**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc167971590)

[1. Анализ предметной области 5](#_Toc167971591)

[1.1. Общая характеристика 5](#_Toc167971592)

[1.2. Целевая аудитория 6](#_Toc167971593)

[1.3. Анализ конкурентов 7](#_Toc167971594)

[2. Разработка концепции и построение бизнес-модели 9](#_Toc167971595)

[2.1. Концепция веб-сервиса 9](#_Toc167971596)

[2.2. Бизнес-модель 10](#_Toc167971597)

[3. Разработка дизайна 11](#_Toc167971598)

[3.1. Дизайн главной страницы 11](#_Toc167971599)

[3.2. Элементы входа и регистрации 14](#_Toc167971600)

[3.3. Дизайн личного кабинета 16](#_Toc167971601)

[4. Реализация веб-приложения 19](#_Toc167971602)

[4.1. Разработка клиентской части 19](#_Toc167971603)

[4.1.1. Выбор среды разработки 19](#_Toc167971604)

[4.1.2. Разработка компонентов 20](#_Toc167971605)

[4.1.3. Разработка маршрутизации 22](#_Toc167971606)

[4.1.4. Оформление стилей 22](#_Toc167971607)

[4.2. Разработка серверной части 24](#_Toc167971608)

[4.2.1. Выбор среды разработки 24](#_Toc167971609)

[4.2.2. Разработка концептуальной модели БД 24](#_Toc167971610)

[4.2.3. Разработка RESTful API 26](#_Toc167971611)

[4.2.4. Разработка среды администрирования 28](#_Toc167971612)

[5. Безопасность и защита данных 33](#_Toc167971613)

[5.1. Меры безопасности 33](#_Toc167971614)

[5.2. Защита личной информации пользователя 35](#_Toc167971615)

[6. Тестирование веб-приложения 37](#_Toc167971616)

[6.1. Тестирование серверной части 37](#_Toc167971617)

[6.2. Тестирование клиентской части 38](#_Toc167971618)

[ЗАКЛЮЧЕНИЕ 41](#_Toc167971619)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 42](#_Toc167971620)

[ПРИЛОЖЕНИЕ А 45](#_Toc167971621)

[ПРИЛОЖЕНИЕ Б 46](#_Toc167971622)

[ПРИЛОЖЕНИЕ В 47](#_Toc167971623)

# **ВВЕДЕНИЕ**

Для реализации новых бизнес-идей сайты становятся неотъемлемой частью стратегии, поскольку они способствуют увеличению потока пользователей. Все больше людей обращаются к интернету для поиска нужных товаров и услуг, что делает наличие качественного веб-сервиса необходимостью для успешного функционирования компании.

**Актуальность темы.** В нынешнее время электроника является неотъемлемой частью нашей жизни, и доступ к современным и функциональным гаджетам становится всё более сложным. Это связано с быстрым технологическим прогрессом, который стимулирует постоянное обновление аппаратного и программного обеспечения. Для многих людей, особенно в условиях экономической нестабильности, приобретение новейших устройств может быть недоступной роскошью. В этом контексте аренда электроники представляет собой ценное решение, позволяющее пользоваться необходимыми устройствами на временной основе без необходимости полной покупки.

Актуальность темы выпускной квалификационной работы определила цели, предмет, объект, задачи исследования и ожидаемые результаты.

**Цель работы.** Спроектировать и разработать веб-сервис для аренды цифровой электроники, который будет отвечать современным требованиям рынка и предпочтениям пользователей.

**Объект исследования.** Создание инновационного сервиса аренды электроники, максимально адаптированного к потребностям пользователей и условиям рыночной экономики.

**Предмет исследования –** сервис аренды электроники

**Задачи исследования:**

* Провести комплексный анализ рынка аренды цифровой электроники.
* Выявить ключевые потребности и предпочтения целевой аудитории.
* Разработать и валидировать концепцию предлагаемого веб-сервиса.
* Спроектировать и реализовать функциональные возможности сервиса, включая интуитивно понятный пользовательский интерфейс.
* Обеспечить высокий уровень безопасности обработки и хранения пользовательских данных.

**Метод исследования.** В выпускной квалификационной работе исследование основывается на комбинации общетеоретических и эмпирических методов. В процессе работы будет выполнен анализ существующих аналогов, определены основные компоненты системы и технологические решения для её реализации. Особое внимание уделено архитектуре базы данных, методам обеспечения безопасности данных, а также стратегиям внедрения и сопровождения веб-сервиса.

**Ожидаемые результаты.** Получение глубоких знаний и практических навыков в области проектирования и разработки веб-сервисов, что станет значимым вкладом в профессиональное развитие автора в сфере информационных технологий. Разработанный сервис будет способствовать оптимизации расходов потребителей на электронику и повышению доступности высокотехнологичных устройств для широкой аудитории.

**Структура работы.** Выпускная квалификационная работа состоит из введения, шести глав основного текста на 40 страницах, заключения, списка использованных источников, содержащего 28 библиографических источников. В работе 25 рисунков, 5 таблиц, 3 приложения, в которых представлены дополнительные материалы, использованные при написании теоретических и практических частей.

1. **Анализ предметной области**
   1. **Общая характеристика**

Веб-сервис аренды электроники предлагает эффективное решение для временного доступа к широкому спектру электронных устройств, от мобильных колонок до высокопроизводительных компьютеров. Платформа обеспечивает пользователей необходимым оборудованием для коротких периодов использования или для оценки перед покупкой, устраняя необходимость в больших капиталовложениях. [[2]](#_СПИСОК_ИСПОЛЬЗОВАННЫХ_ИСТОЧНИКОВ)

В ходе анализа предметной области были изучены потребности рынка аренды электроники, его особенности и тенденции. Основные потребности рынка аренды электроники продемонстрированы в таблице 1.1.

Таблица 1.1 – Основные потребности рынка

|  |  |  |  |
| --- | --- | --- | --- |
| Категория | Потребности | Особенности | Тенденции |
| Выбор устройств | Широкий выбор электронных устройств для аренды | Постоянное обновление ассортимента арендуемых устройств | Увеличение количества арендуемых устройств |
| Конкуренция | Доставка и конкурентоспособность цен | Высокая конкуренция среди арендодателей | Увеличение количества арендодателей и конкуренции |
| Надёжность и качество | Надёжность и качество арендуемых устройств | Необходимость обеспечения надёжности и качества арендуемых устройств | - |
| Доставка | Быстрая и качественная доставка | Необходимость обеспечения быстрой и качественной доставки | - |
| Удобство использования | Удобство и простота использования сервиса | - | Увеличение количества пользователей |

Фокус на потребностях пользователей становится основным фактором успеха на рынке аренды электроники. Понимая и активно удовлетворяя эти потребности, компании могут эффективно привлекать и удерживать клиентов, что позволяет им занимать лидирующие позиции в данной сфере.

* 1. **Целевая аудитория**

Целевой аудиторией сервиса аренды электроники являются широкие категории потенциальных клиентов, объединенных общими потребностями и предпочтениями. Целевая аудитория продемонстрирована на рисунке 1.1



Рисунок 1.1 – Целевая аудитория сервиса аренды электроники

Целевая аудитория формируется с учетом разнообразия потребностей и стилей жизни, что позволяет сервису привлекать широкий спектр клиентов.

* 1. **Анализ конкурентов**

Проведенный SWOT анализ [[13]](#_СПИСОК_ИСПОЛЬЗОВАННЫХ_ИСТОЧНИКОВ) рынка аренды электроники выявил, что конкуренция между нашей, "Мой девайс" и «Мой прокат» компаниями основывается на четырёх ключевых параметрах, продемонстрированных в таблице 1.2.

Таблица 1.2 – SWOT анализ рынка

|  |  |  |  |
| --- | --- | --- | --- |
| SWOT | Наша компания | Конкурент «Мой девайс» | Конкурент «Мой прокат» |
| Сильные стороны | Удобный личный кабинет;  Система бонусов;  Большой ассортимент;  Приятные цены; | Большой ассортимент;  Логистическая компания настроена | Понятный сайт;  Огромный ассортимент |
| Слабые стороны | Нет логистики, для доставки | Нет системы бонусов | Нет системы бонусов |
| Возможности | Расширение по городам;  Увеличения ассортимента; | Конкуренция с новой компанией; | Конкуренция с новой компанией; |
| Угрозы | Конкурент может задавить;  Не будет потока арендаторов; | Конкурент заберёт большую часть арендаторов | Конкурент заберёт большую часть арендаторов |

Исследование также показало, что успешные участники рынка активно используют разнообразные стратегии для привлечения и удержания клиентов. Это включает в себя внедрение технологических решений, улучшение UI (User Interface) и расширение функциональных возможностей платформы, а также работа над UX (User Experience)

Для успешного ведения бизнеса также крайне важно иметь ясное представление о своей целевой аудитории. Для более точного анализа были выбраны те же конкуренты – "Мой девайс" и "Мой прокат". Характеристики ключевого потребителя представлены в таблице 1.3.

Таблица 1.3 – Характеристика ключевого потребителя

|  |  |  |  |
| --- | --- | --- | --- |
| Параметры КП | Наша компания | Конкурент «Мой девайс» | Конкурент «Мой прокат» |
| Возраст | От 18 – 45 лет | | |
| Доход | От 40 до 80+ тыс. рублей | +- 100 тыс. руб. | |
| Критерии выбора сервиса аренды электроники | Удобный личный кабинет | Компания с положительным рейтингом | Большой ассортимент |
| Психографика (AIO) | Умеет экономить  Постоянно в движении  Любит музыку | Учится в универе  Ездит на общественном транспорте | Проживает в большой семье |

Характеристика ключевого потребителя является одним из важных показателей, так как включает в себя множество параметров. Это способствует созданию более глубокому понимаю своего клиента, что, в свою очередь, повышает конкурентоспособность и устойчивость сервиса на рынке. Для нашего сервиса способствующим и важным параметром является AIO [[26]](#_СПИСОК_ИСПОЛЬЗОВАННЫХ_ИСТОЧНИКОВ):

* Activity (деятельность, действия) – очевидный поступок, например, поход в магазин, просмотр телевизора. Сюда относится работа, хобби, социальные сети, отдых и т.д.
* Interest (интерес) – по отношению к какому-либо объекту, событию или теме – это степень возбуждения, сопровождаемая особым и продолжительным вниманием. Сюда относится семья, дом, работа, отдых и т.д.
* Opinion (мнение) – выражение устно или письменно ответ человека на ситуацию, в которой ставится какой-то вопрос. Используется для описания понимания, ожиданий и оценок, таких как убеждения и намерения других людей, предчувствия будущих событий, понимание плюсов и минусов последствий различных вариантов поведения. Сюда относится политика, бизнес, экономика, образование, культура и т.д.

1. **Разработка концепции и построение бизнес-модели**
   1. **Концепция веб-сервиса**

Для разработки концепции веб-сервиса, стоит сначала описать определение. Концепция проекта – это некий документ, в котором описаны основные цели, задачи проекта и способы достижения. Для нашего веб-сервиса определены собственные цели, задачи и способы достижения:

* Удобный и интуитивный пользовательский интерфейс: клиентский опыт будет приоритетным, обеспечивая лёгкость и комфорт при поиске, выборе и оформлении аренды.
* Широкий ассортимент электронных устройств для аренды: разнообразие товаров удовлетворит запросы разных групп пользователей, от студентов до предпринимателей.
* Доступная и конкурентоспособная ценовая политика: разработка гибкой ценовой стратегии с учётом конкуренции на рынке аренды электроники.
* Гарантия надёжности и качества арендуемых устройств: поставка высококачественных электронных устройств, подкреплённая системой гарантий и проверок перед сдачей в аренду.
* Возможность онлайн-оплаты: внедрение удобных и безопасных систем онлайн-оплаты для большего комфорта клиентов.
* Функция отслеживания статуса заказа: предоставление функции отслеживания для информирования клиентов о текущем статусе их заказа.
* Возможность оставлять отзывы и оценки арендуемых устройств: создание платформы для обратной связи, позволяющей клиентам делиться впечатлениями и оценивать арендованные устройства.
  1. **Бизнес-модель**

Бизнес-модель сервиса аренды электроники построена на основе модели Остервальдера, которая описывает девять основных элементов, необходимых для успешного функционирования и развития бизнеса. Эти аспекты включают в себя ключевых партнёров, ключевые активности, ценностные предложения, отношения с клиентами, сегменты потребителей, ключевые ресурсы, каналы поставки, структуру издержек, источники дохода.

Данная бизнес-модель опирается на детализированный анализ и описание каждого из элементов модели Остервальдера ([Приложение А](#_ПРИЛОЖЕНИЕ_А_2))

В частности, она должна включать следующие элементы:

* Аренда электронных устройств за плату: Основной источник дохода - плата за аренду различных электронных устройств.
* Возможность покупки арендуемых устройств: Дополнительная опция для клиентов, предоставляющая возможность выкупа арендованных товаров.
* Возможность дополнительных услуг: Предоставление дополнительных услуг, таких как страхование, доставка и установка, для дополнительного дохода.
* Возможность рекламы и сотрудничества с производителями: Возможности для рекламы электроники и партнерских отношений с производителями для дополнительных источников дохода.

1. **Разработка дизайна**
   1. **Дизайн главной страницы**

Цветовая схема интерфейса сервиса аренды электроники предполагает использование темной темы. Палитра цветов представлена в таблице 3.1.

Таблица 3.1 – Палитра цветов

|  |  |  |
| --- | --- | --- |
| Фоновый | Акцентный | Второстепенный |
| # 101010 | #F77F00 | #0059FF |

Данная цветовая палитра должна сосредоточить внимание пользователя на ключевых элементах интерфейса, обеспечивая эффективное восприятие информации.

Типографика интерфейса сервиса аренды электроники выбрана так, чтобы быть четкой и легко читаемой. Основным шрифтом, используемым в дизайне, является Onest. Этот шрифт применяется для выделения важных элементов, таких как заголовки, подзаголовки и основной текст.

На главной странице предусмотрены различные секции ([Приложение Б](#_ПРИЛОЖЕНИЕ_А_1)). Дизайн главной страницы был неоднократно уточнен для достижения оптимальной композиции.

Дизайн главной страницы был не раз переработан из-за нестыковки в композиции. Конечный вариант имеет строгий вид, необходимую информацию о данном сайте и процессе аренды.

Преимущественным референсом стал стиль Bento Design. Bento Design – это стиль, который произошёл от Японского ланч-бокса, где объединяются несколько секций с едой в один контейнер. В понимании web-дизайна, бенто-стиль разделяет страницы на четко определённые блоки или секции, каждый из который имеет собственный функционал или логику. Пример Bento Design продемонстрирован на рисунке 3.1

Bento Style Presentation Grid System for Adobe InDesign | Sample Layout – Dark Background


Рисунок 3.1 – Bento Design

Для визуальной привлекательности главной страницы была разработана композиция изображения для главной страницы. Для лучшего восприятия и как должна выглядеть композиция, был найден референс 3D изображения, продемонстрированный на рисунке 3.2.

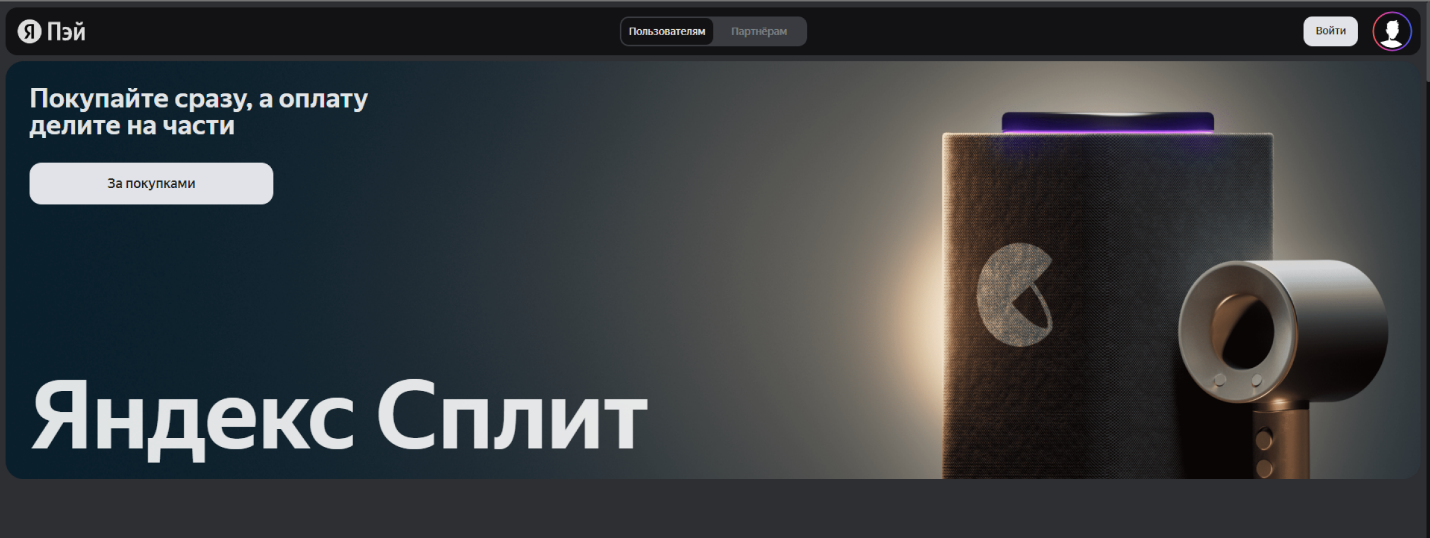


Рисунок 3.2 – 3D изображение

Этот элемент не только украшают страницу, но и подчеркивают её стиль и уникальность, делая интерфейс более динамичным и привлекательным для пользователей. Для реализации композиции, был выбран инструмент разработки Blender. Он позволяет осуществлять 3D визуализацию, скульптинг и много другое. Процесс разработки 3D композиции для сервиса аренды электроники продемонстрирован на рисунке 3.3.

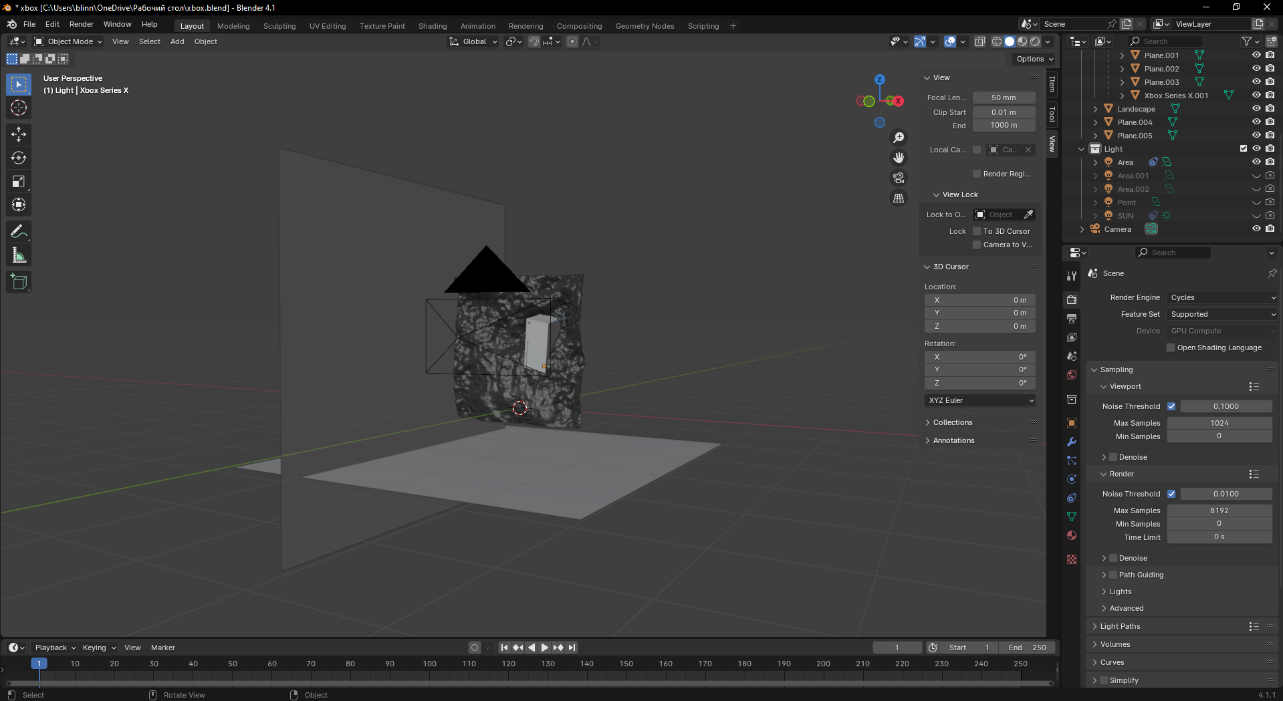


Рисунок 3.3 – Процесс разработки композиции

Акцентам предметом был выбран Xbox Series X, благодаря его строгому и в то же время привлекательному дизайну. Этот выбор позволяет не только подчеркнуть современные технологические достижения, но и привлечь внимание аудитории к эстетической составляющей устройства, делая его визуально выразительным элементом в нашем контенте. На рисунке 3.3 продемонстрированы Viewport Shading’s (Шейдинги Вьюпорта).



Рисунок 3.4 – Viewport Shading (Шейдинги Вьюпорта)

Изначально был разработан ландшафт заднего фона, который создает подходящую атмосферу. Затем была тщательно подобрана детализированная модель Xbox Series X, чтобы обеспечить визуальное соответствие высоким стандартам. В завершение процесса настройки были тонко скорректированы освещение и световые элементы, что позволило идеально подчеркнуть ключевые особенности консоли, улучшая общее визуальное восприятие.

* 1. **Элементы входа и регистрации**

Дальнейшие шаги были направлены на разработку процесса входа и личного кабинета, который также был адаптирован для более удобной верстки. Приведенный, на рисунке 3.5, экран демонстрирует дизайн страницы входа.

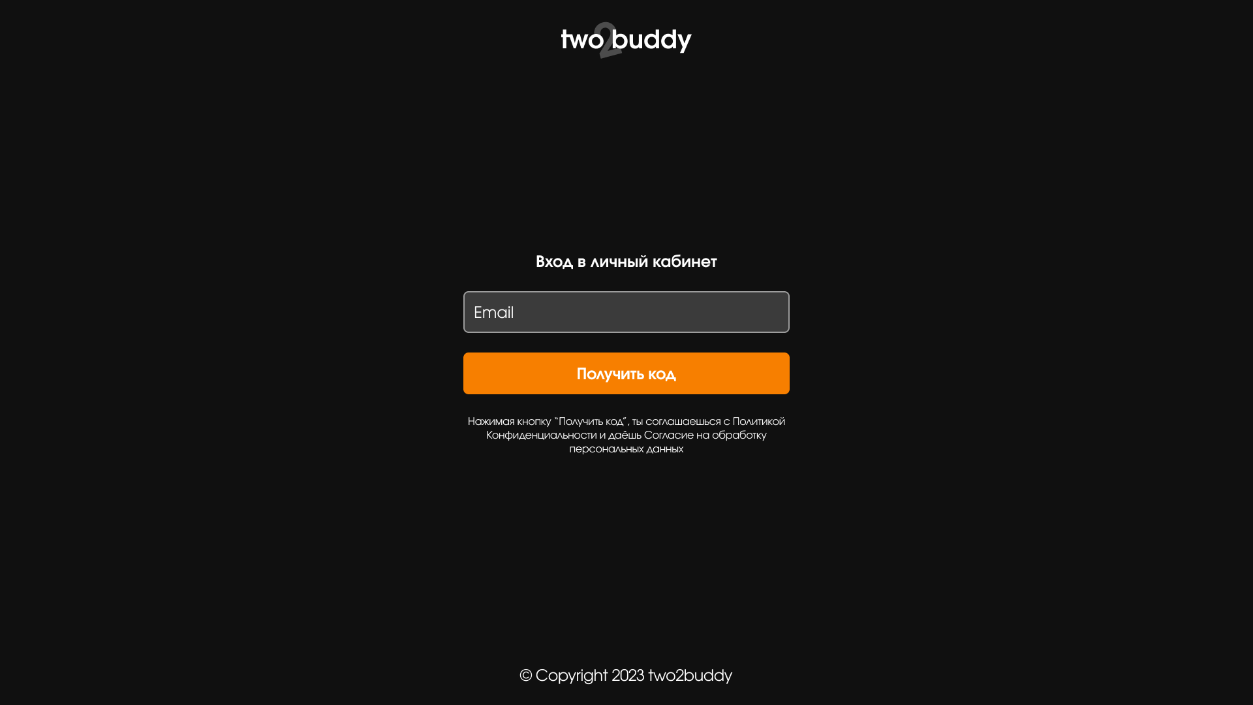


Рисунок 3.5 – Дизайн входа

Концепция разработана с учетом того, что пользователю необходимо будет вводить пароль всего лишь один раз. Вместо того, чтобы сохранять пароль, на указанный адрес электронной почты будет отправлен одноразовый код подтверждения, как представлено на рисунке 3.6.

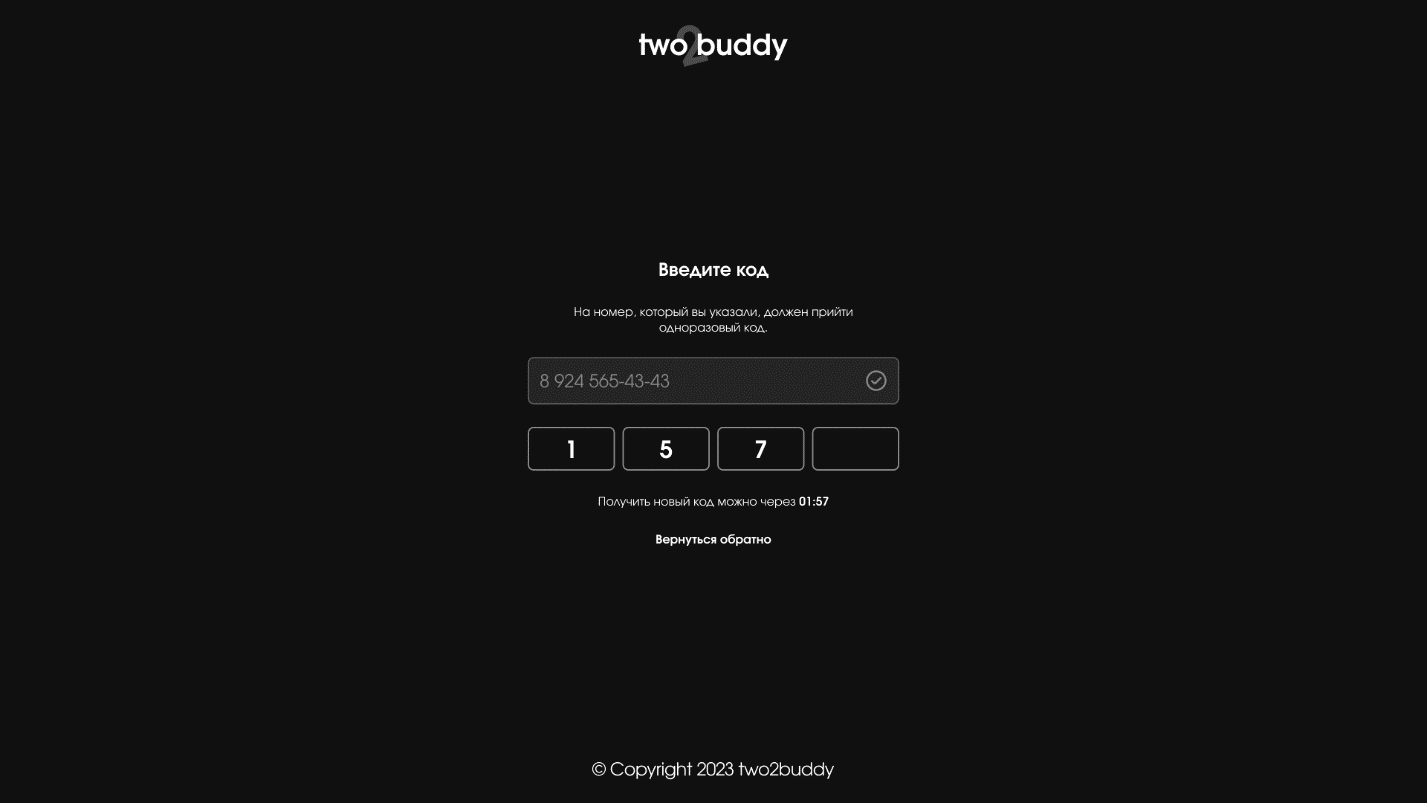


Рисунок 3.6 – Подтверждение входа кодом

* 1. **Дизайн личного кабинета**

После ввода четырёхзначного кода, пользователя перенаправляет на личный кабинет, где первым делом надо заполнить персональные данные. Если пользователь решил не вводить персональные данные, то система отклоняет любую аренду. Дизайн личного кабинета представлен на рисунке 3.7

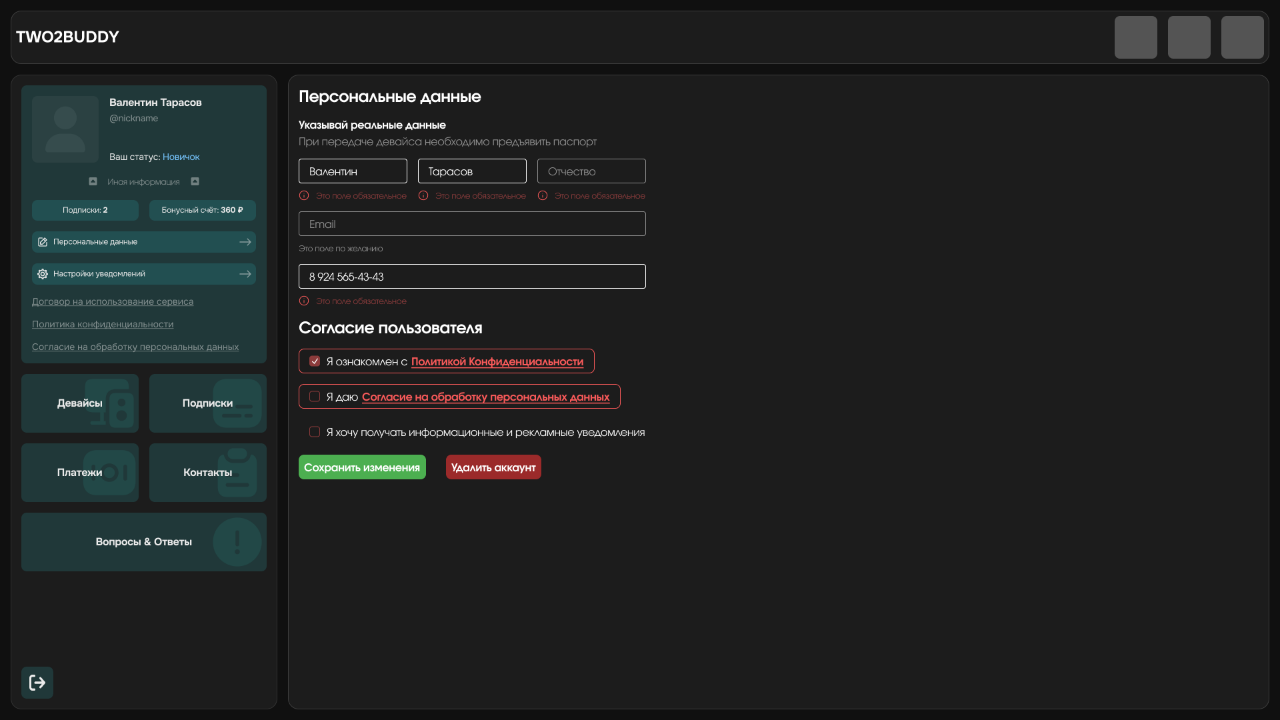


Рисунок 3.7 – Дизайн личного кабинета

Личный кабинет оформлен в соответствии со стилем «Bento Design», обеспечивая удобное и интуитивно понятное распределение элементов. Структура включает в себя боковую панель (aside), главную панель (main) и верхнюю панель (header). Боковая панель содержит всю необходимую навигацию, что позволяет пользователю сосредоточить внимание на основных функциях. Главные навигационные кнопки, такие как "Устройства", "Подписки", "Платежи", "Контакты" и "Вопросы & Ответы", легко доступны для быстрого перехода.

В верхней панели представлена краткая информация о текущем аккаунте, а также предоставлены удобные средства быстрого доступа к ключевым функциям. Этот дизайн призван сделать навигацию по личному кабинету интуитивно понятной и эффективной для пользователей.

Следующей страницей личного кабинета, является процесс оформления аренды. Дизайн процесса оформления аренды представлен на рисунке 3.8.

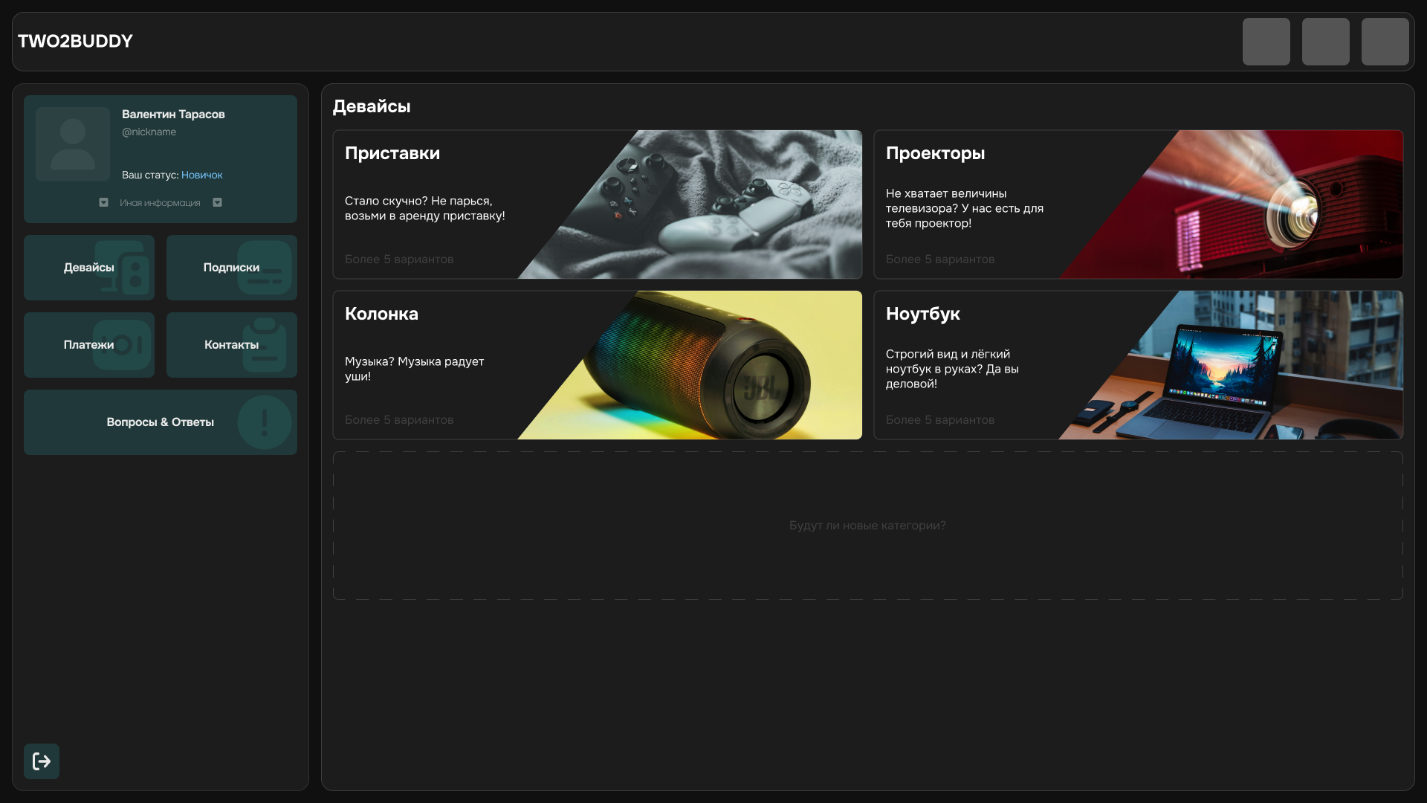


Рисунок 3.8 – Дизайн процесса оформления аренды

Пользователь выбирает категорию аренды и следует на страницу выбора девайса, где предоставляется несколько, на выбор, товаров. Далее пользователь выбирает желаемый срок аренды, который составляет администратор сервиса. После этих действий, пользователь попадает на страницу подтверждения оформления аренды. Дизайн подтверждения оформления аренды представлен на рисунке 3.9.

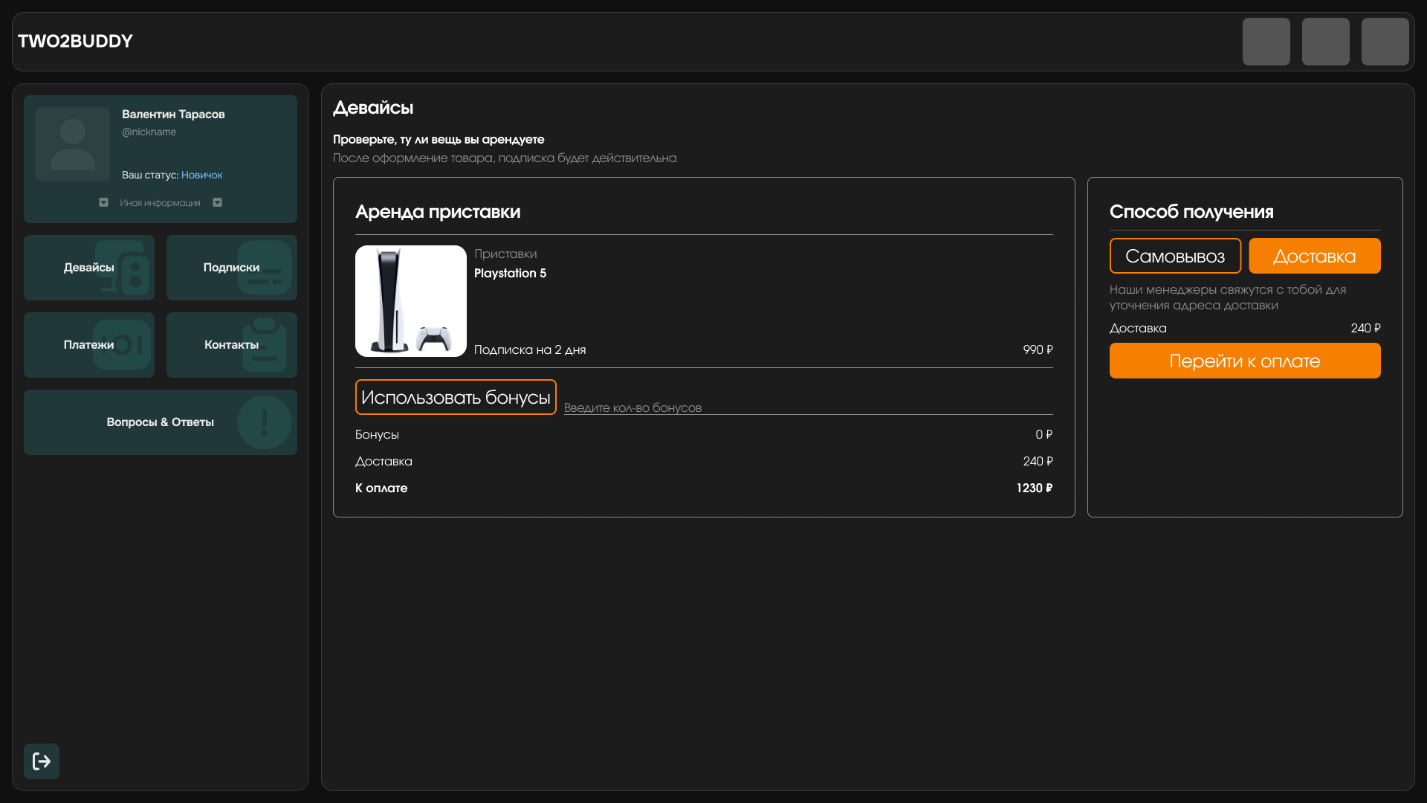


Рисунок 3.9 – Дизайн подтверждения оформления аренды

На данной странице, пользователю необходимо проверить все поля, которые ему представлены, чтобы избежать ненужного оформления аренды. Если пользователь все такие оформил аренду, то он может ещё отменить, так как процесс оплаты происходит после оформления аренды.

Для подробного ознакомления с каждой страницей и элементами личного кабинета рекомендуется ознакомиться с дизайном на платформе ([Figma](https://www.figma.com/design/7CZhKzCEWsD9HH7BvtL9SV/Разработка-сервиса-аренды-электроники?node-id=767-7099&t=utcXpkS9gKDtcHQH-0)). Там представлены все разработанные элементы и страницы, чтобы вы могли полностью оценить внешний вид и функционал вашего личного кабинета.

1. **Реализация веб-приложения**
   1. **Разработка клиентской части** 
      1. **Выбор среды разработки**

Перед началом верстки необходимо определится со средой разработки, таковой является библиотека React и инструмент сборки Vite [[1, 3, 4]](#_СПИСОК_ИСПОЛЬЗОВАННЫХ_ИСТОЧНИКОВ). Этот выбор обеспечил масштабируемость при создании компонентов для повторного использования. Для предотвращения путаницы в большом количестве компонентов и файлов первоначально была выбрана компонентная архитектура, представленная на рисунке 4.1.1.

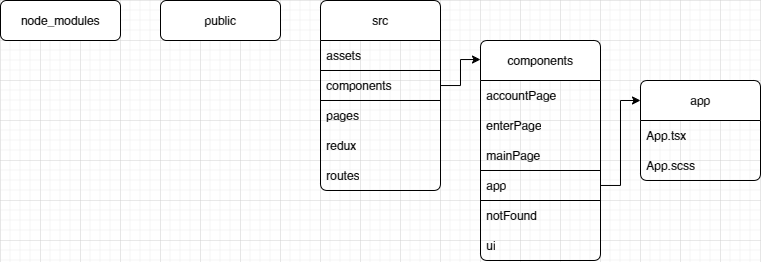


Рисунок 4.1.1 – Компонентная архитектура – неактуальная.

Структура проекта включает множество папок, каждая из которых выполняет свою определенную задачу и содержит соответствующие файлы. Основной папкой для верстки сайта является «Components», где сосредоточены все компоненты, необходимые для построения сайта.

Такая архитектура является базовой для небольших проектов, у которых команды разработчиков составляет не более 5 человек. Для веб-сервиса аренды цифровой электроники была выбрана востребованная, на сегодняшний день, архитектура «Future-Sliced Design» [[28]](#_СПИСОК_ИСПОЛЬЗОВАННЫХ_ИСТОЧНИКОВ) (далее FSD). FSD – это архитектурная методология построения интерфейсных приложений, иначе, это сборник правил и условностей по организации кода. Целью методологии является построить проект более понятным и структурированным в условиях постоянно меняющихся бизнес-требований. На рисунке 4.1.2 представлена FSD архитектура проекта – актуальная.

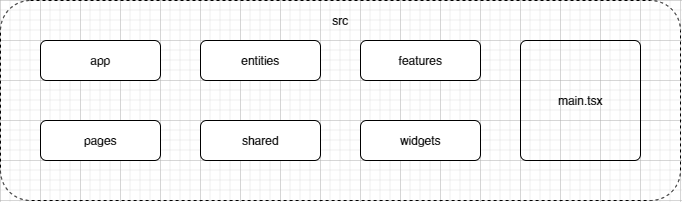


Рисунок 4.1.2 – FSD архитектура проекта – актуальная.

Перейдя на данную архитектуру, мы избавились от захламлённости компонентов в общей папке «Components», что позволило настроить бизнес-логику и оптимизировать общую работу над данным проектом.

* + 1. **Разработка компонентов**

Исходя из документации архитектуры, весь проект — это слой, каждый слой состоит из слайсов и каждый срез состоит из сегментов. На рисунке 4.1.3 продемонстрированы правила построения компонента.

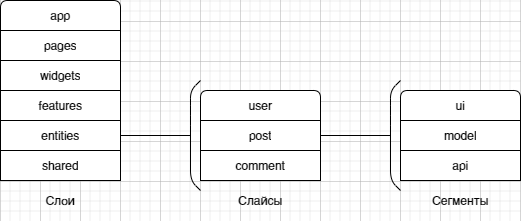


Рисунок 4.1.3 – Правила построения компонента.

Модули на одном уровне могут взаимодействовать только с модулями расположенных строго ниже, допустим, «Shared» не может взаимодействовать с «App», но наоборот возможно. Исходя из версии документации – 2.0.0, всего 6 модулей:

1. App – настройки, общие стили, роутинг, поставщик для всего приложения.
2. Pages – композиционный слой для создания полных из объектов, функций и виджетов.
3. Widgets – композиционный слой для объединения сущностей и функции в значимые блоки, например, профиль пользователя.
4. Features – взаимодействие с пользователем, действия, которые приносят пользу пользователю для бизнеса, например, добавление в корзину, оформления заказа, поиск, отправка комментария и т.д.
5. Entities – бизнес-объекты, например, пользователь, продукт, заказ.
6. Shared – функциональность многократного использования, отдельная от специфики проекта, например, UIKit, библиотеки, API.

Далее слайсы – фрагменты, они разделяют код на бизнес-области. Это существенно упрощает навигацию по кодовой базе, потому что логически связанные модули расположены близко друг к другу. Каждый фрагмент состоит из сегментов. Эти модули предназначены для разделения кода внутри фрагмента по его техническому назначению.

Как говорилось ранее, архитектура FSD позволяет организовано разрабатывать компоненты [[4]](#_СПИСОК_ИСПОЛЬЗОВАННЫХ_ИСТОЧНИКОВ). Это обеспечивает удобство и структурированность в процессе разработки и поддержки кода. Каждый компонент представлен в отдельном файле или папке в соответствии с лучшими практиками именования файлов и структурирования кода.

Каждый модуль содержит необходимую логику и представление для своей функциональности, что делает код более масштабируемым. Это позволяет разработчикам легко находить, понимать и изменять компоненты при необходимости, а также повторно использовать их в различных частях приложения. Данная архитектура способствует удобству внедрения тестирования компонентов, так как каждый компонент может быть протестирован отдельно, что обеспечивает надёжное и стабильное функционирование приложения.

* + 1. **Разработка маршрутизации**

В процессе разработки маршрутизации проекта используется файл «Router.tsx» ([Приложение В](#_ПРИЛОЖЕНИЕ_В))», который играет ключевую роль в определении путей и переходов между различными страницами приложения. Этот файл устанавливает основные правила маршрутизации, определяя, какие компоненты будут загружены при обращении к определенным URL-адресам.

Для определения маршрутов в React-приложении используется «React-Router-DOM» – набор навигационных компонентов, который предоставляет удобный инструмент и способ синхронизации пользовательского интерфейса с URL-адресом, для управления навигацией в веб-приложении.

Внутри файла «Router.tsx» определяются маршруты с помощью компонента родительского компонента «Routes» и дочернего «Route», указывая путь и соответствующий компонент, который должен быть отображен при совпадении URL.

Такой подход обеспечивает эффективное управление навигацией в приложении, делая процесс перехода между страницами непрерывным.

* + 1. **Оформление стилей**

При оформлении стилей мы придерживаемся строгих правил, обеспечивая согласованный и привлекательный дизайн [[21, 22]](#_СПИСОК_ИСПОЛЬЗОВАННЫХ_ИСТОЧНИКОВ). Файлы стилей размещаются в соответствующих папках, что значительно облегчает их обслуживание и дальнейшее развитие. Для лучшего понимания принятых правил создания стилей, представлена диаграмма на рисунке 4.1.4.

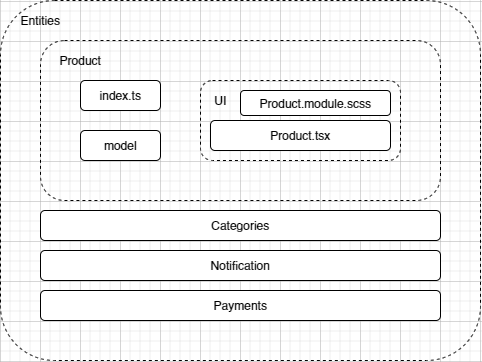


Рисунок 4.1.4 – Правило создания стилей

Для создания стиля компонента, первоначально создаётся общая папка «Products», это означает, что вся бизнес-логика будет находится в этой папке. Далее, в папке UI, создаётся файл или отдельная папка, это позволяет разделить бизнес-логику. На нашем примере, имеется только одна логика «Product».

В процессе оформления стилей мы используем два вида файлов: глобальные, например «App.css» и приватные (модульные), например «Product.module.scss». Глобальные стили применяются ко всем блокам и элементам на сайте, обеспечивая единое визуальное оформление. Приватные стили, также известные как модульные, применяются только к определенному файлу, в котором они были импортированы, и обладают именем файла, содержащим "module". Этот подход предоставляет возможность применять стили к определенным элементам без опасений, что они повлияют на другие компоненты на странице. Такая структура облегчает управление стилями и предотвращает конфликты между ними, что в свою очередь способствует более стабильному и надежному дизайну веб-приложения.

* 1. **Разработка серверной части**
     1. **Выбор среды разработки**

Для разработки серверной части веб-сервиса аренды электроники была выбрана среда выполнения JavaScript программ – Node.js. Данный выбор был обоснован высокой производительностью Node.js, возможностью эффективно обрабатывать асинхронные операции и широким сообществом разработчиков, что обеспечивает доступ к множеству библиотек и решений.

Node.js позволил создать быстрый и масштабируемый backend для обработки запросов пользователей, управления базой данных и обеспечения взаимодействия с клиентской частью приложения. Для создания RESTful API был использован фреймворк Express.js, который обеспечивает удобное маршрутизированние запросов и обработку HTTP-запросов.

* + 1. **Разработка концептуальной модели БД**

Перед разработкой серверной части приложения, была разработана концептуальная модель базы данных, чтобы обеспечить понимание того, как должна выглядеть структура данных и схема приложения.

Концептуальная модель базы данных включала в себя определение основных сущностей (таблиц) и их атрибутов, а также связи между этими сущностями. Была использована методика проектирования ER (Entity-Relationship) для моделирования данных, что позволило лучше понять логику взаимосвязей между различными компонентами приложения.

Основными сущностями веб-сервиса являются пользователи, девайсы, категории, опции подписки, платежи, аренда. Для каждой из этих сущностей были определены соответствующие атрибуты, которые содержат информацию о сущности.

Также были определены связи между сущностями. Например, у каждой аренды может быть один или несколько продуктов, а каждый продукт может принадлежать только одному пользователю. Эти связи позволили понять, как данные будут взаимодействовать друг с другом в нашем веб-сервисе. Концептуальная модель представлена на рисунке 4.2.1 или в подробном формате «[Модель БД](https://dbdiagram.io/d/660fab9003593b6b61411b06)».

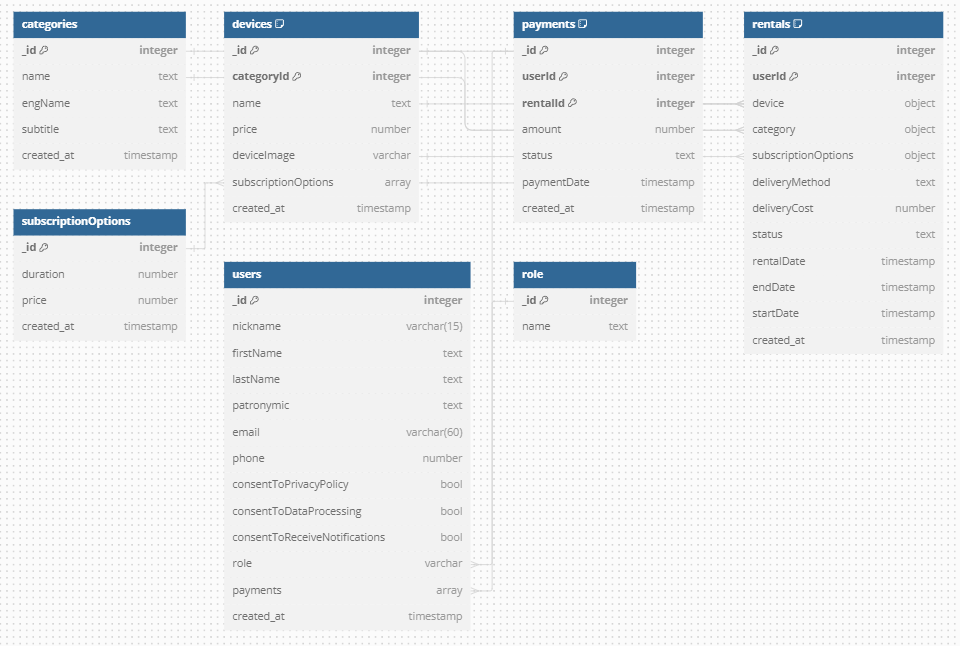
****

Рисунок 4.2.1 – Концептуальная модель БД

Разработка концептуальной модели базы данных [[19, 20]](#_СПИСОК_ИСПОЛЬЗОВАННЫХ_ИСТОЧНИКОВ) перед началом разработки серверной части приложения позволила нам четко определить требования к базе данных и эффективно планировать её структуру и функциональность. Это также обеспечило легкость внесения изменений в структуру данных в случае необходимости в дальнейшем.

* + 1. **Разработка RESTful API**

Для обеспечения взаимодействия между клиентской и серверной частями приложения был разработан RESTful API [[23, 24, 25]](#_СПИСОК_ИСПОЛЬЗОВАННЫХ_ИСТОЧНИКОВ). Для понимания терминологий, разберёмся, что такое REST и API:

* REST (Representational State Transfer) – это архитектурный стиль взаимодействия компонентов распределённого приложения в сети.
* API (Application Programming Interface) – это набор протоколов, с помощью которых клиентская и серверная часть приложения взаимодействует друг с другом.

На рисунке 4.2.2 продемонстрирован пример работы REST API

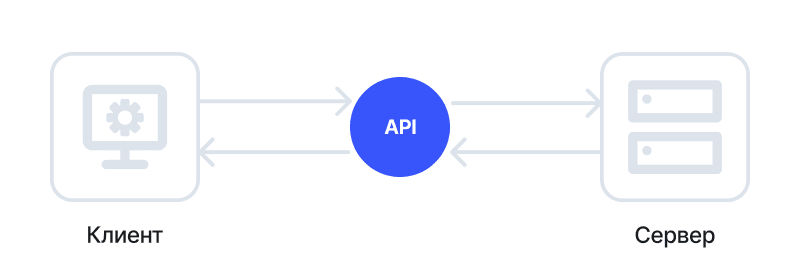


Рисунок 4.2.2 – Пример работы REST API

В качестве маршрутизирования HTTP запросов был использован Express.js – минималистичный и гибкий веб-фреймворк, предоставляющий обширный набор функций для мобильных и веб-приложений. Мы разработали соответствующие маршруты для выполнения операций CRUD (создание, чтение, обновление, удаление) для взаимодействия с сервером.

Примеры некоторых эндпоинтов RESTful API веб-сервиса:

* GET/api/v1/users – получить список всех пользователей;
* GET/api/v1/devices/:id – получить информацию о девайсе с определённым идентификатором;
* PATCH/api/v1/devices/:id – обновить информацию о девайсе с определённым идентификатором;
* DELETE/api/v1/devices/:id – удалить девайс с идентификатором;
* GET/api/v1/rentals/?userId=id – получить список всех заказов с параметром id пользователя.

Большинство запросов к API требует аутентификации, чтобы обеспечить безопасность и конфиденциальность данных, для этого использовали механизм аутентификации и авторизации – сессии. Для обеспечения безопасности на стороне клиента, был разработан приватный роут – «PrivateRouter», который не позволяет перейти на страницу, которая подразумевает предварительную аутентификацию пользователя. Схема работы приватного роута представлена на рисунке 4.2.3.

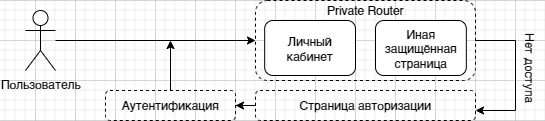


Рисунок 4.2.3 – Схема работы приватного роута

Для полного понимания функциональности и возможностей нашего API сервиса аренды электроники, приглашаем вас ознакомиться с документацией "[Документация web-сервиса](https://documenter.getpostman.com/view/32477520/2sA2rDxgEB#2972be0b-cd66-4905-ae8c-ea4b7f94d8d8)". Документирование от «Postman» предоставляет подробное описание всех эндпоинтов, доступных запросов, параметров и форматов данных, необходимых для взаимодействия с сервисом.

Разработка RESTful API позволила обеспечить эффективное взаимодействие между клиентской и серверной частями приложения, а также обеспечить удобство и гибкость в использовании нашего сервиса для конечных пользователей.

* + 1. **Разработка среды администрирования**

Крайне важным инструментом для web-сервиса аренды цифровой электроники является административная среда, которая позволяет следить за сервисом более информативно. В большинстве случаев, такие среды реализуются для web-платформы, то есть интегрируются прямиком в приложение. Для нашего проекта был выбран путь реализации административной среды через Telegram-бота. Такой метод позволяет разработать только логику на серверной стороне и не писать UI. Преимущества использования бота:

* Удобство использования: Администраторам сервиса нужен только смартфон, чтобы отслеживать статусы или даты оплаты аренды.
* Мгновенные уведомления: Как только клиент оформляет необходимый ему девайс, сообщение сразу же отправляется на бота.
* Возможность с другими сервисам: Telegram-бот легко интегрируется в любой чат, где он необходим, например с системами аналитики.

Разработка бота начинается с Telegram API – программный интерфейсом приложения Telegram, который позволяет разработчикам создавать приложения и ботов, взаимодействующих с этой популярной мессенджер-платформой. Он предоставляет доступ к широкому спектру функций, включая отправку и приём сообщений, управляя чатами, доступ к профилям пользователям и многое другое [[27]](#_СПИСОК_ИСПОЛЬЗОВАННЫХ_ИСТОЧНИКОВ). В первую очередь следует создать бота и запросить токен у «The Botfather» для доступа к HTTP API. Процесс создания и запроса токена продемонстрирован на рисунке 4.2.4.

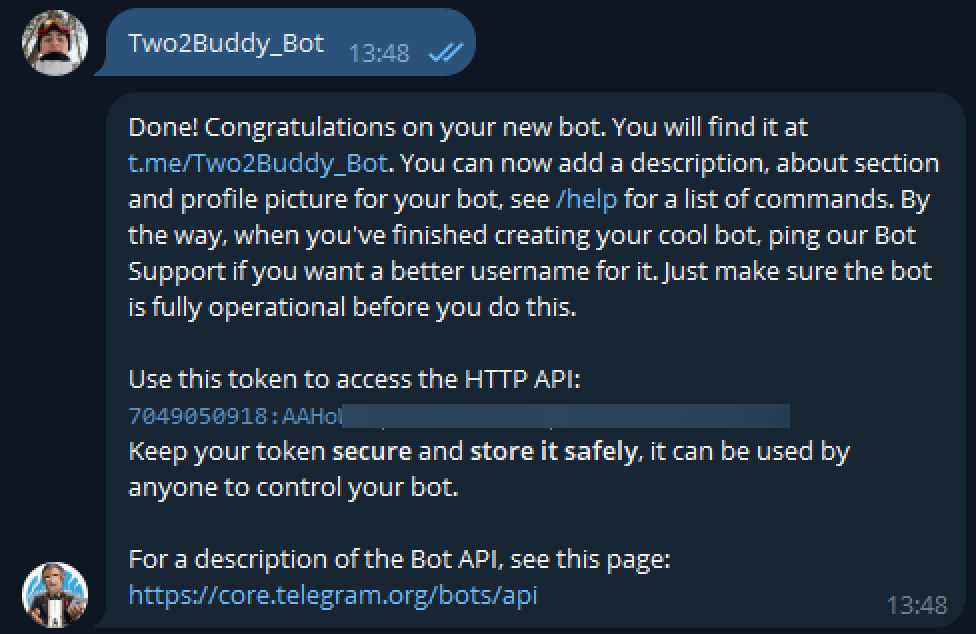


Рисунок 4.2.4 – Процесс запроса токена

Следующее этап, подключение необходимых библиотек и реализация логики телеграмм-бота. Для среды Node.js преимущественно выделяется библиотека «node-telegram-bot-api». Также необходимо структурировать работу над ботом, чтобы в дальнейшем понимать и дополнять её со временем. На рисунке 4.2.5 продемонстрирована структура Telegram-бота.

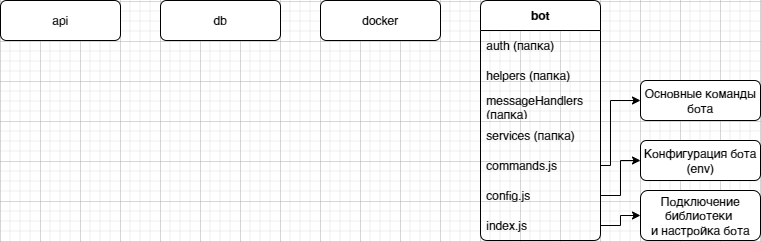


Рисунок 4.2.5 – Структура Telegram-бота

Это базовая структура для многочисленных проектов и не только. Она включает в себя основные папки, которые включают в себя только необходимые по логике файлы, дабы разбить процесс разработки на несколько слоёв. Подключение сторонних библиотек и настройки бота располагается в родительском «index.js» файле. Родительский «index.js» файл представлен на рисунке 4.2.6.



Рисунок 4.2.6 – Родительский «index.js» файл

Процесс тестирования Telegram-бота представляет собой многоразовые запросы на сервер, путём оформления аренды через сервис. На рисунке 4.2.7 продемонстрирована работа Telegram-бота, который присылает сообщения о новой аренде.

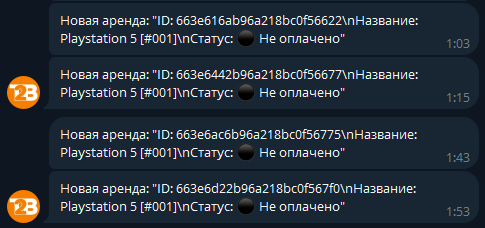


Рисунок 4.2.7 – Работа Telegram-бота.

Данный тест позволяет убедится, что нет иных утечек данных, сообщения доставляются моментально и корректно отображаются данные аренды. Далее, производится тест изменения статуса аренды. На рисунке 4.2.8 продемонстрирован тест изменения статуса аренды.

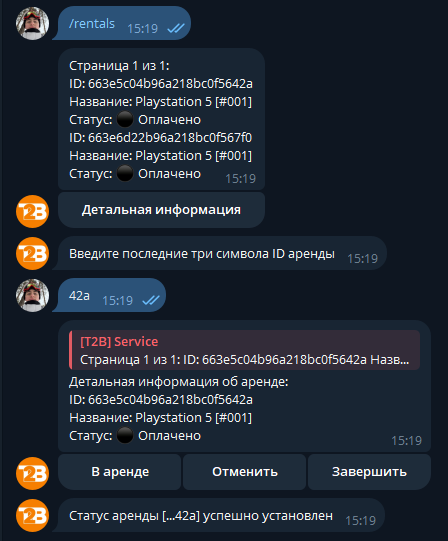


Рисунок 4.2.8 – Тест изменения статуса аренды

Данный тест позволяет убедится, что список аренд и детальная информация выводится, статус корректно изменяется на нужный. Исходя из нынешней версии Telegram-бота, это весь процесс тестирования. В дальнейшем, пункты тестирования будут увеличиваться, чтобы усовершенствовать безопасность Telegram-бота.

1. **Безопасность и защита данных**

С каждым днем технологический прогресс неуклонно продвигается вперед, открывая новые возможности, но при этом также повышая риски кражи персональных данных. В ответ на эти вызовы активно разрабатываются и внедряются новые методы защиты от подобных атак, такие как использование современных алгоритмов шифрования, многоуровневая система аутентификации и постоянный мониторинг безопасности данных.

* 1. **Меры безопасности**

В рамках разработки веб-сервиса аренды цифровой электроники реализуется механизм аутентификации пользователей с использованием сессионного метода и входа по электронной почте с дальнейшей отправкой одноразового кода. Этот метод позволяет повысить безопасность входа в систему, предотвращая несанкционированный доступ.

При попытке входа в систему пользователь вводит свой адрес электронной почты, который является уникальным идентификатором. После этого система генерирует одноразовый код и отправляет его на указанный адрес электронной почты пользователя, пример продемонстрирован на рисунке 5.1. [[16, 17, 18]](#_СПИСОК_ИСПОЛЬЗОВАННЫХ_ИСТОЧНИКОВ)

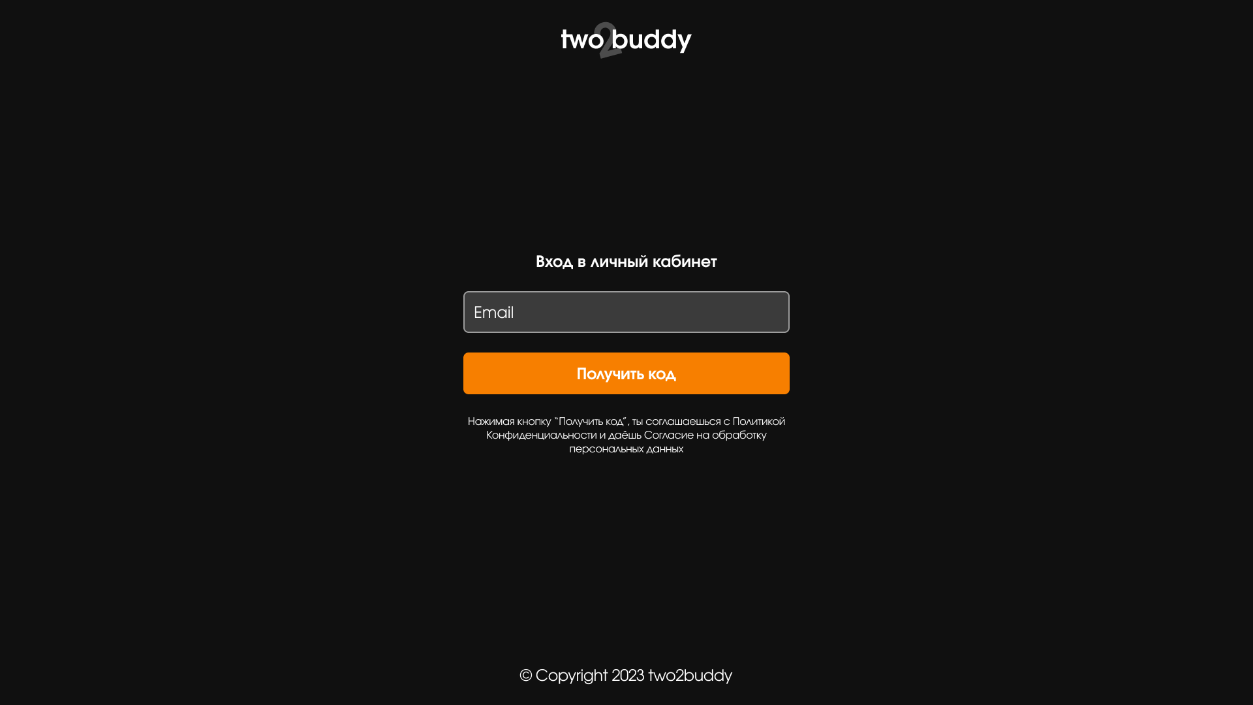


Рисунок 5.1 – Вход пользователя по почте

Для завершения процесса аутентификации пользователь вводит полученный одноразовый код в специальное поле на веб-странице входа. После успешного сопоставления кода с тем, который был отправлен на электронную почту, пользователь получает доступ к личному кабинету, пример продемонстрирован на рисунке 5.2

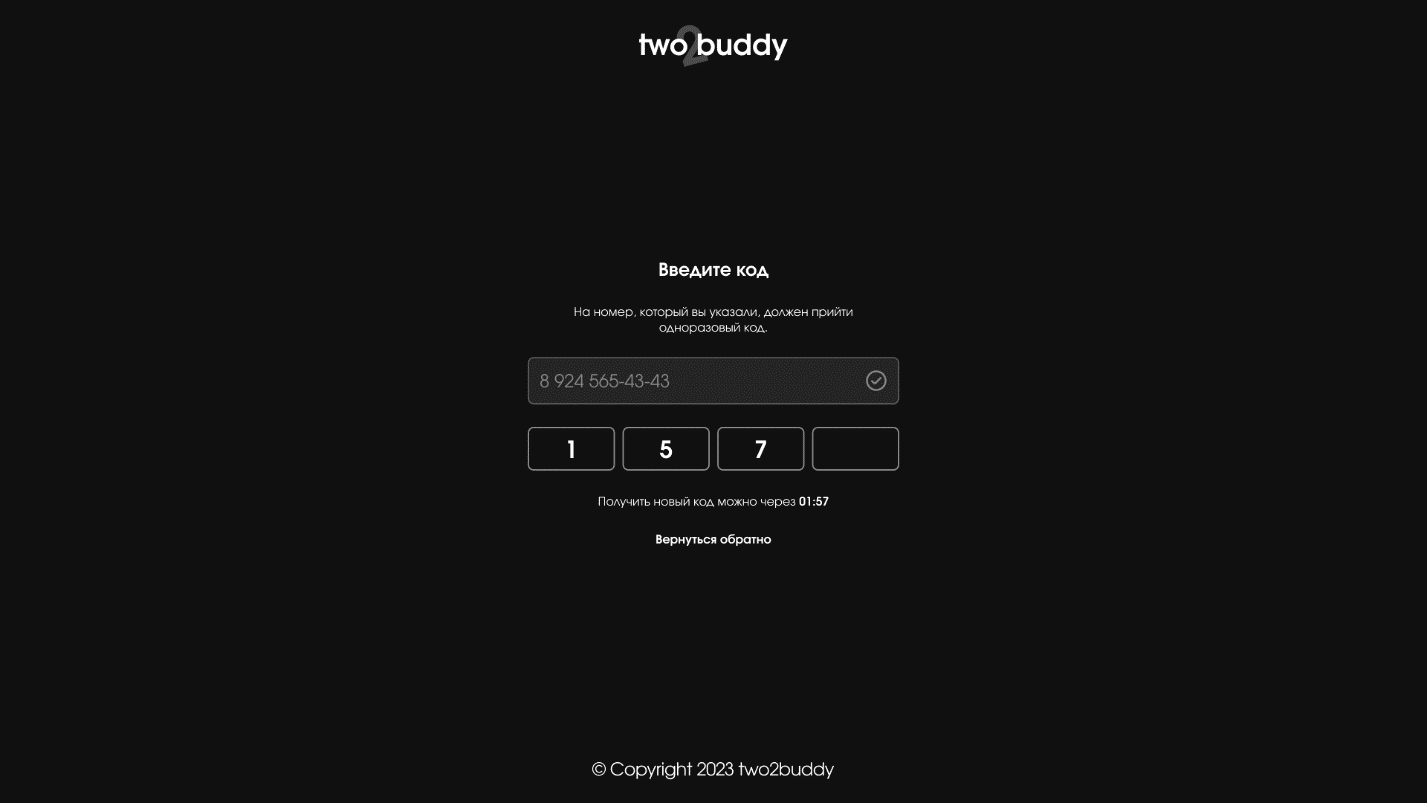


Рисунок 5.2 – Ввод одноразового кода

Этот подход к аутентификации обеспечивает высокий уровень безопасности, так как для входа в систему требуется не только знание пароля, но и физический доступ к почтовому ящику, на который отправляется одноразовый код.

* 1. **Защита личной информации пользователя**

Личная информация пользователей, включая персональные данные и платежные реквизиты, хранится в базе данных MongoDB в соответствии с законодательством о защите персональных данных. MongoDB обеспечивает возможность шифрования данных в покое и в движении, что помогает защитить конфиденциальность информации от несанкционированного доступа. [[10]](#_СПИСОК_ИСПОЛЬЗОВАННЫХ_ИСТОЧНИКОВ)

Шифрование данных в состоянии покоя относится к процессу защиты данных, хранящихся на физических устройствах, таких как жесткие диски, флеш-накопители и резервные ленты. Этот тип шифрования обеспечивает защиту данных от несанкционированного доступа даже в случае физического доступа к устройству. Данные хранятся в зашифрованном виде, и для их прочтения требуется использование ключа расшифровки.

Для повышения безопасности доступа к данным в MongoDB применяются различные меры, включая настройку прав доступа пользователей, многоуровневую аутентификацию и мониторинг активности пользователей. Предусмотрены также средства резервного копирования данных и восстановления после сбоев для обеспечения сохранности информации.

Кроме того, для защиты данных в MongoDB применяются регулярные обновления системы, патчи безопасности и аудит безопасности для выявления и устранения потенциальных уязвимостей. Такие меры обеспечивают надежную защиту личной информации пользователей и обеспечивают соответствие требованиям по обработке персональных данных.

1. **Тестирование веб-приложения**
   1. **Тестирование серверной части**

Для обеспечения надежной работы и соответствия функциональных требований, разработанное API прошло комплексное тестирование с использованием инструмента Postman [[12, 13, 14]](#_СПИСОК_ИСПОЛЬЗОВАННЫХ_ИСТОЧНИКОВ). Postman предоставляет удобный интерфейс для отправки HTTP-запросов к API, а также позволяет автоматизировать процесс тестирования сценариев, коллекций запросов и создания отчетов.

В процессе тестирования API в Postman проверялись следующие аспекты:

* Функциональное тестирование: Были созданы коллекции запросов, включающие в себя различные сценарии использования API. Проводились тесты на создание, чтение, обновление и удаление ресурсов, а также проверялась обработка ошибок и исключений.
* Юзабилити тестирование: Анализировалась удобство использования API для разработчиков. Проверялась структура запросов и ответов, названия эндпоинтов, форматы данных и другие аспекты, влияющие на опыт использования API.
* Кросс-браузерное тестирование: API тестировалось на совместимость с различными браузерами и клиентскими приложениями, используя возможности Postman для эмуляции различных сред.
* Безопасность: Проводилась проверка на уязвимости и защищенность API от атак и несанкционированного доступа с использованием средств Postman для аутентификации и авторизации запросов.

Использование Postman, как показано на рисунке 6.1, значительно облегчило процесс тестирования и позволило быстро обнаруживать и устранять любые выявленные проблемы.

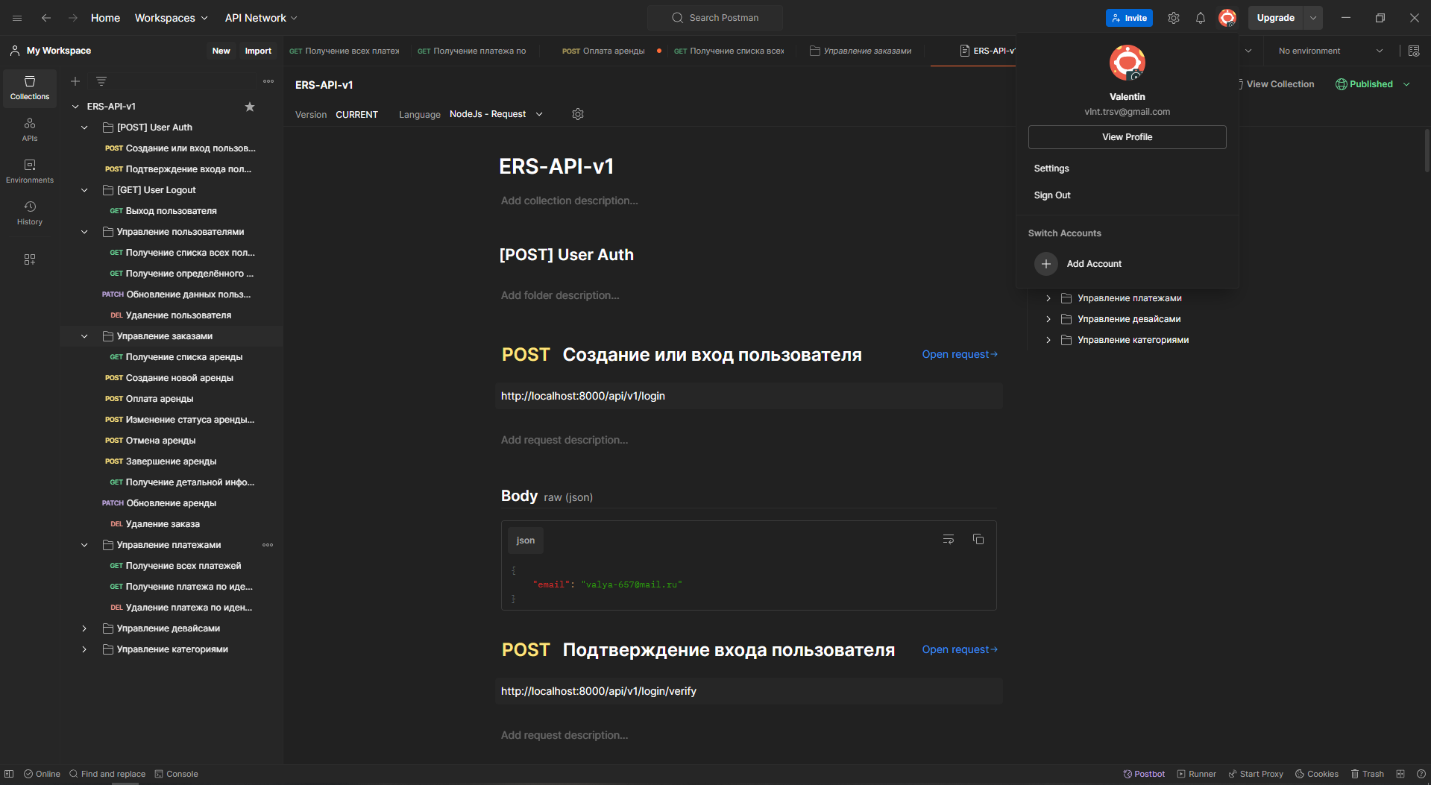


Рисунок 6.1- Тестирование серверной части в «Postman»

Тестирование в Postman также обеспечило документирование результатов тестов и создание отчетов для последующего анализа и отслеживания.

* 1. **Тестирование клиентской части**

Тестирование клиентской части веб-сервиса аренды цифровой электроники важно для обеспечения удобства и функциональности конечным пользователям. Тестирование охватывает несколько ключевых аспектов, каждый из которых направлен на проверку различных элементов системы. Для организации и систематизации процесса тестирования, создаётся чек-лист в виде таблицы или другом удобном формате, что позволяет эффективно отслеживать выполнение каждого теста и убедиться в полном покрытии всех аспектов функциональности и юзабилити сервиса. Пример чек-листа представлен в таблице 6.1. [[13, 14, 15]](#_СПИСОК_ИСПОЛЬЗОВАННЫХ_ИСТОЧНИКОВ)

Таблица 6.1 – Пример чек-листа

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Категория | Тест | Описание | Статус | Заметки |
| Функциональное тестирование | Регистрация пользователя | Проверка процесса регистрации пользователей | Начато | Проверить кнопку |
| Вход в систему | Тестирование функции входа в систему с корректными и некорректными данными | Не начато | Проверить токен |
| Оформление подписки девайса | Проверка функции оформления подписки, с дальнейшем оплатой | Не начато |  |
| Юзабилити тестирование | Интуитивность интерфейса | Оценка легкости навигации по сайту и интуитивности элементов управления | Начато | Проверить отступы у навигации |
| Ошибка в формах | Проверить корректность сообщений об ошибке в формах ввода | Не начато |  |
| Адаптивное тестирование | Отображение на мобильных устройствах | Тестирование отображения интерфейса на различных устройствах | Не начато |  |
| Работа на разных браузерах | Проверка совместимости веб-сервиса с основными типами браузера | Не начато |  |
| И т.д. | И т.д. | И т.д. | И т.д. | И т.д. |

Инструкции по использованию чек-листа:

* Категория и Тест: Описывают, какой аспект системы тестируется.
* Описание: Детализирует, что именно необходимо проверить в рамках данного теста.
* Статус: Отслеживает прогресс тестирования (Не начато, В процессе, Завершено, Требует пересмотра).
* Заметки: Поле для дополнительных комментариев, например, о выявленных проблемах или особенностях выполнения теста.

Правильное исчерпывающее тестирование гарантирует, что сервис будет надёжным, безопасным и удобным для пользователя. Важно регулярно обновлять чек-лист по мере развития и изменения сервиса, добавляя новые тесты и уточняя существующие в соответствии с новыми требованиями и технологиями.

# **ЗАКЛЮЧЕНИЕ**

Проведенный анализ потребностей рынка аренды электроники выявил ряд ключевых факторов, которые определили концепцию разрабатываемого сервиса. Эта концепция включает в себя несколько важных аспектов, направленных на обеспечение максимального удобства и удовлетворения потребностей пользователей. В первую очередь, основное внимание уделяется созданию удобного и интуитивно понятного интерфейса, который позволит пользователям легко и быстро осуществлять аренду электронных устройств.

Кроме того, значительное внимание уделяется разнообразию доступных электронных устройств с конкурентоспособными ценами. Это важно для того, чтобы удовлетворить разнообразные потребности клиентов и сделать аренду доступной для широкого круга пользователей. Надежность и высокое качество арендуемых устройств также становятся приоритетными аспектами, поскольку пользователи ожидают получить качественные и безопасные устройства для своих нужд.

Особое внимание уделяется также быстрой и качественной доставке, поскольку эффективная логистика играет важную роль в обеспечении удобства пользования сервисом.

В разработке дизайна интерфейса учтены различные аспекты, такие как цветовая схема, типографика, иконки и изображения, с целью создания гармоничного и привлекательного внешнего вида. Разработка верстки интерфейса включает в себя не только создание структуры страниц и определение размеров элементов, но и обеспечение адаптивности интерфейса для комфортного использования на различных устройствах и в различных условиях.

В общем и целом, разработка сервиса аренды электроники завершилась успешно, создав продукт, который не только отвечает потребностям рынка, но и способен удовлетворить ожидания пользователей в сфере аренды электроники.

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

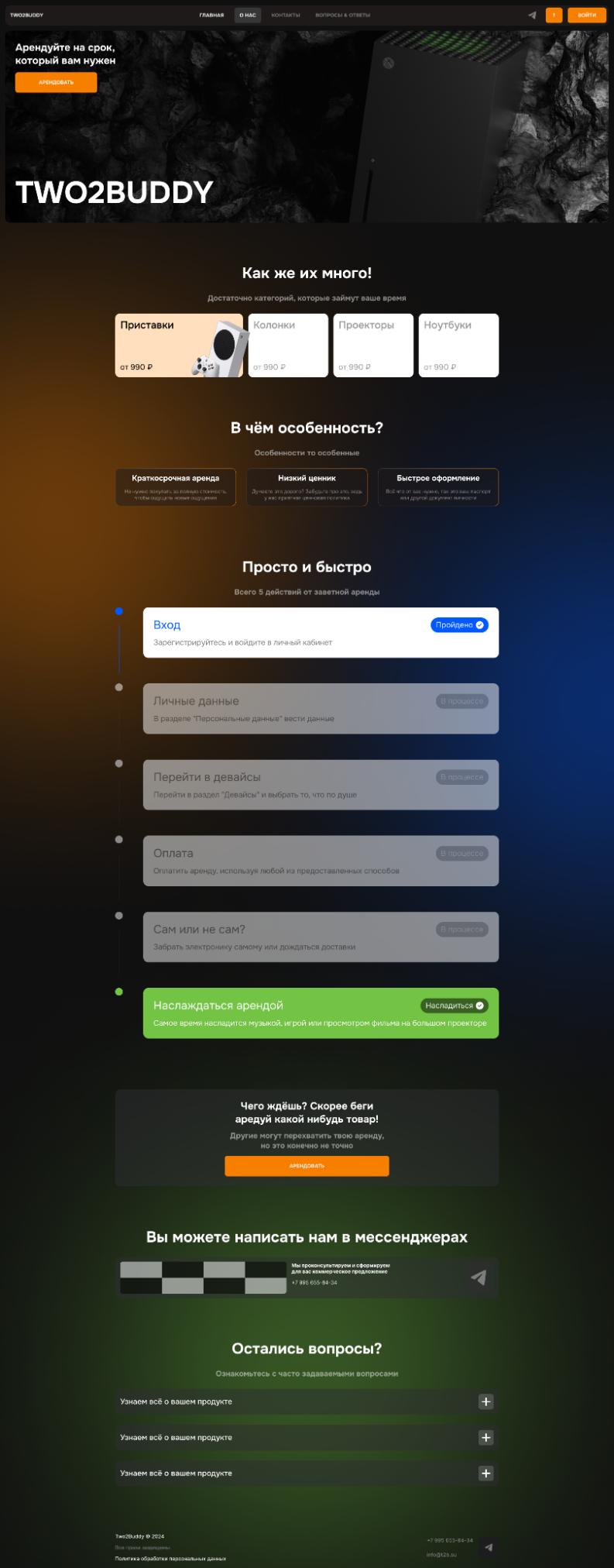
1. Урусов Т.Т. СОЗДАНИЕ ВЕБ-ПРИЛОЖЕНИЯ ИНТЕРНЕТ-МАГАЗИНА С ИСПОЛЬЗОВАНИЕМ СОВРЕМЕННЫХ ИНСТРУМЕНТОВ РАЗРАБОТКИ // Инновации и инвестиции. 2023. №6. URL: https://cyberleninka.ru/article/n/sozdanie-veb-prilozheniya-internet-magazina-s-ispolzovaniem-sovremennyh-instrumentov-razrabotki (дата обращения: 13.04.2024).
2. Тропынина Наталья Евгеньевна, Логинов Павел Олегович СОВРЕМЕННЫЕ ТЕНДЕНЦИИ РАЗВИТИЯ РЫНКА ЭЛЕКТРОННОЙ КОММЕРЦИИ // Инновационная экономика: перспективы развития и совершенствования. 2022. №3 (61). URL: https://cyberleninka.ru/article/n/sovremennye-tendentsii-razvitiya-rynka-elektronnoy-kommertsii (дата обращения: 13.04.2024).
3. Берьянов Максим Сергеевич, Салахов Илья Равильевич, Иванов Михаил Дмитриевич ИССЛЕДОВАНИЕ СОВРЕМЕННОГО СТЕКА REACT РАЗРАБОТКИ В 2022 ГОДУ // Столыпинский вестник. 2022. №8. URL: https://cyberleninka.ru/article/n/issledovanie-sovremennogo-steka-react-razrabotki-v-2022-godu (дата обращения: 24.04.2024).
4. Котов И. Компонентный подход [Электронный ресурс] // Хабр [сайт]. – URL: https://habr.com/ru/sandbox/182978 (дата обращения 11.04.2024).
5. МакГрат М. JavaScript для начинающих. – М.: Эксмо, 2023. – 232 с.
6. Орлова А. 3 способа разработки веб-сайта [Электронный ресурс] // Vc [сайт]. – URL: https://vc.ru/dev/78714-3-sposoba-razrabotki-veb-sayta (дата обращения 20.11.2023).
7. React [Электронный ресурс]. – URL: https://react.dev (дата обращения 22.11.2023).
8. VK Cloud Solution. Как работать с персональными данными и соблюдать 152-Ф3. 2023. URL: <https://cloud.vk.com/promopage/cloud-security/?utm_source=yandex.direct.DB&utm_medium=cpc&utm_campaign=y_cloud_security_umnaya_general1_rf_98415349&utm_content=15364403011&utm_term=cloud%20152&yclid=4164357771529027583> (Дата обращения: 15.04.2024)
9. Данил Кочнев. Анализ конкурентов на практике. 2020. URL: <https://vc.ru/marketing/181297-analiz-konkurentov-na-praktike-10-shagov-shablon>
10. Хабр. Введение в Postman. 2018. <URL:https://habr.com/ru/companies/kolesa/articles/351250/>
11. TestEngineer.ru. Большой гайд по тестированию с Postman для начинающих. 2021. URL: <https://testengineer.ru/gajd-po-testirovaniyu-v-postman/>
12. Postman Documentation [Электронный ресурс]. Официальная документация Postman. 2024. URL: <https://learning.postman.com/docs/publishing-your-api/documenting-your-api/>
13. QA Bible. Тестирование клиентской и серверной части. 2023. URL: <https://vladislaveremeev.gitbook.io/qa_bible/vidy-metody-urovni-testirovaniya/testirovanie-klientskoi-chasti-i-servernoi-frontend-testing-vs.-backend-testing>
14. TestEngineer.ru. Тестирование фронтенда: Большой гайд. 2023. ULR: <https://testengineer.ru/frontend-testing-guide/>
15. Хабр. Знакомство с тестированием веб-приложений. 2022. URL: <https://habr.com/ru/companies/ruvds/articles/676752/>
16. Хабр. Безопасность веб-приложений: анализ методов защиты от атак на уровне Backend. 2024. URL: <https://habr.com/ru/companies/ruvds/articles/676752/>
17. HTML Academy. Как защитить приложение от хакеров. 2022. URL: <https://htmlacademy.ru/blog/soft/web-security>
18. TProger. Как защитить веб-приложение: основные советы, инструменты, полезные ссылки. 2017. URL: <https://tproger.ru/translations/webapp-security>
19. Хабр. Моделирование данных: обзор. 2021. URL: <https://habr.com/ru/articles/556790/>
20. Webonto.ru. Концептуальная модель базы данных – диаграмма связи между объектами. 2023. URL: <https://webonto.ru/kontseptualnaya-model-bazyi-dannyih/>
21. Хабр. Styled Components – идеальная стилизация React-приложения. 2021. URL: <https://habr.com/ru/articles/591381/>
22. Dev.to. Styling React Components with CSS. 2023. URL: <https://dev.to/frank-mwangi/styling-react-components-with-css-a-comprehensive-guide-428f>
23. Хабр. Введение в REST API – RESTful веб-сервисы. 2020. URL: <https://habr.com/ru/articles/483202/>
24. SkillFactory Media. REST API. 2023. URL: <https://blog.skillfactory.ru/glossary/rest-api/>
25. Yandex Cloud. REST API: для чего нужен и как работать. 2024. URL: <https://yandex.cloud/ru/docs/glossary/rest-api>
26. VC.ru. Психографика. Темная сторона маркетинга? 2018. URL: <https://vc.ru/flood/42764-psihografika-temnaya-storona-marketinga>
27. Tproger.ru. Как обеспечить безопасность телеграм-бота. 2023. URL: <https://tproger.ru/articles/obespechenie-bezopasnosti-telegram-botov>
28. Feature-sliced.design [Электронный ресурс]. Официальная документация FSD. 2024. URL: https://feature-sliced.design/docs/get-started/overview

# **ПРИЛОЖЕНИЕ А**

Модель Остервальдера

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ключевые партнёры   1. Производители гаджетов, электроники 2. Иные интернет-магазины | Ключевые активности   1. Дизайн и поставки 2. Маркетинг и продажи 3. Клиентский сервис 4. Брендинг | Ценностные предложения   1. Возможность аренды на короткий, длительный срок 2. Удобный личный кабинет арендодателя 3. Приемлемые цены на электронику 4. Большой ассортимент электроники | Отношения с клиентами   1. Личный кабинет, где можно оставить отзыв, сообщить о проблеме 2. Соцсети | Сегменты потребителей   1. Студенты 2. Бизнесмены 3. Путешественники 4. Экономичные потребители 5. Малый бизнес |
| Ключевые ресурсы   1. Бренд и узнаваемость 2. Сотрудники пунктов выдачи 3. Руководящий состав | Каналы поставки   1. Производители гаджетов, электроники 2. Иные интернет-магазины |
| Структура издержек   1. Заработная плата сотрудникам пункта выдачи 2. Маркетинг 3. Логистика | | | Источники дохода   1. Подписки на электронику 2. Рекламные баннеры на сайте компании 3. Подписки на эксклюзивные товары от партнёров | |

# **ПРИЛОЖЕНИЕ Б**



# **ПРИЛОЖЕНИЕ В**

