2.1 Проектирование моделей данных и бизнес-процессов платформы 2.1.1 UML-диаграмма прецедентов

Диаграмма прецедентов - это инструмент в разработке программного обеспечения, который помогает визуализировать взаимодействие между актерами (пользователями или внешними системами) и функциональностью системы, представленной в виде прецедентов (сценариев использования). Эта диаграмма является частью модели прецедентов, которая используется для описания функций и потребностей системы на концептуальном уровне.

Главная цель диаграммы прецедентов - понять, как пользователи будут взаимодействовать с системой, какие действия они могут выполнить, и какие результаты ожидать. Она помогает определить основные сценарии использования системы и выделить ключевые функциональные возможности.

На диаграмме прецедентов актеры представляются как внешние сущности, которые инициируют прецеденты, то есть определенные действия или сценарии в системе. Прецеденты, в свою очередь, описываются как конкретные действия или операции, которые могут быть выполнены в системе, обычно в ответ на запросы актеров.

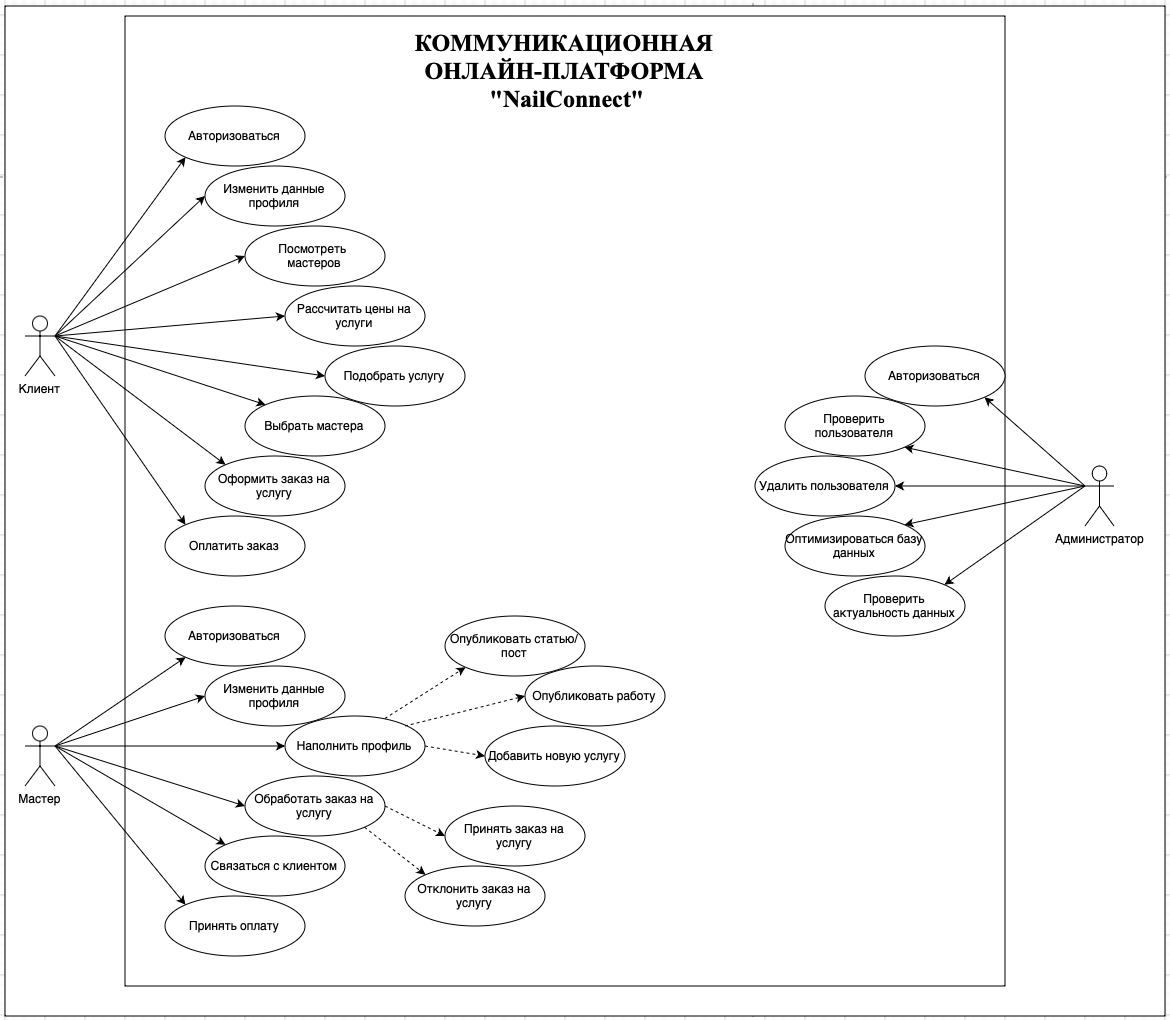


Диаграмма прецедентов интернет-портала «NailConnect»

Разрабатываемая система предполагает несколько уровней доступа

Для пользователей разных ролей предусмотрен разный функционал.

Система обслуживания маникюра предоставляет разнообразный функционал для пользователей с разными уровнями доступа. Вот как это работает:

1. **Заказчик**: Зарегистрированный пользователь с доступом к расширенным функциям. У заказчика есть свой личный кабинет, где он может управлять своими данными, оставлять отзывы, а также просматривать информацию о заказах. В своем личном кабинете заказчик может оформить услугу и получить контактные данные мастера маникюра. До совершения заказа заказчик может изучить профили мастеров, их услуги и даже оценить приблизительную стоимость услуги. Заказчик также имеет возможность комментировать публикации и рецепты мастеров.
2. **Мастер**: Также зарегистрированный пользователь с расширенными возможностями. В своем личном кабинете мастер может редактировать свой профиль и управлять своими услугами и публикациями: добавлять, редактировать и удалять. Он также может отслеживать заказы клиентов, принимать или отклонять их, а также вести учет своих расходов и доходов. Кроме того, мастер имеет возможность просматривать отзывы от клиентов.
3. **Администратор**: Это пользователь с наивысшими привилегиями. В его полномочия входит доступ ко всем таблицам базы данных с возможностью оптимизации данных. Он также ответственен за проверку пользователей и может при необходимости удалять профили за нарушение правил поведения на платформе.
4. **Незарегистрированный пользователь (гость)**: Это посетитель платформы с ограниченным функционалом. Он может просматривать информацию на сайте, но не имеет доступа к расширенным функциям, таким как оформление заказов или добавление комментариев. Однако у гостя есть возможность зарегистрироваться на платформе для получения расширенного доступа к функциям.

2.1.2 BPMN-диаграммы описания бизнес-процессов интернет-площадки

BPMN (Business Process Model and Notation) - это инструмент выбран для создания более подробного обзора бизнес-процессов на площадках. Он помогает визуализировать и описать бизнес-процессы в виде графической схемы, что упрощает понимание последовательности действий, вовлеченных ролей участников, событий и передачу данных внутри этих процессов.

Эти диаграммы могут быть использованы как внутри компании для оптимизации рабочих процессов, так и для взаимодействия с клиентами и другими сторонами. Например, они могут использоваться для документирования процессов взаимодействия с клиентами, что помогает ясно представить этапы обработки заказов или решения проблем клиентов.

Преимущества использования BPMN включают в себя возможность стандартизации процессов, что способствует более эффективному управлению и оптимизации бизнес-процессов. Он также обеспечивает ясное представление о ролях и ответственностях участников процесса, что упрощает согласование и сотрудничество внутри компании и с внешними партнерами.

Диаграмма на Рисунке описывает автоматизированный процесс оформления заказа через интернет-портал, в котором заказчик взаимодействует с интернет-порталом. Процесс включает в себя ряд этапов, начиная от выбора мастера и заканчивая решением по заказу.

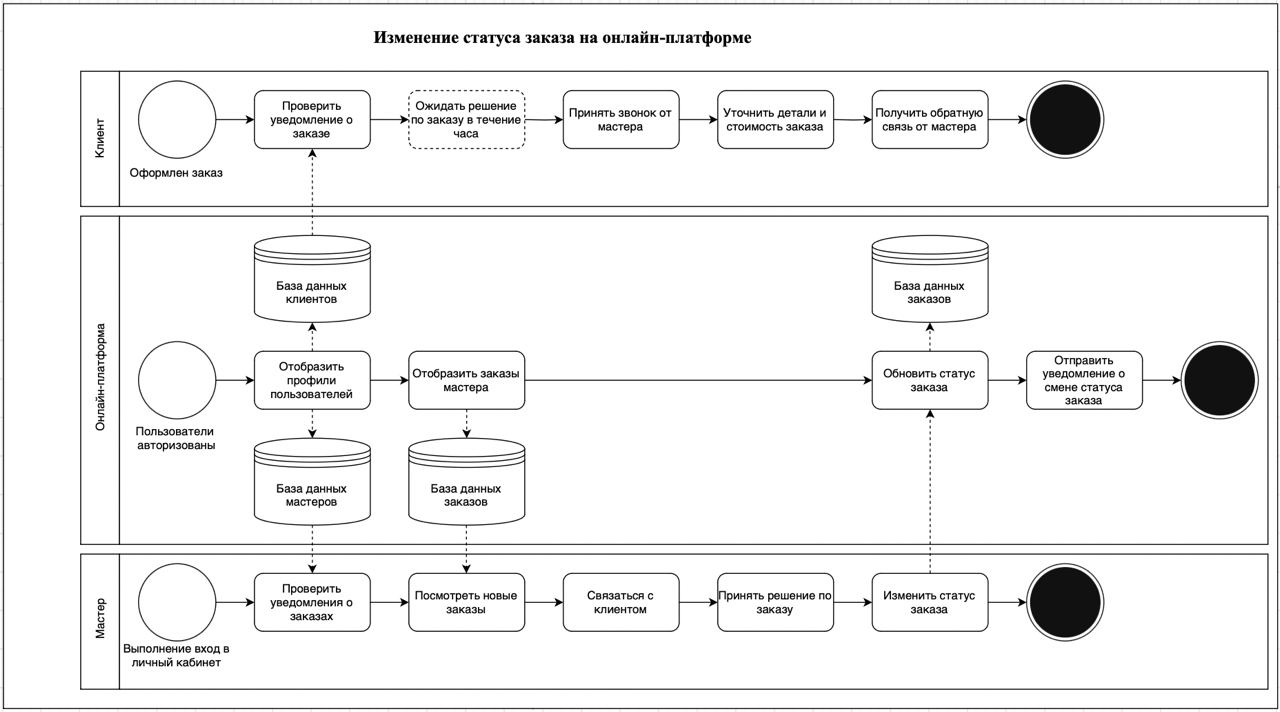


Диаграмма в нотации BPMN бизнес-процесса «Оформление заказа»

Процесс оформления заказа начинается с авторизации заказчика на платформе. Каждое действие заказчика взаимодействует с интернет-площадкой. Например, когда заказчик посещает страницу мастеров, интернет-площадка обращается к базе данных для получения информации о мастерах и возвращает ее заказчику, чтобы он мог выбрать нужного мастера. То же самое происходит на странице мастера, где заказчик выбирает торт и рассчитывает его стоимость.

Для оформления заказа заказчик заполняет поля формы заказа, и платформа записывает данные заказчика в базу данных. Затем интернет-площадка направляет уведомление о новом заказе мастеру. Когда мастер получает уведомление, он может просмотреть заказ, связаться с заказчиком и принять решение относительно заказа: принять его или отклонить.

Была разработана диаграмма в нотации BPMN для процесса принятия и изменения статуса заказа. Этот процесс показан на Рисунке 18. В рамках данного процесса происходит взаимодействие между мастером и клиентом с использованием интернет-портала.

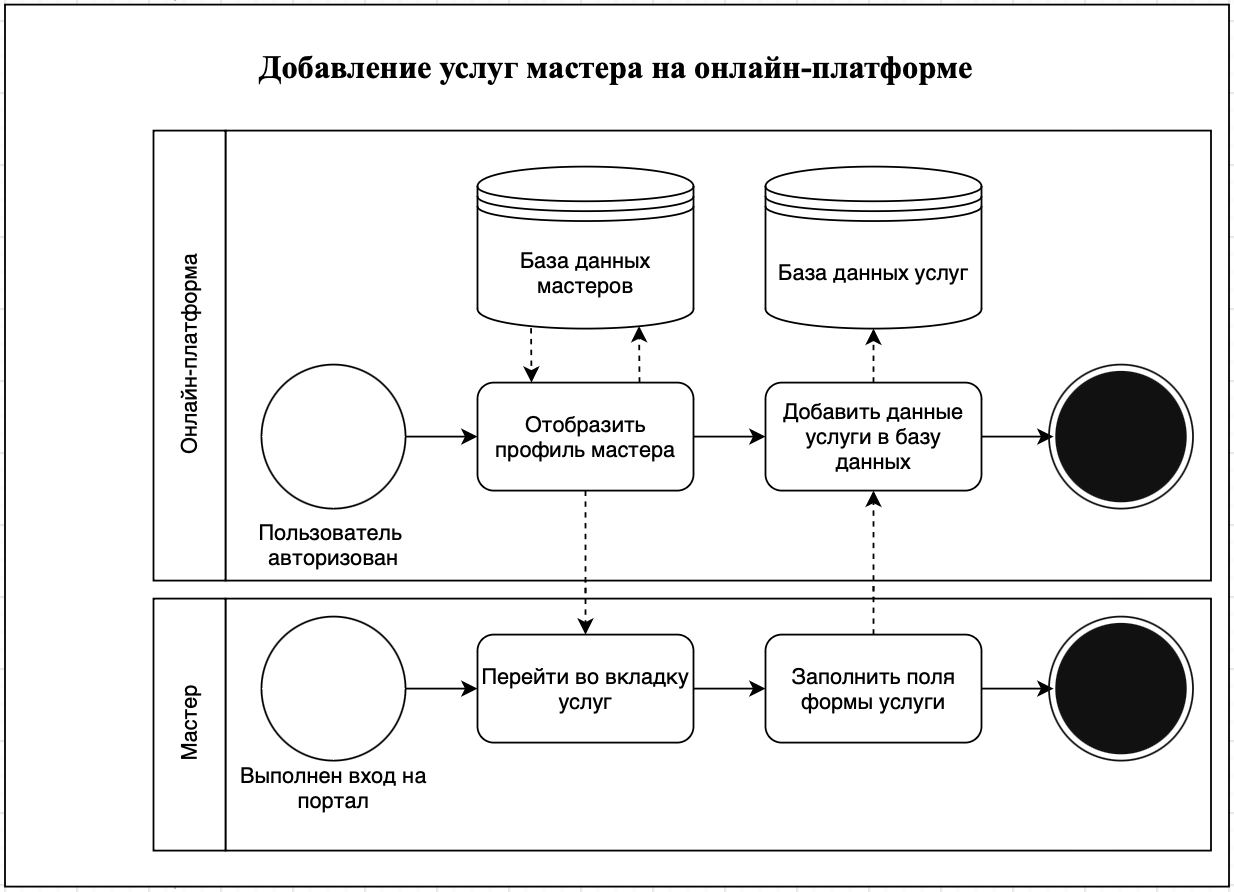


Процесс оформления заказа на интернет-платформе инициируется в случае, если каждый пользователь прошел процедуру аутентификации и заказчик решил разместить заказ. Этот процесс включает в себя несколько шагов, начиная от отображения личных кабинетов клиента и мастера на платформе до изменения статуса заказа и обновления информации в базе данных.

При поступлении заказа платформа обращается к базе данных для получения необходимой информации о заказе и уведомляет как заказчика, так и мастера о новом заказе. В это время, пока заказчик ожидает обновления статуса заказа, мастер анализирует информацию о заказе, предоставленную платформой и полученную из базы данных. После этого мастер связывается с заказчиком для уточнения деталей заказа и принимает решение о его выполнении.

После принятия решения мастер изменяет статус заказа на платформе, что приводит к обновлению информации в базе данных. После этого заказчику отправляется уведомление о новом статусе заказа в его личный кабинет на платформе.

Кроме того, была разработана диаграмма BPMN для еще одного бизнес-процесса: добавления новой услуги мастера. Этот процесс также включает взаимодействие между мастером и интернет-порталом.



2.1.3 DFD диаграмма потока данных

На Рисунке 21 представлена декомпозиция DFD-диаграммы, изображен подробный процесс формирования контента интернет-портала. Можно отметить основные действия принятия, обработки и выгрузки контента на сайт.



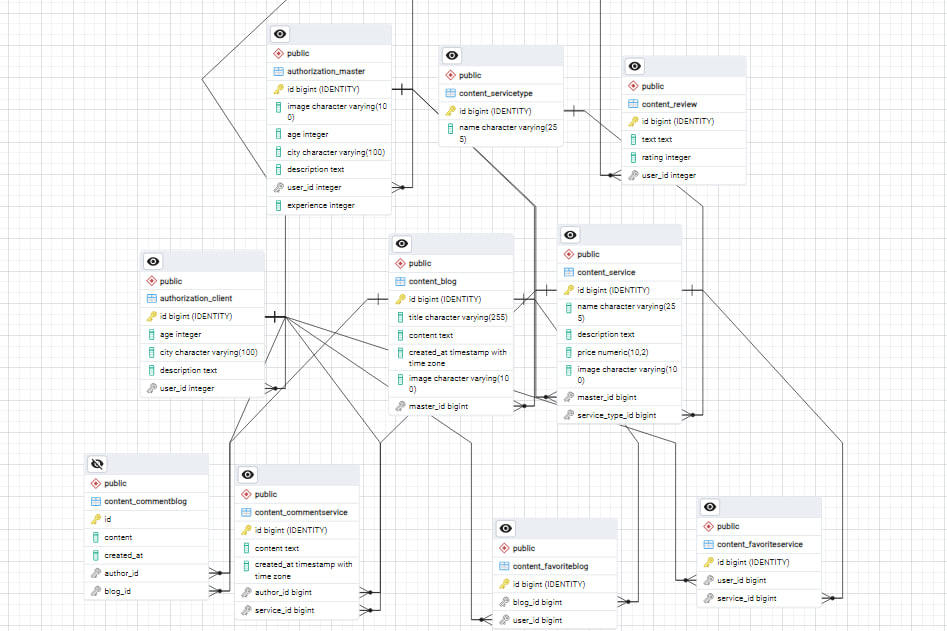
При анализе процесса формирования контента на платформе становится ясно, как он проходит через различные этапы и взаимодействия. Например, когда пользователь добавляет контент через платформу, система отправляет заявку или уведомление модератору о новом добавлении. После получения этих данных модератор начинает обработку контента в соответствии с установленными требованиями и стандартами.

Процесс обработки может включать в себя различные действия, такие как проверка контента на соответствие правилам использования и качеству, редактирование для улучшения читабельности или визуального оформления, а также добавление тегов или метаданных для упрощения поиска и категоризации.

После обработки контент отправляется в базу данных, где он хранится и отображается на онлайн-платформе для доступа пользователям. Таким образом, пользователи сами создают контент для платформы, а модераторы обрабатывают его, чтобы обеспечить соответствие установленным стандартам и требованиям, что в конечном итоге способствует корректному и качественному отображению на площадке.

2.2 Проектирование базы данных интернет-портала

После определения и проектирования основных бизнес-процессов и определения необходимого функционала для разных ролей онлайн-площадки была разработана ER-диаграмма базы данных. ER-модель данных – это диаграмма, которая описывает логическую структуру базы данных. Ее создание очень важно при проектировании базы данных, так как она позволяет лучше понять, как связаны между собой различные таблицы и какие данные необходимо хранить в каждой из них. ER-модель состоит из сущностей, атрибутов и связей между ними. Сущности представляют собой объекты, которые нужно хранить в базе данных, например, услуги, клиенты, мастера и т.д. Каждая сущность имеет свой уникальный идентификатор, который называется первичным ключом. Атрибуты – это свойства сущностей, которые необходимо хранить в базе данных, например, имя клиента, его почта и телефон. Связи между сущностями показывают, какие связи существуют между объектами и какие данные необходимо связать в базе данных. ER модель данных позволяет лучше понять, как хранятся данные в базе данных, какие таблицы необходимо создать и как они будут связаны между собой. Такая диаграмма также позволяет определить, какие данные необходимо хранить в каждой из таблиц, какие связи между ними существуют и какие ограничения нужно наложить на эти данные

.

На Рисунке изображена ER-диаграмма разрабатываемой онлайн площадки. Она содержит 10 сущностей: клиенты, мастера, услуги, блоги, комментарии к блогам, комментарии к услугам, избранные услуги, избранные блоги, отзывы, типы услуг. Самой быстро наполняемой является таблицам мастеров и услуг. Также в таблице мастеров содержится больше всего атрибутов. Таблица мастеров включает в себя следующие атрибуты: индивидуальный номер мастера, который идентифицирует каждую запись и является уникальным для каждой записи, также этот атрибут является первичным ключом для своей таблицы и внешним ключом для таблицы отзывов и заказов. Далее в таблице мастеров есть атрибуты: имя мастера, его город, страна, номер телефона, описание и фото, опыт. У таблицы также есть внешние ключи, такие как: индивидуальный номер услуг, индивидуальный номер блогов, индивидуальный номер публикаций

Логическая модель данных, которая была разработана, представляет собой важный этап в процессе проектирования базы данных. На основе этой модели создается физическая база данных, которая является реальной реализацией представленной логической модели. Физическая база данных состоит из конкретных файлов и структур, которые хранят и обрабатывают данные в реальном времени.

Проектирование физической базы данных включает в себя не только определение таблиц и столбцов, но и разработку различных объектов базы данных, таких как индексы, ограничения и другие. Кроме того, в этом процессе учитывается определение физических атрибутов, таких как размерность, типы данных, а также оптимизация индексов для обеспечения эффективного доступа к данным.

Кроме того, физическое проектирование базы данных включает в себя рассмотрение факторов хранения данных на диске, таких как размещение данных на диске для оптимизации производительности и обеспечения безопасности. Важным аспектом является также управление пространством на диске и оптимизация процессов резервного копирования и восстановления данных.

Таким образом, проектирование физической базы данных требует детального анализа и учета различных аспектов для создания эффективной и надежной системы хранения и обработки данных.

Для удобства взаимодействия с базой данных будут использоваться операции CRUD. CRUD – это аббревиатура, которая означает C (Создание), R (Чтение), U (Обновление) и D (Удаление). CRUD-модель – это основной набор операций, которые могут быть выполнены с данными в базе данных.

Create (Создание) – операция, которая позволяет создавать новые записи в базе данных. Например, создание нового пользователя в базе данных.

Read (Чтение) – операция, которая позволяет читать данные из базы данных. Например, чтение списка пользователей из базы данных.

Update (Обновление) – операция, которая позволяет обновлять данные в базе данных. Например, обновление информации о пользователе в базе данных.

Delete (Удаление) – операция, которая позволяет удалять данные из базы данных. Например, удаление пользователя из базы данных.

CRUD-модель широко используется в веб-разработке для создания приложений, которые работают с данными. Например, веб-приложение для управления списком пользователей может использовать операции CRUD для создания новых пользователей, чтения списка существующих пользователей, обновления информации о пользователях и удаления пользователей из базы данных.

В веб-разработке для реализации операций CRUD обычно используются HTTP-методы: POST для создания записей, GET для чтения данных, PUT или PATCH для обновления данных и DELETE для удаления записей. Также для удобства работы с данными обычно используются формы и интерфейсы для взаимодействия с базой данных.

CRUD-модель является одним из основных принципов проектирования баз данных и позволяет упростить работу с данными в приложениях, а также повысить эффективность разработки.

Createпозволяет добавлять новые строки в таблицу. На примере регистрации пользователя сформирован запрос: INSERT INTO users VALUES

('user.name', 'user.email, 'user.login, 'user.password, 'user.role,)

Readпозволяет извлекать определенные записи и считывать их значения. На примере вывода всех пользователей с именем Елена был написан запрос: SELECT user\_id, name FROM users WHERE (name = 'Елена')

Updateпозволяет изменять существующую запись в таблице. На примере изменения пользователем почты был написан запрос: UPDATE users SET email = ‘valer03@gmail.com’ WHERE user\_id = 23

Deleteпозволяет удалить запись из таблицы. На примере удаления пользователя администратором был написан запрос: DELETE FROM users

WHERE user\_id = 23

2.3 Создание графического контента интернет-площадки

После завершения проектирования бизнес-процессов и базы данных следующим важным шагом является проектирование дизайн-макета пользовательского интерфейса. Дизайн-макет позволяет визуализировать и представить, как будет выглядеть и функционировать проект, что позволяет провести тестирование уже на ранних этапах разработки. Создание дизайн-макета является важным этапом, так как это помогает определить функциональность и компоненты системы более точно, что в свою очередь ускоряет процесс разработки.

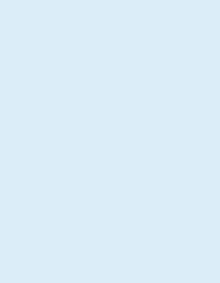
Перед началом разработки дизайн-макета необходимо провести ряд этапов, которые включают выбор цветовой гаммы, определение подходящих шрифтов, разработку логотипа и фирменного стиля проекта. Кроме того, требуется подбор или создание статического графического контента для сайта. Эти шаги не только способствуют созданию привлекательного дизайна, соответствующего бренду, но и формированию единого и узнаваемого образа проекта, что является важным для успешного его продвижения и взаимодействия с пользователями.

Перед тем, как приступить к созданию дизайна сайта, важно тщательно подобрать цветовую гамму. Цвета играют значительную роль, так как они могут создавать определенное настроение и ассоциироваться с содержанием проекта, а также оказывать влияние на эмоциональное состояние пользователей. Цвета должны вызывать определенные ассоциации у посетителей и погружать их в атмосферу проекта.

Для определения цветовой гаммы мы анализировали, какие ассоциации возникают у людей при словах, связанных с нашей тематикой. Hex-коды цветов представлены на соответствующем рисунке для удобства использования.



EDD99C



DBEBF8



#FFC0CB

Цветовая гамма веб-сайта, посвященного маникюру, играет ключевую роль в создании атмосферы, привлекающей посетителей и соответствующей целям проекта. Рассмотрим каждый из предложенных цветов и их применимость к сайту маникюра:

1. **EDD99C (желтый оттенок):**
   * Этот оттенок желтого ассоциируется с солнечным светом и энергией, что может подчеркнуть яркость и радость, связанные с уходом за собой.
   * Желтый также ассоциируется с радостью и оптимизмом, что может создать позитивное впечатление и привлечь внимание посетителей.
   * В контексте маникюра, этот цвет может ассоциироваться с веселыми и яркими дизайнами ногтей, добавляя ощущение лета и солнечного настроения.
2. **DBEBF8 (голубой оттенок):**
   * Голубой цвет ассоциируется с чистотой, прохладой и спокойствием, что может подчеркнуть профессионализм и надежность услуг маникюра.
   * Он также может создать ощущение умиротворения и расслабления, что важно для атмосферы ухода и красоты.
   * Голубой цвет может использоваться для подчеркивания чистоты и свежести ногтей, а также создания впечатления комфорта и безопасности в салоне красоты.
3. **#FFC0CB (розовый оттенок):**
   * Розовый цвет ассоциируется с женственностью, нежностью и красотой, что идеально подходит для сайта, посвященного маникюру и красоте рук.
   * Этот цвет может создать романтическую атмосферу и подчеркнуть элегантность и изысканность предоставляемых услуг.
   * В контексте маникюра, розовый цвет может ассоциироваться с разнообразием дизайнов, включая деликатные узоры и легкие оттенки, что привлечет внимание любителей красоты и стиля.

Эти цвета хорошо дополняют друг друга, создавая гармоничную и привлекательную цветовую гамму, которая подчеркивает эстетику и профессионализм сайта маникюра. Они помогут создать уютную и красивую атмосферу, что важно для привлечения посетителей и установления доверительных отношений с клиентами.

РАЗРАБОТКА ЛОГОТИПА

На сегодняшний день современный логотип, который состоит из контуров и имеет больше минималистичных элементов, может стать залогом узнаваемости и успеха бренда. Логотип часто выступает в роли дизайнерского решения любого проекта. Основная задача логотипа заключается в том, чтобы отражать суть компании, ее деятельность, настрой и, конечно же, формировать бренд. Следующим шагом в разработке фирменного стиля проекта является создание логотипа. Важно обозначить, какого формата необходим логотип, что он будет отображать, будет ли это просто текст или графический элемент, или же все вместе. Современный логотип должен соответствовать меркам сферы графического дизайна. Зная концепцию проекта и его фирменные цвета, можно создать отличный вариант логотип

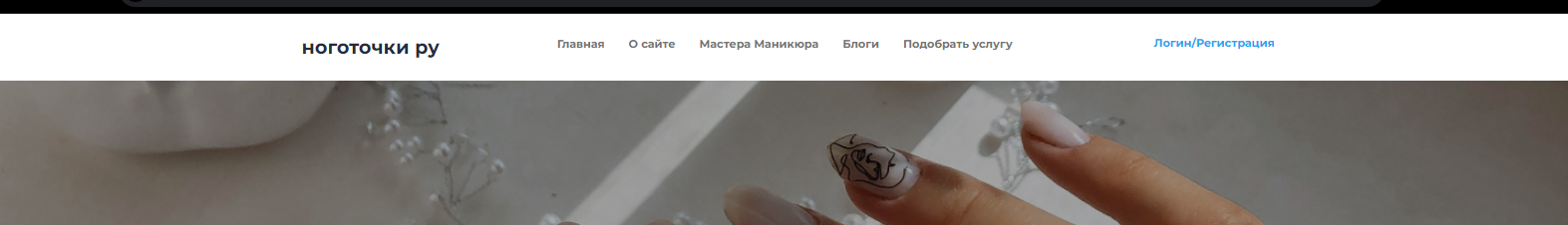


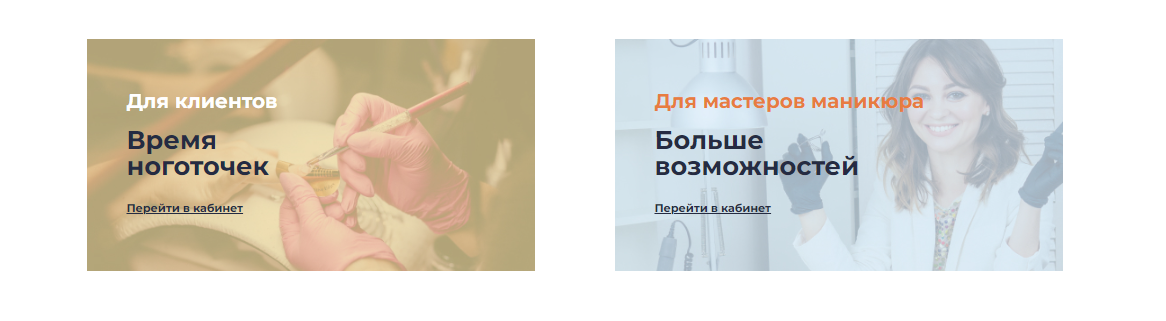
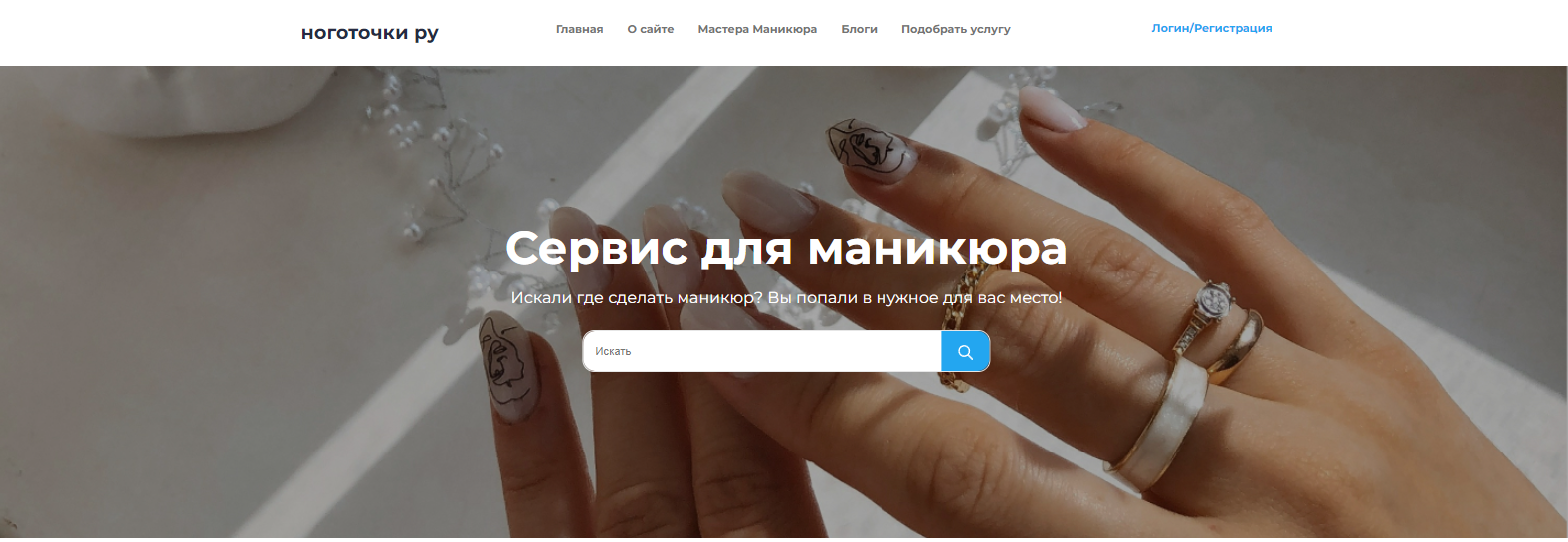
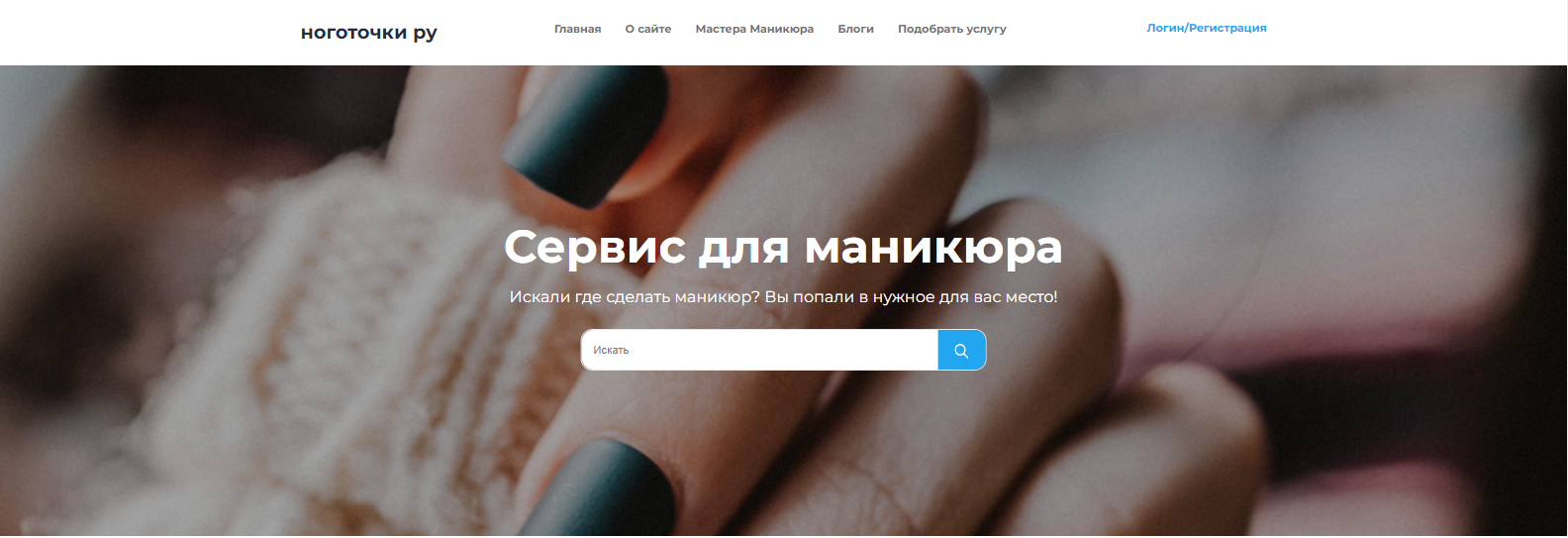
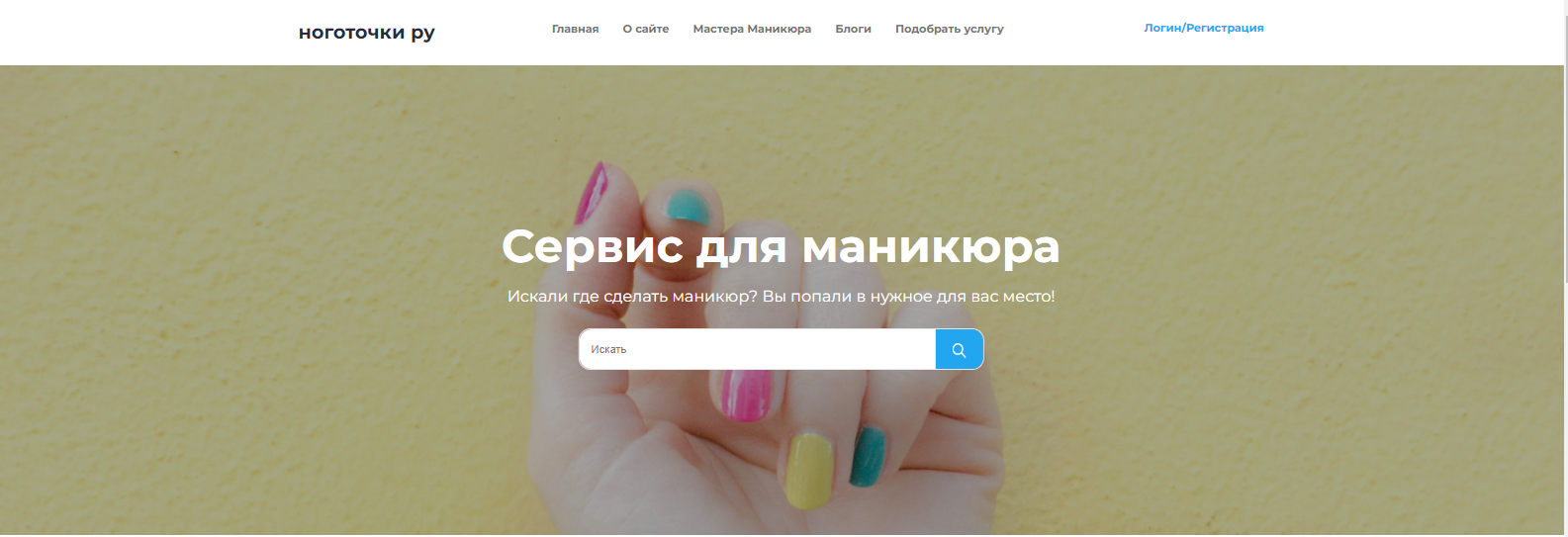
2.4 Создание дизайн-макета пользовательского интерфейса проекта

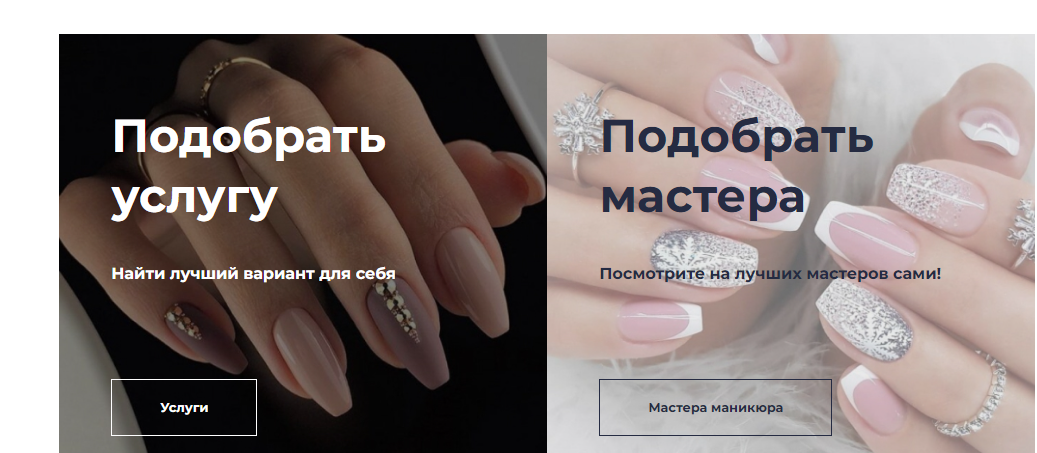
После того, как фирменный стиль, следующий шаг – это проектирование самого дизайн-макета. Средой проектирования дизайн-макета была выбрана онлайн-версия программы Figma.

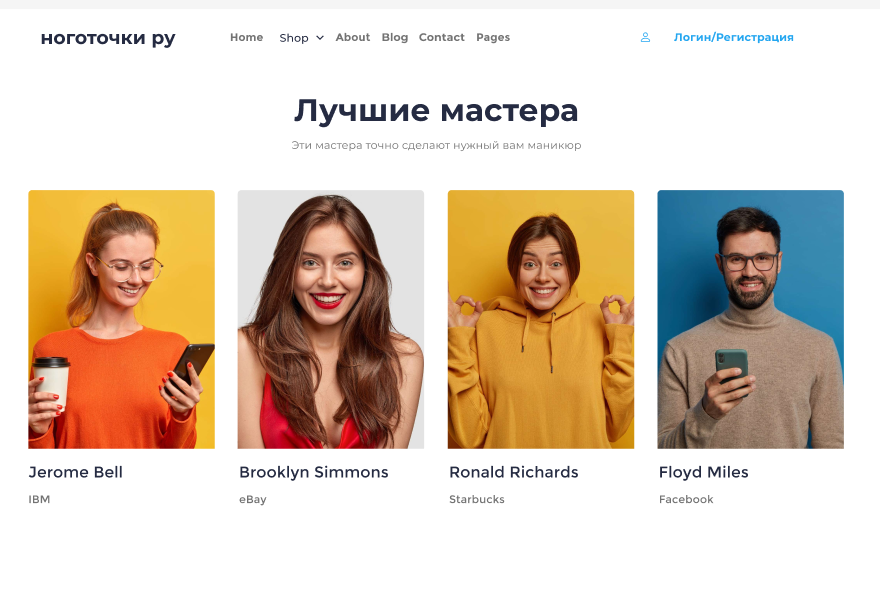
Проектирование макета целесообразно начинать с главной страницы, которая встречает пользователя. Ее внешний вид и функционал должны обратить на себя внимание, и пользователь должен заинтересоваться в изучении портала.

На главной странице размещена шапка сайта с навигацией (рисунок) слайдер с информацией и формой для поиска(Рисунок 26), после чего идет блок для перехода в личный кабинет как для пользователей, так и для мастеров (Рисунок), потом идет навигация для страниц Услуги и Мастера (Рисунок)







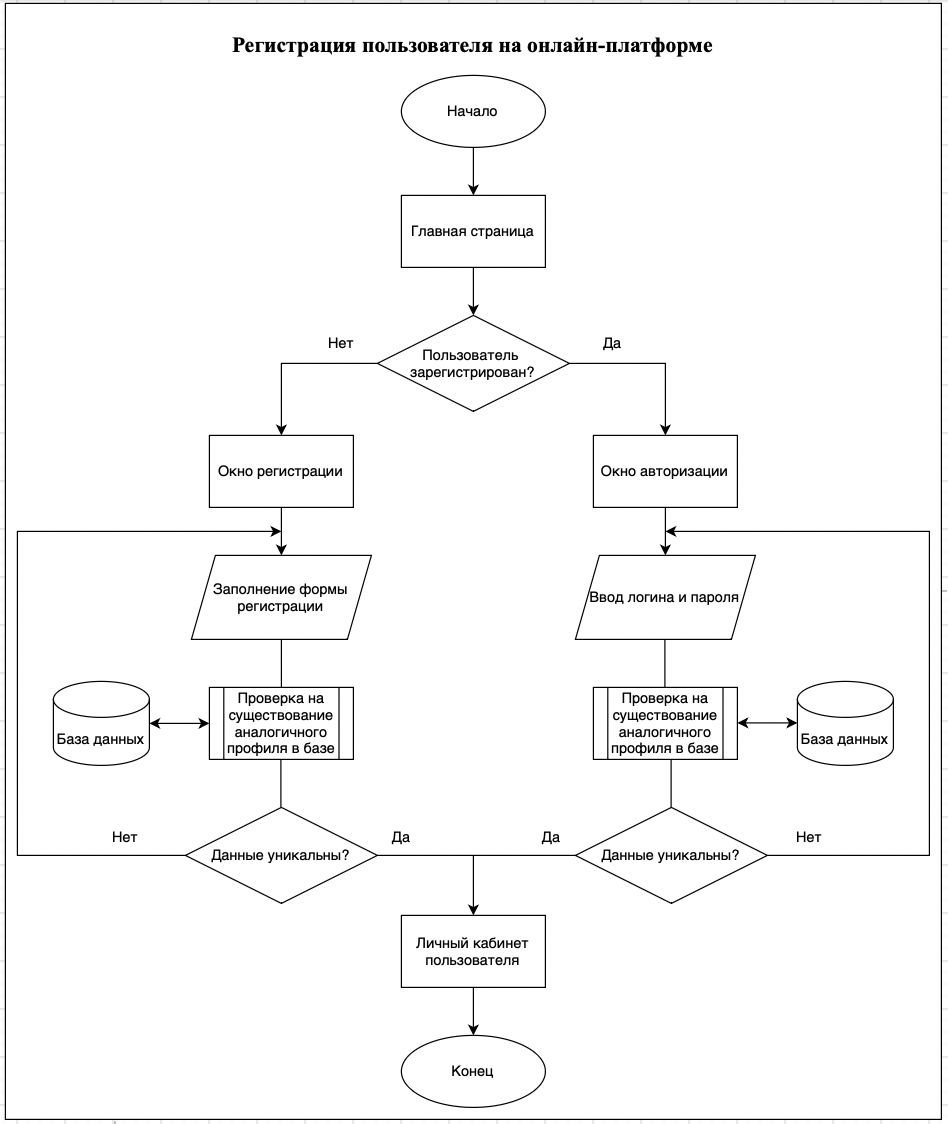


Более подробно ознакомиться с дизайном макета проекта можно по ссылке: <https://www.figma.com/file/sDc1ejJEZlAQuHgkVcKNzW/%D0%BD%D0%BE%D0%B3%D0%BE%D1%82%D0%BE%D1%87%D0%BA%D0%B8-%D1%80%D1%83?type=design&node-id=30-423&mode=design&t=YMmJd8iLm9FPO3aF-0>

2.5 Описание алгоритмов

Алгоритм – это последовательность действий, которые выполняются для решения определенной задачи. Основная задача алгоритмов – это обработка и преобразование данных. Алгоритмы играют важную роль в разработке веб-приложений. Они используются для обработки, анализа и преобразования данных, а также для оптимизации производительности приложения. В веб-разработке также используются алгоритмы маршрутизации, которые позволяют определить, какой контроллер или действие должно быть вызвано в ответ на определенный запрос. Алгоритмы маршрутизации могут использоваться для управления навигацией веб-приложения и определения доступа к определенным страницам [40]. Алгоритмы могут использоваться для решения задач безопасности вебприложений. Например, алгоритмы шифрования используются для защиты данных пользователя, а алгоритмы аутентификации и авторизации используются для контроля доступа к различным функциям приложения. В качестве примера одного из алгоритмов был выбран алгоритм регистрации пользователя в системе и оформления заказа на сервисе. Эти процессы были выбраны не случайно, они являются обязательными и наиболее важными для системы. С их помощью данные пользователей попадают в базу данных. На Рисунке 35 представлен алгоритм регистрации пользователя в виде блок-схемы, которая была спроектирована в онлайн-среде draw.io. Регистрация направленна на создание личной учетной записи в системе с целью получения доступа к большему функционалу. Для этого пользователь может перейти с главной страницы на окно регистрации или авторизации в зависимости от того, есть ли у него профиль или нет.

В процессе регистрации пользователю необходимо корректно ввести свои данные в форму регистрации. Введенные данные пользователя проверяются в базе данных на существование аналогичных, в случае если данные уникальны, создается профиль пользователя. Если же введенные данные не удовлетворяют требованиям, то необходимо проверить и ввести данные заново.



2.6 Архитектура веб-приложения

Архитектура веб-приложения – это описание структуры и взаимодействия различных компонентов приложения, которые обеспечивают 64 его функционирование. Различные архитектурные подходы могут быть использованы в веб-разработке. Логика веб-приложения, также называемая бизнес-логикой, определяет, как приложение будет обрабатывать данные и выполнять определенные задачи. Кроме того, логика веб-приложения может включать проверку и валидацию данных, авторизацию и аутентификацию пользователей, обработку ошибок и многое другое. Для онлайн-площадки была также разработана общая схема архитектуры, которая изображена на Рисунке 37.

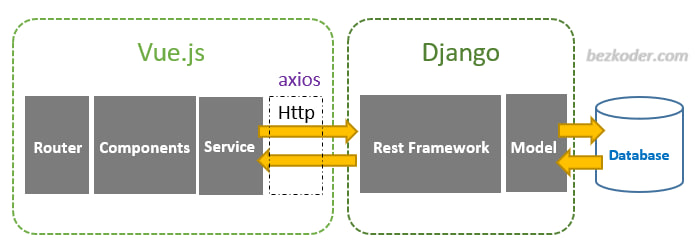
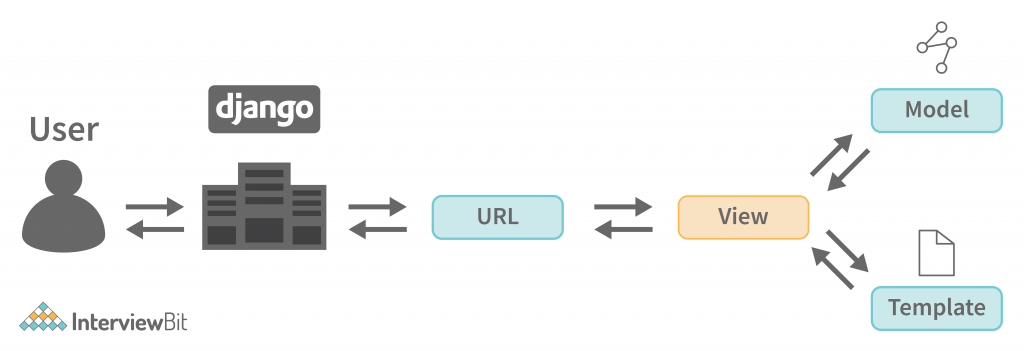


Рисунок 37 – Архитектура веб-приложения онлайн-площадки На схеме видно, что клиентская часть, разработанная с помощью фреймворка Vue.js, в свою очередь состоит из роутеров, которые направляют на компоненты, у которых есть свои сервисы, при выполнении которых формируется запрос, который отправляется на сервер. На сервере запросы проходят по API, которое реализовано с помощью REST Framework. Серверная часть приложения разработана с помощью фреймворка Django. Полученные данные в запросе имеют свои поля в базе данных, на сервере они реализованы через модели данных. Модели (Models) определяют структуру данных приложения и правила их валидации. Они взаимодействуют с базой данных и предоставляют интерфейс для работы с данными. Такая архитектура предоставляет разработчику удобный способ организации кода и разделения логики на независимые компоненты. 65 Стоит отметить, что в Django есть свой механизм администрирования (Admin), который позволяет управлять моделями и данными через встроенный веб-интерфейс.

**2.8 Техническая реализация серверной части**

**2.8.1 Работа архитектуры серверной части**

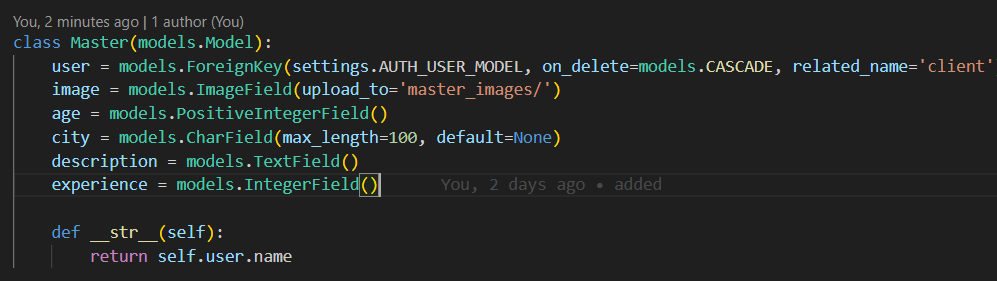
Django – это основанный на Python фреймворк для создания вебприложений. Фреймворк Django REST работает поверх Django. Это библиотека, которую мы можем использовать для быстрого и эффективного 69 создания веб-API с минимальным количеством кода. Django использует шаблон проектирования Model-View-Controller (MVC). Ранее было упомянуто, что backend-фреймворки имеют «слоистую» архитектуру. Подробная схема работы серверной части изображена на Рисунке

Рисунок – Работа серверной части онлайн-площадки 

После того как клиент инициировал запрос, сервер отправляет запрос по url-адресу. URL-адреса определяют, какие представления должны быть вызваны для обработки запросов. Они связывают URL-шаблоны с функциями представлений. Представления (Views) обрабатывают запросы от пользователей и возвращают ответы в виде HTML-страниц, JSON-данных или других форматов. Они используют модели для получения и изменения данных, а также шаблоны для форматирования и отображения этих данных. Сериализаторы (Serializer) позволяют преобразовывать сложные данные, такие как наборы запросов и экземпляры моделей, в собственные типы данных Python, которые затем могут быть легко преобразованы в JSON, XML или другие типы содержимого. Сериализаторы также обеспечивают десериализацию, позволяя преобразовывать разобранные данные обратно в сложные типы после предварительной проверки входящих данных [35]. Модели (Models) определяют структуру данных приложения и правила их валидации. Они взаимодействуют с базой данных и предоставляют интерфейс для работы с данными

**2.8.2 Создание моделей**

Разработка моделей является одним из самых важных этапов, так как, если данные будут неверно смоделированы, это может привести к проблемам в работе приложения, таким как ошибки при сохранении, получении данных и проблемы с производительностью. Модель является единственным источником информации о данных веб-приложения. Она содержит основные поля и поведение данных, которые необходимо хранить. Как правило, каждая модель отображается в одну таблицу базы данных. Все атрибуты модели имеют свой тип данных, в зависимости от информации, которую они описывают. Каждая модель представляет собой класс Python, который является подклассом django.db.models.Model. Для этого необходимо создать класс, который будет наследоваться от models.Model, каждое поле указывается как атрибут класса, и каждый атрибут отображается в столбец базы данных. По коду видно, что будет создана таблица Master, в которой будет шесть столбцов. User, который является внешним ключом таблицы User. Это означает, что при удалении пользователя удалиться автоматически информация и о его профиле. Столбец City имеет текстовый тип, максимальную длину 100 и значение по умолчанию, если вдруг пользователь не укажет данную информацию. Столбец image хранит в себе ссылку на фотографию пользователя и также имеет фото по умолчанию и адрес папки, куда будут загружаться фото. В столбце description хранится описание мастера, где он может рассказать, например, о своем опыте работы, experience – содержит информацию о количестве опыта

****Рисунок – Создание модели Мастера

**Выводы**

Работа по созданию интернет-портала "NailConnect" представляет собой комплексный процесс, включающий разработку концепции пользовательского интерфейса и программное решение. В ходе этой работы были достигнуты следующие ключевые результаты:

1. **Модель прецедентов и бизнес-процессы:**
   * Была разработана модель прецедентов, представленная в виде диаграммы UML, которая отражает отношения между ролями пользователей и вариантами использования системы.
   * Спроектированы диаграммы в нотации BPMN для таких ключевых бизнес-процессов, как оформление заказа, изменение статуса заказа и добавление услуги мастера.
2. **Алгоритмы и функциональность:**
   * Описаны алгоритмы регистрации и оформления заказа, обеспечивающие персонализированный доступ к расширенному функционалу портала.
   * Разработаны основные функциональные возможности веб-портала, включая возможность просмотра профилей мастеров, оформления заказов и изменения статуса заказов.
3. **База данных:**
   * Создана ER-диаграмма базы данных, включающая 10 таблиц для хранения данных, обеспечивающих эффективное хранение и обработку информации о пользователях, заказах, мастерах и услугах.
4. **Дизайн и пользовательский интерфейс:**
   * Спроектирован дизайн-макет интернет-портала "Master Manicure" с использованием принципов UX-дизайна и онлайн-сервиса Figma.
   * Уделено особое внимание цветовой палитре, шрифтам и визуальным элементам, чтобы создать привлекательный и удобный для пользователей интерфейс.
5. **Архитектура веб-приложения:**
   * Представлена архитектура разработанного веб-приложения как со стороны клиента, так и со стороны сервера, обеспечивающая эффективное взаимодействие между фронтендом и бэкендом.
   * Разработана клиентская и серверная части интернет-портала в среде разработки Visual Studio Code с использованием фреймворков Vue.js и Django, что обеспечивает надежность и масштабируемость системы.

В целом, выполнение данного проекта позволило создать современный и функциональный интернет-портал для маникюра, который удовлетворяет потребностям пользователей и обеспечивает эффективное взаимодействие между клиентами и мастерами.