МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Ижевский государственный технический университет имени М.Т. Калашникова» (ФГБОУ ВО «ИжГТУ имени М.Т. Калашникова»)

Институт «Информатика и вычислительная техника»

Кафедра «Автоматизированные системы обработки информации и управления»

Работа защищена с оценкой\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата\_\_\_\_\_\_\_\_\_\_\_\_\_

Подпись\_\_\_\_\_\_\_\_\_\_\_\_\_

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе

по дисциплине «Программная инженерия»

на тему «Автоматизация деятельности салона красоты»

Выполнили

студенты гр. Б23-782-1 Зорина Е.А, Юрушев И.П.

Принял

к.т.н., доцент каф. АСОИУ Касимов Д.Р.

Рецензия:

степень достижения поставленной цели работы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

полнота разработки темы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

уровень самостоятельности работы обучающегося\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

недостатки работы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ижевск 2024

**СОДЕРЖАНИЕ**

[СОДЕРЖАНИЕ 2](#_Toc185274567)

[ВВЕДЕНИЕ 4](#_Toc185274568)

[1 Разработка спецификации требований 5](#_Toc185274569)

[1.1 Требования 5](#_Toc185274570)

[1.2 User Story 6](#_Toc185274571)

[1.3 Use Cases 6](#_Toc185274572)

[2 Планирование программного проекта 16](#_Toc185274573)

[2.1 Mindmap проекта 16](#_Toc185274574)

[2.2 Команда разработчиков 17](#_Toc185274575)

[2.3 Постановка задач для команды разработки 19](#_Toc185274576)

[2.4 Диаграмма Ганта 20](#_Toc185274577)

[2.5 Стоимость проекта 21](#_Toc185274578)

[3 Проектирование программного продукта 22](#_Toc185274579)

[3.1 UML-диаграмма состояний 22](#_Toc185274580)

[3.2 UML-диаграмма последовательности 23](#_Toc185274581)

[3.3 UML-диаграмма классов 25](#_Toc185274582)

[4 Описание репозитория 27](#_Toc185274583)

[5 Оценка качества программного продукта 28](#_Toc185274584)

[5.1 Чек листы 28](#_Toc185274585)

[5.2 Баг-репорты 29](#_Toc185274585)

[5.3 Тест кейсы 32](#_Toc185274586)

[5.4 Структура проекта 32](#_Toc185274587)

[5.5 Оценка качества сайта 33](#_Toc185274588)

[ЗАКЛЮЧЕНИЕ 36](#_Toc185274589)

[Приложение А 37](#_Toc185274585)

[Приложение Б 38](#_Toc185274585)

**ВВЕДЕНИЕ**

В современном мире, где цифровые технологии проникают во все сферы жизни, наличие сайта для бизнеса становится не просто желательным, а необходимым условием успешного функционирования. Это особенно актуально для маникюрных салонов, которые стремятся привлечь новых клиентов и удержать существующих. В современном мире, где цифровые технологии проникают во все сферы жизни, наличие сайта для бизнеса становится не просто желательным, а необходимым условием успешного функционирования. Это особенно актуально для маникюрных салонов, которые стремятся привлечь новых клиентов и удержать существующих. Создание сайта маникюрного салона позволяет значительно увеличить его видимость в интернете. Сайт предоставляет клиентам возможность ознакомиться с услугами и ценами в любое время, не выходя из дома. Сайт позволяет маникюрному салону продемонстрировать свои услуги и работы. Сайт предоставляет возможность клиентам оставлять отзывы и комментарии, что способствует улучшению качества услуг.

Целью работы является разработка эффективной онлайн-платформы, которая станет не только мощным инструментом для привлечения новых клиентов, но и важным средством для улучшения взаимодействия с существующими клиентами, повышения уровня обслуживания и формирования устойчивого имиджа маникюрного салона. Эта платформа будет интегрирована с современными технологиями, что позволит создать удобный интерфейс для пользователей, обеспечивая легкость навигации и доступ к информации о услугах и расписании.

Для разработки сайта маникюрного салона необходимо провести выявление и анализ требований к проекту, что позволит четко определить цели и задачи. Затем следует этап планирования, на котором разрабатывается структура и основные функции сайта. После этого осуществляется проектирование, включающее создание макетов и дизайна. На этапе реализации происходит непосредственная разработка сайта, где все запланированные функции и элементы дизайна воплощаются в жизнь. Завершающим этапом является тестирование, в ходе которого проверяется работоспособность сайта и устраняются возможные ошибки.

**1 Разработка спецификации требований**

Первый этап разработки программного продукта включает в себя сбор, анализ и формализацию требований к создаваемой системе. Эта работа является ключевой, поскольку она задает основу для дальнейшего проектирования и реализации программного обеспечения

* 1. **Требования**

**Функциональные требования:**

1. На главном экране приложения для клиента должна отображаться клавиша: “Прайс-лист”.
2. На главном экране приложения для клиента должна отображаться клавиша: “Запись на услугу”.
3. На главном экране приложения для клиента должны отображаться клавиша: “График работы”.
4. На главном экране приложения для администратора должна отображаться клавиша: “Запись клиентов”.
5. На главном экране приложения для администратора должна отображаться клавиша: “График работы мастеров”.
6. На главном экране приложения для администратора должна отображаться клавиша: “Изменение записи”.
7. На главном экране приложения для администратора должна отображаться клавиша: “Заметки”.

**Нефункциональные требования:**

1. На главном экране приложения должно отображаться название жирным шрифтом;
2. На главном экране приложения должен отображаться адрес, строго под названием с меньшим шрифтом;
3. В записях на услугу, забронированное время должно отображаться тёмным цветом, свободное ярким.
4. График работы салона с днями недели (аббревиатурой) и временем (в формате hh:mm) должен отображаться внизу сайта
5. Сайт должен поддерживать браузеры Chrome, Яндекс, Brave, Opera, Edge, Safari, Firefox, Atom, Thorium, DuckDuckGo

**1.2 User Story**

Для описания функциональности с точки зрения клиента и администратора, мы использовали User Story.

Для клиента:

1. Я как клиент, хочу увидеть прайс лист с ценами, чтобы определиться с выбором услуги.
2. Я как клиент, хочу увидеть график записей, чтобы записаться на услугу.
3. Я как клиент, хочу увидеть график работы салона, чтобы прийти вовремя.

Для администратора:

1. Я как администратор, хочу увидеть график записи клиентов, чтобы быть готовым к встрече посетителей.
2. Я как администратор, хочу увидеть график работы мастеров, чтобы распределить обязанности.
3. Я как администратор, хочу увидеть заметки, чтобы отвечать на вопросы клиентов

**1.3** **Use Cases**

Для описания взаимодействий пользователя с системой, были использованы Use Cases.

Таблица 1.3.1 – Содержание Use Case 1

|  |  |
| --- | --- |
| Название | Запись на услугу |
| Описание | Клиент может выбрать свободные время и дату, подходящие под его запрос, используя онлайн систему записи |
| Акторы | Клиент |

Продолжение таблицы 1.3.1

|  |  |
| --- | --- |
| Предусловия | Клиент зашел на сайт и нажал клавишу: “Запись на услугу” |
| Основной поток событий | 1. Клиент заходит на сайт.  2. Клиент нажимает клавишу: “Запись на услугу”.  3. Система отображает календарь.  4. Клиент выбирает необходимый месяц.  5. Система отображает дни недели с числом выбранного месяца, подсвечивая свободные ярким цветом, забронированные, слегка тёмным серым цветом.  6. Клиент выбирает подходящий свободный день недели.  7. Система отображает время возможной записи, подсвечивая свободные часы ярким, забронированные, слегка тёмно-серым.  8. Клиент выбирает подходящее ему время.  9. Система отображает подтверждение записи. |
| Альтернативный поток событий | 1. На шаге 5 система не отображает список доступных дней недели.  2. Клиент изменяет параметры поиска дней недели.  3. Система отображает новый список доступных дней недели.  4. Клиент продолжает процесс записи. |

Продолжение таблицы 1.3.1

|  |  |
| --- | --- |
| Постусловия | Клиент успешно записался на выбранное время и день, после чего получил подтверждение записи. |
| Расширенные атрибуты | 1. Если клиент выбирает забронированное время, система сообщает об ошибке и предлагает исправить его.  2. Клиент может выбрать время и дату записи.  3. Система отмечает выбранные данные, тёмно-серым цветом и закрывает к ним доступ, тем самым отмечая запись клиентов. |
| Диаграмма Use Case | Пример use case диаграммы «Запись на услугу» |

Продолжение таблицы 1.3.1

|  |  |
| --- | --- |
| Рекомендации по реализации | 1. Реализовать удобный и интуитивно понятный интерфейс для выбора подходящей даты и времени.  2. Отобразить доступные записи в удобном для клиента виде (подсвечивая ярким цветом свободные, тёмным забронированные) |

Таблица 1.3.2 – Содержание Use Case 2

|  |  |
| --- | --- |
| Название | График работы мастеров |
| Описание | Администратор может посмотреть график работы каждого мастера, для консультирования клиентов и отслеживания работы мастеров. |
| Акторы | Администратор |
| Предусловия | Администратор зашёл в график работы мастеров |
| Основной поток событий | 1. Администратор заходит в график работы мастеров.  2. Система выдаёт календарь с месяцем.  3. Администратор выбирает подходящую дату.  4. Система выдаёт часы работы и имена мастеров.  5. Администратор изучает часы работы мастеров. |

Продолжение таблицы 1.3.2

|  |  |
| --- | --- |
| Альтернативный поток событий | 1. На шаге 4 система выдаёт часы работы и имена мастеров.  2. Администратор, изучив часы работы мастеров, изменяет график в связи с остатками незабронированных часов. |
| Постусловия | Администратор успешно изучил или изменил в случае необходимости график |
| Расширенные атрибуты | Если администратор изменяет время на нерабочие часы, система выдаёт ошибку. |
| Диаграмма Use Case | Пример use case диаграммы «График работы мастеров» |

Продолжение таблицы 1.3.2

|  |  |
| --- | --- |
| Рекомендации по реализации | 1. Реализовать удобный и интуитивно понятный интерфейс графика работы мастеров.  2. Отобразить календарь, дни недели и время в удобном для понимания администратором виде. |

Таблица 1.3.3 – Содержание Use Case 3

|  |  |
| --- | --- |
| Название | Изменение записей клиентов. |
| Описание | Администратор может изменить запись клиента в связи с просьбой клиента или мастера. |
| Акторы | Администратор |
| Предусловия | Администратор зашёл в изменение записей клиентов. |
| Основной поток событий | 1. Администратор заходит в изменение записей клиентов  2. Система выдаёт календарь с месяцем.  3. Администратор выбирает подходящую дату.  4. Система выдаёт часы работы, имена мастеров и клиентов, записанных на определённое время (в случае, если время свободно, система выдаёт пустую строку со временем).  5. Администратор выбирает необходимое для изменения время.  6. Администратор стирает запись и добавляет нового клиента  7. Система сохраняет данные |

Продолжение таблицы 1.3.3

|  |  |
| --- | --- |
| Основной поток событий | 1. Администратор заходит в изменение записей клиентов  2. Система выдаёт календарь с месяцем.  3. Администратор выбирает подходящую дату.  4. Система выдаёт часы работы, имена мастеров и клиентов, записанных на определённое время (в случае, если время свободно, система выдаёт пустую строку со временем).  5. Администратор выбирает необходимое для изменения время.  6. Администратор стирает запись и добавляет нового клиента  7. Система сохраняет данные |
| Постусловия | Администратор успешно изменил запись клиентов в случае необходимости. |
| Расширенные атрибуты | Если администратор указывает запись клиента на нерабочие часы, система выдаёт ошибку. |

Продолжение таблицы 1.3.3

|  |  |
| --- | --- |
| Диаграмма Use Case | Пример use case диаграммы «Изменение записей клиентов» |
| Рекомендации по реализации | 1. Реализовать удобный и интуитивно понятный интерфейс изменения записей клиентов.  2. Отобразить календарь, дни недели и время в удобном для понимания администратором виде. |

На рисунке 1.3.1 и 1.3.2 изображен прототип интерфейса, разработанный в программе Canva.

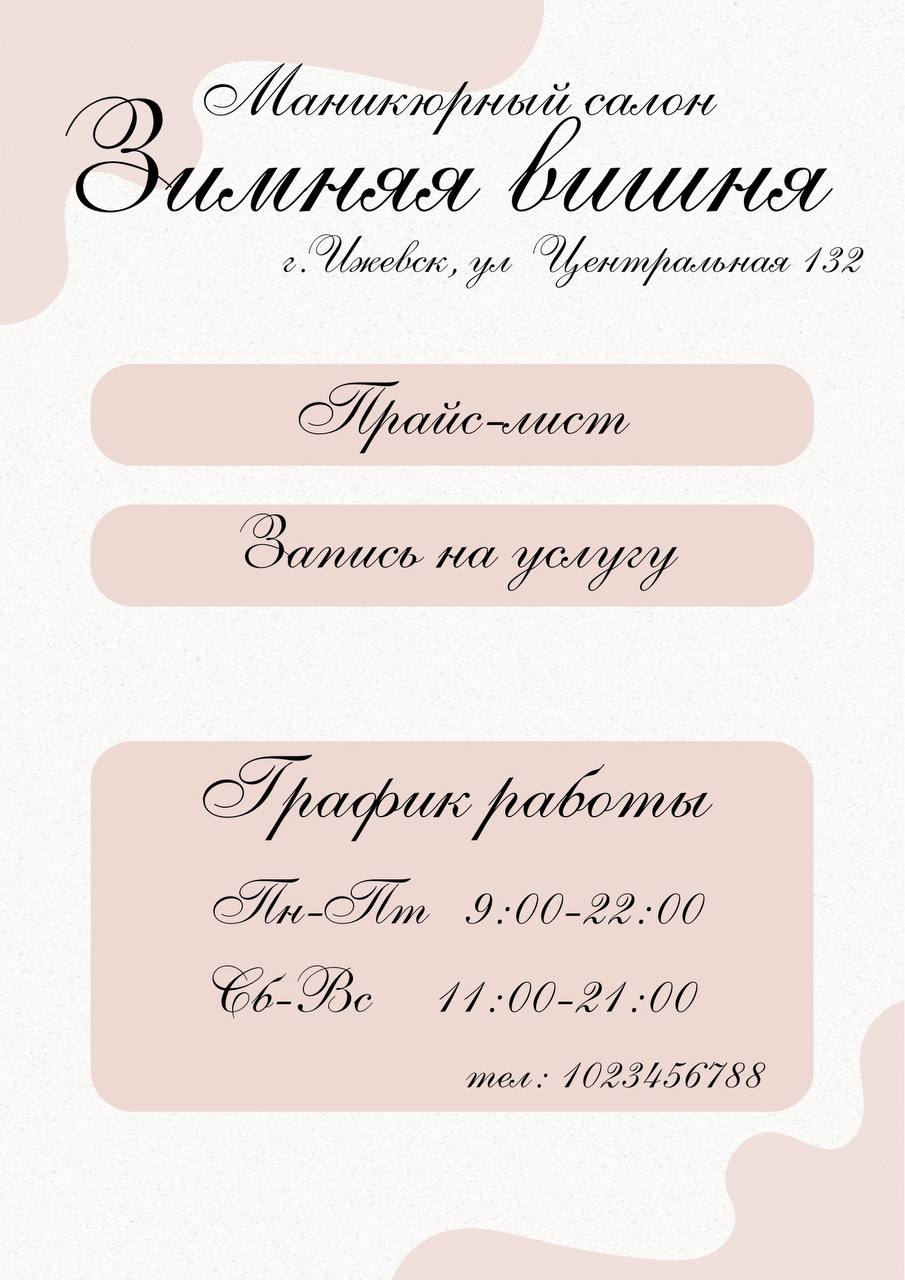


Рисунок 1.3.1- Прототип интерфейса



Рисунок 1.3.2- Прототип интерфейса

Создание прототипа интерфейса необходимо для понимания стиля сайта и начала разработки дизайна.

На данном этапе была выполнена разработка требований и вариант экранов сайта для клиента и администратора. Это позволило:

1. Определить ключевые функции и характеристики сайта.
2. Сформировать задачи для дальнейшей реализации.

Это позволило нам обеспечить четкое понимание того, каким должен быть конечный продукт и как он будет взаимодействовать с пользователями.

**2 Планирование программного проекта**

На данном этапе мы начали формировать структуру проекта, что включает в себя разработку MindMap, установку задач для команды и определение состава команды. Этот процесс помогает выявить основные направления работы, а также оценить объем работ и ресурсы, необходимые для старта проекта.

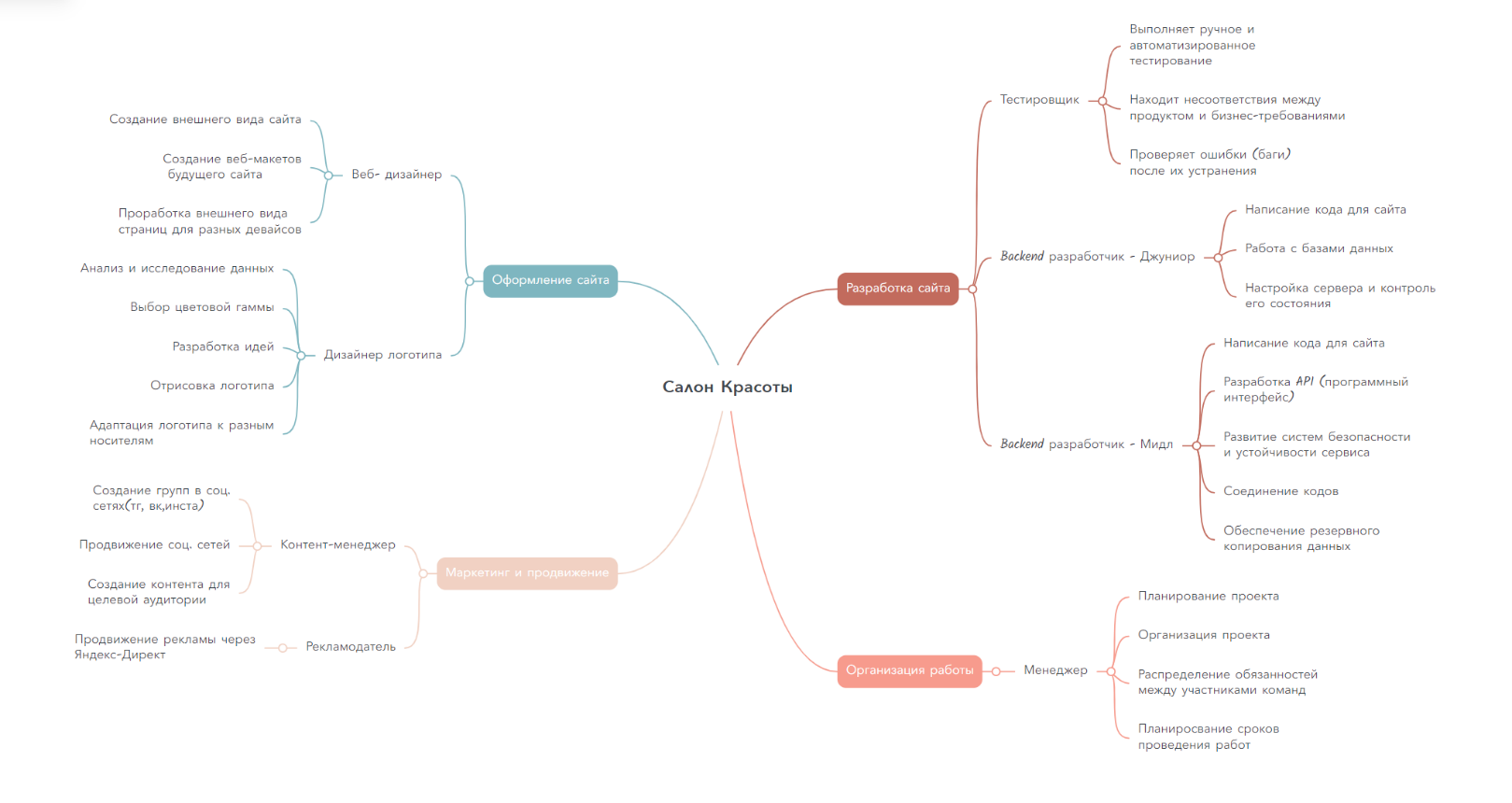
**2.1 Mindmap проекта**

Рисунок 2.1.1 – Mindmap проекта

Основные этапы реализации проекта.

1. Оформление сайта.

* Создание дизайна – разработка макетов страниц.
* Выбор цветовой схемы и шрифтов – подбор стиля, соответствующего салону.
* Разработка логотипа – создание уникального логотипа бренда.

1. Разработка сайта.

* Выбор платформы – создание сайта на CMS или с нуля.
* Создание структуры сайта – определение навигации.
* Разработка функционала – реализация задач сайта.
* Оптимизация под любые браузеры и операционные системы.

1. Маркетинг и продвижение.

* Создание социальных сетей для продвижения сайта.
* Создание рекламных кампании в Яндекс Директ.

1. Организация работы.
   * Определение услуг и цен. Составление каталога.
   * Набор команды для разработки сайта.
   * Определение сроков разработки.
   * Распределение бюджета.
   * Организация работы.
   * Тестирование.

**2.2 Команда разработчиков**

Для успешного создания сайта маникюрного салона была собрана команда, состоящая из специалистов разных областей. Были учтены задачи и требования проекта, обеспечивающие оптимальный баланс между высоким качеством услуг и разумными затратами.

Тестировщик (1 Junior): отвечает за выявление ошибок и проверку функциональности сайта. Проект не требует сложного функционала, поэтому Junior-специалист сможет выполнить тестирование качественно, обеспечивая приемлемый уровень надежности.

Менеджер (1 Middle): руководит процессом разработки, координирует команды и следит за соблюдением сроков. Профессионал среднего уровня способен эффективно организовать работу и урегулировать возможные конфликты в команде.

Дизайнер (1 Senior): создает уникальный визуальный стиль сайта, включая логотип и графические элементы. Опытный дизайнер обеспечивает оригинальность и запоминаемость бренда, что важно для привлечения клиентов в маникюрный салон.

Веб-дизайнер (1 Middle): отвечает за создание удобного и понятного интерфейса сайта. Проект требует знание современных стандартов и принципов UX/UI, поэтому специалист среднего уровня обеспечит высокое качество разработки пользовательского интерфейса.

Backend разработчик – Джун (1 Junior): занимается разработкой серверной части и баз данных. Начальный уровень позволяет справляться с базовыми задачами, что достаточно для проекта с невысокой нагрузкой.

Backend разработчик – Мидл (1 Middle): обеспечивает разработку сложной логики сервера и API. Специалист среднего уровня способен поддерживать баланс между производительностью и надежностью, что важно для обработки заявок из сайта.

Рекламщик (1 Middle): разрабатывает стратегии продвижения сайта и увеличения его посещаемости. Профессионал среднего уровня способен осуществить маркетинговые кампании и анализировать их эффективность, что существенно влияет на привлечение клиентов.

Контент-менеджер (1 Junior): создает и обновляет текстовый и визуальный контент на сайте. Начальный уровень позволяет регулярно обновлять информацию, что поддерживает актуальность сайта и интерес к услугам маникюра.

Роли, от которых отказались:

1. Аналитик. Функции и дизайн составляются самостоятельно основываясь на собственных потребностях.
2. Фронтенд-разработчик. Наличие опытного Веб-дизайнера в команде, позволяет отказаться от фронтенда.

Для расчета стоимости часа работы каждого специалиста были найдены средние зарплаты и пересчитаны на часовую оплату, которая представлена в таблице 2.2.1:

Распределение обязанностей между участниками команды позволило оптимизировать процесс разработки и достичь необходимых результатов в минимальные сроки.

|  |  |
| --- | --- |
| **Должность** | **Заработная плата** |
| Тестировщик | 75 000 (/168) 446 р/ч |
| Менеджер | 120 000 (/168) 714 р/ч |
| Дизайнер | 60 000 (/168) 357 р/ч |
| Веб-дизайнер | 90 000 (/168) 535 р/ч |
| Backend разработчик – Мидл | 130 000 (/168) 773 р/ч |
| Frontend-разработчик | 130 000 (/168) 773 р/ч |
| Рекламщик | 70 000 (/168) 416 р/ч |
| Контент-менеджер | 70 000 (/168) 416 р/ч |

Таблица 2.2.1 – Заработная плата

* 1. **Постановка задач для разработчиков**

Для организации работы и последующей её реализации, для каждого участника команды, были составлены задачи, показанные на рисунках 2.3.1 и 2.3.2

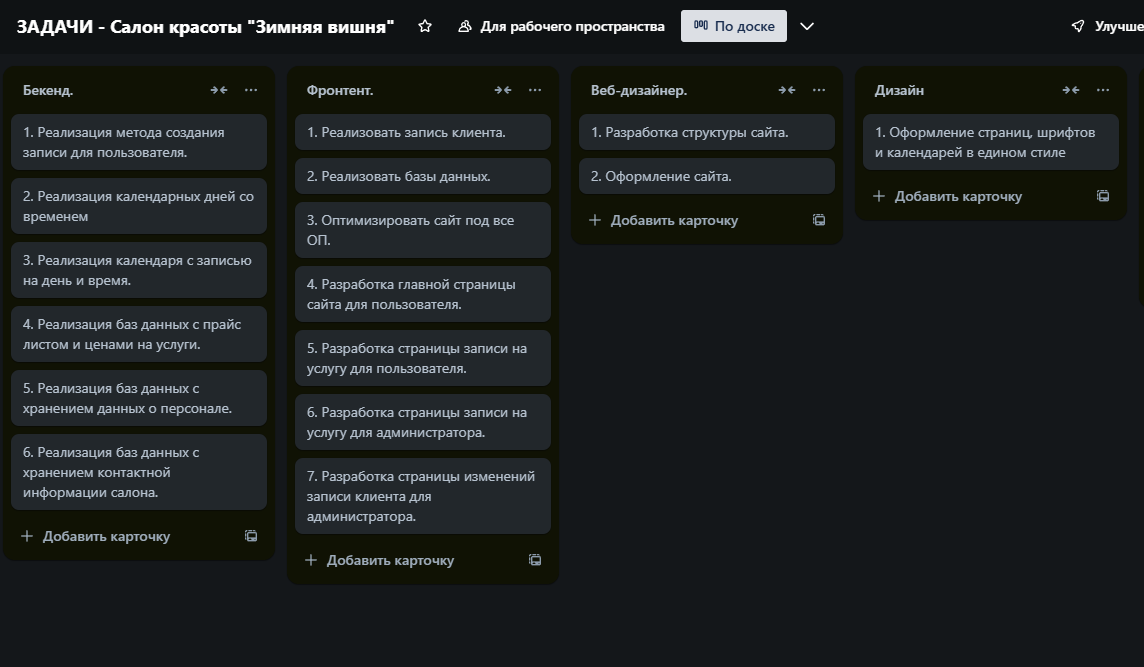


Рисунок 2.3.1- Задачи для разработчиков

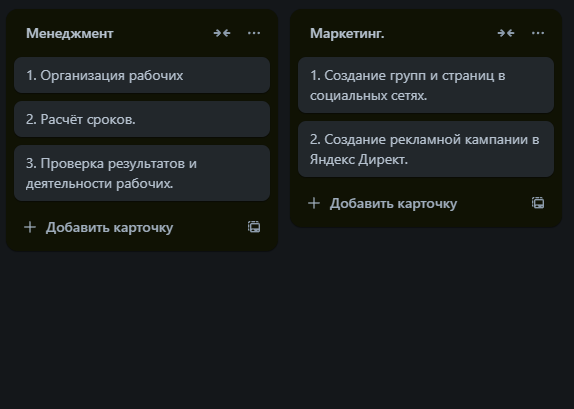


Рисунок 2.3.2- Задачи для разработчиков

В связи с тем, сто проект требует грамотного составления и ограничен в сроках, мы расставили для каждой их задач приоритет и время на выполнение.

Задачи, их приоритет и время изображены на рисунке 2.3.3.

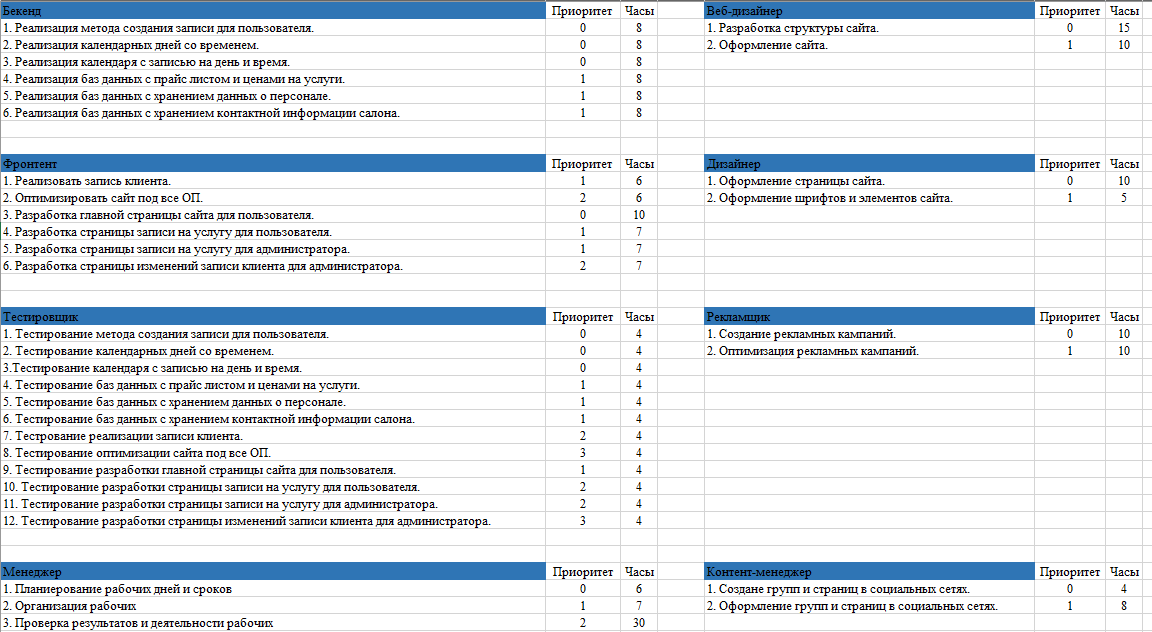


Рисунок 2.3.3- Задачи, их приоритет и время

Благодаря созданию плана задач, были распределены обязанности между всеми участниками команды, это повлияло на организацию работы. Риски пропуска организационных элементов сведены к минимуму, Распределена нагрузка между работниками, процесс разработки ускорен.

* 1. **Диаграмма Ганта**

Диаграмма Ганта позволила визуализировать сроки выполнения задач между сотрудниками. Это позволяет отслеживать выполнение проекта.

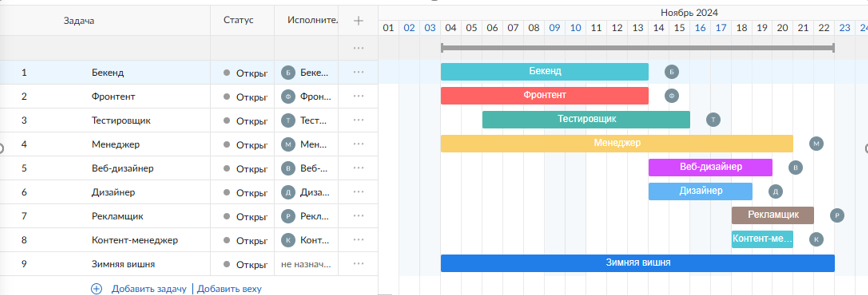


Рисунок 2.4.1 – Диаграмма Ганта

Цели применения диаграммы:

1. Управление временем: позволяет установить сроки выполнения каждого разработчика.
2. Визуализация зависимостей: позволяет увидеть, какие задачи зависят друг от друга.
3. Командная координация: обеспечивает ясность для всех участников проекта, позволяя им видеть, кто за что отвечает и когда.
   1. **Стоимость проекта**

Для распределения ресурсов и реализации проекта, был произведён процесс подсчёта стоимости разработки. Для подсчёта заработной платы сотрудников учитывались время в часах и почасовая оплата труда.

Вычисленные данные приведены на рисунке 2.5.1

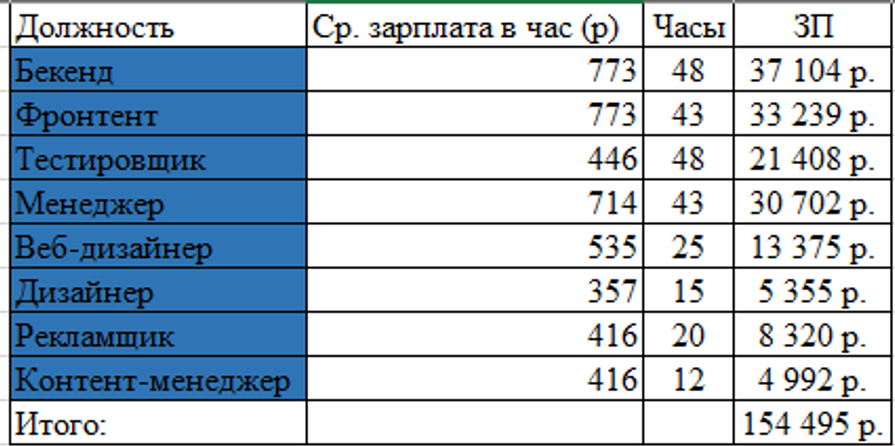


Рисунок 2.5.1 – Стоимость проекта

Общая стоимость проекта составила 154 495 руб.

Точная оценка стоимости позволяет:

1. Эффективно планировать бюджет, выделяя необходимые средства на реализацию проекта.
2. Более точно планировать сроки выполнения задач, учитывая финансовые ограничения.
3. Управлять рисками - идентифицировать финансовые риски и разрабатывать

**3 Проектирование программного продукта**

Проектирование продукта является важным этапом в разработке, который позволяет зафиксировать требования к продукту, упростить процесс разработки, выявить риски и минимизировать их.

**3.1 Составление диаграммы состояний**

Диаграмма состояний, реализованная в ходе выполнения работы, позволяет сохранять данные записи на услугу. Диаграмма состояний представлена на рисунке 3.1.1

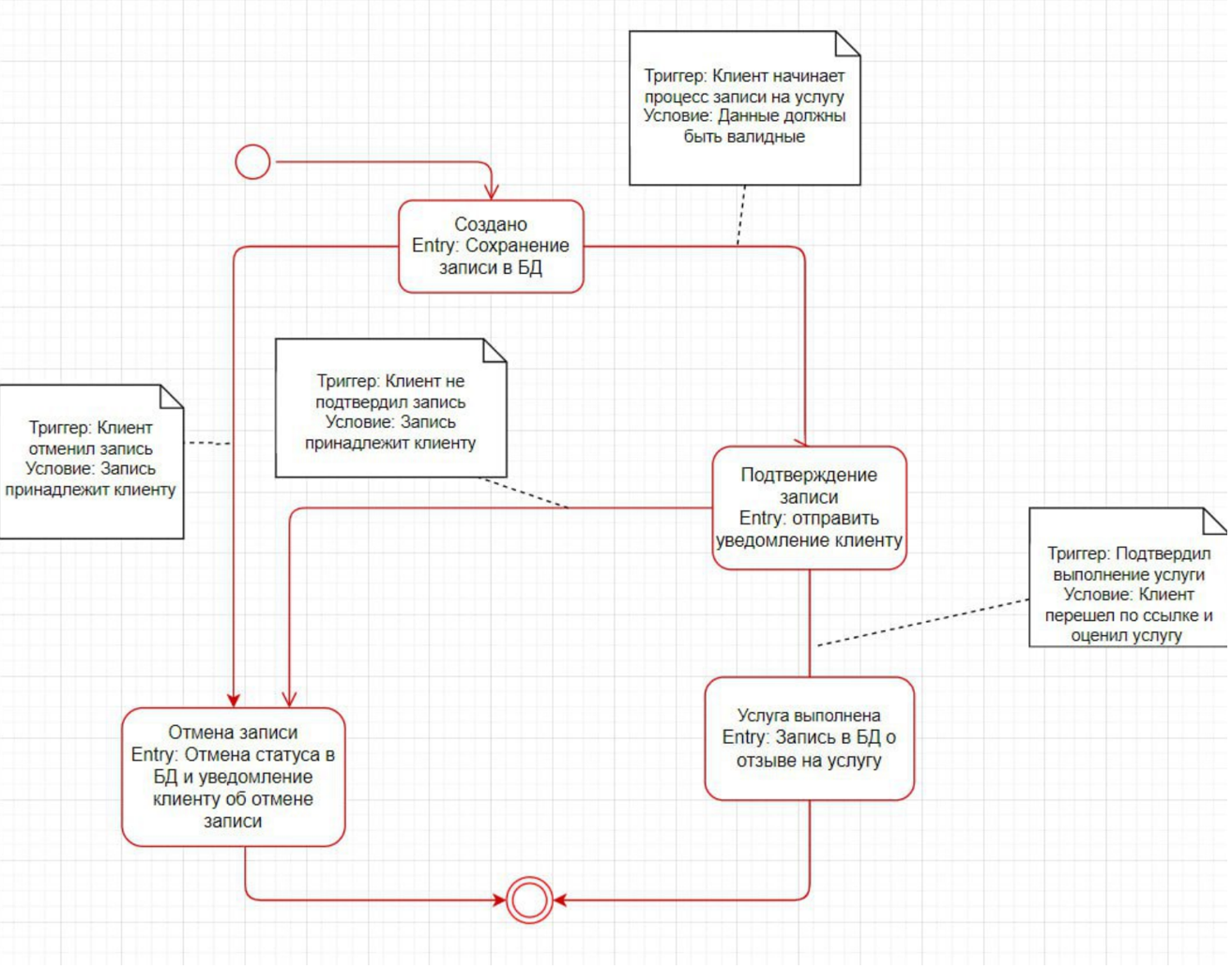


Рисунок 3.1.1 – Диаграмма состояний

Разработанная диаграмма отображает взаимодействие клиента с приложением для осуществления записи на выбранную услугу. Она показывает, какие действия должны осуществляться при выборе клиентом определённых задач.

Диаграмма состояний позволяет:

1. Моделировать поведение - понять, как объект ведет себя в ответ на события, что позволяет лучше спроектировать логику приложения.

2. Упрощать анализ сложных систем, разбивая их на более управляемые состояния и переходы.

3. Выявлять потенциальные ошибки в логике переходов и состояний на этапе проектирования.

Диаграмма состояний является важным инструментом для проектирования и анализа динамического поведения систем.

* 1. **UML-диаграмма последовательности**

Составленная нами UML-диаграмма последовательности иллюстрирует оплату услуги.

Она показывает взаимодействие пользователя с системой. На диаграмме указаны действия, совершаемые системой для реализации оплаты. UML-диаграмма представлена на рисунке 3.2.1

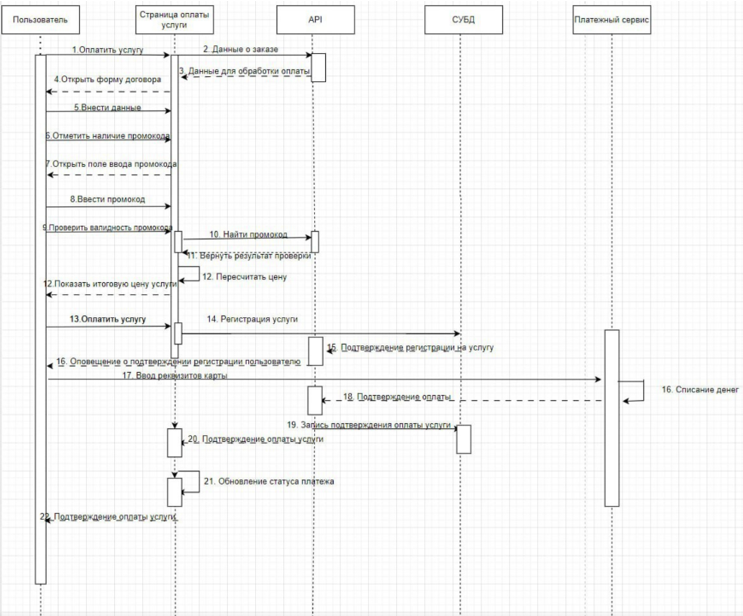


Рисунок 3.2.1 – UML – диаграмма последовательности

Составляющие диаграммы:

1. Пользователь: осуществляет оплату услуги.
2. Страница оплаты услуги: отвечает за интерфейс, позволяет пользователю взаимодействовать с сайтом.
3. API: принимает данные о заказе, осуществляет поиск промокода, отвечает за подтверждение оплаты услуги.
4. СУБД: регистрирует услугу, подтверждает регистрацию на услугу, осуществляет запись подтверждения на услугу.
5. Платёжный сервис: отвечает за платёжные операции.

Диаграмма последовательности – UML является важным этапом проектирования, она объясняет реализацию оплаты услуги, оказанной пользователю.

* 1. **UML – диаграмма классов**

UML-диаграмма классов используется для визуального представления структуры системы записи в салон, описывая классы, их атрибуты, методы и отношения между ними. UML – диаграмма предоставлена на рисунке 3.3.1.

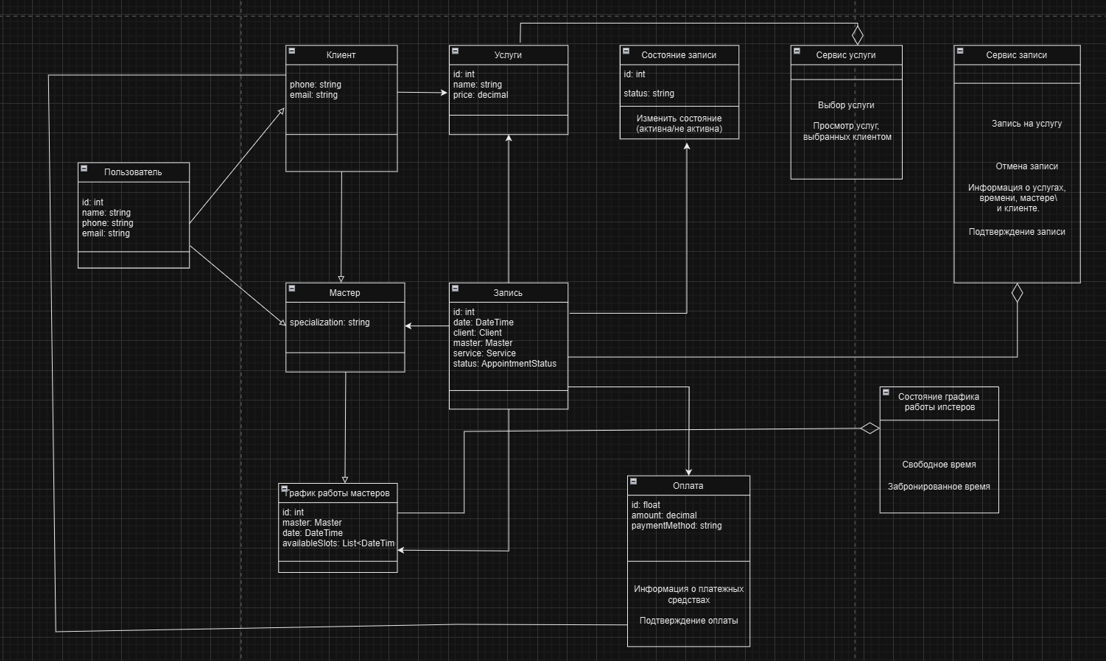


Рисунок 3.3.1 – UML – диаграмма классов

Классы UML – диаграммы:

1. Пользователь.

Атрибуты: id, name, phone, email.

Особенности: наследуется классами клиент и мастер.

1. Клиент.

Атрибуты: phone, email.

Связи: связан с классом “Услуги”.

1. Мастер.

Атрибуты: specialization.

Связи: связан с классом “График работы мастеров”

1. График работы мастеров.

Атрибуты: id, master, date, availableSlots.

Связи: связан с классом «Состояние графика работы мастеров»

Функции: позволяет мастеру посмотреть график работы.

1. Состояние графика работы мастеров.

Функции: позволяет мастеру посмотреть свободное и забронированное время для составления графика дня.

1. Услуги.

Атрибуты: id, name, price.

Связи: связаны с классом «Сервис услуг».

1. Сервис услуг.

Функции: позволяют клиентам выбирать услуги маникюрного салона.

1. Состояние записи.

Атрибуты: id, status.

Функции: информирует у состоянии записи (активна/не активна)

1. Сервис записи.

Функции: осуществляет запись на услугу, отмену записи в случае необходимости, подтверждение записи, хранит информацию о услугах, времени, мастере и клиенте.

1. Оплата.

Атрибуты: id, amount, paymentMetod.

Связи: связана с классом «клиент».

Функции: содержит информацию о платёжных средствах, осуществляет оплату услуги.

1. Запись.

Атрибуты: id, date, client, master, service, status.

Связи: связана с классами: “Услуги”, “Состояние записи”, “Оплата”, “Мастер”, “График работы мастеров”.

UML – диаграмма классов позволяет чётко определить объекты и их атрибуты, создать связи между сущностями для обеспечения логики системы, составить основу для дальнейшей реализации кода.

1. **Описание репозитория**

Для данного проекта был создан репозиторий на платформе GitHab. Это позволяет отслеживать изменения, управлять проектом и редактировать его нескольким разработчикам. На рисунке 4.1 приведен скриншот.

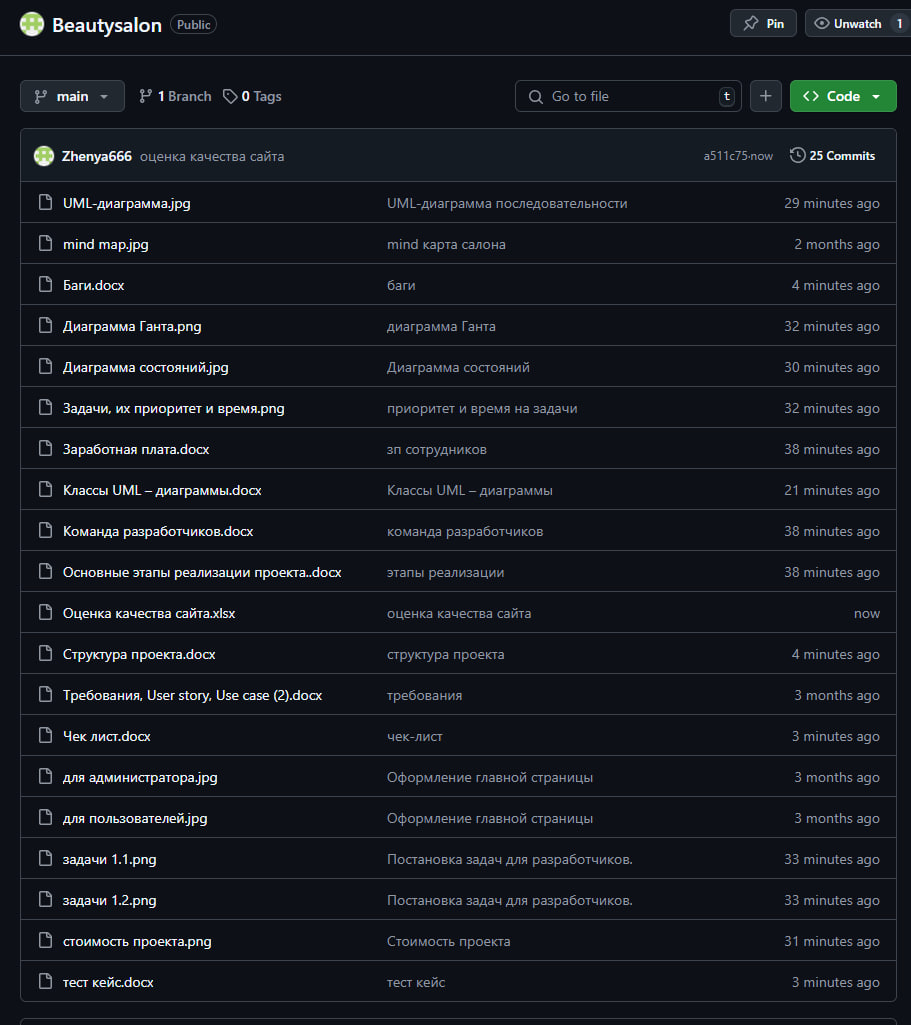


Рисунок 4.1 – Структура репозитория

1. **Оценка качества программного продукта**

Оценка качества программного продукта была проведена для определения соответствия разработанного программного обеспечения установленным требованиям и ожиданиям.

**5.1 Чек листы**

Чек-лист для проверки функциональных требований

1. "Прайс-лист" на главном экране клиента

Проверьте, что клавиша "Прайс-лист" отображается на главном экране приложения для клиента.

Убедитесь, что клавиша работает и ведет на соответствующую страницу.

1. "Запись на услугу" на главном экране клиента

Проверьте, что клавиша "Запись на услугу" отображается на главном экране приложения для клиента.

Убедитесь, что клавиша работает и ведет на соответствующую страницу.

1. **"**График работы" на главном экране клиента.

Проверьте, что клавиша "График работы" отображается на главном экране приложения для клиента.

Убедитесь, что клавиша работает и ведет на соответствующую страницу.

1. "Запись клиентов" на главном экране администратора

Проверьте, что клавиша "Запись клиентов" отображается на главном экране приложения для администратора.

Убедитесь, что клавиша работает и ведет на соответствующую страницу.

1. "График работы мастеров" на главном экране администратора

Проверьте, что клавиша "График работы мастеров" отображается на главном экране приложения для администратора.

Убедитесь, что клавиша работает и ведет на соответствующую страницу.

1. "Изменение записи" на главном экране администратора

Проверьте, что клавиша "Изменение записи" отображается на главном экране приложения для администратора.

Убедитесь, что клавиша работает и ведет на соответствующую страницу.

1. **"**Заметки" на главном экране администратора

Проверьте, что клавиша "Заметки" отображается на главном экране приложения для администратора.

Убедитесь, что клавиша работает и ведет на соответствующую страницу.

**Чек-лист для проверки нефункциональных требований**

1. Название на главном экране приложения

Проверьте, что название приложения отображается жирным шрифтом на главном экране.

1. Адрес под названием

Убедитесь, что адрес отображается строго под названием с меньшим шрифтом.

1. Отображение забронированного и свободного времени

Проверьте, что забронированное время отображается темным цветом.

Убедитесь, что свободное время отображается ярким цветом.

1. График работы салона

Проверьте, что график работы отображается внизу сайта с днями недели (аббревиатурами) и временем в формате hh:mm.

1. Поддержка браузеров

Убедитесь, что сайт корректно отображается и функционирует в следующих браузерах: Chrome, Яндекс, Brave, Opera, Edge, Safari, Firefox, Atom, Thorium, DuckDuckGo.

* 1. **Баг - репорты**

Таблица 5.2.1- Критический

|  |  |
| --- | --- |
| Заголовок | Не работает функция записи на прием |
| Приоритет и серьезность | Критический. Не осуществляется запись на услуги, что приводит к потере клиентов, другие функции приложения работают. |

Продолжение таблицы 5.2.1

|  |  |
| --- | --- |
| Шаги воспроизведения | Запустить сайт.  Перейти в раздел записи.  Выбрать услугу и дату.  Нажать кнопку "Записаться". |
| Фактический результат | Появляется сообщение об ошибке "Не удалось записаться на прием". |
| Ожидаемый результат | Пользователь должен успешно записаться на прием и получить подтверждение. |
| Окружение | Android 11, приложение версии 1.2.3. |
| Оценить срок исправления и назначить члена команды | Ожидаемый срок исправления - 2 дня. Backend – разработчик. |
| Дополнительная информация | Очищали кэш и перезагружали сайт, проблема сохраняется. |

Таблица 5.2.2- Высокоприоритетный

|  |  |
| --- | --- |
| Заголовок | Некорректное отображение цен на услуги |
| Приоритет и серьезность | Высокоприоритетный. Цены отображаются неверно, что вызывает недовольство клиентов. (Есть возможность уточнения цен у администратора.) |
| Шаги воспроизведения | Запустить сайт.  Перейти в раздел "Услуги".  Просмотреть список услуг и их цены. |
| Фактический результат | Цены на услуги отображаются с ошибками (например, маникюр стоит 1000 рублей, а отображается 10000 рублей) |

Продолжение таблицы 5.2.2

|  |  |
| --- | --- |
| Ожидаемый результат | Цены должны отображаться корректно в соответствии с установленными тарифами. |
| Окружение | iOS 14, приложение версии 1.2.3. |
| Оценить срок исправления и назначить члена команды | Ожидаемый срок исправления - 3 дня.  Frontend - разработчик. |
| Дополнительная информация | Проблема наблюдается только на iOS, на Android цены отображаются корректно. |

Таблица 5.2.3-Низкоприоритетный

|  |  |
| --- | --- |
| Заголовок | Ошибка в дизайне кнопки "Записаться" |
| Приоритет и серьезность | Низкоприоритетный. Дизайнерская ошибка не влияет на функциональность приложения, но может ухудшить пользовательский опыт. |
| Шаги воспроизведения | Запустить сайт.  Перейти в раздел "Запись на прием".  Обратить внимание на кнопку "Записаться". |
| Фактический результат | Кнопка "Записаться" имеет неправильный цвет и не соответствует общему стилю приложения. |
| Ожидаемый результат | Кнопка должна быть оформлена в соответствии с дизайном приложения. |
| Окружение | Android 11, приложение версии 1.2.3. |
| Оценить срок исправления и назначить члена команды | Ожидаемый срок исправления – 3 дня.  Веб – дизайнер. |
| Дополнительная информация | Проблема не влияет на функциональность, но может быть замечена пользователями, что важно для общего восприятия приложения. |

* 1. **Тест кейсы**

Для проверки функциональности и производительности продукта, мы использовали тест кейсы. Нами были применены три разновидности.

**Позитивный -** проверяет, что программный продукт работает корректно и выполняет заданные функции в соответствии с требованиями. Он направлен на подтверждение того, что система реагирует ожидаемым образом на корректные входные данные и условия. Позитивные тест кейсы помогают убедиться, что основные функции приложения работают так, как задумано, и что пользователи могут успешно выполнять свои задачи без ошибок.

**Негативный -** проверяет, как система или приложение реагируют на неправильные или неожиданные входные данные. Цель негативного тестирования — убедиться, что программа обрабатывает ошибки корректно и не допускает сбоев или неправильного поведения при получении некорректной информации.

**Деструктивный** - направлен на проверку устойчивости системы к критическим ошибкам или сбоям.

Таблицы по тест кейсам вынесены в приложение Б.

* 1. **Структура проекта**

Название проекта: Салон красоты- Зимняя вишня.

Цель: Создание веб-сайта для просмотра расписания, каталога услуг, мастеров и осуществления записи на выбранные услуги.

**Файлы репозитория**

1. Дизайн.png – изображения примерного вида сайта.
2. Требования.docs – автоматизация деятельности салона.
3. User story, use case – описание основного функционала сайта.
4. Менеджменд.xlsx – распределение обязанностей.
5. Бухгалтерия.xlsx – подсчёт затрат на проект.
6. Разработчики.docs – состав команды, необходимый для реализации проекта.
7. Задачи.docs – задачи, распределённые между участниками, необходимые для реализации проекта.
8. Диаграмма состояния.drawio – описывает поведение системы.
9. Диаграмма последовательности.drawio – визуализация взаимодействия объектов в системе.
10. Диаграмма классов.drawio – моделирование структуры системы.

**Файлы программы**

Файлы, содержащиеся в программе изображены на рисунке 5.4.1

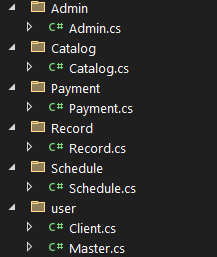


Рисунок 5.4.1 – Файлы программы

* 1. **Оценка качества сайта**

Для оценки качества программного продукта, была использована 5-ти бальная система. Оценка проводилась по таким показателям как: функциональность, надёжность, удобство применения, сопровождаемость, эффективность, переносимость. Благодаря этому были выявлены сильные места системы, а также слабые, требующие доработки. Оценка качества сайта изображена на рисунках (5.5.1-5.5.3)

Рисунок 5.5.1 – Оценка качества сайта

****Рисунок 5.5.2 – Оценка качества сайта

Рисунок 5.5.3 – Оценка качества сайта

**ЗАКЛЮЧЕНИЕ**

В ходе выполнения курсовой работы по созданию сайта маникюрного салона была достигнута поставленная цель — разработка эффективной онлайн-платформы, обеспечивающей пользователям комфортный доступ к услугам салона, включая просмотр каталога, запись на процедуры и осуществление оплаты.

Анализ требований на начальном этапе позволил детально определить нужды целевой аудитории и сформулировать ясные и достижимые задачи. Этап планирования способствовал созданию структурированной схемы сайта, а также обозначению его основных функциональных блоков, что значительно упростило последующее проектирование.

В процессе проектирования мы смогли разработать привлекательные макеты и интерфейс, обеспечивающие легкость навигации и интуитивно понятное взаимодействие с пользователями. Реализация проекта позволила интегрировать все запланированные функции, включая мобильную версию, что повышает доступность сервиса для пользователей различных устройств.

Тестирование, проведенное на завершающем этапе, подтвердило стабильность работы платформы и устранило выявленные недостатки. В результате была создана надежная и функциональная онлайн-платформа, способная значительно улучшить взаимодействие с клиентами и обеспечить высокое качество обслуживания.

В целом, разработка сайта маникюрного салона не только достигла заявленных целей, но и подготовила основу для дальнейшего развития онлайн-сервиса, позволяя внедрять новые функции и улучшения в будущем.

**ПРИЛОЖЕНИЕ А**

**Термины, аббревиатуры и их определения.**

API - Интерфейс программирования приложений, позволяющий взаимодействовать между компонентами системы.

Use case — описание того, как пользователи взаимодействуют с системой для достижения определенной цели. Включает в себя сценарии, предусловия, постусловия которые показывают, как система должна реагировать на действия пользователей или других систем.

Use Case — описание взаимодействия между пользователем (или "актером") и системой для достижения конкретной цели. Он фокусируется на том, как пользователь будет использовать систему, и включает в себя последовательность действий, которые должны быть выполнены, чтобы достичь желаемого результата.

Диаграмма Ганта — инструмент для визуального представления графика выполнения проектов. Она позволяет отображать задачи проекта на временной шкале, показывая, когда каждая задача начинается и заканчивается, а также их продолжительность.

UML (Unified Modeling Language) - стандартный язык моделирования, используемый для визуализации, спецификации, конструирования и документирования артефактов программного обеспечения.

СУБД (Система Управления Базами Данных) - программное обеспечение, которое используется для создания, управления и манипуляции базами данных. СУБД позволяет пользователям выполнять операции, такие как создание, чтение, обновление и удаление данных, а также обеспечивает механизмы для обеспечения целостности данных, безопасности и одновременного доступа.

**ПРИЛОЖЕНИЕ Б**

**Тест кейсы**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Идентификатор | Приоритет | Связанное с тест-кейсом требование | Модуль приложения | Подмодуль приложения | Заглавие | Ожидаемый результат по каждому шагу |

Таблица 5.3.1- Позитивный

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| UG\_U1.12 | Высокий | Пользователь должен иметь возможность просматривать прайс-лист и осуществлять запись. | Прайс - лист | Выбор мастера и услуги | Проверка корректного отображения прайс-листа при выборе мастера и услуги.  Открыть прайс лист.  Выбрать услугу.  Выбрать мастера.  Оставить комментарий мастеру.  Нажать кнопку подтверждения.  Вывод пользователю окна подтверждения с услугами и мастером. | Прайс лист становится активен.  Услуга добавляется в БД пользователя.  Мастер добавляется в БД пользователя.  Сохранение комментария в БД пользователя.  Вывод выбранных услуг пользователю для их уточнения и подтверждения.  Сохранение записи в базе данных |

Таблица 5.3.2-Негативный

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| UG\_U1.13 | Средний | Система должна обрабатывать некорректные данные. | Прайс - лист | Выбор мастера и услуги | Проверка реакции системы на выбор некорректной услуги и мастера.  Открыть прайс лист.  Выбор мастера без наличия свободной записи.  Отсутствие выбора услуги.  Отсутствие выбора мастера. | Прайс лист становится активен  Вывод ошибки и ее причина.  Вывод ошибки и напоминания о выборе услуги.  Вывод ошибки и предложенного мастера на данную услугу. |

Таблица 5.3.3-Деструктивный

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| UG\_U1.14 | Низкий | Система должна быть защищена от вредоносных действий. | Прайс - лист | Выбор мастера и услуги | Проверка на возможность ввода вредоносных команд.  Открыть прайс лист.  Вод в поле SQL команд, запрашивающих доступ к базам данных.  Открытие кода страницы сайта.  Выбор максимального количества услуг и мастеров за минимальное время (Спам). | Прайс лист становится активен  Система игнорирует команду и не сохраняет комментарий, сообщая пользователю о некорректном вводе.  Запрет открытия кода сайта.  Защита от спама. Отклонение бронирования услуг. |