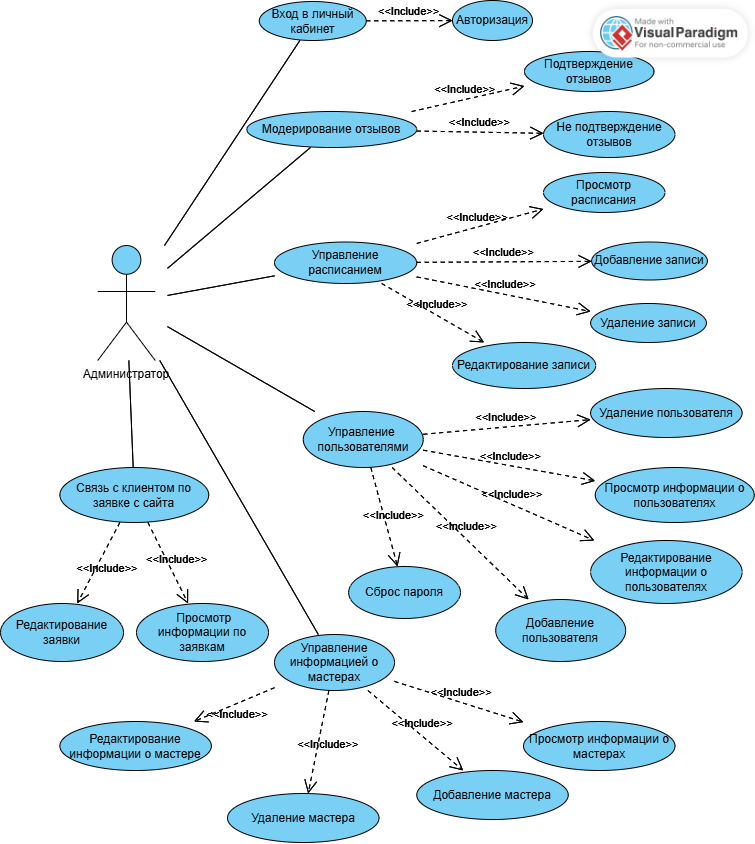


Рисунок 21 – UML Use Case диаграмма администратора

Прецеденты для администратора системы: вход в систему, модерирование отзывов, редактирование информации о мастере, удаление мастера, добавление мастера, просмотр информации о мастере, корректирование записи, добавление записи, удаление записи, просмотр записей, просмотр заявки на связь с клиентом, редактирование заявки на связь с клиентом, просмотр информации о пользователе, редактирование информации о пользователе, удаление пользователя, добавление пользователя, сброс пароля. Более детальное описание данных прецедентов в таблицах 17-33:

Таблица 17 – Вход в систему

| **Основной сценарий:** |
| --- |
| 1. Администратор вводит логин, пароль |
| 1. Администратор нажимает на «Войти» |
| 1. Администратор авторизован в системе |
| **Расширение** |
| 1а. Администратор вводит некорректные данные  .1: появляется уведомление об ошибке с учетными данными |
| 1б Администратор оставляет поле логин/пароль пустым |
| .1: появляется уведомление, что не все поля заполнены |

Таблица 18 – Модерирование отзывов

| **Основной сценарий:** |
| --- |
| 1. Администратор заходит в раздел «Reviews» |
| 1. Администратор нажимает на новый отзыв в статусе «pending» |
| 1. Администратор проверяет отзыв |
| 1. Администратор меняет статус отзыва на «approved» |
| 1. Администратор нажимает на «save» |
| 1. Система переводит администратора на страницу с отзывами |

|  |
| --- |
| 1. Система отображает новый отзыв в разделе «Отзывы» |
| 1. Система отображает уведомление, что статус отзыва изменен |
| 1. Система отображает новый отзыв в разделе «Отзывы» |
| **Расширение** |
| 4а. Отзыв содержит нецензурные слова/спам:  .1: Администратор меняет статус отзыва на «disapproved» |
| .2: администратор нажимает на «save» |
| .3: система переводит администратора на страницу с отзывами |
| .4: система отображает уведомление, что статус отзыва изменен |
| .3: отзыв не отображается в разделе «Отзывы» |
| **Расширение** |
| 5а. Администратор не нажимает на «Save»:  .1: администратор закрывает текущую страницу  .2: изменения не сохраняются |

Окончание Таблицы 18

Таблица 19 – Редактирование информации о мастере

| **Основной сценарий:** |
| --- |
| 1. Администратор заходит в портале администратора в раздел «Masters» |
| 1. Система отображает список мастеров |
| 1. Администратор нажимает на определенного мастера |

Окончание Таблицы 19

|  |
| --- |
| 1. Система отображает информацию о мастере |
| 1. Администратор корректирует информацию о мастере |
| 1. Администратор нажимает на «Save» |
| 1. Система сохраняет новые данные о мастере |
| 1. Система переводит администратора на страницу с мастерами |
| 1. Система отображает уведомление, что данные о мастере изменены |
| 1. На платформе груминг-салона отображается обновлённая информация о мастере |
| **Расширенный сценарий** |
| 6а. Администратор не нажал на «save»  .1: Администратор перешел на другую страницу  .2: Новые данные о мастере не изменились |

Таблица 20 – удаление мастера

| **Основной сценарий:** | |
| --- | --- |
| 1. Администратор заходит в раздел «Masters» | |
| 1. Администратор нажимает на нужного мастера | |
| 1. Администратор нажимает на «Delete» | |
| 1. Появляется вопрос, действительно ли администратор хочет удалить мастера | |
| 1. Администратор нажимает «yes, I’m sure» | |

Окончание Таблицы 20

|  |
| --- |
| 1. Система удаляет мастера |
| 1. Система переводит администратора на страницу с мастерами |
| 1. На экране появляется уведомление, что мастер удален |
| **Расширение** |
| 5а. Администратор нажал на «No, take me back»  .1: Система не удаляет мастера  .2: система возвращает администратора на страницу с мастерами |

Таблица 21 – добавление мастера

| **Основной сценарий:** |
| --- |
| 1. Администратор заходит в раздел «Masters» |
| 1. Администратор нажимает «Add master» |
| 1. Администратор заполнят информацию о новом мастере |
| 1. Администратор нажимает на «save» |
| 1. Система сохраняет данные о новом мастере |
| 1. Система переводит администратора на страницу с мастерами |
| 1. На экране появляется уведомление, что новый мастер создан |

Окончание Таблицы 21

|  |
| --- |
| **Расширенный сценарий** |
| 3а. Администратор не заполняет все обязательные поля  .1: администратор нажимает на «save»  .2: система выводит уведомление что не все поля заполнены |
| 4а. Администратор не нажимает на «Save»  .1: администратор переходит на другую страницу  .2: новый мастер не сохранен |

Таблица 22 – просмотр информации о мастере

|  |
| --- |
| **Основной сценарий** |
| 1. Администратор заходит в раздел «Masters» |
| 1. Администратор нажимает на нужного мастера |
| 1. Система отображает информацию о мастере |

Таблица 23 – Корректирование записи

| **Основной сценарий:** |
| --- |
| 1. Администратор заходит в раздел «Appointments» |
| 1. Администратор нажимает на запись, которую хочет откорректировать |
| 1. Администратор корректирует запись |
| 1. Администратор нажимает на «Save» |
| 1. Система сохранят корректировки записи |

Окончание таблицы 23

|  |
| --- |
| 1. Система переводит администратора на страницу с записями |
| 1. Система отображает уведомление, что запись скорректирована |
| **Расширенный сценарий** |
| 3а. Администратор не заполняет все обязательные поля в записи:  .1: Администратор нажимает на «Save»  .2: отображается уведомление что не все обязательные поля заполнены |
| 4а. Администратор не нажимает на «save»  .1: Администратор переходит на другую страницу  .2: изменения в записи не сохраняются |

Таблица 24 – Добавление записи

| **Основной сценарий:** |
| --- |
| 1. Администратор заходит в раздел «Appointments» |
| 1. Администратор нажимает «Add appointment» |
| 1. Администратор заполняет информацию о новой записи |
| 1. Администратор нажимает на «save» |
| 1. Система сохраняет данные о новой записи |
| 1. Система переводит администратора на страницу с записями |
| 1. На экране появляется уведомление, что новая запись создана |
| 1. Новая запись отображается на экране с записями |

Окончание таблицы 24

|  |
| --- |
| **Расширение** |
| 3а. Администратор не заполнил все обязательные поля  .1: администратор нажимает на «save»  .2: система выводит уведомление что не все поля заполнены |
| 4а. Администратор не нажимает на «Save»  .1: администратор переходит на другую страницу  .2: новая запись не сохранена |

Таблица 25 – удаление записи

| **Основной сценарий:** |
| --- |
| 1. Администратор заходит в раздел «Appointments» |
| 1. Администратор нажимает на нужную запись |
| 1. Администратор нажимает на «Delete» |
| 1. Появляется уведомление, действительно ли администратор хочет удалить запись |
| 1. Администратор нажимает «yes, I’m sure» |
| 1. Система удаляет запись |
| 1. Система переводит администратора на страницу с записями |
| 1. На экране появляется уведомление, что запись удалена |

Окончание Таблицы 25

|  |
| --- |
| **Расширенный сценарий** |
| 5а. Администратор нажал на «No, take me back»  .1: Система не удаляет запись  .2: система возвращает администратора на страницу с записью |

Таблица 26 – просмотр записей

|  |
| --- |
| **Основной сценарий** |
| 1. Администратор нажимает на «Appointments» |
| 1. На экране система отображает список записей |

Таблица 27 – Просмотр заявки на связь с клиентом

| **Основной сценарий:** |
| --- |
| 1. Администратор заходит в раздел «Callback requests» |
| 1. Администратор нажимает на заявку |
| 1. Система открывает экран с информацией по заявке |

Таблица 28 – Редактирование заявки на связь с клиентом

| **Основной сценарий:** |
| --- |
| 1. Администратор заходит в раздел «Callback requests» |
| 1. Администратор нажимает на заявку |
| 1. Система открывает экран с информацией по заявке |

Окончание Таблицы 28

|  |
| --- |
| 1. Администратор редактирует заявку |
| 1. Администратор нажимает «Save» |
| 1. Система сохраняет информацию по заявке |
| 1. Система переводит администратора на экран с заявками |
| 1. На экране отображается уведомление что заявка изменена |
| **Расширение** |
| 4а. Администратор не заполняет все обязательные поля:  .1: администратор нажимает на «Save»  .2: система выводит ошибку, что не все поля заполнены |
| 5а. Администратор не нажимает на «Save»:  .1: Администратор переходит на другую страницу  .2: Система не сохраняет изменения в заявке |

Таблица 29 – Просмотр информации о пользователе

| **Основной сценарий:** |
| --- |
| 1. Администратор заходит в раздел «Users» |
| 1. Система отображает список пользователей |
| 1. Администратор нажимает на определённого пользователя |
| 1. Система отображает информацию о выбранном пользователе |

Таблица 30 – Редактирование информации о пользователе

| **Основной сценарий:** |
| --- |
| 1. Администратор заходит в раздел «Users» |
| 1. Система отображает список пользователей |
| 1. Администратор нажимает на определённого пользователя |
| 1. Система отображает информацию о пользователе |
| 1. Администратор редактирует информацию |
| 1. Администратор нажимает «Save» |
| 1. Система сохраняет данные |
| 1. Система переводит Администратора на страницу с пользователями |
| 1. На экране отображается уведомление, что данные сохранены |
| **Расширение** |
| 4а. Администратор не заполняет все обязательные поля:  .1: администратор нажимает на «Save»  .2: система выводит ошибку, что не все поля заполнены |
| 5а. Администратор не нажимает на «Save»:  .1: Администратор переходит на другую страницу  .2: Система не сохраняет изменения в заявке |

Таблица 31 – Удаление пользователя

| **Основной сценарий:** |
| --- |
| 1. Администратор заходит в раздел «Users» |
| 1. Администратор нажимает на нужного пользователя |
| 1. Администратор нажимает на «Delete» |
| 1. Появляется уведомление, действительно ли администратор хочет удалить пользователя |
| 1. Администратор нажимает «yes, I’m sure» |
| 1. Система удаляет пользователя |
| 1. Система переводит администратора на страницу с пользователями |
| 1. На экране появляется уведомление, что пользователь удален |
| **Расширение** |
| 5а. Администратор нажал на «No, take me back»  .1: Система не удаляет запись  .2: система возвращает администратора на страницу с записью |

Таблица 32 – добавление пользователя

| **Основной сценарий:** |
| --- |
| 1. Администратор заходит в раздел «Users» |
| 1. Администратор нажимает «Add user» |
| 1. Администратор заполнят информацию о новом пользователе |

Окончание таблицы 32

|  |
| --- |
| 1. Администратор нажимает на «save» |
| 1. Система сохраняет данные о новой записи |
| 1. Система переводит администратора на страницу с пользователями |
| 1. На экране появляется уведомление, что новый пользователь создан |
| **Расширенный сценарий** |
| 3а. Администратор не заполняет все обязательные поля |
| .1: администратор нажимает на «save»  .2: система выводит уведомление что не все поля заполнены |
| 4а. Администратор не нажимает на «Save»  .1: администратор переходит на другую страницу  .2: новый пользователь не сохранен |

Таблица 33 – Сброс пароля

| **Основной сценарий** |
| --- |
| 1. Администратор заходит в раздел «Users» |
| 1. Администратор нажимает на нужного пользователя |
| 1. Администратор нажимает «Reset password» |
| 1. Администратор вводит новый пароль |
| 1. Администратор вводит подтверждение пароля |
| 1. Администратор нажимает на «Change password» |

Окончание таблицы 33

|  |
| --- |
| 1. Система сохраняет новый пароль |
| 1. Система переводит администратора на экран с пользователями |
| 1. Система отображает уведомление, что пароль изменен |
| **Расширение** |
| 4а. Администратор вводит пароль, не соответствующий требованиям:  .1: Администратор подтверждает пароль  .2: Администратор нажимает на «Change password»  .3: Система выводит ошибку, что пароль не соответствует требованиям |
| 5а. Администратор вводит некорректный пароль в поле для подтверждения пароля:  .1: Администратор нажимает на «Change password»  .2: Система выводит уведомление, что пароли не совпадают |

# **2.2 Функциональное моделирование**

Воспользуемся методологией IDEF0[[5]](#_Список_используемых_источников:) для анализа бизнес-процессов и рассмотрим процесс записи в груминг-салон на примере записи через мессенджер «Telegram». На рисунке 22 представлена контекстная диаграмма IDEF0 «Как есть»:

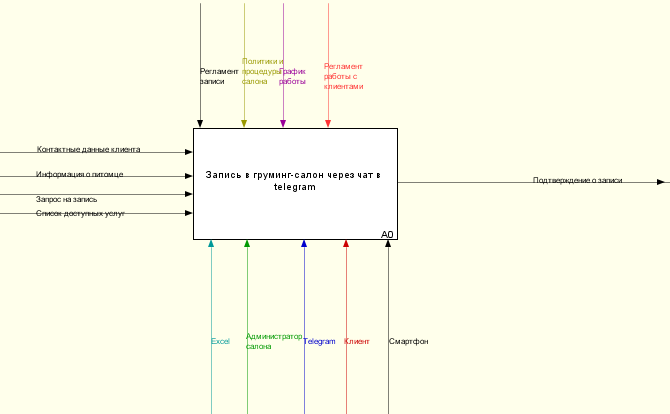


Рисунок 22 – Контекстная диаграмма IDEF0 – «Как есть»

В данной контекстной диаграмме описывается процесс записи в груминг-салон через чат с администратором салона в мессенджере «Telegram». Входные данные включают контактную информацию клиента (например, имя, контактные данные), данные о питомце (порода), запрос на запись с желаемыми датой, временем и мастером, список доступных услуг.

В процессе осуществляется управление соглашениями и правовыми аспектами: Политики и процедуры салона, график работы, регламент работы с клиентами, регламент записи.

Механизмы процесса включают администратора салона, Telegram, клиент, смартфон и excel, в котором администратор ведет расписание записей.

Выходным результатом является подтверждение записи (сообщение от администратора салона в “Telegram”).

Далее подробнее рассмотрим декомпозицию диаграммы IDEF0 - «Как есть». (см. рисунок 23).

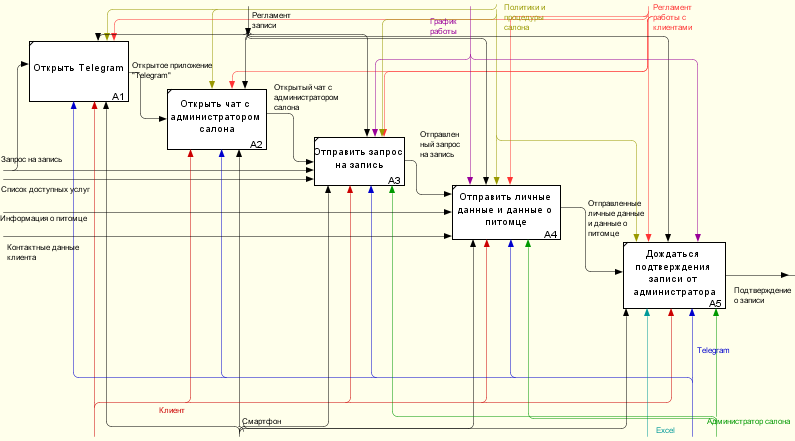


Рисунок 23 – Декомпозиция диаграммы IDEF0 «Как есть»

Представленная декомпозиция диаграммы IDEF0 "Как есть" подробно описывает процесс записи в груминг-салон. Вот последовательность шагов, представленных в декомпозиции:

1. Пользователь открывает telegram.
2. Пользователь открывает чат с администратором салона.
3. Пользователь отправляет запрос на запись.
4. Пользователь отправляет личные данные и данные о питомце.
5. Пользователь ожидает подтверждения записи от администратора.

На рисунке 24 представлена диаграмма IDEF0 «Как будет», которая позволит автоматизировать процесс записи в груминг-салон.

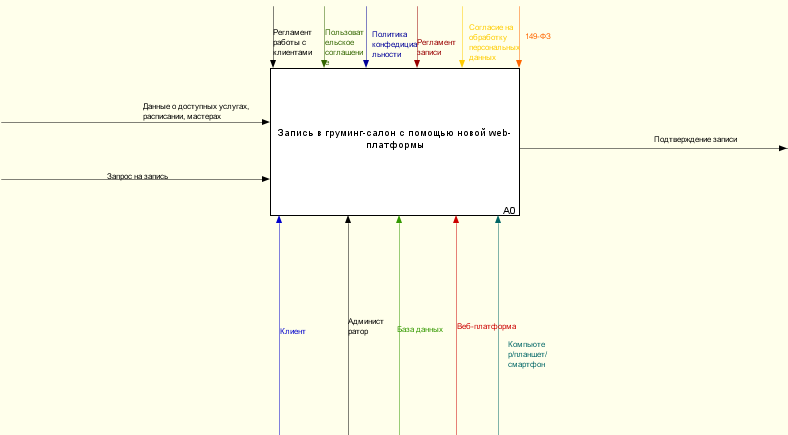


Рисунок 24 – Контекстная диаграмма IDEF0 – «Как будет»

В данной контекстной диаграмме описывается процесс записи в груминг-салон при помощи новой web-платформы. Входные данные включают запрос на запись и данные о доступных услугах, времени на запись и мастерах.

В процессе осуществляется управление пользовательским соглашением, политикой конфиденциальности, согласием на обработку персональных данных, 149-ФЗ, регламентом работы с клиентами и регламентом записи салона.

Механизмы процесса включают в себя пользователей, администраторов, базу данных, веб-платформу и эл. устройство (смартфон, планшет или компьютер). Эти компоненты используются для автоматизированной записи в груминг-салон, предоставления и хранения информации о салоне, его услугах и мастерах.

Выходным результатом является подтверждение записи.

Далее подробнее рассмотрим декомпозицию диаграммы IDEF0 - «Как будет». (см. рисунок 25).

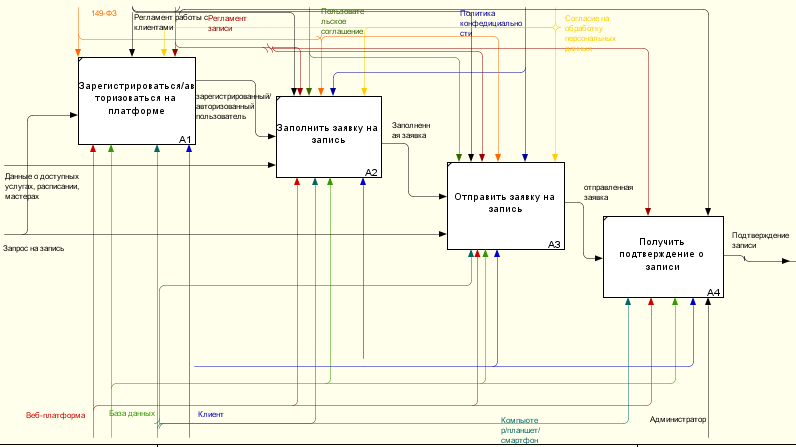


Рисунок 25 – Декомпозиция диаграммы IDEF0 «Как будет»

В представленной декомпозиции диаграммы IDEF0 "Как будет" описывается процесс записи в груминг-салон. Как видно на диаграмме, шагов для осуществления записи меньше, чем на диаграмме “Как есть”. Пользователь сначала проходит регистрацию или авторизацию на платформе, следом заполняет заявку на запись, отправляет её в салон и ожидает подтверждения администратора.

**2.3 Формирование требований к информационной системе** Для формирования требований к разрабатываемой информационной системе мной была выбрана методология FURPS, так как она является подходящей для данной задачи и способна учесть все необходимые аспекты реализации. Методология FURPS широко применяется для классификации требований к информационным системам и имеет долгую и успешную историю использования. Она также соответствует международным стандартам классификации.

**Функциональные требования:**

Система должна включать в себя следующие функции:

* Авторизация/регистрация в системе пользователей - возможность регистрации и входа в систему с использованием уникальных учетных записей.
* Онлайн – запись - возможность пользователям записываться самостоятельно онлайн в салон.
* «Быстрая запись» - возможность пользователям осуществлять «Быструю запись» в виде отправки заявки менеджеру на обратный звонок без авторизации в системе
* Просмотр услуг салона – возможность пользователю ознакомиться с услугами салона, их стоимостью и длительностью
* Просмотр информации о салоне и его контактах – возможность пользователю ознакомиться с информацией о салоне, такой как история, награды лицензии и его контакты
* Возможность публикации и просмотра отзывов о салоне – возможность пользователю просматривать отзывы и оставлять их после авторизации в системе.
* Отображение информации о мастерах салона – возможность ознакомиться с информацией о мастерах салона (об их специализации, образовании и опыте работы)
* Отмена записи – возможность пользователю самостоятельно отменять запись в личном кабинете
* Редактирование личных данных – возможность пользователю редактировать данные в личном кабинете
* Удаление аккаунта – возможность пользователю удалять свой аккаунт

**Требования к удобству использования:**

* Многократное использование ранее введенных данных
* Эффективная навигация на экранном пространстве (Расположение общих разделов сверху, основные действия производятся в центре экрана);
* Языковая доступность. Клиентский интерфейс полностью на русском языке;
* Цвета, применяемые к различным элементам системы, должны соотноситься между собой и быть приятными для восприятия глазами;
* Количество кликов до перехода к какой-либо функции или к модулю не должно превышать 2-3 нажатий;

**Требования к надежности:**

* Готовность системы должна находиться в пределах 90–99%
* Должна быть обеспечена отказоустойчивость в 99% аварийных случаев.
* Минимизация последствий ошибок пользователей путем возможности восстановления после отказа системы, возврата к предыдущему состоянию и контроля системных ресурсов.
* Длительность выполнения задач, количество ошибок, объем потерянного времени и данных должны стремиться к минимуму;
* Количество разрешенных проблем и процентное отношение успешно завершенных операций к их общему числу, должны находиться в пределах 95–100%;

**Требования к производительности:**

* Допустимая средняя задержка в границах от 0,1 до 2 секунд
* Скорость обработки информации. Время обработки запроса соответствует сложности запроса.
* Минимальное потребление ресурсов. Возможность использования сервиса на устройстве любой мощности.
* Максимальное время для запуска и завершения работы системы – не более 10 минут;

**Требования к поддержке:**

* Объем изменений должен соответствовать ровно одному модулю, связанному с текущей задачей;
* Расширение функционала системы;
* Постоянная поддержка работоспособности: обновление данных и устранение ошибок;
* Наращивание дополнительного функционала системы происходит регулярно и без лишних затрат денежных средств, времени, без приостановки работы системы;
* Для использования информационной системы требуется наличие любого современного браузера (например, Яндекс. Браузер, Google Chrome, Safari, Opera, Mozilla Firefox и другие);
* При выводе продукта в эксплуатацию он будет поддерживать русский язык;

**Выводы по 2 главе**

Исходя из концептуальной модели управления записью в груминг-салон, были определены взаимодействие различных компонентов и разработан идеальный сценарий работы информационной системы. Функциональное моделирование, включающее контекстные диаграммы и декомпозиции "как было" и "как будет", наглядно демонстрирует изменения в работе системы записи в груминг-салон.

Применение UML-диаграммы вариантов использования позволяет описать весь функционал и поведение, доступные пользователю и администратору при взаимодействии с приложением для записи в груминг-салон. С использованием классификации требований FURPS+ были определены функциональные требования, требования к удобству использования, надежности, производительности и поддержке, а также учтены различные ограничения.

# **3. ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ**

# 

# **3.1 Проектирование архитектуры информационной системы**

В процессе проектирования информационной системы было решено использовать классическую трехуровневую клиент-серверную архитектуру. Эта архитектура предусматривает разделение системы на три уровня: (см. рисунок 26).

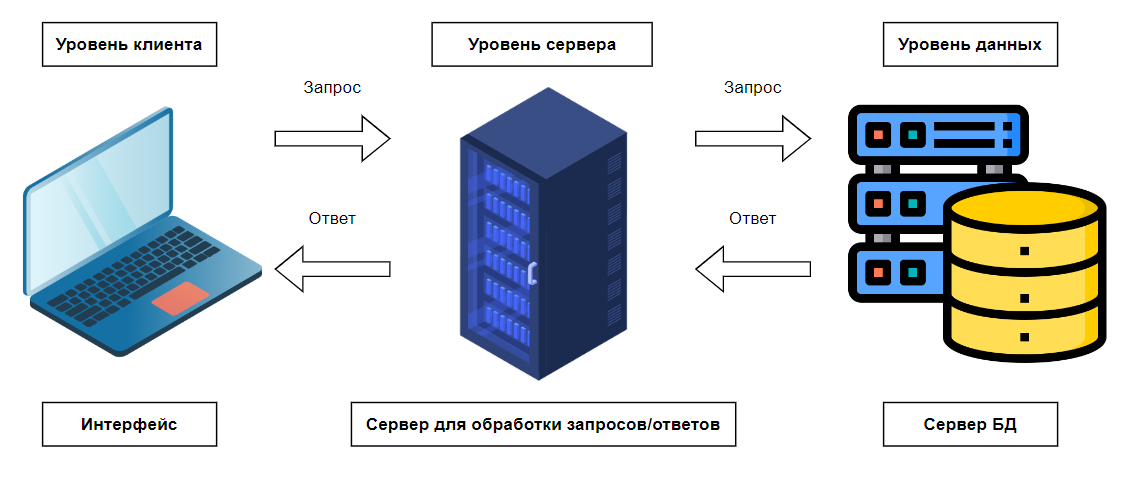


Рисунок 26 – Схема трехуровневой архитектуры приложения

Данная архитектура состоит из трех уровней: уровень клиента (представление), уровень сервера (бизнес-логика), уровень данных (хранилище).

Трехуровневая клиент-серверная архитектура позволяет достичь модульности, масштабируемости и разделения ответственности в системе. Она широко применяется при разработке информационных систем [6].

В связи с выбором трехуровневой архитектуры информационной системы было решено из множества, распространенных сегодня, архитектурных паттернов выбрать MVT (Model-View-Template)-паттерн [7], являющийся адаптацией классического MVC для фреймворка Django, за его широкую популярность и применимость в различных типах приложений. MVT-паттерн обеспечивает модульность, разделение ответственности и удобство в разработке и поддержке информационных систем. Используемый архитектурный паттерн MVT изображен на рисунке 27:



Рисунок 27 – MVT-паттерн

Описание MVT:

1. Модель (Model): описывает данные, используемые в приложении. Отдельные классы, как правило, соответствуют таблицам в базе данных
2. Представление (View): получает запрос, обрабатывает его и отправляет пользователю некоторый ответ. Если для обработки запроса необходимо обращение к модели и базе данных, то View взаимодействует с ними. Для создания ответа может применять Template или шаблоны. В архитектуре MVC этому компоненту соответствуют контроллеры (но не представления)
3. Шаблон (Template): представляет логику представления в виде сгенерированной разметки html. В MVC этому компоненту соответствует View, то есть представления.

На рисунке 28 представлена диаграмма классов [8] основных компонентов «views»:

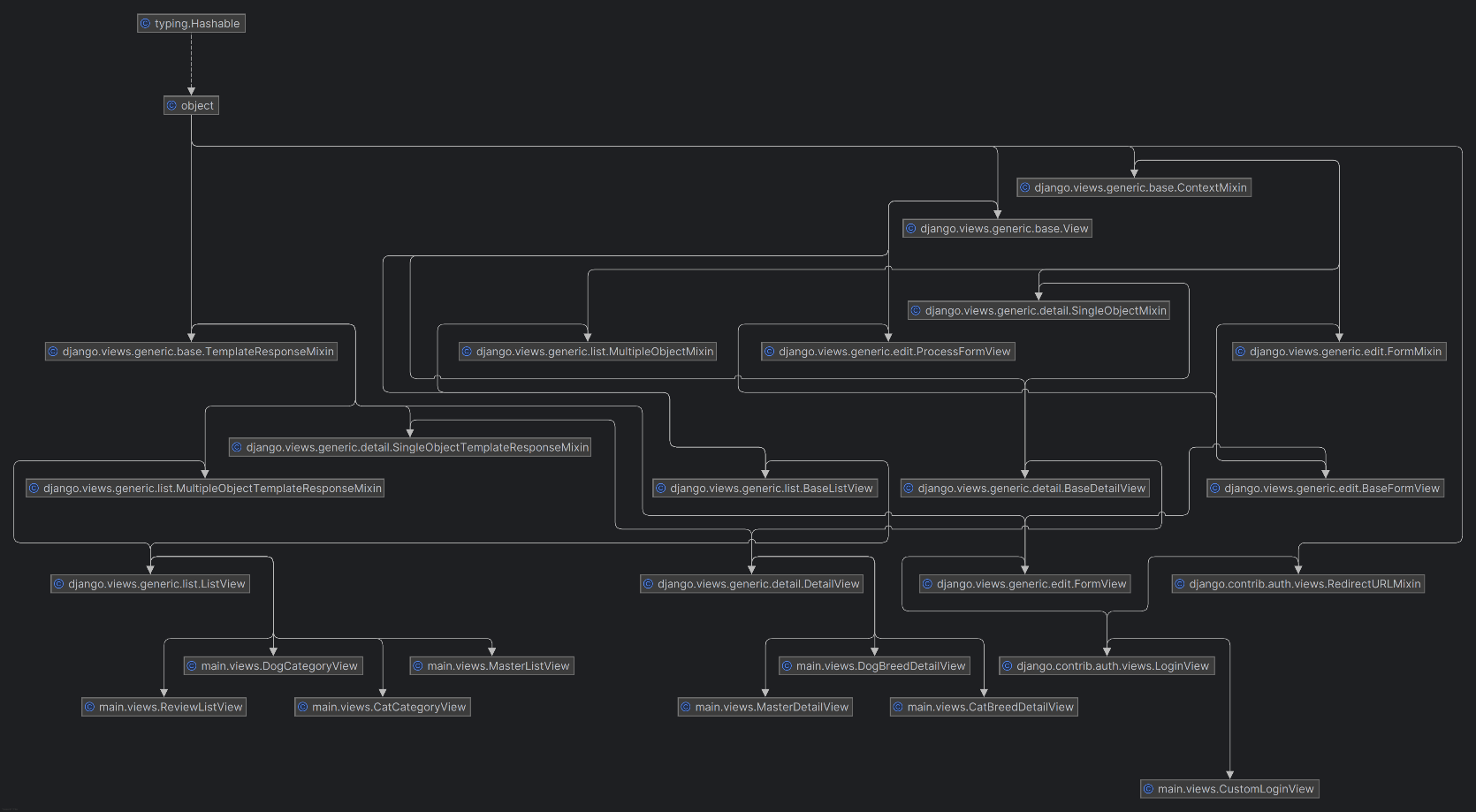


Рисунок 28 – Диаграмма классов «views»

Далее приведено описание классов «views»:

* Typing.Hashable: интерфейс, который определяет, что объект может быть хэширован.
* Object: представляет объекты в модели данных.
* django.views.generic.base.TemplateResponseMixin: класс в Django, который используется для обработки ответов на основе шаблонов. Используется для создания представлений, которые возвращают страницы с формами, информацией об услугах, отзывами клиентов и т.д.
* django.views.generic.list.MultipleObjectTemplateResponseMixin : класс в Django, который используется в представлениях (views) для обработки множественных объектов и их отображения на основе шаблона. Он используется в сочетании с классом ListView, к примеру, для отображения списка услуг, мастеров, записей клиентов.
* django.views.generic.list.listView: класс в Django, который предоставляет функциональность для отображения списка объектов в веб-приложении. Используется для отображения списка услуг, доступных для груминга, списков животных, записей клиентов и т.д.
* main.views.ReviewListView: класс, который обрабатывает представление (view) для отображения списка отзывов пользователей о сервисе.
* main.views.DogCategoryView: класс, который является представлением (view), отвечающее за отображение информации о категориях собак в груминг-салоне (мелкие, средние, крупные).
* main.views.CatCategoryView: класс, который является представлением (view), отвечающее за обработку запросов, связанных с категориями котов (короткошерстные, средне шерстные, длинношерстные).
* main.views.MasterListView: класс, который является представлением (view),  ответственым за отображение списка мастеров (грумеров), работающих в салоне.
* Django.views.generic.list.BaseListView: базовый класс представления в Django, который позволяет создавать представления для отображения списков объектов. Он предоставляет функциональность, например, для списка доступных услуг, мастеров или записей клиентов.
* Django.views.generic.detail.SingleObjectTemplateResponseMixin: один из миксинов в Django, который используется для обработки запросов, связанных с отображением одного объекта. Используется для отображения информации о конкретной записи, клиенте или услуге.
* Django.views.generic.list.MultipleObjectTemplateResponseMixin: класс в Django, который используется для обработки представлений, возвращающих несколько объектов. Он предоставляет возможность рендеринга шаблона с множеством объектов.
* Django.views.generic.detail.DetailView: один из классов представлений (views) в Django, который предоставляет функциональность для отображения деталей одного конкретного объекта модели. Например, используется для отображения информации о конкретной услуге, грумере или клиенте.
* main.views.MasterDetailView: класс являющийся представлением (view), которое отвечает за отображение деталей мастера (грумера) и предоставление пользователю возможности записаться на услугу.
* main.views.DogBreedDetailView: представляет собой класс-представление (view), который отвечает за отображение подробной информации о конкретной породе собак.
* main.views.CatBreedDetailView: класс - представление (view) для отображения деталей конкретной породы кошек в веб-приложении для записи в груминг-салон.
* Django.views.generic.detail.BaseDetailView: класс в Django, который входит в модуль django.views.generic.detail. Он предназначен для обработки запросов на отображение детальной информации об объекте в веб-приложении.
* django.views.generic.edit.FormView: класс представления в Django, который предназначен для обработки форм. Например, такая форма включает поля для ввода породы собаки, выбора услуги (стрижка, купание и т.д.), указания времени записи и др.
* django.contrib.auth.views.LoginView: встроенный класс Django, который предоставляет функциональность для аутентификации пользователей через форму входа в систему.
* main.views.CustomLoginView: класс представляет собой кастомное представление (view) для обработки процесса входа пользователя в систему.
* django.contrib.auth.views.RedirectURLMixin: это вспомогательный класс в Django, который предоставляет функциональность для перенаправления пользователя на определённый URL после выполнения какого-либо действия, например, после успешного входа в систему или выхода из неё.
* main.views.generic.edit.BaseFormView - класс в Django, который предоставляет базовую функциональность для представлений (views), обрабатывающих формы.
* django.views.generic.edit.FormMixin: класс в Django, который. Используется для создания представлений, которые обрабатывают формы записи клиентов.
* django.views.generic.base.ContextMixin: это полезный инструмент для управления данными, которые нужно передать в шаблон в представлениях на основе классов. Используется для легкого предоставления информации о доступных услугах клиентам
* django.views.generic.edit.ProcessFormView: абстрактный класс в Django, который предназначен для обработки форм. Например, для создания представления, которое обрабатывает запись клиентов на услуги груминга.
* django.views.generic.detail.SingleObjectMixin:  предоставляет базовую функциональность для работы с одним объектом в представлениях на основе классов в Django. Этот миксин используется в представлениях, которые требуют обработки и отображения конкретного объекта из базы данных.
* django.views.generic.base.View: базовый класс для создания представлений (views). Используется для обработки HTTP-запросов и возврата соответствующих ответов.

На рисунке 29 представлена диаграмма классов основных компонентов «models»:

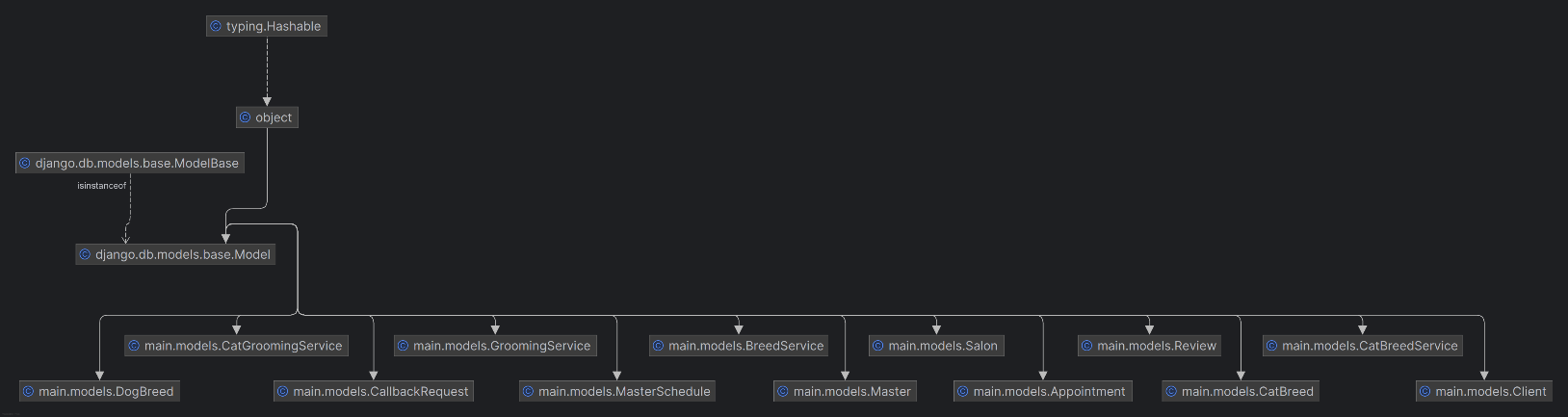


Рисунок 29 – диаграмма классов «models»

Описание классов «models»:

* Typing.Hashable: интерфейс, который определяет, что объект может быть хэширован.
* Object: представляет объекты в модели данных.
* django.db.models.base.ModelBase: метакласс для моделей, определяемых в приложении. Он отвечает за создание и настройку классов моделей, которые используются для взаимодействия с базой данных.
* Django.db.models.base.Model: является базовым классом для всех моделей в Django.
* main.models.DogBreed: класс main.models.DogBreed представляют собой модель, которая описывает породы собак.
* main.models.CatGroomingService: представляет собой модель, которая описывает услуги груминга для кошек.
* main.models.CallbackRequest: представляет собой модель, которая используется для хранения информации о запросах на обратный звонок от клиентов.
* main.models.GroomingService: представляет модель, которая описывает услугу груминга, предоставляемую в груминг-салоне.
* main.models.MasterSchedule: представляет собой модель, которая отвечает за хранение информации о расписании мастеров (грумеров) в салоне.
* main.models.BreedService: представляет собой модель, которая описывает связь между породой собаки и услугами, которые доступны для этой породы в груминг-салоне.
* main.models.Master:представляет собой модель, описывающую мастера (грумера) в груминг-салоне.
* main.models.Salon: представляет собой модель данных, которая описывает структуру таблицы в базе данных, связанной с салоном.
* main.models.Appointment: представляет собой модель, которая описывает запись на прием в груминг-салон.
* main.models.Review: представляет собой модель, которая используется для хранения отзывов клиентов о предоставленных услугах.
* main.models.CatBreed: представляет собой модель, которая описывает породу кошек.
* main.models.CatBreedService: представляет собой модель, которая описывает связь между типом кошки и услугами, которые доступны для этого типа в груминг-салоне.
* main.models.Client: представляет собой модель, которая описывает клиентов салона.

На рисунке 30 представлена диаграмма классов основных компонентов «forms»:



Рисунок 30 – диаграмма классов «forms»

Описание классов «forms»:

* Typing.Hashable: интерфейс, который определяет, что объект может быть хэширован.
* Object: представляет объекты в модели данных.
* django.forms.utils.RenderableMixin: класс в Django, который используется для упрощения рендеринга форм и их компонентов.
* django.forms.utils.RenderableFormMixin: представляет собой вспомогательный класс, который используется для упрощения работы с формами.
* django.forms.forms.BaseForm: базовый класс для создания форм в Django. Используется для создания форм, которые будут обрабатывать пользовательский ввод, например, для записи на услуги груминга.
* typing.Iterator: класс используется для аннотирования типов, связанных с итераторами.
* typing.Iterable: класс предоставляет инструменты для аннотирования типов.
* django.forms.forms.Form: используется для создания форм в веб-приложении.
* django.contrib.auth.forms.AuthenticationForm: это встроенная форма Django, предназначенная для аутентификации пользователей.
* main.forms.CustomLoginForm: пользовательская форма для аутентификации пользователей (логина) в систему.
* main.forms.QuickAppointmentForm : форма, которая используется для быстрого создания записи на услугу.
* django.forms.models.BaseModelForm: базовый класс для создания форм, основанных на моделях Django.
* Django.forms.forms.DeclarativeFieldsMetaclass: метакласс, используемый в Django для создания форм.
* Django.forms.models.ModelFormMetaclass — это метакласс, используемый в Django для создания форм, основанных на моделях.
* django.forms.models.ModelForm: класс в Django, который упрощает создание форм на основе моделей.
* main.forms.ClientProfileForm: представляет собой форму, используемую для сбора и валидации данных профиля клиента.
* django.contrib.auth.forms.BaseUserCreationForm: базовая форма для создания пользователя в Django. Она предоставляет функциональность для создания новых учетных записей пользователей, включая валидацию и обработку данных.
* django.contrib.auth.forms.UserCreationForm: встроенная форма в Django, которая используется для создания новых пользователей.
* main.forms.ClientRegistrationForm: форма для регистрации клиента.
* main.forms.ReviewForm: форма, используемая для сбора отзывов от пользователей о груминг-салоне.
* main.forms.AppointmentForm: форма, связанная с процессом записи клиента на услуги груминга.

На рисунке 31 представлена диаграмма классов основных компонентов admin:

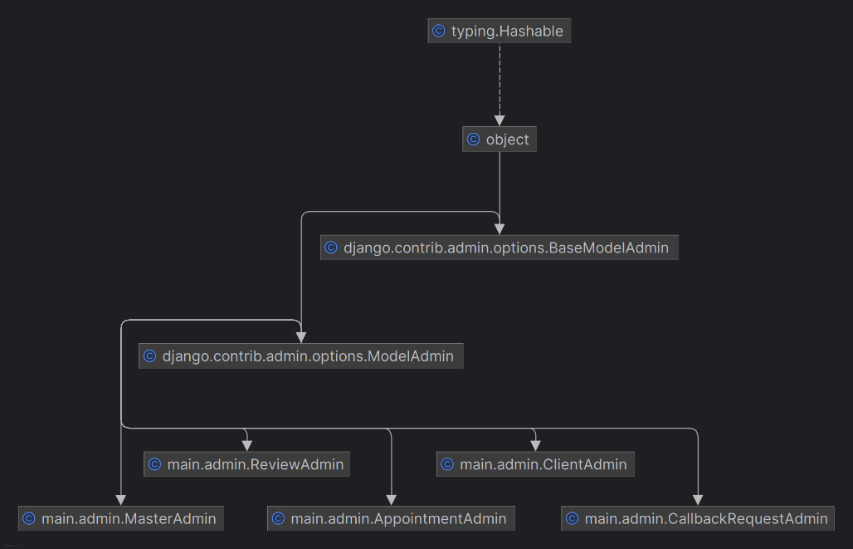


Рисунок 31 – диаграмма классов «admin»

Описание классов «admin»:

* Typing.Hashable: интерфейс, который определяет, что объект может быть хэширован.
* Object: представляет объекты в модели данных.
* django.contrib.admin.options.BaseModelAdmin: базовый класс для создания административных интерфейсов в Django. Используется для создания удобного интерфейса для администраторов, которые будут управлять записями, клиентами, отзывами и другими аспектами бизнеса.
* Класс main.admin.MasterAdmin: представляет собой класс администратора для управления объектами модели "Мастера" в административной панели Django.
* main.admin.ReviewAdmin: класс, который управляет отображением и функциональностью модели отзыва (например, отзыва о груминге) в административной панели Django.
* main.admin.AppointmentAdmin: класс используется для настройки административной панели Django (Django Admin) для управления записями на груминг.
* main.admin.ClientAdmin: относится к административной части приложения, которая управляет моделью клиента (Client).
* main.admin.CallbackRequestAdmin: административный интерфейс для управления запросами на обратный звонок, которые поступают от клиентов в груминг-салон.

На рисунке 32 представлена диаграмма классов основных компонентов apps:

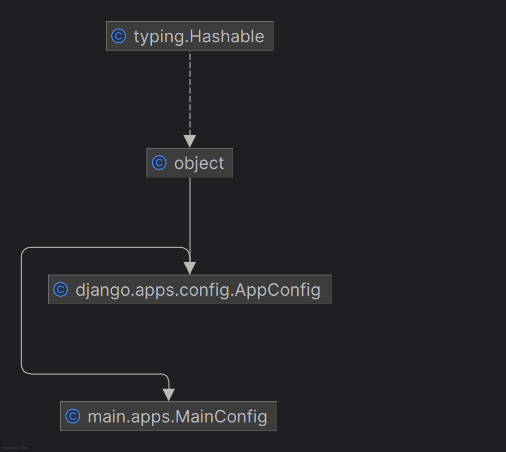


Рисунок 32 – диаграмма классов «apps»

Описание классов «apps»:

* Typing.Hashable: интерфейс, который определяет, что объект может быть хэширован.
* Object: представляет объекты в модели данных.
* django.apps.config.AppConfig: используется для конфигурации приложений в проекте.
* Main.apps.MainConfig: класс конфигурации приложения, служит для настройки параметров приложения. Отвечает за настройку приложения "Main" внутри проекта Django.

На рисунке 33 представлена диаграмма классов основных компонентов backends:

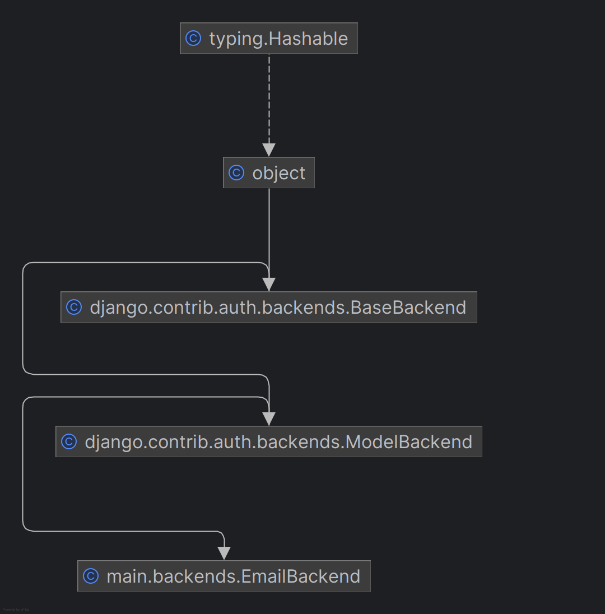


Рисунок 33 – диаграмма классов «backends»

Описание классов «backends»:

* Typing.Hashable: интерфейс, который определяет, что объект может быть хэширован.
* Object: представляет объекты в модели данных.
* django.contrib.auth.backends.BaseBackend: абстрактный класс, который предоставляет базовую реализацию для пользовательских бэкендов аутентификации в Django.
* django.contrib.auth.backends.ModelBackend: класс, предоставляемый Django, который отвечает за аутентификацию пользователей.
* main.backends.EmailBackend: пользовательский класс аутентификации, который использует адрес электронной почты вместо стандартного поля username для входа пользователей.

На рисунке 19 представлена диаграмма классов основных компонентов «migrations»:

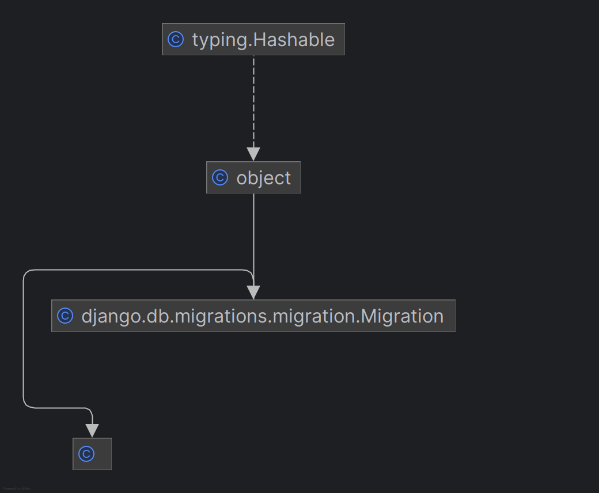


Рисунок 34 – диаграмма классов «migrations»

Описание классов «migrations»:

* Typing.Hashable: интерфейс, который определяет, что объект может быть хэширован.
* Object: представляет объекты в модели данных.
* Django.db.migrations.migration.Migration: отвечает за управление изменениями в структуре базы данных приложения. Миграции позволяют легко вносить изменения в модели (например, добавление, изменение или удаление полей в таблицах) и синхронизировать эти изменения с базой данных.

На рисунке 35 представлена диаграмма развертывания. На ней представлена визуализация компонентов и элементов системы:

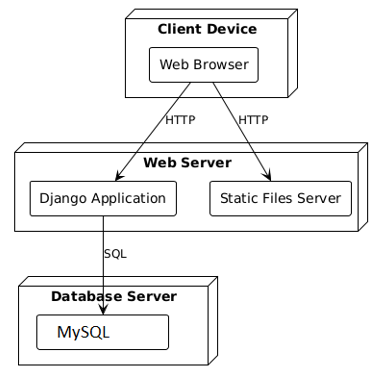


Рисунок 35 – UML диаграмма развёртывания

На стороне клиента находится пользователь, который запускает гаджет с минимальными системными требованиями и доступом в Интернет. Затем пользователь открывает любой современный браузер, такой как Mozilla Firefox, Google Chrome и т. д., и переходит на веб-сервис для записи в груминг-салон по доменному имени. Все данные, запрашиваемые или внесенные пользователем, сохраняются в базе данных MySQL.

# **3.2 Инфологическое проектирование**

В ходе проектирования информационной системы была разработана контекстная DFD диаграмма, которая позволяет визуализировать как движение данных между объектами системы, так и преобразования данных, которые могут применяться на разных шагах процесса. На рисунке 36 представлена контекстная DFD диаграмма [9].

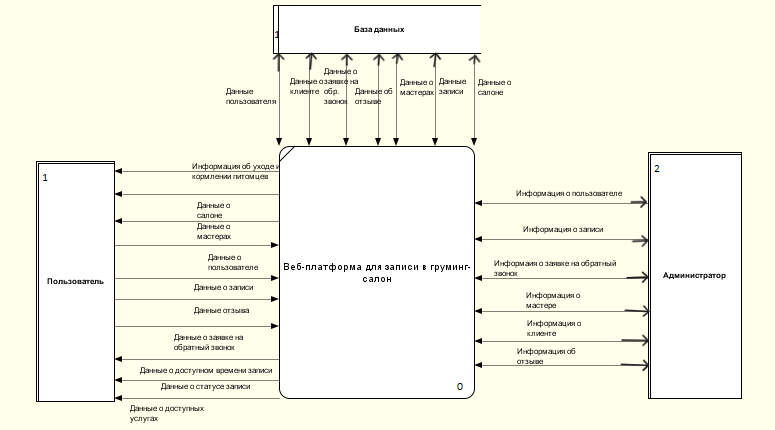


Рисунок 36 – Контекстная DFD диаграмма

На рисунке 37 представлена инфологическая модель реляционной базы данных для веб-платформы для записи в груминг-салон, реализованной с помощью СУБД MySQL, которая была выбрана по нескольким причинам:

1. Производительность: MySQL хорошо справляется с большими объемами данных и высокими нагрузками. Если груминг-салон будет обрабатывать много записей и запросов, MySQL может обеспечить необходимую производительность.
2. Гибкость и масштабируемость: MySQL позволяет легко масштабировать базу данных, как вертикально (увеличение ресурсов сервера), так и горизонтально (распределение нагрузки между несколькими серверами). Это может быть полезно, если бизнес вырастет.
3. Интеграция с Django: Django имеет отличную интеграцию с MySQL через библиотеку «mysqlclient». Это позволяет использовать все преимущества Django ORM, обеспечивая удобный и эффективный доступ к данным.
4. Сообщество и поддержка: MySQL является одной из самых популярных систем управления базами данных, поэтому можно найти множество ресурсов, документации и активное сообщество, которое поможет решить возможные проблемы.

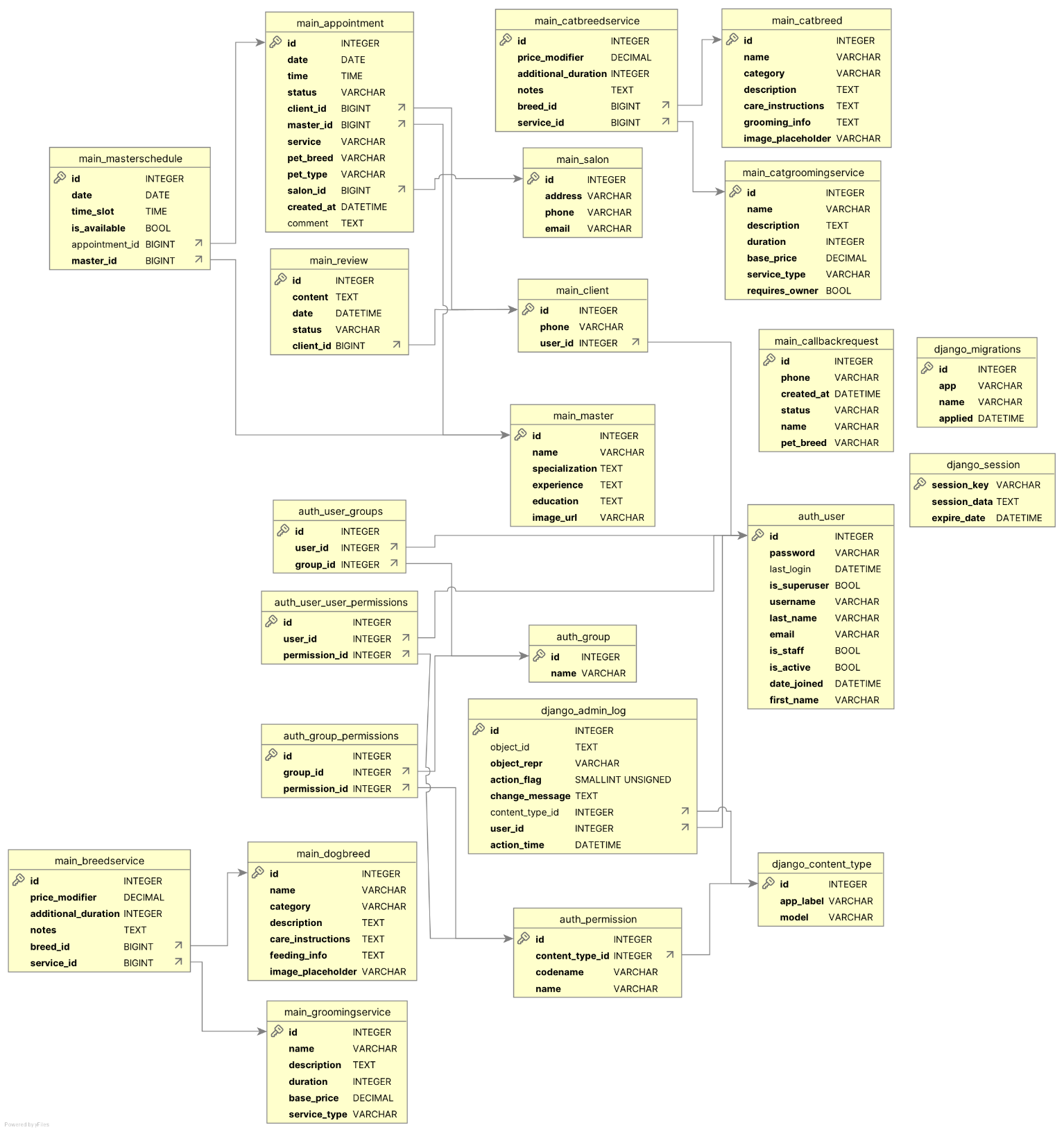


Рисунок 37 – Инфологическая модель базы данных

Структура спроектированной ER-диаграммы описана в таблицах 34-56.

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id | INTEGER | ID клиента, первичный ключ таблицы |
| phone | VARCHAR | Телефон клиента |
| User\_id | INTEGER | ID пользователя, внешний ключ таблицы auth\_user |

Таблица 34 – Структура таблицы main\_client в базе данных

Таблица 35 – Структура таблицы main\_master в базе данных

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id | integer | ID мастера, первичный ключ |
| name | varchar | Имя мастера |
| specialization | text | Специализация мастера |
| experience | text | Опыт работы мастера |
| education | text | Образование мастера |
| Image\_url | varchar | URL-адрес фото мастера |

Таблица 36 – Структура таблицы main\_salon в базе данных

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id | integer | Id салона, первичный ключ |
| address | varchar | Адрес салона |
| phone | varchar | Номер телефона салона |
| email | varchar | Эл. Адрес салона |

Таблица 37 – Структура таблицы main\_review в базе данных

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id | INTEGER | ID отзыва |
| content | TEXT | Текст отзыва |
| date | DATETIME | Дата и время составления отзыва |

Окончание Таблицы 37

|  |  |  |
| --- | --- | --- |
| status | VARCHAR | Статус отзыва |
| Client\_id | BIGINT | ID клиента, внешний ключ таблицы main\_client |

Таблица 38 – Структура таблицы main\_masterschedule в базе данных

|  |  |  |
| --- | --- | --- |
| id | integer | ID расписания, первичный ключ |
| date | date | Дата работы мастера |
| Time\_slot | time | Время, в которое работает мастер |
| Is\_available | bool | Флаг доступности мастера |
| Appointment\_id | bigint | id записи, внешний ключ таблицы main\_appointment |
| Master\_id | bigint | ID мастера, внешний ключ таблицы main\_master |

Таблица 39 – Структура таблицы main\_appointment в базе данных

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id | integer | Id записи, первичный ключ |
| date | DATE | Дата записи |
| time | TIME | Время записи |
| status | VARCHAR | Статус записи |
| client\_id | BIGINT | Id клиента, внешний ключ таблицы main\_client |
| master\_id | BIGINT | ID мастера, внешний ключ таблицы main\_master |
| service | VARCHAR | Наименование услуги |
| pet\_breed | VARCHAR | Порода питомца |
| pet\_type | VARCHAR | Тип породы питомца |
| salon\_id | BIGINT | ID салона, внешний ключ таблицы main\_salon |
| Created\_at | DATETIME | Дата создания записи |
| comment | TEXT | Комментарий к записи |

Таблица 40 – Структура таблицы main\_catbreed в базе данных

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id | integer | Id породы кошек, первичный ключ |
| name | varchar | Порода кошек |
| category | varchar | категория |
| description | text | описание |
| Care\_instructions | text | Рекомендации по уходу за данной породой |
| Grooming\_info | text | Информация по грумингу |
| Image\_placeholder | varchar | Изображение |

Таблица 41 – Структура таблицы main\_catgroomingservice в базе данных

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id | integer | Id услуги для определенной породы кошек, первичный ключ |
| name | varchar | Наименование услуги |
| description | text | описание |
| duration | integer | Длительность услуги в минутах |
| Base\_price | decimal | Цена в рублях |
| Service\_type | varchar | Тип сервиса |
| Requires\_owner | bool | Надо ли владельцу присутствовать на услуге |

Таблица 42 – Структура таблицы main\_callbackrequest в базе данных

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id | integer | Id заявки на обратный звонок, первичный ключ |

Окончание таблицы 42

|  |  |  |
| --- | --- | --- |
| phone | varchar | Телефон клиента |
| Created\_at | datetime | Дата и время создания |
| notes | text | комментарий |
| status | varchar | Статус заявки |
| name | varchar | Имя отправителя |
| Pet\_breed | varchar | Порода питомца |

Таблица 43 – Структура таблицы main\_catbreedservice в базе данных

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id | integer | Id услуги для определенной породы кошек, первичный ключ |
| Price\_modifier | decimal | Цена услуги |
| Additional\_duration | integer | Длительность услуги по времени |
| notes | text | комментарий |
| Breed\_id | bigint | id породы, внешний ключ таблицы main\_catbreed |
| Service\_id | bigint | Id услуги, внешний ключ таблицы main\_catgroomingservice |

Таблица 44 – Структура таблицы auth\_user в базе данных

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id | INTEGER | ID пользователя, первичный ключ таблицы |
| password | VARCHAR | Пароль пользователя |
| Last\_login | DATEIME | Дата и время последней авторизации |
| Is\_superuser | BOOL | Флаг пользователя со всеми правами |

Окончание таблицы 44

|  |  |  |
| --- | --- | --- |
| username | VARCHAR | Имя пользователя в системе |
| Last\_name | VARCHAR | Фамилия пользователя |
| email | VARCHAR | Эл. Адрес пользователя |
| Is\_staff | BOOL | Флаг пользователя с правами администратора |
| Is\_active | BOOL | Флаг авторизации пользователя |
| Date\_joined | DATETIME | Дата регистрации |
| firstname | VARCHAR | Имя пользователя |

Таблица 45 – Структура таблицы auth\_user\_groups в базе данных

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| User\_id | integer | Id сотрудника, внешний ключ таблицы auth\_user |
| id | integer | id группы пользователей, первичный ключ таблицы |
| group\_id | integer | Id группы, внешний ключ таблицы auth\_group |

Таблица 46 – Структура таблицы auth\_group в базе данных

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id | integer | Id группы, первичный ключ |
| name | varchar | Наименование группы |

Таблица 47 – Структура таблицы auth\_user\_user\_permissions в базе данных

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| user\_id | integer | id пользователя используется для связи между пользователями и разрешением, внешний ключ таблицы auth\_user |
| permission\_id | integer | id разрешения связанный с пользователем, внешний ключ таблицы auth\_permission |
| id | integer | id записи, первичный ключ таблицы |

Таблица 48 – Структура таблицы auth\_group\_permissions в базе данных

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| group\_id | integer | id группы связанный с разрешением, внешний ключ таблицы auth\_group |
| permission\_id | integer | id разрешения связанный с группой, внешний ключ таблицы auth\_permission |
| id | integer | id записи, первичный ключ таблицы |

Таблица 49 – Структура таблицы auth\_ permission в базе данных

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| name | varchar | Название разрешения |
| content\_type\_id | integer | id типа содержимого, внешний ключ таблицы django\_content\_type |
| codename | varchar | Кодовое имя разрешения |
| id | integer | id разрешения, первичный ключ таблицы |

Таблица 50 – Структура таблицы django\_content\_type в базе данных

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| app\_label | varchar | Используется для идентификации и организации приложений |
| model | varchar | Определяет структуру и поведение данных в базе данных |
| id | integer | id разработанной структуры, первичный ключ таблицы |

Таблица 51 – Структура таблицы django\_admin\_log в базе данных PetFinderVision

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| action\_time | timestamp with time zone | Время выполнения действия администратора |
| object\_id | text | id объекта, с которым связано действие |

Окончание таблицы 51

|  |  |  |
| --- | --- | --- |
| object\_repr | varchar | Строковое представление объекта, с которым связано действие |
| action\_flag | Smallint unsigned | Флаг действия, указывающий на тип действия (1 для добавления, 2 для изменения, 3 для удаления и т.д.). |
| change\_message | text | Сообщение об изменениях или комментарий, связанный с действием. |
| content\_type\_id | integer | Идентификатор типа содержимого (Content Type) объекта. Внешний ключ к таблице django\_content\_type |
| user\_id | integer | id пользователя, выполнившего действие. Внешний ключ к таблице auth\_user |
| id | integer | id записи, первичный ключ таблицы |

Таблица 52 – Структура таблицы main\_dogbreed в базе данных

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id | varchar | Id породы, первичный ключ |
| name | varchar | Название породы |
| category | varchar | Категория породы |
| description | text | описание |
| Care\_instructions | text | рекомендации по уходу |
| Feeding\_info | text | Рекомендации по кормлению |
| Image\_placeholder | varchar | Изображение |

Таблица 53 – Структура таблицы main\_groomingservice в базе данных

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id | integer | Id услуги, первичный ключ |
| name | varchar | Наименование услуги |
| description | text | Описание услуги |

Окончание Таблицы 53

|  |  |  |
| --- | --- | --- |
| duration | integer | Длительность услуги в минутах |
| Base\_price | decimal | Базовая цена |
| Service\_type | varchar | Тип сервиса |

Таблица 54 – Структура таблицы main\_breedservice в базе данных

|  |  |  |
| --- | --- | --- |
| id | integer | Id услуги для определенной породы, первичный ключ |
| Price\_modifier | decimal | Цена услуги в рублях |
| Additional\_duration | integer | длительность |
| notes | text | комментарий |
| Breed\_id | bigint | id породы, внешний ключ main\_dogbreed |
| Service\_id | bigint | Id услуги, внешний ключ таблицы main\_groomingservice |

Таблица 55 – Структура таблицы django\_migrations в базе данных

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| app | varchar | Название приложения, к которому относится миграция |
| name | varchar | Название разрешения |
| applied | timestamp with time zone | Дата и время применения миграции |
| id | integer | id записи, первичный ключ таблицы |

Таблица 56 – Структура таблицы django\_session в базе данных

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| session\_data | text | Сериализованные данные сессии |
| expire\_date | timestamp with time zone | Дата и время истечения сессии |

Окончание таблицы 56

|  |  |  |
| --- | --- | --- |
| session\_key | varchar | id сессии, первичный ключ таблицы |

# **Проектирование интерфейса**

На рисунке 38, слева изображена главная страница веб-платформы для записи в груминг-салон. Справа имеется страница регистрации на веб-платформе. Чтобы зарегистрироваться, пользователю необходимо на странице регистрации ввести свой контактный телефон, эл. адрес, имя, пароль, подтвердить пароль и нажать «Зарегистрироваться»:

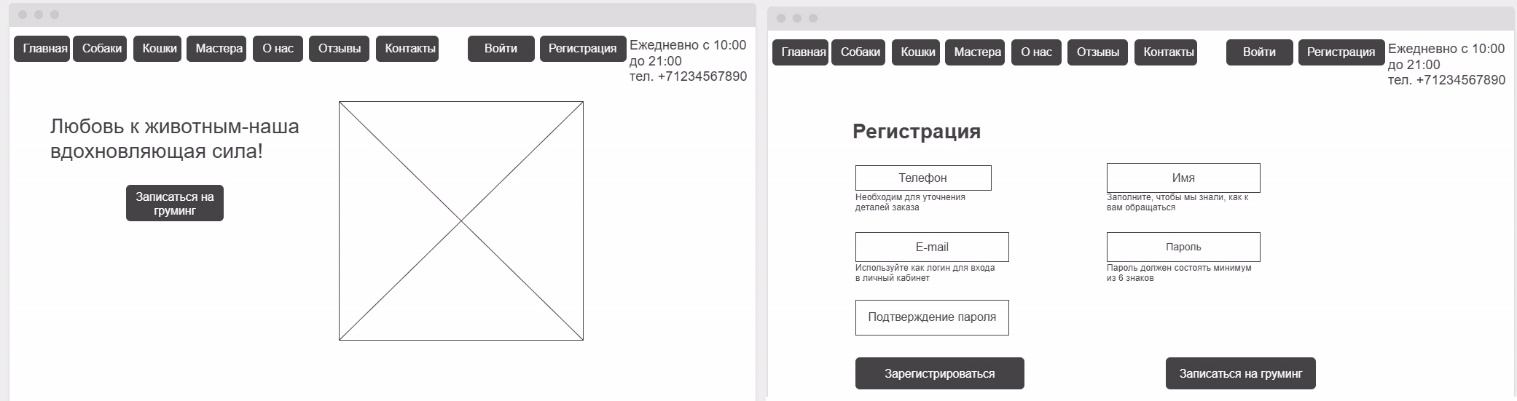


Рисунок 38 – Начальная страница перед входом на платформу.

После регистрации система переведет клиента на страницу авторизации, которая изображена на рисунке 39, слева. На данной странице клиенту надо ввести свой эл. Адрес и пароль и нажать на «Войти».

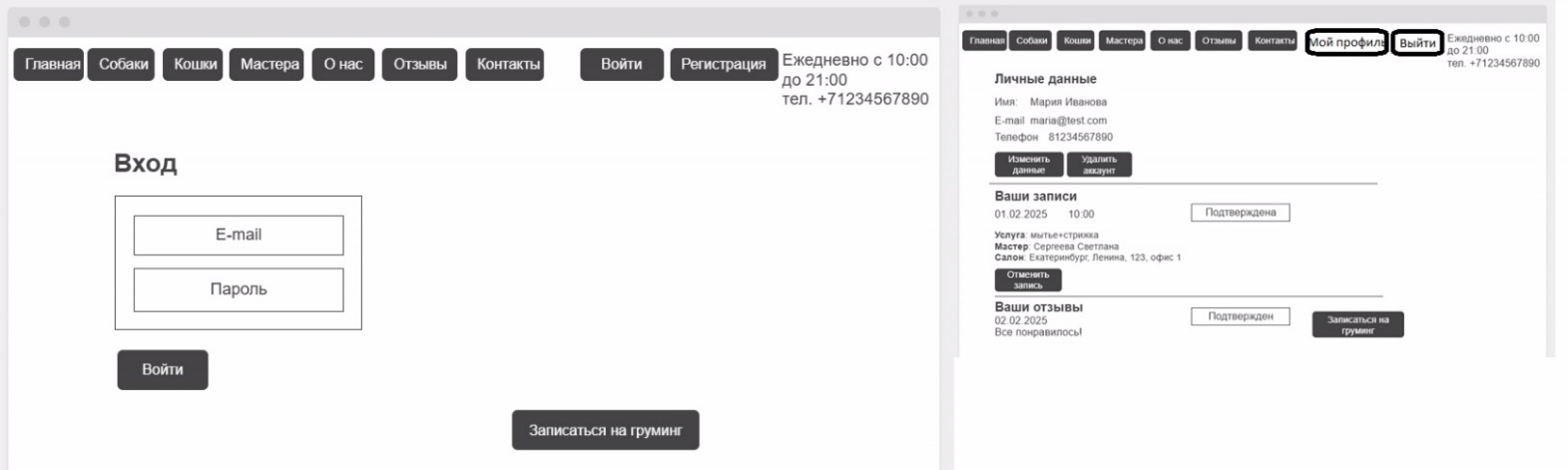


Рисунок 39 – Страница авторизации

На рисунке 39 справа представлена страница «Мой профиль», которая открывается после авторизации в системе. На этой старице пользователь может посмотреть или изменить личные данные, удалить аккаунт, посмотреть список записей и их статусы, отменить запись при необходимости, посмотреть оставленные ранее им отзывы.

На рисунке 40, слева изображен раздел «Собаки». На данной странице представлены все услуги для собак, их длительность и стоимость. Также пользователь может переключаться между видами пород собак : «Мелкие», «Средние», «Крупные» и нажать на нужную породу, чтобы открыть страницу с предлагаемыми для выбранной породы услугами и почитать рекомендаии по уходу и кормлению за данной породой. Страница с услугами для выбранной породы изображена на Рисунке 40, справа:



Рисунок 40 – Страница «Собаки»

На рисунке 41 справа избражена станица раздела «Кошки». Если пользователь выбирает определенный вид/породу кошки, то у него отобразятся услуги, предлагаемые для данного вида, что отображено на Рисунке 41 слева:

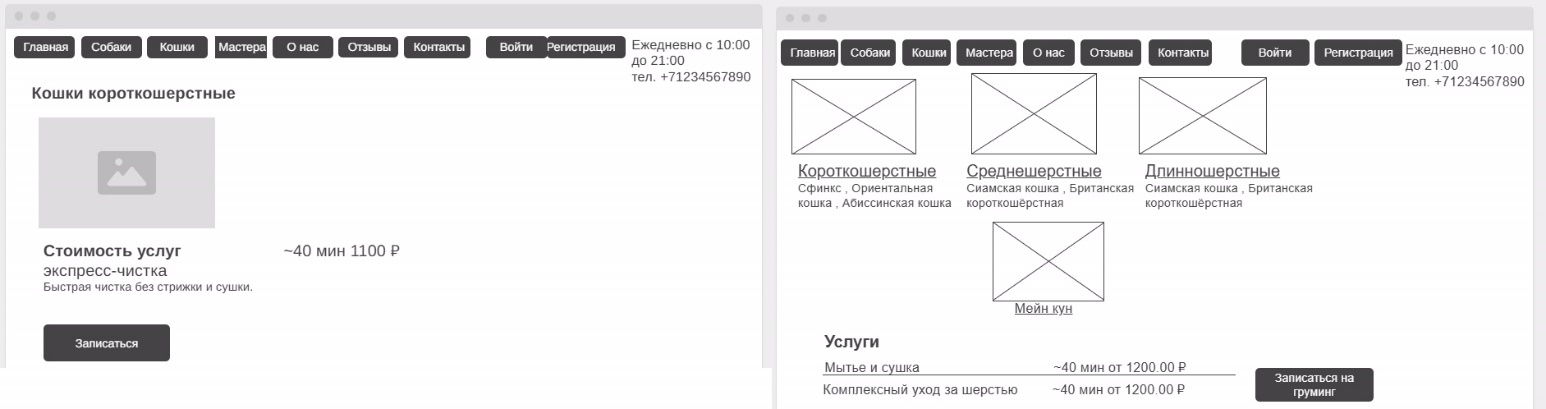


Рисунок 41 – Страница «Кошки короткошерстные»

На рисунке 42 слева изображена страница раздела «Мастера». На ней представлены мастера салона и краткая информация о них:

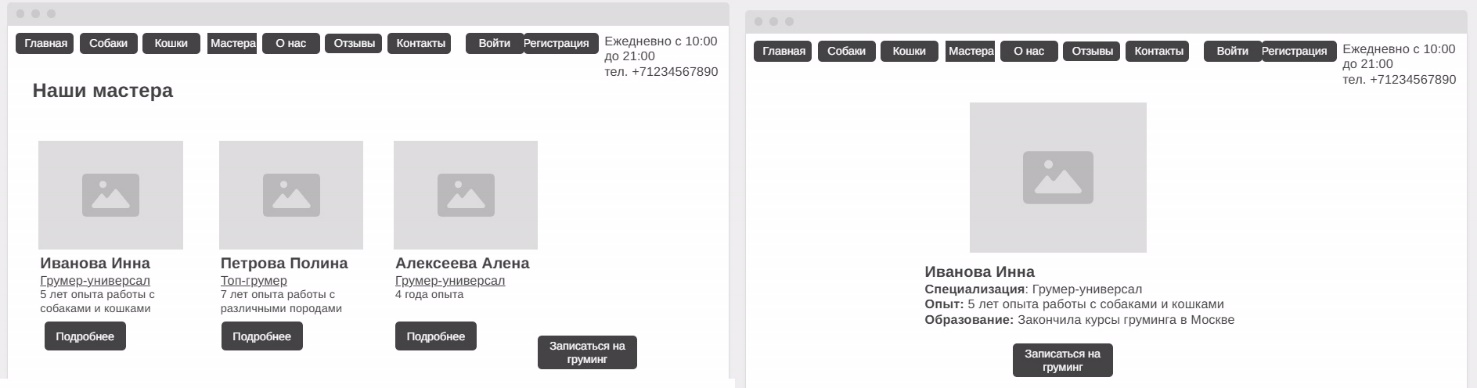
****

Рисунок 42 – Страница «Мастера»

Пользователь может нажать на «Подробнее», чтобы болеее детально ознакомиться с опытом работы и образованием мастера, как показано на Риснке 42, справа.

На рисунке 43 слева показана страница «О нас» с информацией о салоне (история, лицензии, сертификаты и награды). Справа изображена страница «Отзывы», на которой клиент может ознакомиться с отзывами и, после авторизации, оставить собственный:

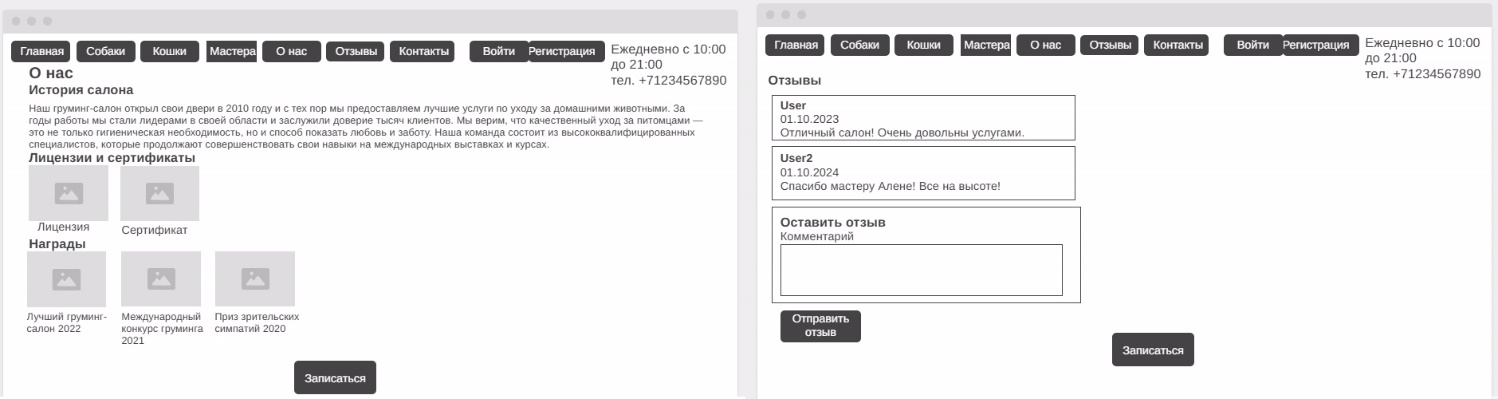


Рисунок 43 – Страница «О нас»

В разделе «Контакты», изображенном на рисунке 44 слева, пользователь может ознакомиться с контактами салона:

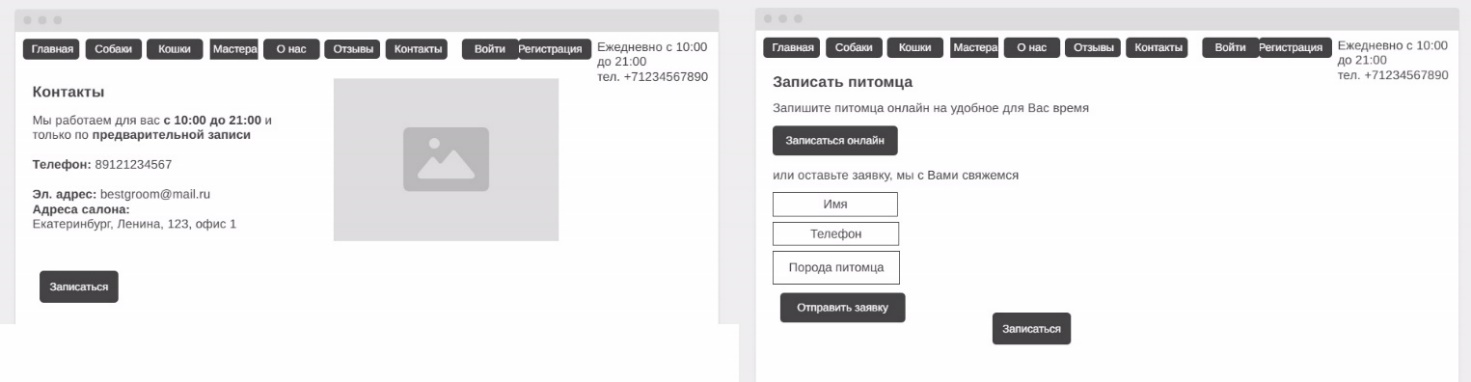


Рисунок 44 – Страница записи

Справа отображена страница, открывающаяся после нажатия клиентом на кнопку «Записаться». На ней пользователь может выбрать формат онлайн записи, нажав на «Записаться онлайн» или оставить заявку на обратный звонок администратора, заполнив форму для заявки н нажав на «Отправить заявку».

В случае, если пользователь выбирает «Записаться онлайн», открывается страница с формой для онлайн-записи, которая показана на рисунке 45:

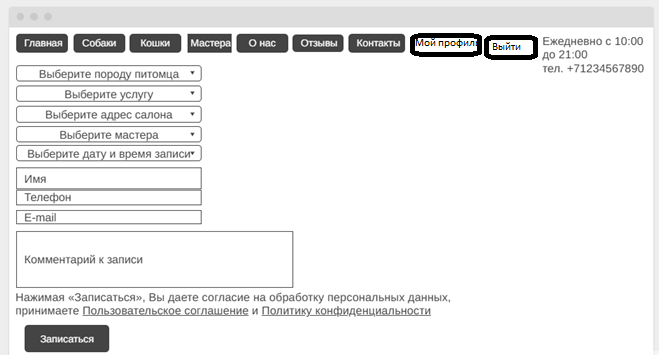


Рисунок 45– Страница онлайн-записи

На странице онлайн-записи клиенту надо выбрать породу питомца, услугу, адрес салона, мастера, дату и время записи. Имя, телефон и e-mail будут автоматически заполнены. Также клиент может оставить комментарий к записи. Пользователь может ознакомиться с Пользовательским соглашением и политикой конфиденциальности, нажав на них.

# **3.4 Программная реализация**

# **3.4.1 Выбор используемых технологий**

Стек используемых технологий представлен в таблице 57:

Таблица 57 стек технологий

|  |  |
| --- | --- |
| File:PyCharm Icon.svg - Wikimedia Commons | **PyCharm** — это кроссплатформенная интегрированная среда разработки для языка программирования Python, разработанная компанией JetBrains на основе IntelliJ IDEA. |
| Python Logo transparent PNG - StickPNG | **Python** — это высокоуровневый язык программирования, который обладает простым и понятным синтаксисом. Он позволяет разработчикам эффективно писать код, создавать разнообразные программы и решать различные задачи. |

Окончание таблицы 57

|  |  |
| --- | --- |
| HTML - Wikipedia | **HTML** – это стандартизированный язык гипертекстовой разметки документов для просмотра web-страниц в браузере |
| CSS3 Logo PNG Transparent & SVG Vector - Freebie Supply | **CSS** - это формальный язык декорирования и описания внешнего вида web-страниц и документов |
|  | **JavaScript** - это мультипарадигменный язык программирования, используемый для создания интерактивных web-страниц и элементов на них. |
|  | **Git** — это развитая система контроля версий с активной поддержкой и открытым исходным кодом, которую используют тысячи разработчиков из разных точек мира. |
|  | **Python Virtualenv** - это инструмент, используемый для создания изолированного рабочего пространства для приложения Python. |
| upload.wikimedia.org/wikipedia/commons/thumb/7/... | **Django** — это популярный фреймворк для разработки веб-приложений на языке Python. Он предоставляет набор инструментов и функций, упрощающих создание высокопроизводительных и масштабируемых веб-приложений. |
| MySQL Logo and symbol, meaning, history, sign. | **MySQL –** это свободная реляционная система управления базами данных. Использовалась для хранения и управления базой данных веб-сервиса |

# **Скриншоты экранов приложения**

На рисунке 46 представлена главная страница веб-платформы:

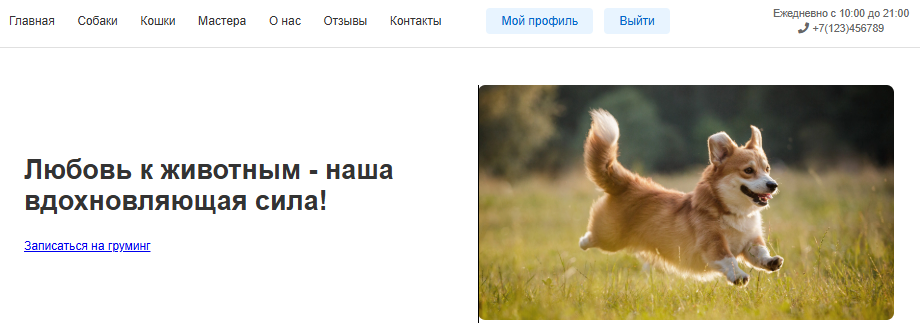


Рисунок 46 – главная страница ИС

На рисунке 47 изображена страница регистрации нового пользователя:

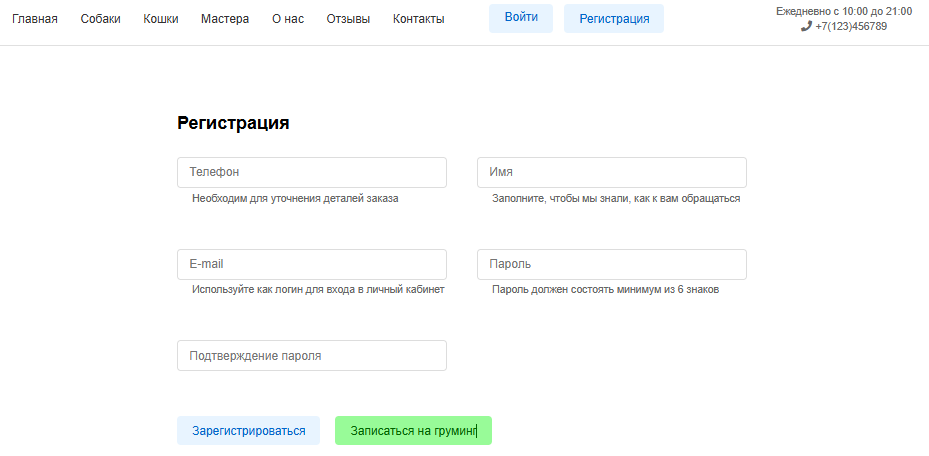


Рисунок 47 – страница регистрации

На рисунке 48 изображена страница авторизации:

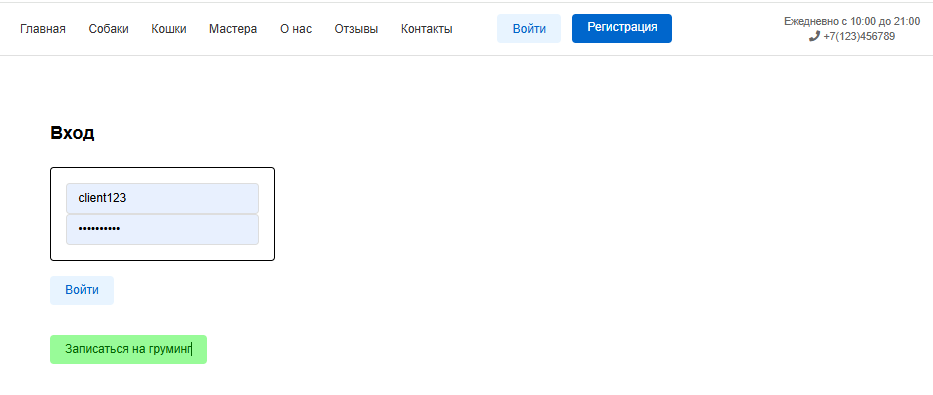


Рисунок 48 – страница Авторизации

На рисунке 49 представлена страница «Мой профиль»:

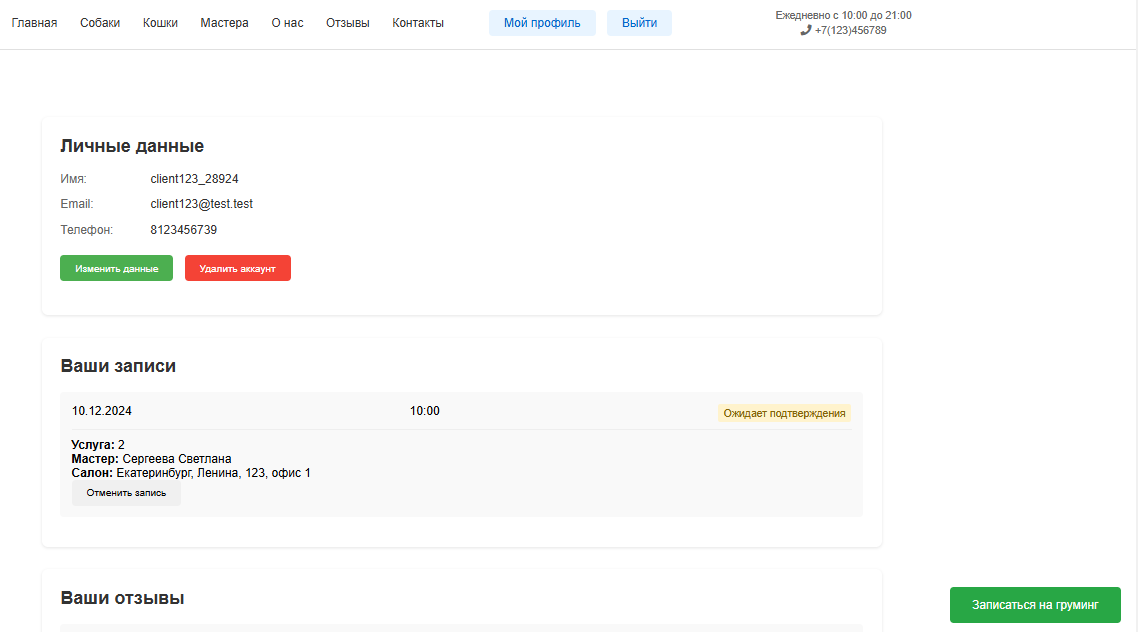


Рисунок 49 – страница «Мой профиль»

На рисунке 50 изображена страница выбора варианта записи в груминг-салон:

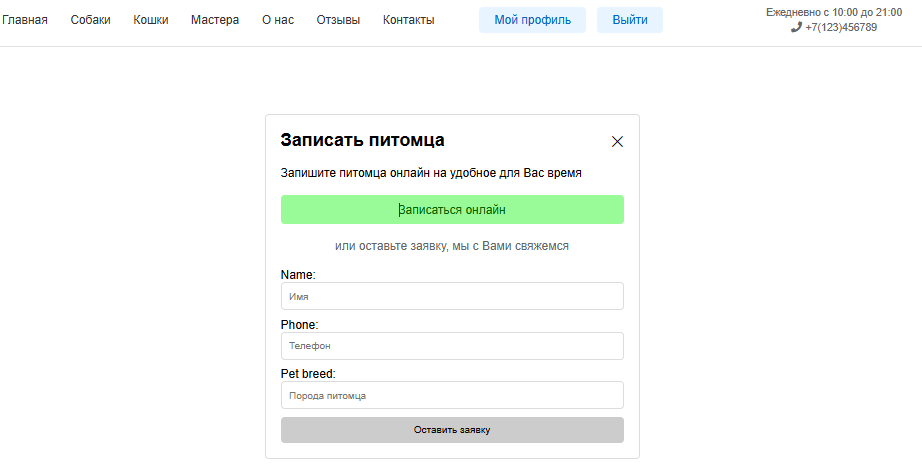


Рисунок 50 – страница выбора варианта записи

На рисунке 51 изображена страница с формой для онлайн-записи:

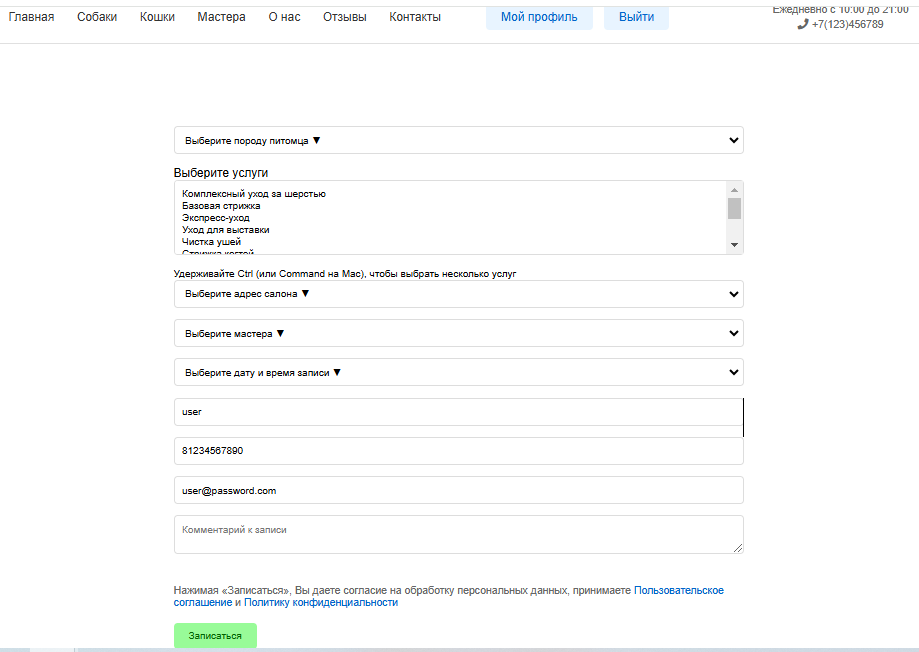


Рисунок 51 – форма онлайн-записи

На рисунке 52 изображена страница с предлагаемыми услугами для собак:



Рисунок 52 – страница «Собаки»

На рисунке 53 отображена страница с предлагаемыми услугами для кошек:

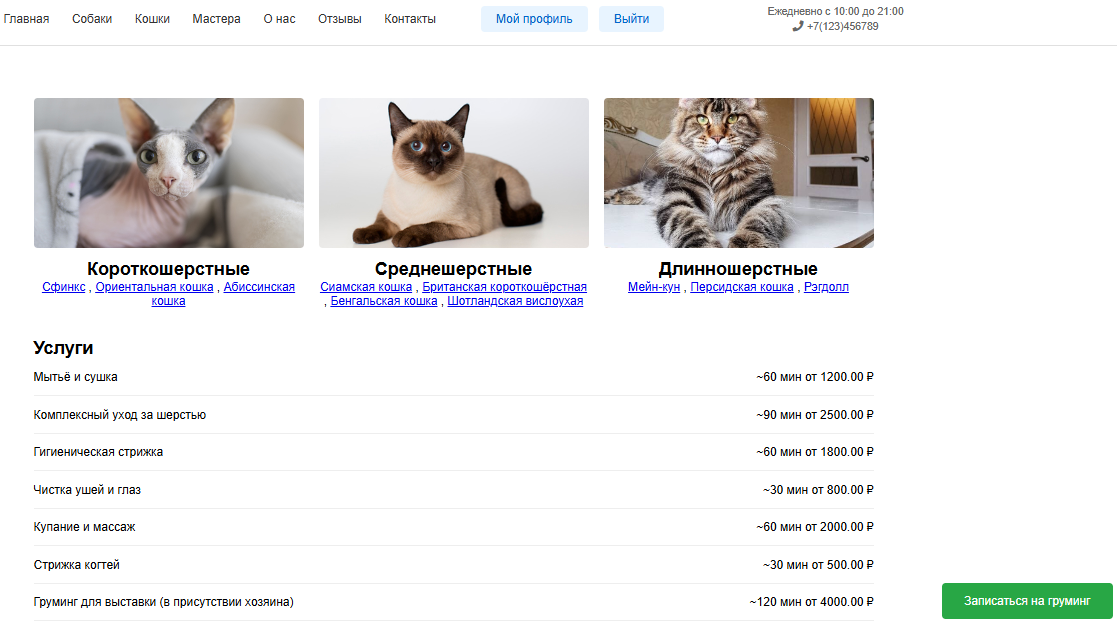


Рисунок 53 – страница «Кошки»

На рисунке 54 изображена страница с мастерами салона:

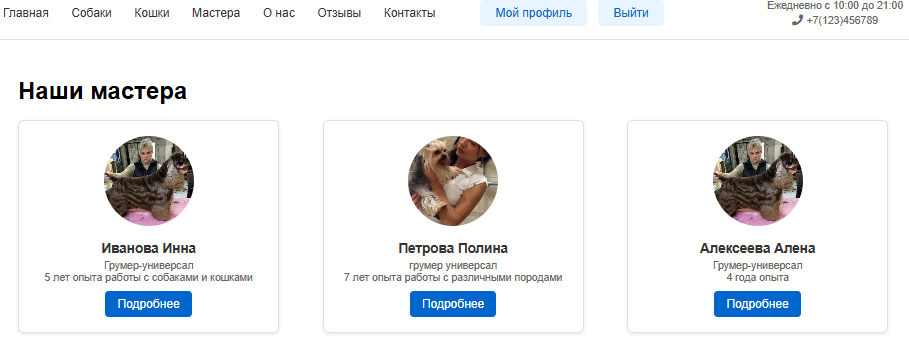


Рисунок 54 – страница «Мастера»

На рисунке 55 изображена страница с информацией о салоне:

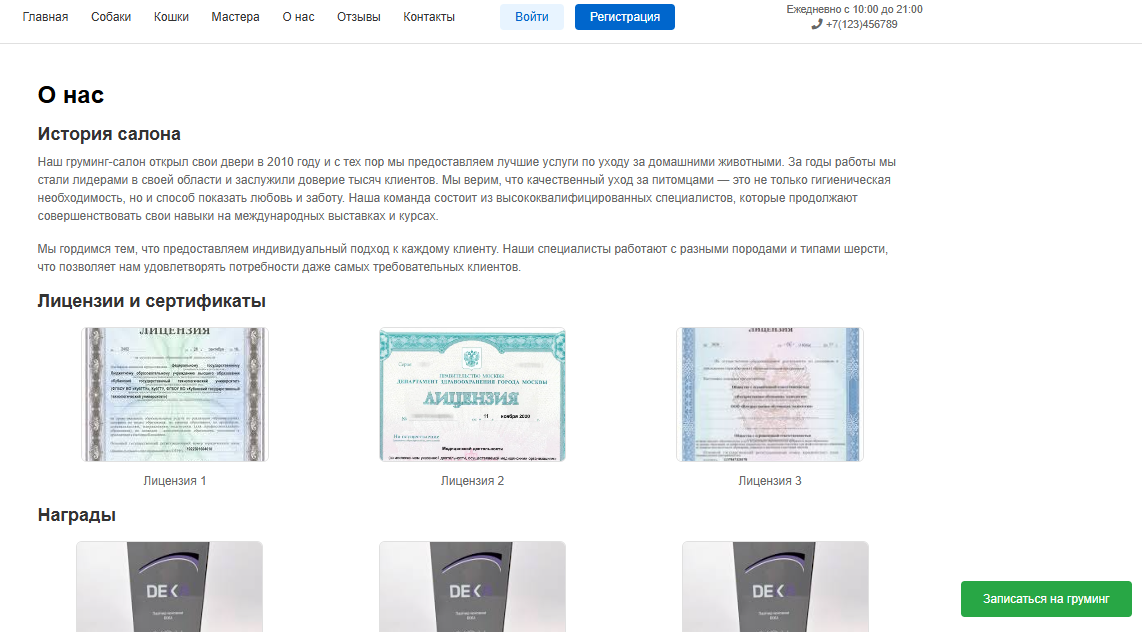


Рисунок 55 – страница «О нас»

На рисунке 56 изображена страница с отзывами:

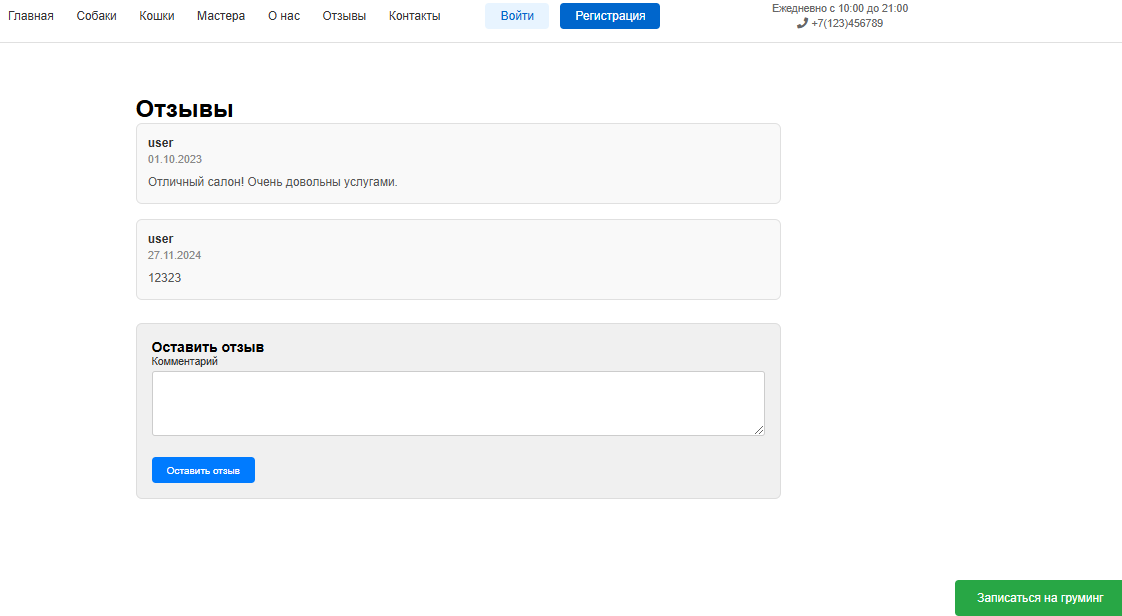


Рисунок 56 – страница «Отзывы»

На рисунке 57 изображена страница с контактными данными салона:

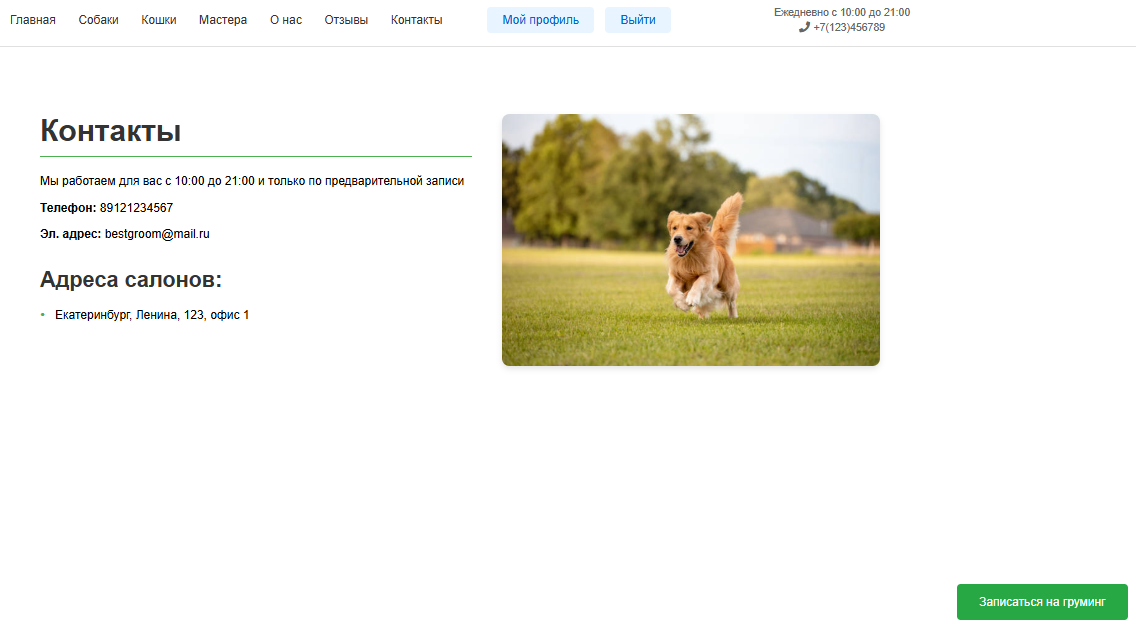


Рисунок 57 – страница «Контакты»

# **3.4.3 Структура разработанного проекта**

Модуль, разработанный в рамках выпускной квалификационной работы, структура которого представлена на рисунке 58 создавался в PyCharm с использованием Django фреймворка. На рисунке 58 изображены папки проекта: grroming, main и templates

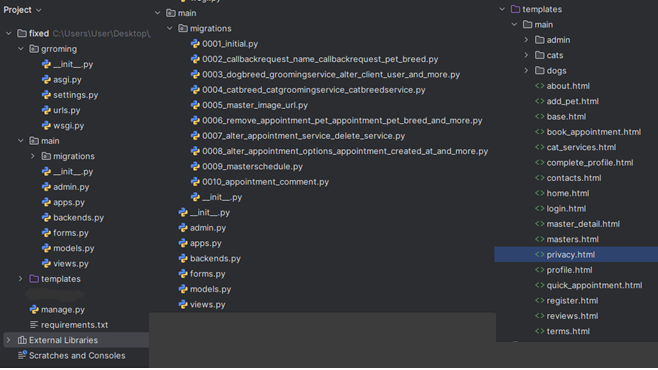


Рисунок 58 - Структура проекта

Папка grooming – основной пакет приложения. В ней находится файл инициализации пакета \_\_init\_\_.py, файл конфигурации ASGI asgi.py, файл, создаваемый по умолчанию settings.py с настройками проекта и в котором реализовано подключение к базе данных, файл urls.py с основными URL-маршрутами сайта с привязкой к views.py, файл wsgi.py с конфигурацией WSGI.

В папке main реализовано основное приложение. В нем есть также файл \_\_init\_\_.py , admin.py с настройками admin-панели, в которую переносятся (регистрируются) все созданные классы моделей models.py для дальнейшего редактирования данных с помощью форм, apps.py с конфигурацией приложения, backends.py с реализацией авторизации клиента с проверкой на корректность ввода учетных данных, forms.py с реализацией форм для ввода данных, models.py с сущностями базы данных, views.py – с представлениями, реализацией функций файла моделей. Также, в данной папке имеется папка migrations.py с файлами миграций базы данных.

В папке temaplates представлены html-шаблоны, отвечающие за разметку страниц веб-платформы.

manage.py – реализация утилиты управления проектом, отвечает за то, что main будет запускаться по умолчанию.

# **3.4.4 Листинги**

В рамках выпускной квалификационной работы было разработано множество функций. Одной из ключевых функций является функция записи на услуги груминг-салона book\_appointment, реализация которой представлена на Листинге 1. Данная функция контролирует логику записи клиентов, предлагает доступное время мастеров для записи и обрабатывает формы.

Листинг 1. Fixed/main/view.py

@login\_required

def book\_appointment(request):

    client = Client.get\_or\_create\_client(request.user)

    if not client.phone:

        messages.warning(request, 'Пожалуйста, заполните свой профиль для продолжения')

        return redirect('complete\_profile')

    if request.method == 'POST':

        post\_data = request.POST.copy()

        datetime\_value = post\_data.get('datetime')

        master\_id = post\_data.get('master')

        selected\_services\_ids = post\_data.getlist('services')

        if datetime\_value and master\_id:

            datetime\_obj = datetime.strptime(datetime\_value, '%Y%m%d%H%M')

            post\_data['date'] = datetime\_obj.date()

Продолжение Листинга 1

            post\_data['time'] = datetime\_obj.time()

            # Вычисляем общую длительность выбранных услуг

            selected\_services = GroomingService.objects.filter(id\_\_in=selected\_services\_ids)

            total\_duration = sum(service.duration for service in selected\_services)

            time\_end = datetime\_obj + timedelta(minutes=total\_duration)

            # Проверяем занятость мастера

            existing\_appointments = Appointment.objects.filter(

                master\_id=master\_id,

                date=datetime\_obj.date(),

                status\_\_in=['pending', 'confirmed', 'completed']

            )

           for appointment in existing\_appointments:

                # Вычисляем время окончания существующей записи

                appointment\_services = appointment.services.all()

                appointment\_duration = sum(service.duration for service in appointment\_services)

appointment\_end\_time = datetime.combine(appointment.date, appointment.time) + timedelta(minutes=appointment\_duration)

                # Проверяем пересечение времён

     if not (time\_end.time() <= appointment.time or datetime\_obj.time() >= appointment\_end\_time.time()):

                    messages.error(request, 'Выбранное время для этого мастера уже занято. Пожалуйста, выберите другое время или другого мастера.')

                    return redirect('book\_appointment')

form = AppointmentForm(post\_data)

if form.is\_valid():

            appointment = form.save(commit=False)

            appointment.client = client

            appointment.save()

            appointment.services.set(selected\_services)

            messages.success(request, 'Запись успешно создана!')

            return redirect('home')

else:

            messages.error(request, 'Пожалуйста, исправьте ошибки в форме')

    # Получаем текущее время

Продолжение Листинга 1

current\_datetime = datetime.now()

    available\_slots = []

    # Получаем всех мастеров

    masters = Master.objects.all()

    master\_appointments = {}

    # Устанавливаем шаг интервала

    default\_interval = 30  # Шаг в минутах

    # Создаем словарь занятых слотов для каждого мастера

    for master in masters:

        master\_appointments[master.id] = []

        occupied\_slots = Appointment.objects.filter(

            master=master,

            status\_\_in=['pending', 'confirmed', 'completed']

        )

        for appointment in occupied\_slots:

            # Вычисляем время окончания с учетом длительности услуги

            appointment\_services = appointment.services.all()

            appointment\_duration = sum(service.duration for service in appointment\_services)

appointment\_end\_time = datetime.combine(appointment.date, appointment.time) + timedelta(minutes=appointment\_duration)

master\_appointments[master.id].append((appointment.date, appointment.time, appointment\_end\_time.time()))

   # Генерируем базовые временные слоты

    for day\_offset in range(14):

        current\_date = current\_datetime.date() + timedelta(days=day\_offset)

        # Рабочие дни (например, пн-пт)

        if current\_date.weekday() < 5:

            start\_time = time(10, 0)

            end\_time = time(20, 30)

            slot\_time = datetime.combine(current\_date, start\_time)

while slot\_time.time() <= end\_time:

                slot\_end\_time = slot\_time + timedelta(minutes=default\_interval)

                # Проверяем, не пересекается ли слот с занятыми слотами мастера

                is\_free = True

Окончание Листинга 1

for occupied\_slot in master\_appointments.get(master.id, []):

                    occupied\_date, occupied\_start, occupied\_end = occupied\_slot

              if current\_date == occupied\_date and not (slot\_end\_time.time() <= occupied\_start or slot\_time.time() >= occupied\_end):

                        is\_free = False

                        break

                if is\_free:

                    available\_slots.append({

                        'id': slot\_time.strftime('%Y%m%d%H%M'),

                        'datetime': slot\_time.strftime('%d.%m.%Y %H:%M')

                    })

                slot\_time += timedelta(minutes=default\_interval)

    context = {

        'breeds': list(CatBreed.objects.values('id', 'name')) + list(DogBreed.objects.values('id', 'name')),

        'services': GroomingService.objects.all(),

        'salons': Salon.objects.all(),

        'masters': Master.objects.all(),

        'available\_slots': available\_slots,

        'master\_appointments': master\_appointments,

    }

    return render(request, 'main/book\_appointment.html', context)

Описание кода:

1. **Декоратор @login\_required**:  
   Функция защищена декоратором @login\_required, что означает, что доступ к ней возможен только для авторизованных пользователей. Если пользователь не авторизован, он будет перенаправлен на страницу входа.
2. **Получение клиента**

client = Client.get\_or\_create\_client(request.user)

Здесь мы получаем или создаем объект клиента на основе текущего пользователя.

1. **Проверка профиля клиента:**

if not client.phone:

messages.warning(request, 'Пожалуйста, заполните свой профиль для продолжения')

return redirect('complete\_profile')

Если у клиента нет номера телефона, выводится предупреждение, и пользователь перенаправляется на страницу заполнения профиля.

1. **Обработка POST - запроса:**

if request.method == 'POST':

post\_data = request.POST.copy()

Если запрос является POST-запросом, мы копируем данные из формы.

1. **Получение данных из формы:**

datetime\_value = post\_data.get('datetime')

master\_id = post\_data.get('master')

selected\_services\_ids = post\_data.getlist('services')

Извлекаем дату и время записи, ID мастера и выбранные услуги.

1. **Проверка занятости мастера:**

Прежде чем продолжить, мы проверяем, есть ли уже записи для выбранного мастера в указанное время. Если запись пересекается с уже существующей, выводится сообщение об ошибке, и пользователь перенаправляется назад на страницу записи.

1. **Создание записи:**

form = AppointmentForm(post\_data)

if form.is\_valid():

appointment = form.save(commit=False)

appointment.client = client

appointment.save()

appointment.services.set(selected\_services)

Если форма валидна, создается новая запись на услугу, которая связывается с клиентом и выбранными услугами.

1. **Генерация доступных временных слотов**:

Код генерирует доступные временные слоты для записи на основе текущего времени и статуса существующих записей. Слоты создаются для рабочих дней (например, с понедельника по пятницу) с шагом в 30 минут.

1. **Контекст для рендеринга**:

context = {

'breeds': list(CatBreed.objects.values('id', 'name')) + list(DogBreed.objects.values('id', 'name')),

'services': GroomingService.objects.all(),

'salons': Salon.objects.all(),

'masters': Master.objects.all(),

'available\_slots': available\_slots,

'master\_appointments': master\_appointments,

}

Все необходимые данные передаются в шаблон для рендеринга: породы животных, доступные услуги, салоны, мастера и доступные временные слоты.

1. **Рендеринг шаблона**:

return render(request, 'main/book\_appointment.html', context)

В конце функция возвращает рендеринг шаблона с переданным контекстом.

Таким образом, функция book\_appointment реализует логику записи в груминг-салон, включая проверку профиля клиента, обработку формы записи, проверку занятости мастеров и генерацию доступных временных слотов. Функция также учитывает возможность обработки ошибок и информирует пользователя о статусе его запроса.

**Выводы по 3 главе**

Для проектируемой веб-платформы была выбрана клиент-серверная архитектура, основанная на паттерне проектирования MVT (Model-View-Template). Этот паттерн разделяет приложение на три основных компонента: модель (Model), представление (View) и Шаблон (Template), что обеспечивает логическую раздельность и удобство разработки.

Для хранения данных приложения была выбрана база данных MySQL, которая обеспечивает надежность, безопасность и высокую производительность системы. Для реализации серверной части веб-платформы был выбран высокоуровневый язык программирования Python с использованием фреймворка Django, который предоставляет множество инструментов и функций для быстрой и удобной разработки веб-приложения.

Для разработки пользовательского интерфейса платформы были выбраны JavaScript, CSS и HTML являющиеся мощными и гибкими инструментами для создания интерактивных пользовательских интерфейсов, разметки сайта и их дизайна.

Для определения требований и визуализации интерфейса веб-платформы было решено выбрать сайт <https://wireframe.cc/>, который позволяет создавать скетчи интерфейса, что помогает определить структуру и компоненты пользовательского интерфейса ещё на ранних этапах разработки.

# 

# **Заключение**

В процессе выполнения данной выпускной квалификационной работы были проанализированы многие уже имеющиеся сервисы, и исходя из их недостатков и достоинств, было принято решение о создание веб-платформы, которая сможет удовлетворить потребности пользователя.

Основная цель работы заключалась в упрощении и ускорении процесса записи в груминг-салон «Домовенок».

Для достижения этой цели были поставлены и выполнены следующие задачи:

**Задача проектирования и реализации самостоятельной записи в груминг-салон:** на основе анализа существующих решений на рынке по части пользовательских интерфейсов был разработан удобный и ориентированный на пользователя интерфейс, позволяющий клиентам легко и удобно записываться в груминг-салон с подключением к нему базы данных для хранения и предоставления информации о салоне, его услугах, отзывах и мастерах, а также был разработан алгоритм, позволяющий клиентам самостоятельно выбирать в системе услугу, дату, время и мастера для записи из доступных на текущий момент.

**Задача проектирования и реализации отправки заявки на обратный звонок:** был разработан удобный пользовательский интерфейс для отправки заявки администратору груминг-салона на обратный звонок. Также был сделан механизм передачи данных из заявки с контактами клиента и информацией в панель администратора для последующей его связи с клиентом.

**Задача проектирования и реализации публикации отзывов:** был создан удобный пользовательский интерфейс для предоставления пользователям возможности оставлять отзывы о салоне, также администратору системы предоставлены права модератора отзывов, а неавторизованным пользователям был ограничен доступ на оставление отзывов.

**Задача проектирования и реализации личного кабинета администратора:** был реализован личный кабинет администратора с удобным интерфейсом и правами на просмотр и редактирование информацией из базы данных (записи, другие пользователи, мастера и др.), сброса пароля клиенту.

**Проектирование и реализация личного кабинета пользователя:** был разработан личный кабинет пользователя с удобным интерфейсом с возможностью просмотра и редактирования личных данных, удаления своего аккаунта, совершения записи на услуги, оставления отзывов, отмены записей. Также всем пользователям предоставлена возможность совершать регистрацию в системе и авторизовываться.

**Задача проектирования и разработки отображения полезной информации салона на сайте:** реализация отображения информации об услугах салона, мастерах, отзывах, информации о салоне и его контактах для всех пользователей (в т.ч. и для неавторизованных) с помощью удобного интерфейса. Реализовано разделение услуг салона на услуги для кошек, собак, с разделением на категории пород животных (мелкие/средние/крупные – для собак и короткошерстные/средне шёрстные/длинношерстные – для кошек) для удобного поиска нужной породы животного с предлагаемыми для нее услугами и рекомендациями по уходу на сайте.

Созданная веб-платформа предоставляет пользователям удобный и быстрый инструмент для самостоятельной записи в груминг-салон. Она объединяет функционал онлайн записи, отправки заявки на обратный звонок, публикации отзывов, регистрации и авторизации в личном кабинете, а также просмотр полезной информации о салоне и его услугах. Благодаря этому груминг-салон сможет увеличить количество довольных клиентов и свою прибыль.

В дальнейшем в данном проекте будет реализовано следующее:

- добавление функционала чата с пользователями

- интеграция с социальными сетями

- интеграция с платежными системами для оплаты услуг онлайн

- возможность покупки подарочного сертификата

- адаптация платформы под мобильные устройства для удобства пользователей

- галерея работ

- раздел с акциями, скидками и специальными предложениями

- возможность подписки на рассылку новостей и акций

- инструменты для анализа посещаемости, популярности услуг и работы грумеров

- раздел FAQs с ответами на часто задаваемые вопросы

- программы лояльности и скидок для постоянных клиентов

# **Список используемых источников:**

1. Егорян В. В., Калугин А. В. РОЛЬ АРХИТЕКТУРНОГО ПОДХОДА ПРИ РАЗРАБОТКЕ ПРОГРАММНЫХ ПРИЛОЖЕНИЙ // Столыпинский вестник. 2022. №9. URL: https://cyberleninka.ru/article/n/rol-arhitekturnogo-podhoda-pri-razrabotke-programmnyh-prilozheniy (дата обращения: 01.11.2024).

2. Методология Питера Чена / [Электронный ресурс] [URL: https://sites.google.com/site/anisimovkhv/learning/pris/lecture/tema7/tema7\_5] (Дата обращения 01.11.2024).

3. Фаулер М. UML. Основы, 3-е издание. – Пер. с англ. – СПб: Символ-Плюс, 2004. – 126 с.

4. Фаулер М. UML. Основы, 3-е издание. – Пер. с англ. – СПб: Символ-Плюс, 2004. – 124 с

5. Бистерфельд О.А. Методология функционального моделирования IDEF0: учебно-методическое пособие / О.А. Бистерфельд; Ряз. гос. ун-т им. С.А. Есенина. - Рязань, 2008. - 48 с.

6. Ларин, С. Н. Модели, методы, показатели, характеристики и метрики, применяемые в экспертных системах оценки качества разработки и создания инновационных программных проектов / С. Н. Ларин, Л. Ю. Лазарева, Т. С. Ларина // Региональная экономика: теория и практика. – 2017. – Т. 15, № 6(441). – С. 1189-1190. – DOI 10.24891/re.15.6.1187. – EDN YRRKSN.

7. Объясни значение Model View Teamplay (MVT) [Электронный ресурс] // easyoffer. URL : <https://easyoffer.ru/question/534> (дата обращения: 01.11.2024)

8. Фаулер М. UML. Основы, 3-е издание. – Пер. с англ. – СПб: Символ-Плюс, 2004. – 62 с.

9. НОТАЦИЯ DFD [Электронный ресурс] // proCoding. URL: https://pcoding.ru/gost/dfd.pdf (дата обращения: 01.11.2024)