**Министерство образования Иркутской области**

Государственное бюджетное профессиональное

образовательное учреждение Иркутской области

«Иркутский авиационный техникум»

(ГБПОУИО «ИАТ»)

КП.09.02.07-3.24.222.14 ПЗ

ВЕБ-ПРИЛОЖЕНИЕ «КОМПЬЮТЕРНЫЙ КЛУБ»

|  |  |  |
| --- | --- | --- |
| Председатель ВЦК: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | (А.С. Александрова) |
|  | (дата, подпись) |  |
| Руководитель: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | (Н.Р. Карпова) |
|  | (дата, подпись) |  |
| Студент: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | (Н.А. Курашов) |
|  | (дата, подпись) |  |

Иркутск 2025

Содержание

[Введение 3](#_Toc197905055)

[1 Предпроектное исследование 4](#_Toc197905056)

[1.1 Описание предметной области 4](#_Toc197905057)

[1.2 Обзор инструментальных средств для разработки 5](#_Toc197905058)

[2 Техническое задание 9](#_Toc197905059)

[3 Проектирование веб-приложения 10](#_Toc197905060)

[3.1 Архитектура веб-приложения 10](#_Toc197905061)

[3.2 Функциональное проектирование 12](#_Toc197905062)

[3.3 Проектирование базы данных 15](#_Toc197905063)

[3.4 Проектирование пользовательского интерфейса 19](#_Toc197905064)

[3.4.1 Разработка прототипов интерфейса 19](#_Toc197905065)

[3.4.2 Выбор цветовой гаммы и шрифтов 32](#_Toc197905066)

[3.4.3 Разработка элементов интерфейса 33](#_Toc197905067)

[3.4.4 Разработка дизайн макетов 34](#_Toc197905068)

[4 Разработка веб-приложения 47](#_Toc197905069)

[4.1 Разработка интерфейса веб-приложения 47](#_Toc197905070)

[4.2 Разработка базы данных веб-приложения 57](#_Toc197905071)

[4.3 Разработка веб-приложения 63](#_Toc197905072)

[Приложение А 74](#_Toc197905073)

# **Введение**

В современном мире компьютерные клубы стали очень популярны, как среди простых людей, так и среди профессиональных игроков. С увеличением популярности киберспорта и онлайн-игр появилась необходимость в разработке сайтов для компьютерных клубов. Веб-приложение позволит улучшить качество обслуживания клиентов, станет намного проще бронировать игровые места, производить оплату и управлять учетными записями.

Актуальность данного проекта заключается в том, что компьютерные клубы используются как в качестве места, где можно отдохнуть с друзьями, так и местом, где возможно провести соревнования по компьютерным играм. Разработка веб-приложения позволит сделать работу клуба более эффективной и удобной для клиентов и администраторов клуба.

Целью курсового проекта является проектирование и разработка веб-приложения для компьютерного клуба, которое обеспечит удобное управление бронированием, оплатой.

Для достижения этой цели необходимо решить следующие задачи:

1. Провести исследование предметной области;
2. Провести обзор инструментов и технологий для разработки веб-приложения;
3. Сформировать техническое задание;
4. Разработать архитектуру веб-приложения;
5. Разработать пользовательский интерфейс;
6. Подготовить руководство пользователя.

Таким образом, данный проект направлен на создание эффективного и простого веб-приложения, способствующего автоматизации процессов управления компьютерным клубом и улучшению взаимодействия с клиентами.

# **1 Предпроектное исследование**

## **1.1 Описание предметной области**

Веб-приложение «Компьютерный клуб» предназначен для управления работой компьютерного клуба, предоставления информации о доступных услугах и рабочих местах, а также упрощения процесса бронирования времени пользователями.

Веб-приложение включает в себя функционал для администраторов клуба, пользователей, а также данные о доступных тарифах. Веб-приложение позволит автоматизировать основные процессы, такие как управление бронированием, отслеживание занятости рабочих мест, управление тарифами.

Объектами данной предметной области являются: администратор, пользователь, рабочее место и тариф.

Администратор имеет доступ ко всему функционалу веб-приложения, где он может, управлять списком рабочих мест (добавлять, редактировать или удалять записи), настраивать тарифы клуба. Также он может отслеживать занятость рабочих мест в определенное время и управлять бронированием. Таким образом, администратор обеспечивает корректную работу системы и актуальность данных для пользователей.

Пользователь – это клиент, который использует информационную систему для взаимодействия с компьютерным клубом.

Пользователь может просматривать доступные рабочие места и их характеристики (комплектующие ПК, периферию), бронировать рабочее место на выбранное время, выбирать удобный тариф для пользования

Рабочее место – это объект, представляющий физическое место в клубе, оборудованное компьютером. Каждое рабочее место имеет уникальный номер, технические характеристики оборудования и статус (свободно/занято/сломано).

Тариф – набор условий, определяющих стоимость использования рабочего места. Тариф может быть почасовыми, так и дневными. Администратор может устанавливать различные тарифы в зависимости от времени суток, дня недели или длительности использования.

Веб-приложение «Компьютерный клуб» работает за счет взаимодействия этих объектов, что позволяет автоматизировать управление клубом и сделать его использование удобным как для сотрудников, так и для клиентов.

Таким образом, веб-приложение помогает улучшить качество обслуживания, минимизирует количество ошибок со стороны рабочего персонала, и упрощает взаимодействие с клиентами.

* 1. Обзор инструментальных средств для разработки

Для создания веб-приложения будут использованы:

– HTML

– CSS

– PhpMyAdmin

– MySQL

– MySQL Workbench

– Draw.io

– Figma

– PHP

– Laravel

– Visual Studio Code

HTML – язык гипертекстовой разметки. Любой документ на языке HTML представляет собой набор элементов, причём начало и конец каждого элемента обозначается специальными пометками - тегами. Язык разметки дает браузеру необходимые инструкции о том, как отображать тексты и другие элементы страницы на мониторе. Язык HTML интерпретируется браузерами и отображается в виде документа, в удобной для человека форме.

CSS – каскадные таблицы стилей (англ. Cascading Style Sheets). CSS используется для определения стилей (правил) оформления документов - включая дизайн, вёрстку и вариации макета для различных устройств и размеров экрана.

PhpMyAdmin – веб-приложение с открытым кодом, написанное на языке PHP и представляющее собой веб-интерфейс для администрирования СУБД MySQL. PhpMyAdmin позволяет через браузер и не только осуществлять администрирование сервера SQL MySQL, запускать команды и просматривать содержимое таблиц и баз данных. Приложение пользуется большой популярностью у веб-разработчиков, так как позволяет управлять СУБД MySQL без непосредственного ввода SQL команд.

Веб-приложение будет содержать в себе информацию. Её необходимо хранить, изменять, и использовать. Это реализуется благодаря базам данных.

MySQL – это популярная реляционная СУБД, которая отличается высокой скоростью, надежностью и широким функционалом. Она поддерживает стандарт SQL, предоставляет различные движки хранения данных (InnoDB, MyISAM) и легко интегрируется с большинством языков программирования. В свою очередь разработку и поддержку реализовывает корпорация Oracle, получившая права на торговую марку вместе с поглощённой Sun Microsystems, которая ранее приобрела шведскую компанию MySQL AB. Продукт распространяется как под GNU General Public License, так и под собственной коммерческой лицензией.

Проектировать структуру веб-приложения удобно через MySQL Workbench и Draw.io, а его дизайн – через онлайн-сервис Figma. Веб-приложение будет состоять из двух частей – клиентская и серверная.

MySQL Workbench – инструмент для визуального проектирования баз данных, интегрирующий проектирование, моделирование, создание и эксплуатацию БД в единое бесшовное окружение для системы баз данных MySQL. В проекте используется для создания ER-диаграммы БД.

Draw.io – это удобное бесплатное онлайн-приложение для создания диаграмм для рабочих процессов, BPM, организационных, сетевых диаграмм, блок-схем и UML В проекте используется для создания прототипа страниц.

Figma – бесплатный удобный онлайн-сервис для дизайнеров, веб-разработчиков и маркетологов. Он предназначен для создания прототипов сайтов или приложений, иллюстраций и векторной графики. В редакторе можно настроить совместную работу, вносить и обсуждать правки, причём как в браузере, так и через приложение на компьютере. Популярен, в частности, для разработки прототипа и дизайна сайта или приложения. В проекте используется для создания наглядного ожидаемого дизайна проекта, а также уникальных SVG-элементов в едином стиле.

Для взаимосвязи баз данных и северной части продукта необходимо использовать серверный язык – PHP.

PHP – скриптовый язык общего назначения, интенсивно применяемый для разработки веб-приложений. В настоящее время поддерживается подавляющим большинством хостинг-провайдеров и является одним из лидеров среди языков, применяющихся для создания динамических веб-сайтов. Действует, как и самостоятельно, так и с фреймворками.

PHP не ориентирован на создание десктопных приложений, но есть потребность в создании интерфейсов для настройки серверов, беспрерывного выполнения, отладки скриптов (сценариев), управления локальными и тестовыми серверами. Из-за этого и возникли решения данной проблемы.

Laravel – бесплатный веб-фреймворк с открытым кодом, предназначенный для разработки с использованием архитектурной модели MVC (англ. Model View Controller – модель-представление-контроллер). Laravel выпущен под лицензией MIT.

Для разработки программного продукта будет использоваться Visual Studio Code.

Visual Studio Code (VS Code) – текстовый редактор, разработанный Microsoft для Windows, Linux и macOS. Позиционируется как «лёгкий» редактор кода для кроссплатформенной разработки веб и облачных приложений. Включает в себя отладчик, инструменты для работы с Git, подсветку синтаксиса, IntelliSense (система автозавершения кода) и средства для рефакторинга. Имеет широкие возможности для кастомизации: пользовательские темы, сочетания клавиш и файлы конфигурации. Распространяется бесплатно, разрабатывается как программное обеспечение с открытым исходным кодом, но готовые сборки распространяются под проприетарной лицензией.

VS Code имеет поддержку плагинов, доступных через Visual Studio Marketplace. Они могут включать в себя дополнения к редактору, поддержку дополнительных языков программирования, статические анализаторы кода.

# **2 Техническое задание**

В начале разработки создавалось техническое задание, в котором указывались основные требования. Для создания технического задания использовался стандарт ГОСТ 34.602-2020. Согласно ГОСТ 34.602-2020 техническое задание должно включать следующие разделы:

1. Введение.
   1. Общие сведения.
   2. Цели и задачи.
2. Основания для разработки.

2.1. Нормативные документы.

2.2. Проектные документы.

1. Назначение системы.
   1. Общее описание.
   2. Преимущества и новизна.
2. Требования к системе.
   1. Функциональные требования.
   2. Технические требования.
   3. Эксплуатационные требования.
3. Требования к техническому обеспечению.
   1. Оборудование.
   2. Сетевые требования.
4. Требования к программному обеспечению.
   1. Программные компоненты.
   2. Интерфейсы.
5. Организационно-технические требования.
   1. Этапы разработки.

Техническое задание на разработку веб-приложения представлено в приложении А.

# **3 Проектирование веб-приложения**

## **3.1 Архитектура веб-приложения**

Архитектура программного обеспечения – совокупность важнейших решений об организации программной системы, включая выбор структурных элементов и их интерфейсов, а также архитектурный стиль, обеспечивающий оптимизацию производительности, безопасности и управляемости.

Для работы приложения было решено использовать клиент-серверную архитектуру, в соответствии с рисунком 1. Работа клиент-серверной архитектуры построена на взаимодействии между клиентом и сервером. Клиент отправляет запросы на получение данных с сервера, а сервер проводит проверку доступа к информации и формирует запрос в базу данных для получения, обработки и отправки необходимой информации клиенту.

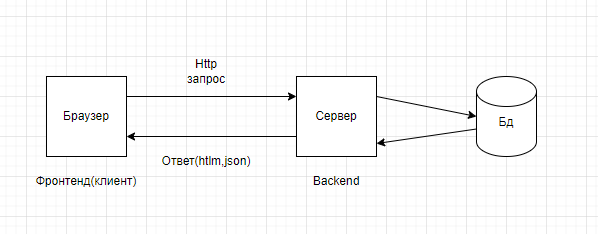


Рисунок 1 – Архитектура веб-приложения «Компьютерный клуб»

Клиентская часть реализует пользовательский интерфейс, формирует запросы к серверу и обрабатывает ответы от него.

Сервер – специальное системное оборудование, который предоставляет запрашиваемую информацию или услуги пользователям. Он выполняет работы сервисного обслуживания по клиентским запросам, предоставляет пользователям доступ к определенным системным ресурсам, сохраняет данные или БД.

Параметры, которые могут реализоваться на стороне сервера:

1. Хранение и доступ к данным;
2. Работа с поступающими клиентскими запросами;
3. Процесс отправки ответа клиенту.

Клиент – локальный компьютер на стороне виртуального пользователя, который выполняет отправку запроса к серверу для возможности предоставления данных или выполнения определенной группы системных действий.

Параметры, которые могут реализоваться на стороне клиента:

1. Формулировка запроса к серверу и его последующая отправка;
2. Получение итогов запроса

Таким образом, в архитектуре «клиент-сервер» клиент посылает запрос на предоставление данных и получает только те данные, которые действительно были затребованы.

Model-View-Controller (MVC) – схема разделения данных приложения, и она управляет логикой, разделяя ее на три отдельных компонента: модель, представление и контроллер – таким образом, что модификация каждого компонента может осуществляться независимо. Модель представлена на рисунке 2.

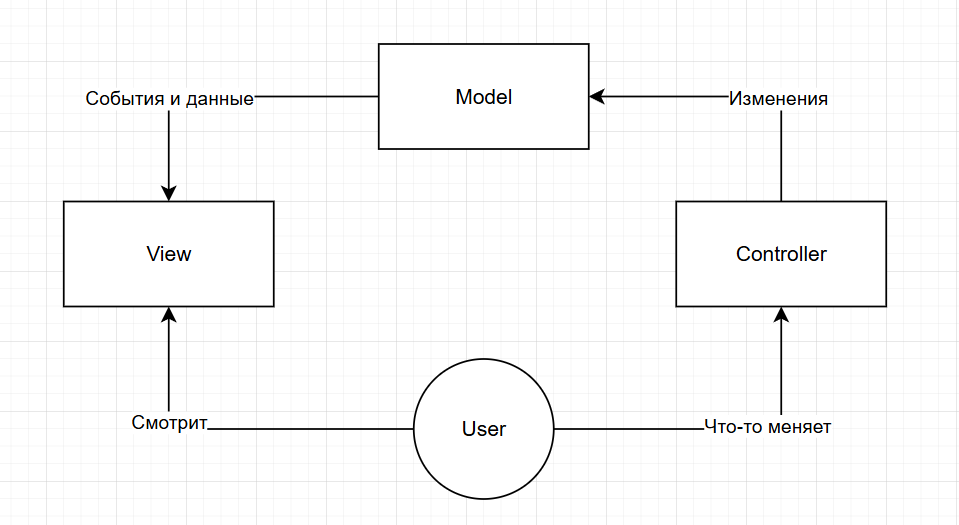


Рисунок 2 – Модель MVC

Модель предоставляет данные и методы работы с ними: запросы в базу данных, проверка на корректность. Модель не зависит от представления (не знает, как данные визуализировать) и контроллера (не имеет точек взаимодействия с пользователем), просто предоставляя доступ к данным и управлению ими.

Представление отвечает за получение необходимых данных из модели и отправляет их пользователю. Представление не обрабатывает введённые данные пользователя

Контроллер обеспечивает «связь» между пользователем и системой. Контролирует и направляет данные от пользователя к системе и наоборот. Использует модель и представление для реализации необходимого действия.

**3.2 Функциональное проектирование**

Проектирование программных продуктов выполняется поэтапно с использованием различных наглядных диаграмм.

Проектирование веб-приложения начинается с построения диаграммы прецедентов в рамках предметной области. Это позволит детально продемонстрировать предметную область и возможности работы веб-приложения. На рисунке 3 представлена диаграмма прецедентов Uses-CASE. Она содержит 2 актёра, которые могут выполнять суммарно 10 функций.

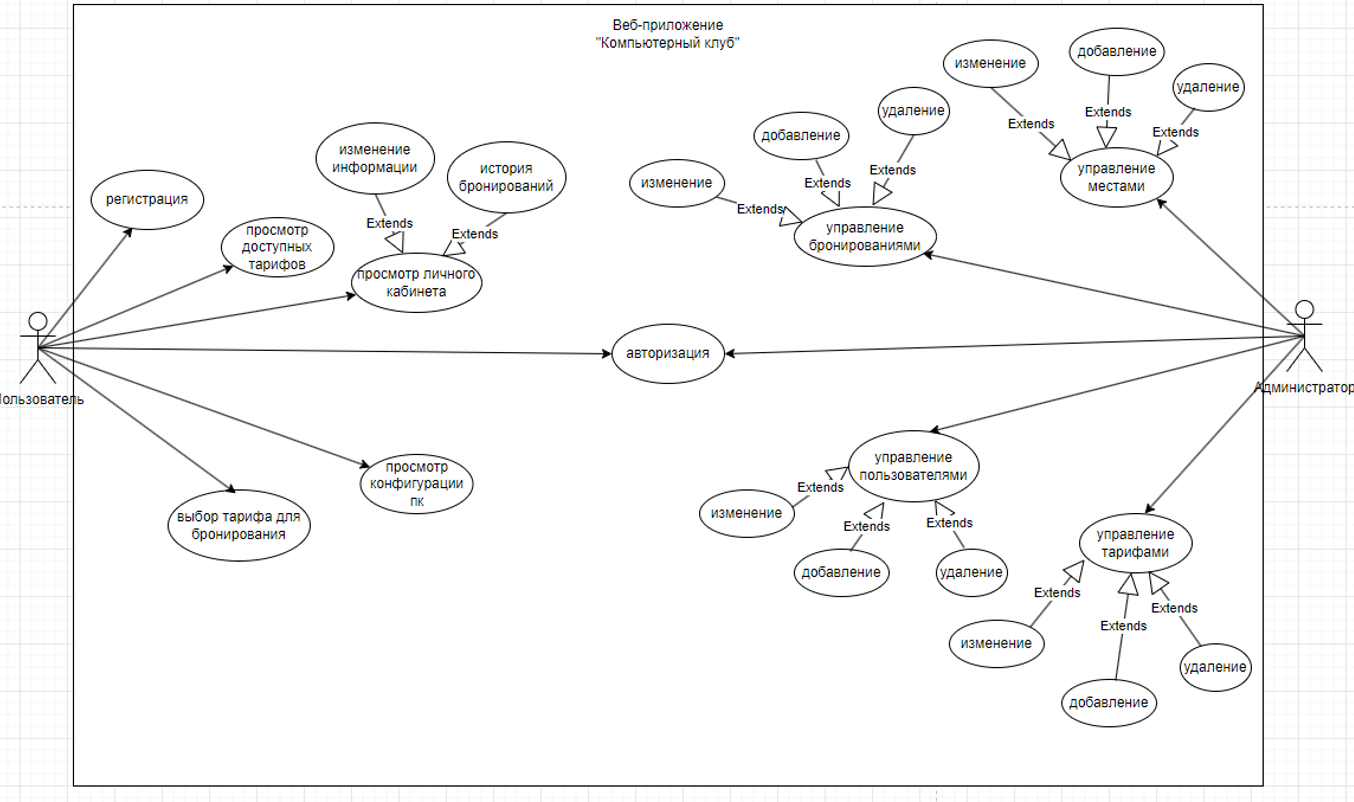


Рисунок 3 – Диаграмма прецедентов

Диаграмма потоков данных DFD отображает потоки данных в системе и процессы их обработки. Диаграмма состоит из блоков, представляющих функции и процессы, и стрелок, представляющих потоки данных между ними.

На рисунке 4 представлена диаграмма потоков данных DFD, которая помогает лучше понять, как данные перемещаются в веб-приложении и как они обрабатываются каждым процессом.

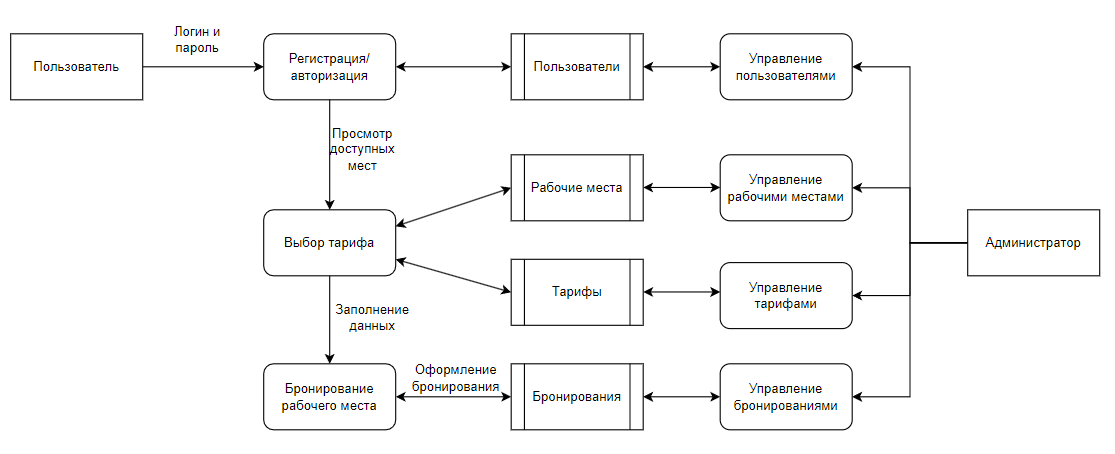


Рисунок 4 –Диаграмма потоков данных

Диаграмма классов – это способ визуализации структуры классов и их взаимосвязей в системе. Классы описывают состояние (атрибуты) и поведение (методы). Диаграмма классов на рисунке 5 отображает объекты в системе, их взаимодействие и содержание данных.

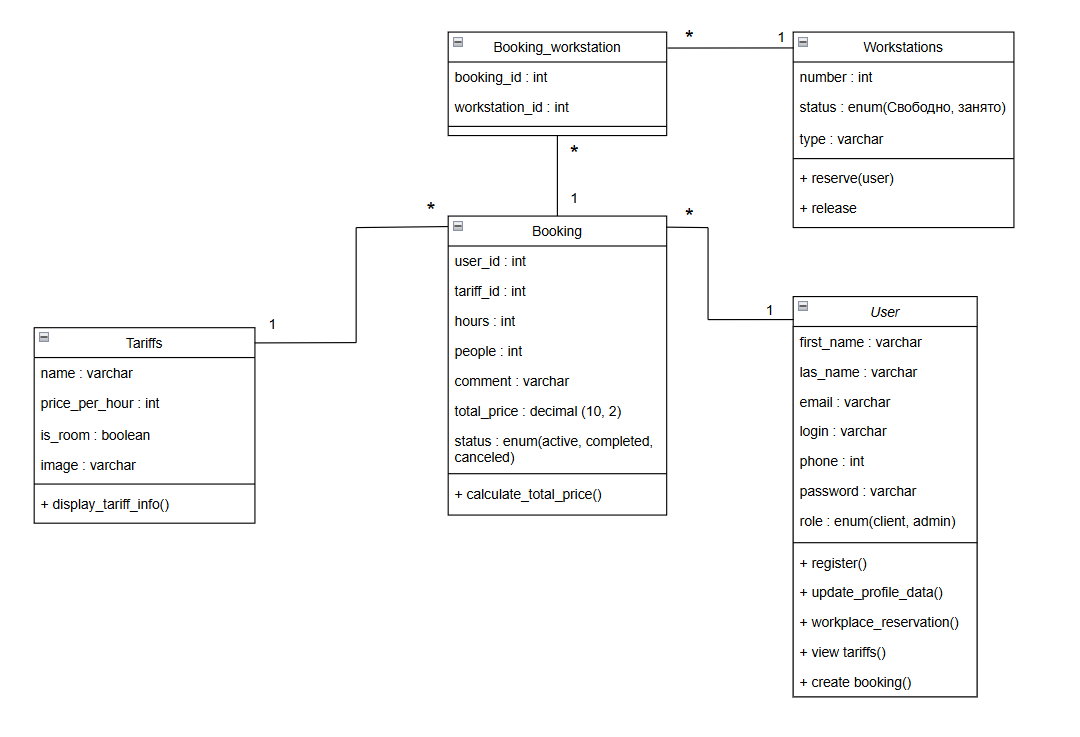


Рисунок 5 – Диаграмма классов

Следующим этапом будет построение диаграммы последовательностей, которая представлена на рисунке 6. В них обычно отображаются объекты, взаимодействующие в рамках сценария, сообщения, которые они передают друг другу, а также результаты, связанные с этими сообщениями.

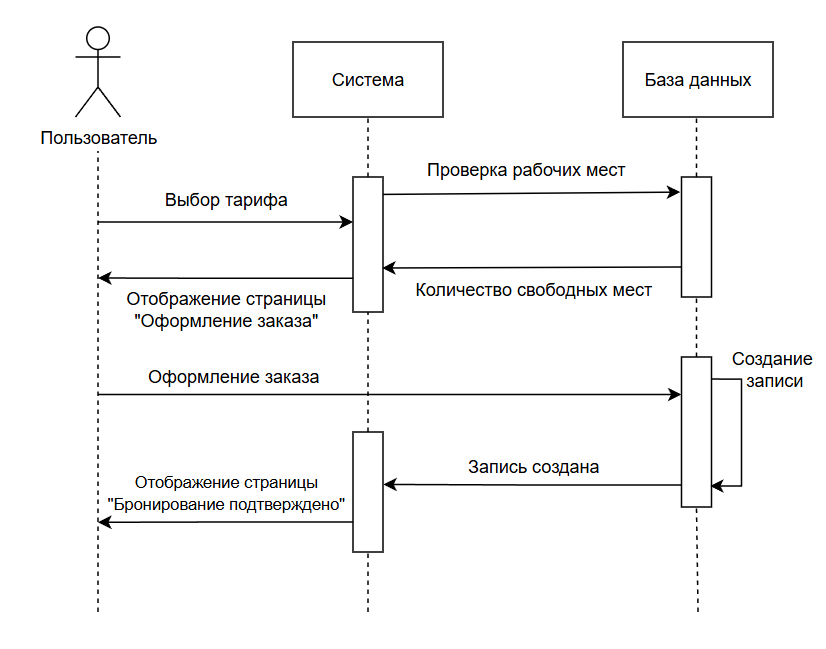


Рисунок 6 – Диаграмма последовательности

Таким образом, каждая из данных диаграмм играет важную роль в процессе проектирования веб-приложения «Компьютерный клуб», помогая лучше понять его структуру, функции и взаимодействие между элементами. Опираясь на эти диаграммы и их объяснение, проводится анализ потребностей пользователей и разрабатывается эффективное и безопасное веб-приложение.

## **3.3 Проектирование базы данных**

База данных – совокупность организованной информации, относящейся к определённой предметной области, предназначенная для длительного хранения во внешней памяти компьютера и постоянного применения.

Основное требование к базам данных – удобство доступа к данным, возможность оперативно получить исчерпывающую информацию по любому интересующему вопросу (важно не только то, что информация содержится в базе, важно и то, насколько она хорошо структурирована и целостна).

Согласно современным требованиям к базам данных, информация, содержащаяся в них, должна быть:

1. непротиворечивой (не должно быть данных, противоречащих друг другу);
2. неизбыточной (следует избегать дублирования информации в базе, это может привести к противоречивости – например, если какие-то данные изменяют, а их копию в другой части базы изменить забыли);
3. целостной (все данные должны быть связаны, не должно быть ссылок на данные, не существующие в базе).

В таблицах была устранена избыточность и дублирование информации.

После всех этапов проектирования необходимо создать ER-модель базы данных, которая представлена на рисунке 7.

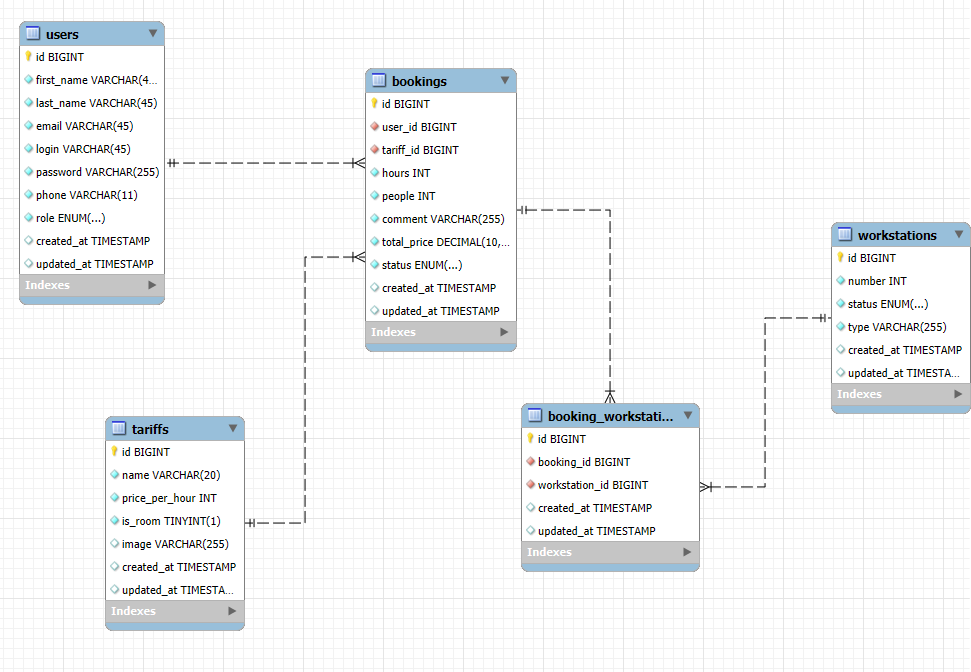


Рисунок 7 – ER-модель

ER-модель состоит из 5 таблиц, таблица пользователя содержит его имя, фамилию, электронную почту, логин, пароль, номер телефона и роль (пользователь или администратор), таблица рабочих мест содержит в себе номер, тип (Стандартный, VIP), таблица тарифа содержит его название, цена за час, значение «Комната» (если да, то VIP) и картинку, таблица бронирования содержит данные о пользователе, рабочем месте, используемом тарифе, начало бронирования, окончание бронирования, и статус (Активно, завершено, отменено), таблица связи бронирований и рабочих мест, которая включает в себя данные о бронировании и рабочих местах.

Таблица 1 – Users

|  |  |  |
| --- | --- | --- |
| Атрибут | Тип данных | Описание атрибута |
| id | Int | Id пользователя |
| first\_name | Varchar (45) | Имя пользователя |
| last\_name | Varchar (45) | Фамилия пользователя |
| email | Varchar (45) | Email пользователя |
| login | Varchar (45) | Логин пользователя (уникальный) |
| password | Varchar (255) | Пароль пользователя |
| phone | Varchar (11) | Номер телефона пользователя |
| role | Enum('client', 'admin') | Роль пользователя (клиент или администратор) |

Таблица 2 – workstations

|  |  |  |
| --- | --- | --- |
| Атрибут | Тип данных | Описание атрибута |
| id | Int | Id рабочего места |
| number | Int | Номер рабочего места |
| status | Enum ('Свободно', 'Занято') | Статус рабочего места |
| type | Varchar (255) | Тип рабочего места |

Таблица 3 – tariffs

|  |  |  |
| --- | --- | --- |
| Атрибут | Тип данных | Описание атрибута |
| id | Int | Id тарифа |
| name | Varchar (20) | Название тарифа |

Продолжение таблицы 3

|  |  |  |
| --- | --- | --- |
| price\_per\_hour | Decimal (10, 2) | Стоимость использования рабочего места за час |
| is\_room | Boolean | Комната |
| image | Varchar (255) | Картинка |

Таблица 4 – bookings

|  |  |  |
| --- | --- | --- |
| Атрибут | Тип данных | Описание атрибута |
| id | Int | Id бронирования |
| user\_id | Int | Id пользователя |
| tariff\_id | Int | Id тарифа |
| hours | Int | Количество часов |
| people | Int | Количество человек |
| comment | Varchar (255) | Комментарий |
| total\_price | Decimal (10, 2) | Итоговая стоимость бронирования |
| status | Enum ('active', 'completed', 'cancelled') | Статус бронирования |

Таблица 5 – booking\_workstation

|  |  |  |
| --- | --- | --- |
| Атрибут | Тип данных | Описание атрибута |
| id | Int | Id связи |
| booking\_id | Int | Id бронирования |
| workstation\_id | Int | Id рабочего места |

## **3.4 Проектирование пользовательского интерфейса**

### **3.4.1 Разработка прототипов интерфейса**

Схема навигации показывает основные страницы веб-приложения «Компьютерный клуб» и их взаимосвязь, представлена на рисунке 8. Пользователь начинает взаимодействие с главной страницы, откуда может спуститься чуть ниже и выбрать один из нужных тарифов, который перенесет его на страницу бронирования, где может оформить заказ после перейдет на страницу «Бронирование подтверждено» После он сможет перейти в личный кабинет, где сможет посмотреть историю бронирований или перейти на главную страницу. Также если пользователь является администратором, он сможет перейти в админ-панель из личного кабинета.

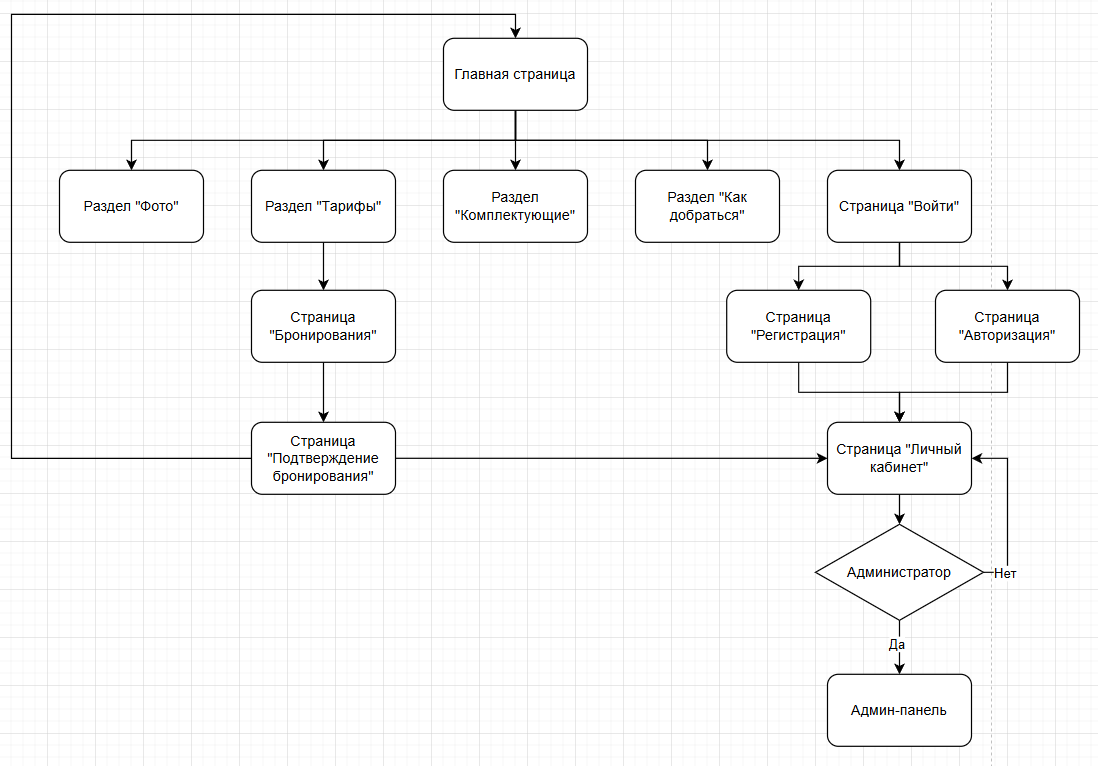


Рисунок 8 – Схема навигации веб-приложения «Компьютерный клуб»

Разработка удобного пользовательского интерфейса – это один из важнейших этапов в процессе создания веб-приложения «Компьютерный клуб» для этого необходимо построить прототип интерфейса веб-приложения.

Интерфейс – это набор инструментов, который позволяет пользователю взаимодействовать с программой. В более широком смысле термин обозначает любые инструменты для соприкосновения между разными системами и сущностями.

Прототип был разработан с помощью редактора Figma. На рисунке 9 представлен прототип главной страницы веб-приложения, который включает в себя навигационная панель с логотипом, наименования разделов, блок с тарифами, раздел с комплектующими, карту, и подвал с навигационной панелью



Рисунок 9 – Прототип главной страницы

На рисунке 10 представлен прототип страницы бронирования места, он включает в себя навигационные панели, а также два блока, в одном из которых указывается вся основная информация о бронировании, а в другом указывается комментарий, его можно оставить по желанию.

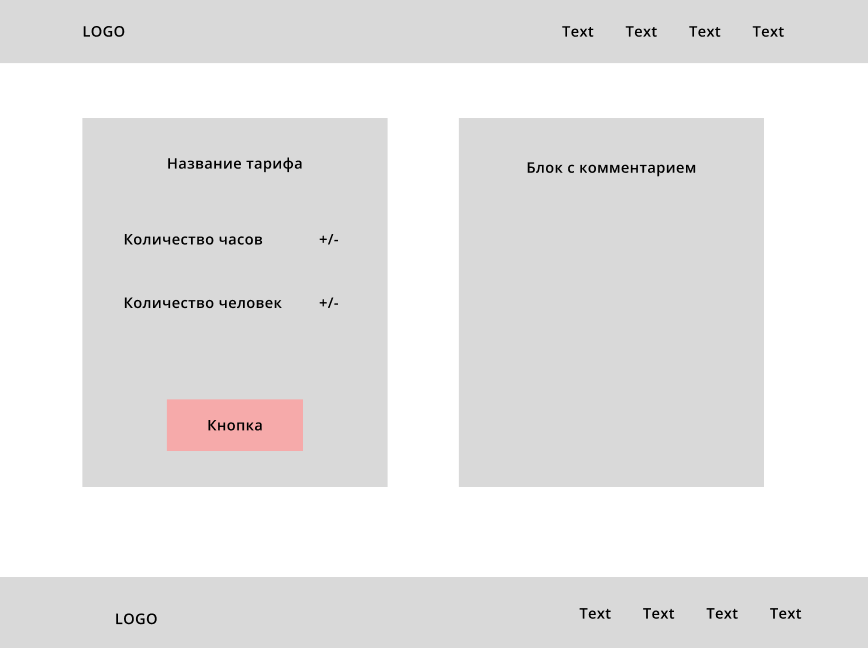


Рисунок 10 – Прототип страницы бронирования

Прототип страницы регистрации представлен на рисунке 11, который включает в себя форму, в которой пользователь должен заполнить поля: имя, фамилия, логин, адрес электронной почты (email), номер телефона, пароль и подтверждение пароля. Ниже расположена кнопка «Зарегистрироваться», по нажатию на которую происходит отправка данных для создания новой учетной записи. Чуть ниже присутствует текст с предложением перейти на страницу авторизации в случае, если у пользователя уже есть зарегистрированный аккаунт.

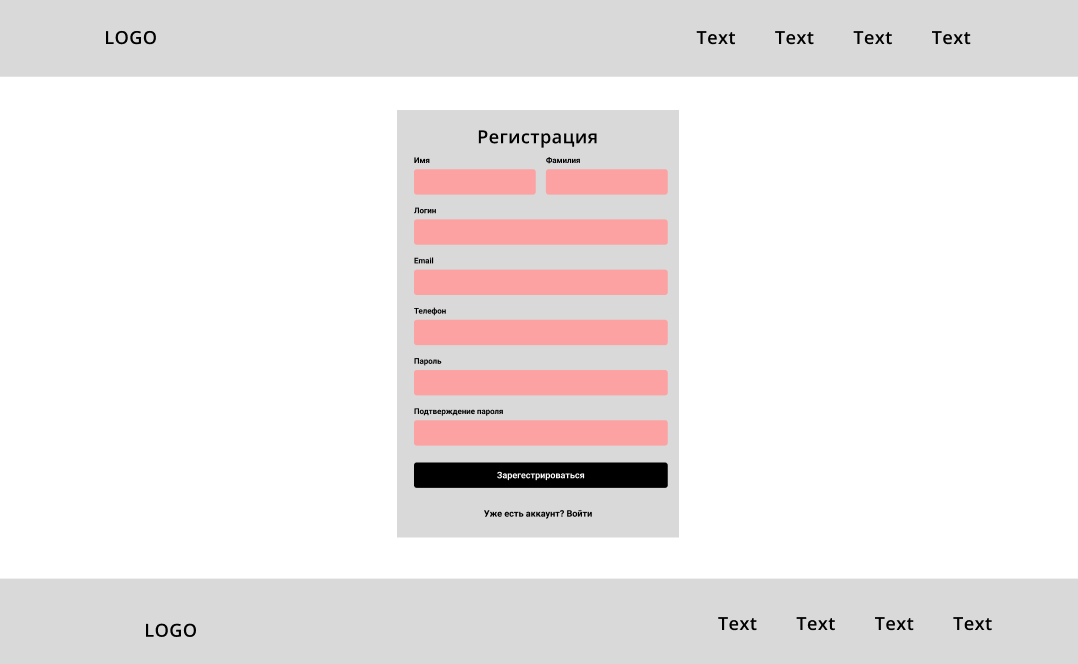


Рисунок 11 – Прототип страницы регистрации

Прототип страницы авторизации представлен на рисунке 12. На ней также есть форма, в которой пользователь должен заполнить поля: адрес электронной почты (email и пароль, чтобы войти в свой аккаунт. Также присутсвует кнопка «Войти» и текст для перехода на страницу регистрации, если у пользователя нет учетной записи.



Рисунок 12 – Прототип страницы авторизации

Прототип личного кабинета представлен на рисунке 13, который включает в себя две карточки, на которых отражено общее количество бронирований и количество часов. Также есть возможность просмотра истории бронирования, смены личных данных и пароля.

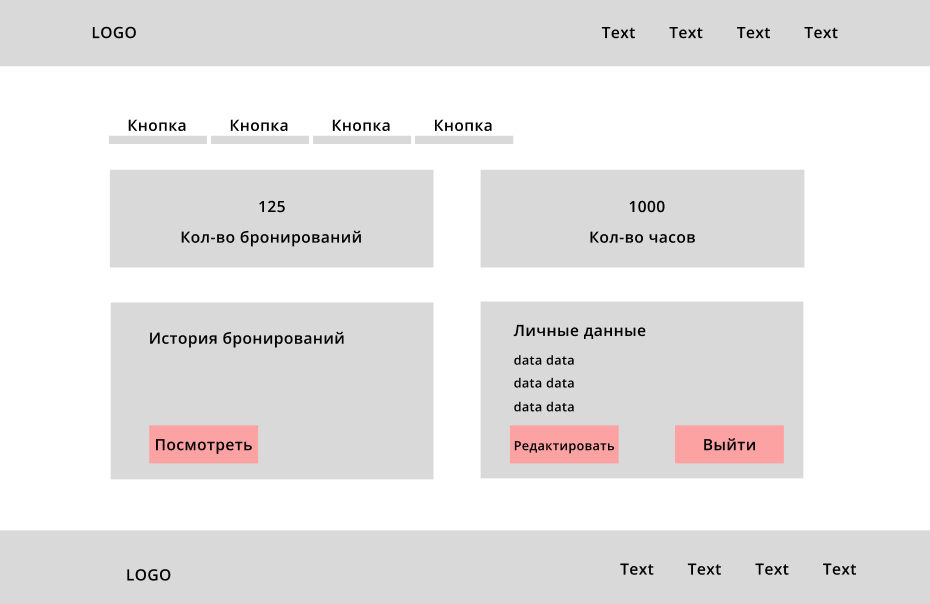


Рисунок 13 – Прототип личного кабинета

На рисунке 14 представлен прототип страницы «Бронирование подтверждено». На данную страницу пользователь попадет после нажатия по кнопке «Оформить заказ» на странице бронирования. На этой странице будут видны общие данные, данные об используемом тарифе, информация о самом клиенте, и небольшой блок об оплате



Рисунок 14 – Прототип страницы «Бронирование подтверждено»

Следующим прототипом стала страница админ-панели, представленной на рисунке 15, на которой администраторы системы смогут взаимодействовать с пользователями, тарифами, рабочими местами и бронированиями. На данной странице изображена боковая панель, с переходом на разные таблицы, которые есть в базе данных, чуть ниже располагаются кнопка перехода на сайт и кнопка выхода из учетной записи, в основной секции располагается сама таблица, в которой будут отображаться данные, кнопка для создания какой-либо записи, и кнопки для перехода по страницам, если записей окажется очень много.

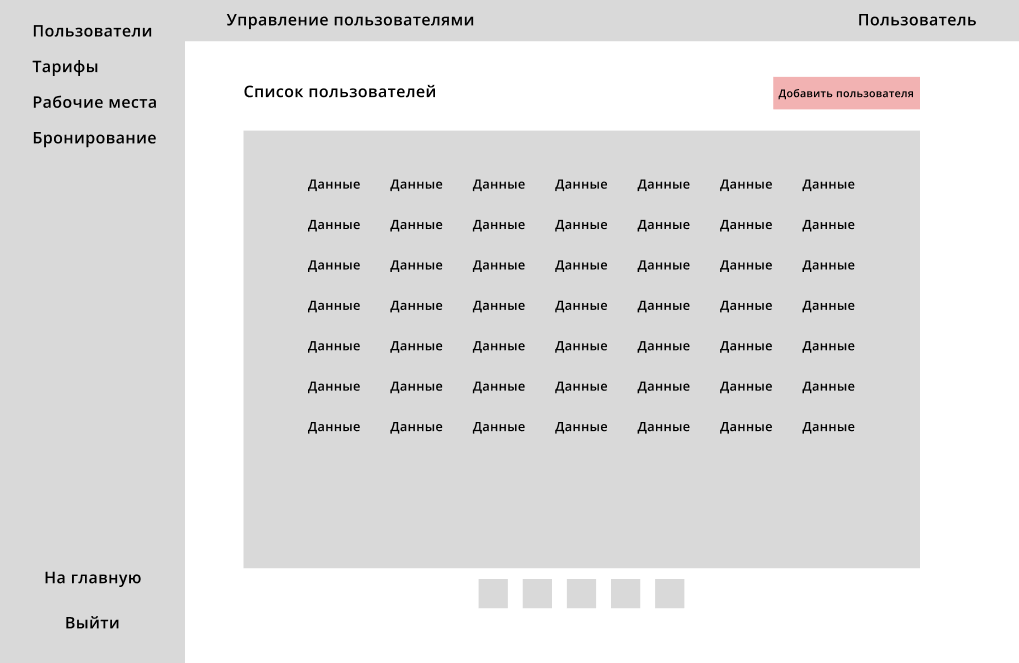


Рисунок 15 – Прототип админ-панели

Следующим этапом стала разработка прототипов под мобильные устройства. Интерфейс сохраняет всю основную структуру, адаптируясь под меньшие размеры экранов. Основным изменением является навигационная панель – она сворачивается в компактное раскрывающееся меню, доступное по нажатию на иконку с тремя горизонтальными линиями. Дизайн прототипов под мобильные устройства представлены на рисунках 16, 17, 18, 19, 20, 21



Рисунок 16 – Адаптивный прототип главной страницы



Рисунок 17 – Адаптивный прототип страницы бронирования



Рисунок 18– Адаптивный прототип страницы «Бронирование подтверждено»

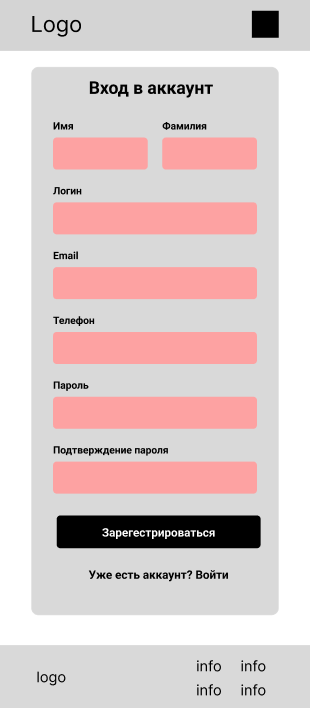


Рисунок 19 – Адаптивный прототип страницы регистрации

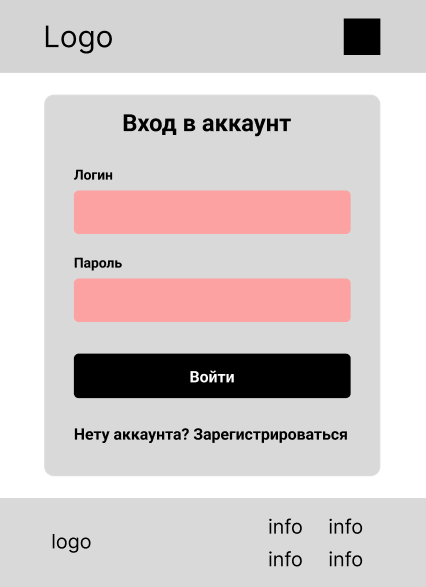


Рисунок 20 – Адаптивный прототип страницы авторизации

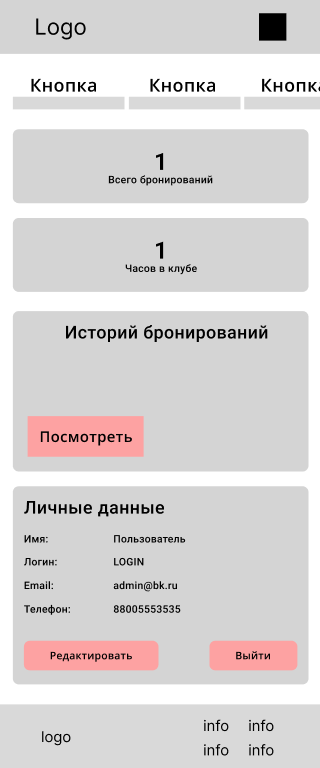


Рисунок 21 – Адаптивный прототип личного кабинета

### **3.4.2 Выбор цветовой гаммы и шрифтов**

Цветовая гамма включает в себя четыре цвета – черный, белый, синий и фиолетовый. Данные цвета сочетаются с тематикой компьютерного клуба. Фиолетовый, черный и белый в основном используется в виде фона страниц, каких-то отдельных блоков или карточках. Черный и белый является основным цветом текста. Синий цвет используется в кнопках. В остальных случаях используются оттенки данных цветов для создания различных акцентов, чтобы сохранить стилистику веб-приложения. Цветовая гамма представлена на рисунке 22

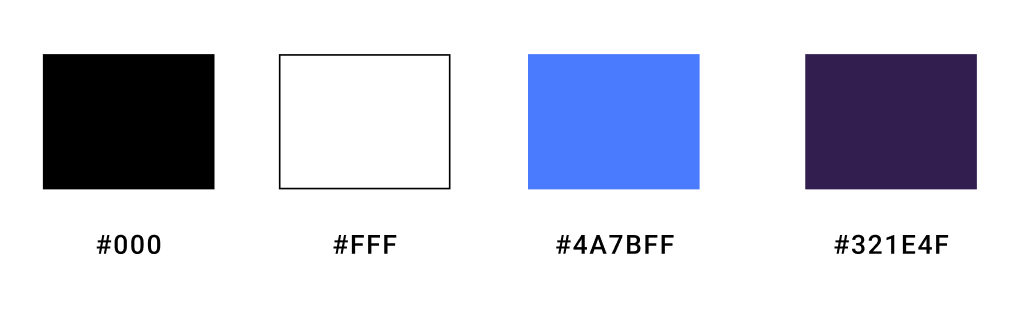


Рисунок 22 – Цветовая гамма

На страницах используется два шрифта, такие как Orbitron и Roboto, представлены на рисунке 23. Roboto –основной шрифт интерфейса, применяемый во всем тексте: заголовках, описаниях, кнопках и прочих элементах. Он обладает хорошей читаемостью, современным видом и поддержкой различных начертаний.

Orbitron используется точечно — для выделения названий, логотипа, акцентных заголовков и других ключевых элементов. Данный шрифт используется в проектах, связанных с технологиями, из-за чего он популярен в контексте, требующих цифрового или футуристического стиля. Виды начертаний в совокупности с цветом создают наибольший акцент на ключевые элементы.



Рисунок 23 – Шрифты «Orbtiron» и «Roboto»

### **3.4.3 Разработка элементов интерфейса**

На рисунке 24 представлен элемент карточки тарифа для дальнейшего перехода на страницу бронирования.



Рисунок 24 – Карточка тарифа

На рисунке 25 представлена кнопка «Оформить заказ» для дальнейшего перехода на страницу «Бронирование подтверждено»

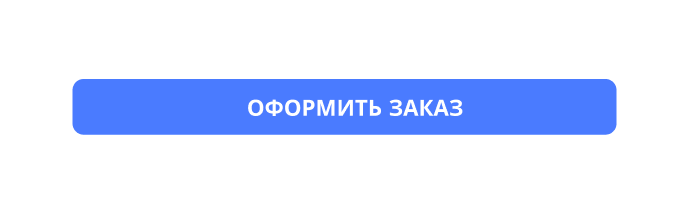


Рисунок 25 – Кнопка «Оформить заказ»

Логотип представлен на рисунке 26. Он представляет собой неоновое изображение компьютера в киберпанк стиле. Логотип символизирует киберспортивную тематику.



Рисунок 26 – Логотип «Cyber Arena»

### **3.4.4 Разработка дизайн макетов**

На основе прототипов были спроектированы макеты страниц веб-приложения для персональных компьютеров и смартфонов.

На рисунке 27 представлен макет главной страницы, в который входит навигационная панель, содержащая логотип и разделы такие как, «Фото», «Тарифы», «Комплектующие», «Как добраться, и кнопка «Войти», секция «приветствия пользователя», секция фотографий из компьютерного клуба, секция «Тарифы», включающий карточки с тарифами и их стоимостью, секция «Комплектующие», и секция с картой. В подвале странице находится логотип, основная информация и контакты

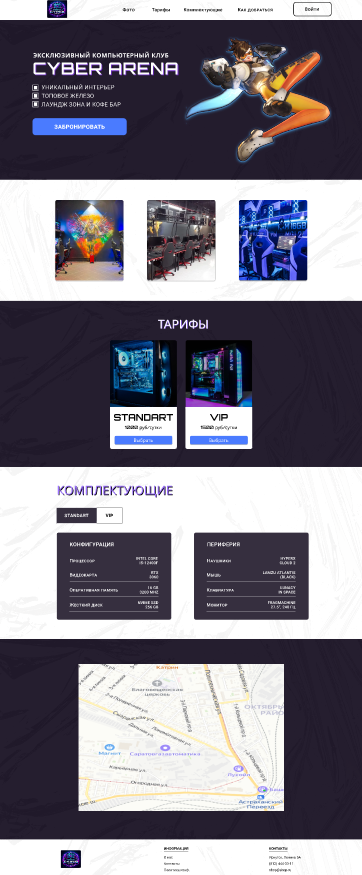


Рисунок 27 – Макет главной страницы

На рисунке 28 представлен макет страницы бронирования, в котором присутствуют навигационные панели, и две секции. В первой находится, название тарифа, количество свободных мест, блоки с указанием итоговой цены, количеством человек и количеством часов в бронировании, также присутствуют переключатели, и кнопка «Оформить заказ», справа блок «Оставить комментарий» и кнопка для быстрого бронирования на 4 часа.

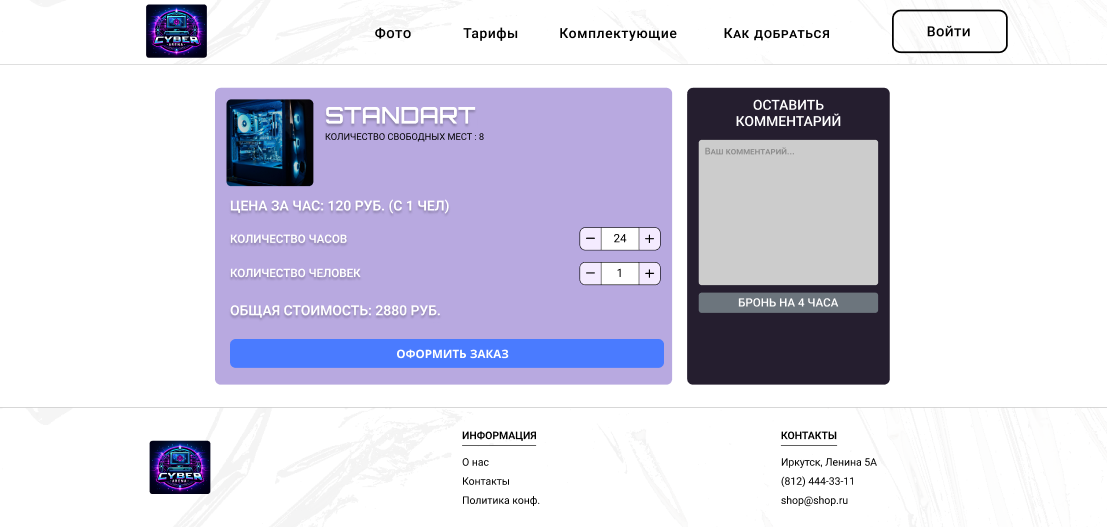


Рисунок 28 – Макет страницы бронирования

Макеты страниц регистрации и авторизации представлены на рисунках 29, 30. На каждом рисунке присутствует форма для заполнения, в которой пользователю нужно ввести свои данные. В случае с регистрацией это – имя, фамилия, логин, email, номер телефона, пароль, подтверждение пароля. В случае с авторизацией, это адрес электронной почты и пароль, который был указан при регистрации. В обоих случаях есть кнопки, «Зарегистрироваться», «Войти», и ниже альтернативный вариант, если есть аккаунт, то перейти на страницу авторизации, если же нету аккаунта, перейти на страницу регистрации.

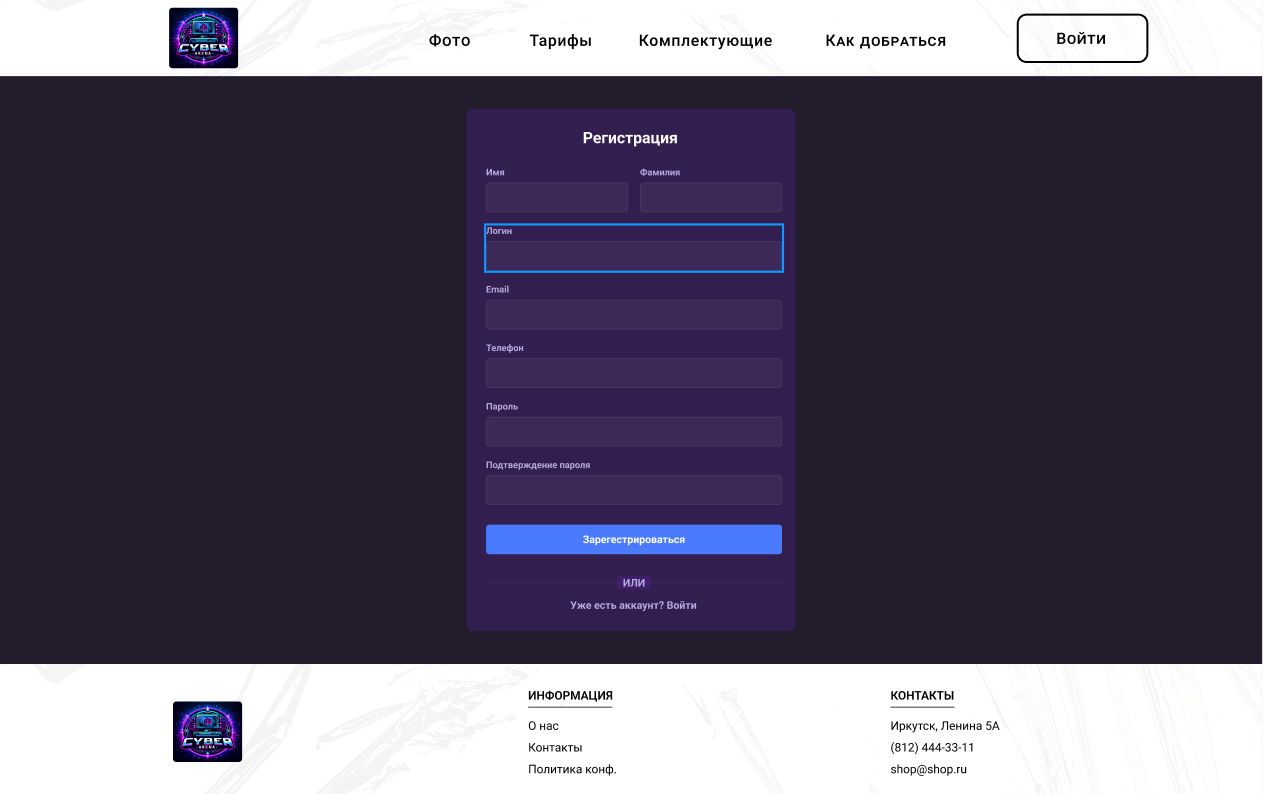


Рисунок 29 – Макет страницы регистрации

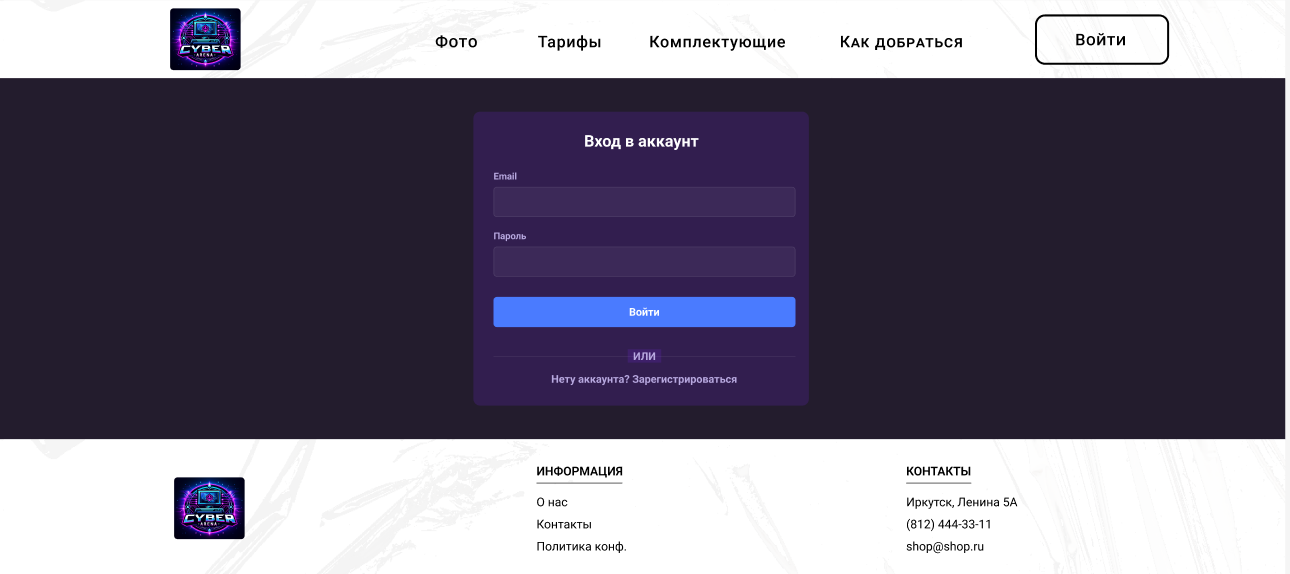


Рисунок 30 – Макет страницы авторизации

Макет страницы личного кабинета представлен на рисунке 31. Он включает в себя несколько блоков информации, на которых изображено две небольших карточки с общим количеством бронирований и количеством часов. Два блока чуть ниже, в которых отражена информация о последних бронированиях и личных данных. Также присутствуют небольшие кнопки для перехода по секциям «Обзор», «История бронирований», «Личные данные», где пользователь сможет изменить свои данные, и «Безопасность», где пользователь сможет изменить свой пароль

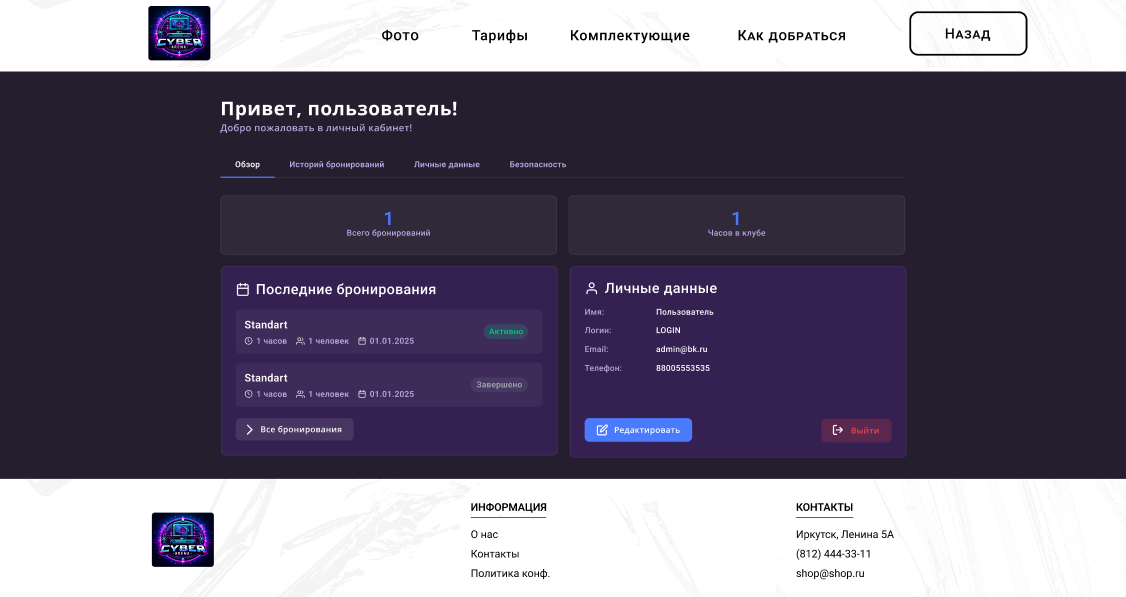


Рисунок 31 – Макет страницы личного кабинета

На рисунке 32 представлен макет страницы «Бронирование подтверждено», где изображены детали бронирования, которые включают в себя номер бронирования, дату создания и статус. Чуть ниже блок с информацией о тарифе, который включает в себя само название тарифа, указывается номер(а) рабочих мест, количество забронированных часов, и количество человек. В следующем блоке изображена информация о самом клиенте, это его имя и фамилия, электронная почта и номер телефона. Заключительным блоков является «Оплата», где написана итоговая стоимость и небольшой текст, что «Оплата производится на месте перед началом игры.», и чуть ниже располагаются две кнопки, для перехода в личный кабинет, или на главную страницу

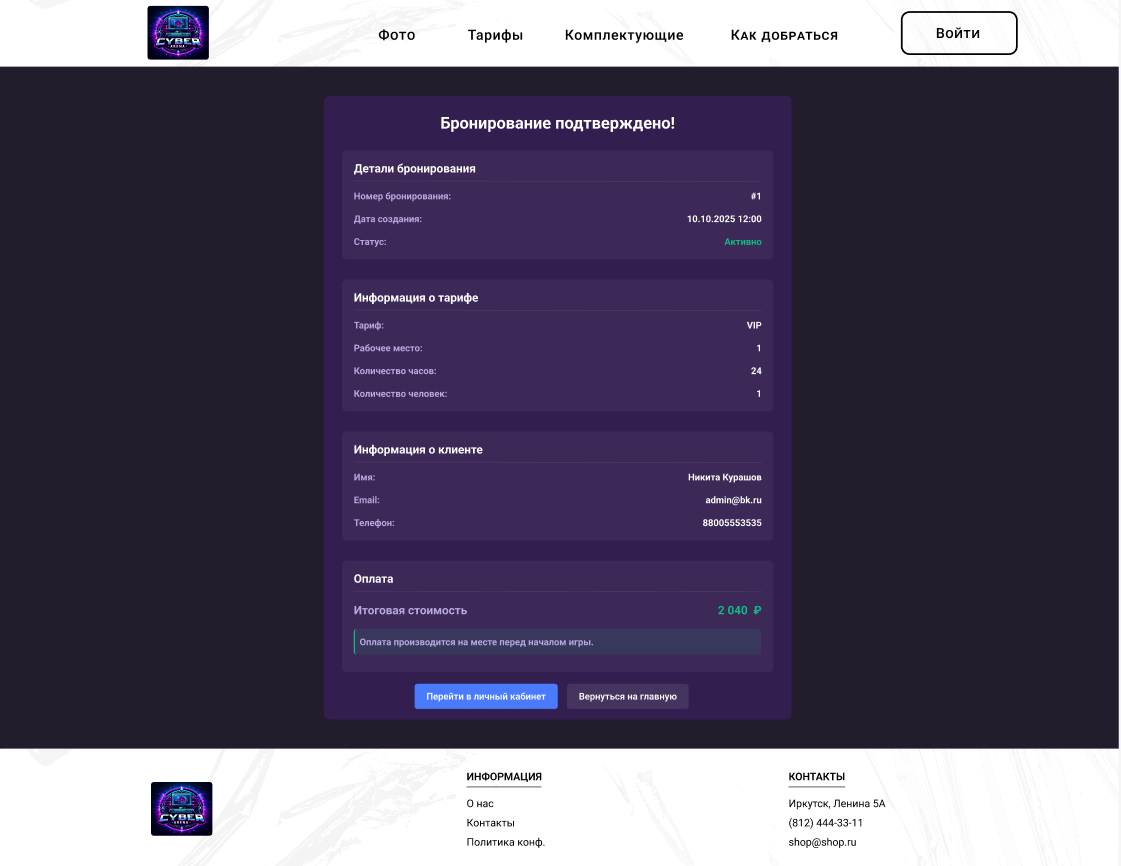


Рисунок 32 – Макет страницы «Бронирование подтверждено»

На рисунке 33 изображен макет админ-панели, который содержит в себе боковую панель с кнопками перехода на разные таблицы, а также кнопками перехода «На сайт» или «Выход», которая позволяет выйти из учетной записи. Также имеется шапка, где написано над какой таблицей ведется управление, а также имя администратора. В основной секции изображена информация о списке используемой таблице, кнопке, которая будет меняться в зависимости о таблице, кого нужно добавить, и сама таблица, где отражена основная информация о пользователе, тарифе, рабочих местах или бронированиях. Чуть ниже самой таблицы видны кнопки для перехода между страницами, если записей окажется слишком много.

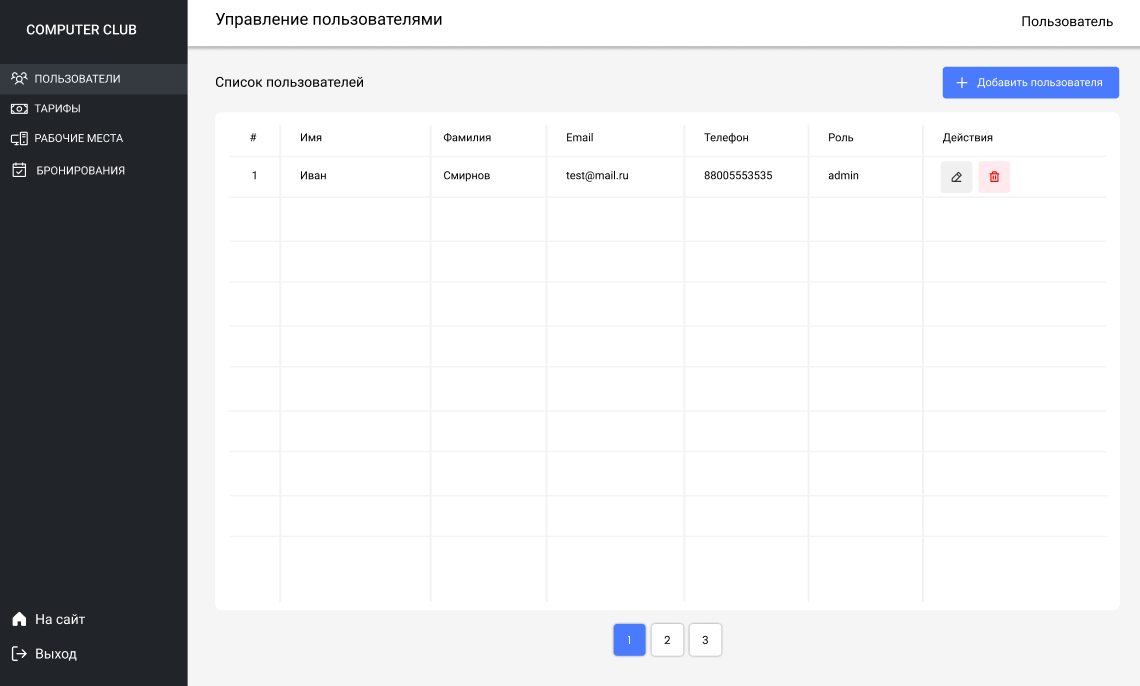


Рисунок 33 – Макет админ-панели

Следующим этапом проектирования стала разработка макетов для мобильных устройств. На рисунках 34, 35, 36 представлен макет главной страницы, адаптированный под смартфон. На странице присутствует выпадающее меню навигации, логотип, и также остальные секции, описанные ранее.

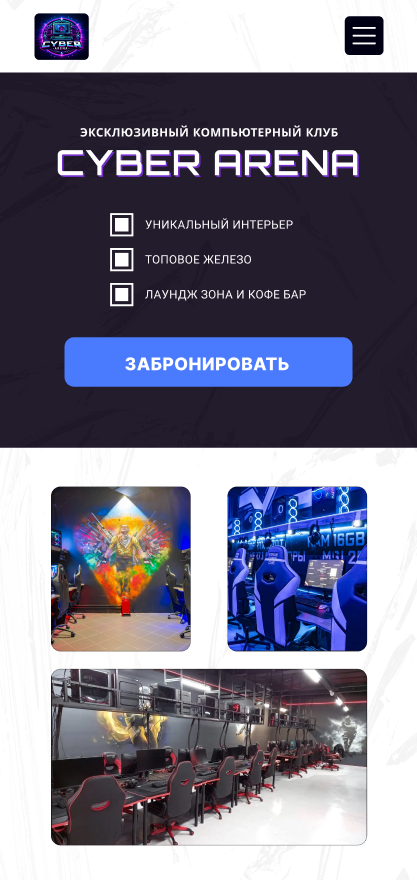


Рисунок 34 – Адаптивный макет главной страницы

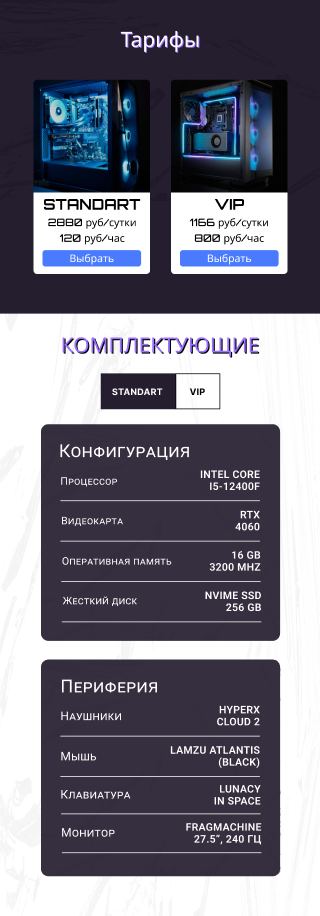


Рисунок 35 – Адаптивный макет главной страницы

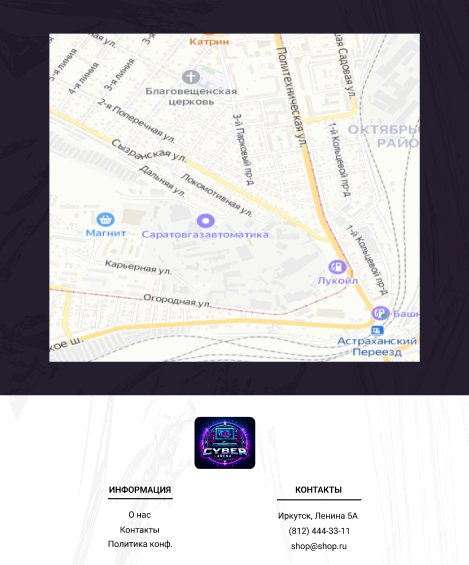


Рисунок 36 – Адаптивный макет главной страницы

На рисунке 37 представлен адаптивный макет страницы бронирования, который включает в себя выпадающее меню навигации с логотипом, основной информацией о бронировании, и возможностью оставить комментарий к бронированию



Рисунок 37 – Адаптивный макет страницы бронирования

На рисунке 38 представлен адаптивный макет страницы «Бронирование подтверждено», на котором присутствует выпадающее меню с логотипом, и секция с основной информацией о бронировании.

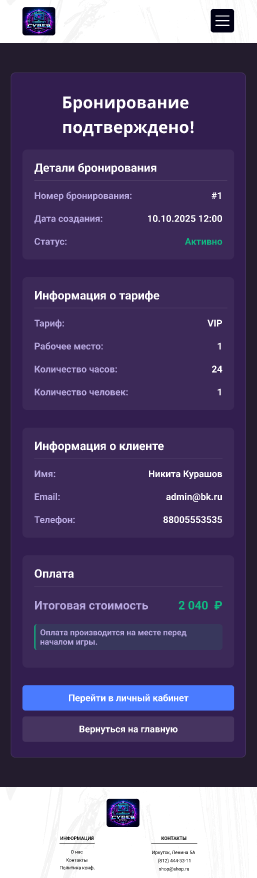


Рисунок 38 – Адаптивный макет страницы «Бронирование подтверждено»

На рисунке 39 представлен адаптивный макет страницы личного кабинета.

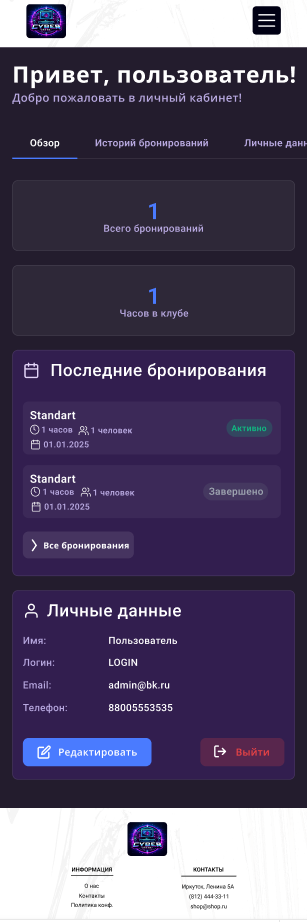


Рисунок 39 – Адаптивный макет страницы личного кабинета

# **4 Разработка веб-приложения**

## **4.1 Разработка интерфейса веб-приложения**

Разработка пользовательского интерфейса веб-приложения осуществлялась с использованием PHP-фреймворка Laravel и его встроенного шаблонизатора Blade, который обеспечивает удобное разделение логики приложения и представления данных. Для реализации адаптивного дизайна применялись технологии HTML5, CSS3 и JavaScript. Стилизация компонентов выполнена с использованием кастомных CSS-стилей.

На рисунках 40, 41 представлены два переиспользуемых компонента веб-приложения – это шапка и подвал сайта. В шапке присутствует логотип и навигационное меню для быстрого перехода между разделами на главной странице, а также кнопка входа в личный кабинет. В подвале также присутствует логотип, и два основных раздела: информация и контакты.



Рисунок 40 – Шапка веб-приложения



Рисунок 41 – Подвал веб-приложения

Следующим этапом станет рассмотрение ключевых представлений, с которыми будет взаимодействовать пользователь.

Главная страница предназначена для информирования пользователей о клубе и включает в себя несколько секций. Первая из них, это приветствие, представлена на рисунке 42, которая содержит название компьютерного клуба, преимущества данного клуба, картинку персонажа из игры, а также кнопку для перехода по якорной ссылке.

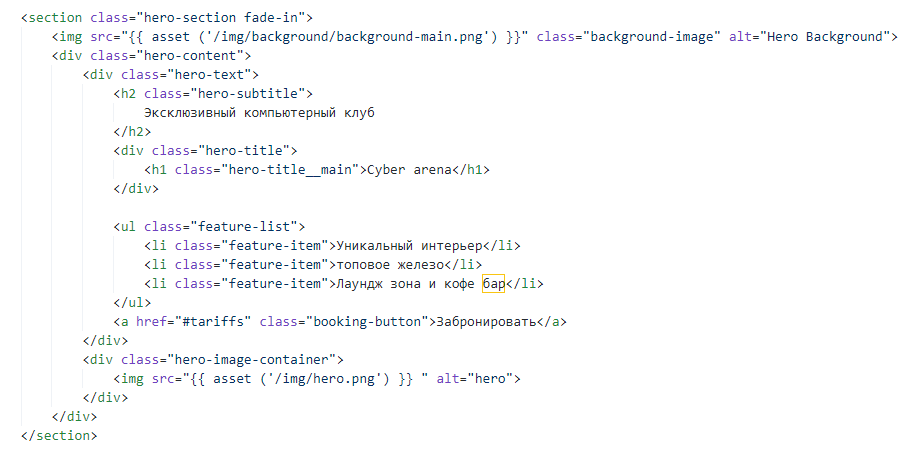


Рисунок 42 – секция «Приветствие»

Вторая секция – это галерея, в котором присутствуют несколько фотографий из данного компьютерного клуба, в соответствии с рисунком 43



Рисунок 43 – секция «Галерея»

Следующая секция – тарифы, которая реализована с помощью цикла foreach, который позволяет отображать все доступные тарифы из базы данных на основе массива, переданного в представление из контроллера. Каждый тариф выводится в виде отдельной карточки с изображением, названием, стоимостью в час, и в сутки, а также кнопкой перехода на страницу «Бронирование». Секция тарифы представлена на рисунке 44.



Рисунок 44 – секция «Тарифы»

Комплектующие – это следующая секция на главной странице веб-приложения. Она предоставляет информацию о конфигурации игровых мест, разделенных на категории Standart и VIP. Для переключения используются соответствующие кнопки с текстом Standart и VIP. На карточках представлены конфигурация: процессор, видеокарта, оперативная память, SSD-диск; и периферия: наушники, мышь, клавиатура, монитор, которые используется за компьютерами. Секция комплектующие представлена на рисунках 45, 46.

Функционал кнопок переключения, т.е. смены карточек с характеристиками, реализован при помощи JavaScript, в соответствии с рисунком 47



Рисунок 45 – Секция комплектующие

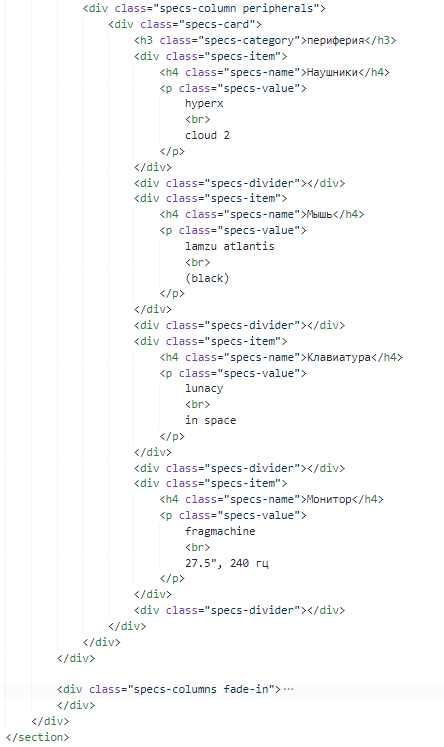


Рисунок 46 – Секция комплектующие



Рисунок 47 – Функционал переключения кнопок в секции комплектующие

Заключительной секцией главной страницы стала карта. Она реализована через iframe от Google Maps, где изображено местоположение компьютерного клуба, представлена на рисунке 48

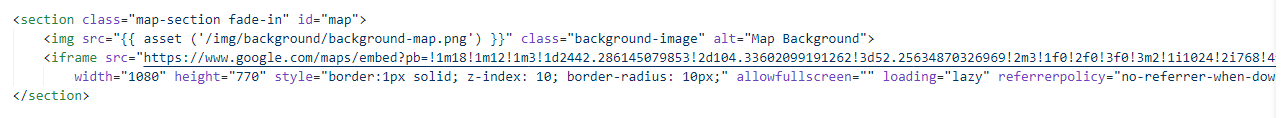


Рисунок 48 – Секция карта

Следующим представлением является «Бронирование», в ней присутствует шапка, основной контент, с которым будет взаимодействовать пользователь, а также подвал. Данное представление дает возможность пользователю оформить заказ выбранного тарифа. Оно состоит из двух колонок: форма оформления и ее основная информация (слева), и дополнительная информация (справа), которая содержит блок, где можно указать комментарий и кнопку для быстрого бронирования. Секция бронирование представлено на рисунке 49.



Рисунок 49 – Секция бронирование

В заголовке формы указано количество свободных мест, которое берется из переменной availableSpots, которое прописано в контроллере, и оно в свою очередь берет значение из базы данных по тарифу. Ниже в форме указывается цена за час, которая также берется из базы данных, и в зависимости от значения в тарифе, указывается, за что будет платить пользователь (отдельное место или VIP-комната).

Ниже представлены кнопки «-» и «+», которые нужны для уменьшения или увеличения количества человек или часов. Эти кнопки связаны с JavaScript-функциями, которые изменяют значения в соответствующих полях и автоматически пересчитывают итоговую стоимость бронирования. Стоимость отображается в реальном времени на экране пользователя. Расчёт стоимости зависит от типа тарифа: если это общий зал – цена умножается на количество часов и человек, если это отдельная VIP-комната – учитывается только количество часов.

Кроме того, реализована функция быстрого бронирования — кнопка «Бронь на 4 часа» автоматически устанавливает значение поля «количество часов» в 4 и обновляет цену. Это упрощает процесс бронирования для пользователей, которые хотят выбрать стандартный временной слот без ввода данных вручную.

Все расчеты производятся на стороне клиента с помощью JavaScript, а уже при оформлении заказа, т.е. создании бронирования, все это осуществляется на стороне сервера. JavaScript код представлен на рисунках 50, 51



Рисунок 50 – JavaScript код для представления «Бронирование»

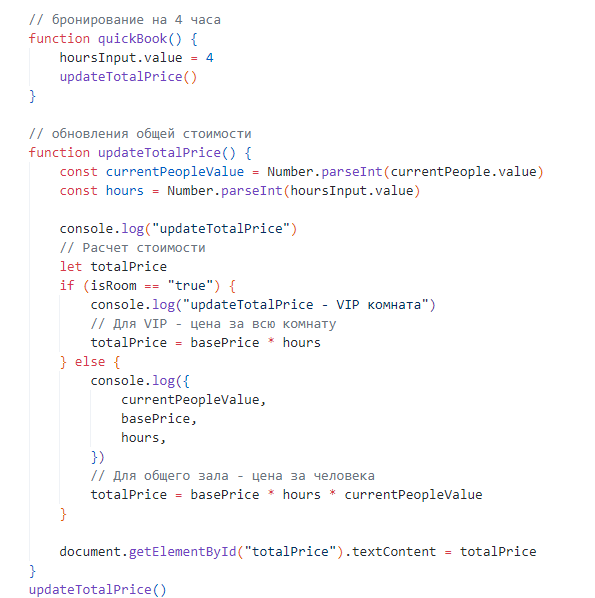


Рисунок 51 – JavaScript код для представления «Бронирование»

## **Разработка базы данных веб-приложения**

Вторым из этапов разработки веб-приложения является разработка базы данных. Она разработана с использованием миграций. Каждая такая миграция создает отдельные таблицы, в которой указываются все необходимые поля, их типы данных, ограничения и связи.

Одной из ключевых таблиц является users – таблица пользователей, предназначенная для хранения данных о пользователях. В таблице присутствуют поля: first\_name, last\_name, email, login, password, phone, а также role, определяющее роль пользователя (администратор или клиент). Поля email и login имеют уникальные ограничения. Также таблица автоматически создает временные метки при помощи метода timestamps(). Миграция users представлена на рисунке 52.

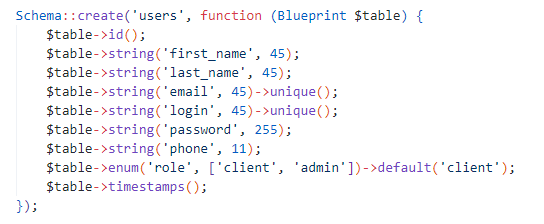
  
Рисунок 52 – Миграция users

Таблица workstations содержит данные о рабочих местах. Каждое рабочее место имеет уникальный номер (number), статус (например, "Свободно" или "Занято") и тип (например, "Standart" или "VIP"). Это позволяет администратору управлять доступностью оборудования в режиме реального времени. Миграция workstations представлена на рисунке 53.

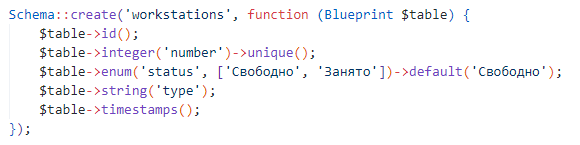


Рисунок 53 – Миграция workstations

Тарифы, по которым осуществляется бронирование, хранятся в таблице tariffs. В ней присутствуют поля name, price\_per\_hour, is\_room (значение, указывающее, бронируется ли VIP-комната), и image, которое хранит путь к изображению. Миграция tariffs представлена на рисунке 54.

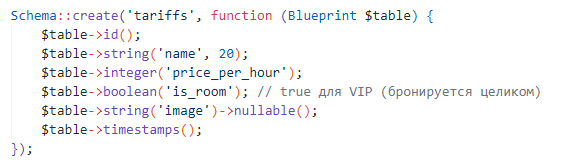


Рисунок 54 – Миграция tariffs

Основная таблица для оформления заказов – bookings. В ней хранятся внешние ключи на таблицы users и tariffs, а также такие поля, как количество часов, количество человек, комментарий к заказу, итоговая стоимость и статус брони (active, completed, cancelled). Миграция bookings представлена на рисунке 55.

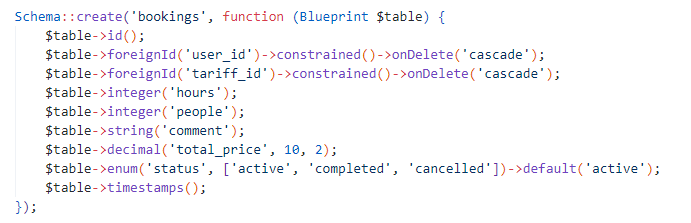


Рисунок 55 – Миграция bookings

Связь между бронированиями и рабочими местами реализована через промежуточную таблицу booking\_workstation. Она обеспечивает связь «многие ко многим» между бронированиями и рабочими местами. Миграция booking\_workstation представлена на рисунке 56.

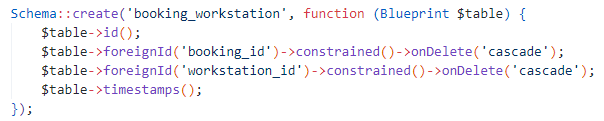


Рисунок 56 – Миграция booking\_workstation

Для тестового наполнения базы данных использовались сидеры, создающие начальные данные, необходимые для работы приложения. Первым сидером стал TariffSeeder, который автоматически добавляет два тарифа: стандартный и VIP, для отображения на главной странице. В каждом тарифе задаются название, стоимость, тип (комната или общий зал) и путь к изображению, в соответствии с рисунком 57.



Рисунок 57 – Сидер TariffSeeder

Следующим сидером стал UserSeeder, который предназначен для создания пользователя с ролью администратора. Это позволяет сразу после запуска проекта входить в админ-панель без необходимости ручной регистрации, в соответствии с рисунком 58. Даже при создании записи через сидер, пароль хешируется.

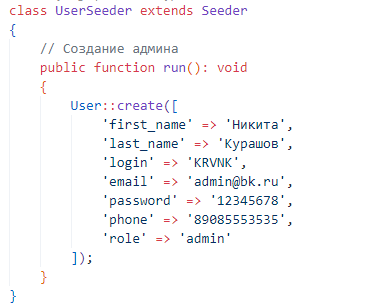


Рисунок 58 – Сидер UserSeeder

Заключительным сидером стал WorkStationSeeder, в котором создаются 15 рабочих мест, при помощи цикла, у которых указывается номер, статус и тип. Первые 10 рабочих мест относятся к типу Standart, а оставшиеся 5 – к типу VIP.

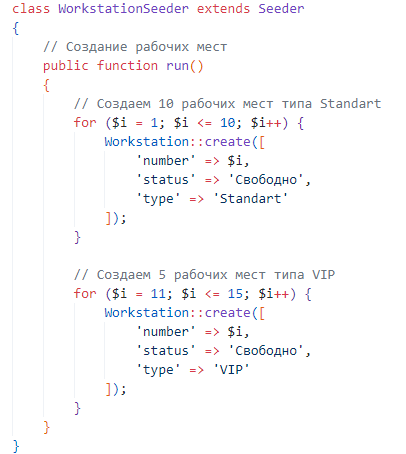


Рисунок 59 – Сидер WorkStationSeeder

После выполнения миграций и запуска сидеров, вся структура базы данных была создана в системе СУБД MySQL. Чуть ниже представлены таблицы, которые были сформированы.

На рисунке 60 представлена таблица users, в которой отображаются записи пользователей. Видно, что таблица содержит поля id, first\_name, last\_name, login, password (захешированный), email, phone, role, а также временные метки.

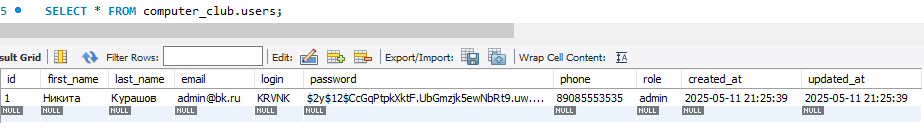


Рисунок 60 – Таблица Users

На рисунке 61 видно таблицу tariffs с двумя записями, созданными сидером TariffSeeder. Указаны названия тарифов, стоимость за час, значение is\_room и путь к изображению, а также временные метки.

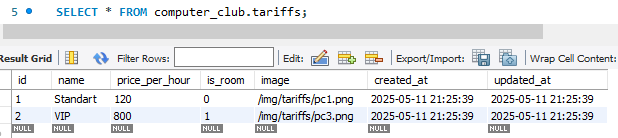


Рисунок 61 – Таблица tariffs

На рисунке 62 показана таблица workstations, содержащая 15 рабочих мест: 10 стандартных и 5 VIP. Каждое рабочее место имеет уникальный номер, статус и тип.

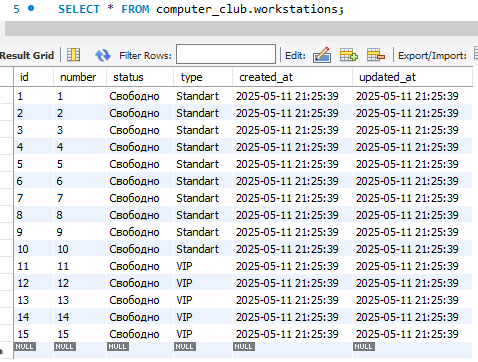


Рисунок 62 – Таблица WorkStations

## **4.3 Разработка веб-приложения**

Заключительным этапом разработки веб-приложения является разработка серверной части, обеспечивающей обработку логики веб-приложения, взаимодействие с базой данных, управление маршрутами, а также защиту и управление доступом.

В рамках серверной части были разработаны модели, контроллеры, middleware и маршруты. Модели описывают структуру данных и связи между сущностями, контроллеры описывают логику обработки запросов и управления ответами приложения, маршруты описывают, какие действия должны выполняться при обращении к определённым URL-адресам веб-приложения, а middleware обеспечивает защиту и контроль доступа к различным разделам сайта, используя маршруты.

В проекте использовались следующие модели: User, Tariff, Workstation, Booking. Каждая модель соответствует отдельной таблице в базе данных и определяет связи между сущностями, которые впоследствии упрощают выборку и обработку данных.

Модель User описывает пользователя и содержит поля: имя, фамилию, логин, электронную почту, номер телефона, пароль и роль. В модели реализована связь «один ко многим» с моделью Booking, что позволяет получать все бронирования конкретного пользователя. Также реализован метод для определения автоматического преобразования атрибутов, в данном случае для хеширования пароля, что избавляет от необходимости дублировать код в контроллерах, в соответствии с рисунком 63.



Рисунок 63 – Модель User

Модель Tariff описывает тариф и содержит поля: название, цена в час, значение is\_room и путь к картинке. В данной модели реализована связь «один ко многим» с моделью Booking, в соответствии с рисунком 64

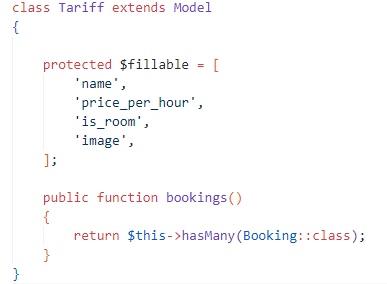


Рисунок 64 – Модель Tariff

Модель Workstation описывает рабочие места и содержит поля номер, статус и тип, а также реализует связь «многие ко многим» c моделью Booking, в соответствии с рисунком 65

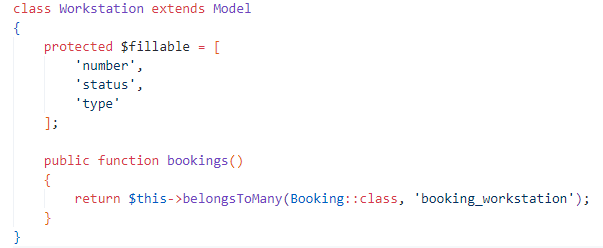


Рисунок 65 – Модель Workstation

Модель Booking представляет процесс бронирования и содержит такие поля, как user\_id, tariff\_id (внешние ключи), количество часов, количество человек, комментарий, итоговая стоимость и статус. Она связана с пользователем, тарифом и рабочими местами, в соответствии с рисунком 66.



Рисунок 66 – Модель Booking

Следующим этапом будет описание контроллеров. Обработка логики бронирования реализована в BookingController. В этом контроллере присутствует 3 метода. Первый из них отвечает за логику отображения информации на странице бронирования. В методе show, мы находим соответствующий тариф из базы данных и определяет количество доступных рабочих мест в зависимости от типа тарифа. Для VIP-комнаты проверяется наличие свободных мест типа «VIP», а для обычного тарифа подсчитывается количество свободных рабочих мест типа «Standart». Метод также определяет максимальное количество человек для бронирования, в соответствии с рисунком 67

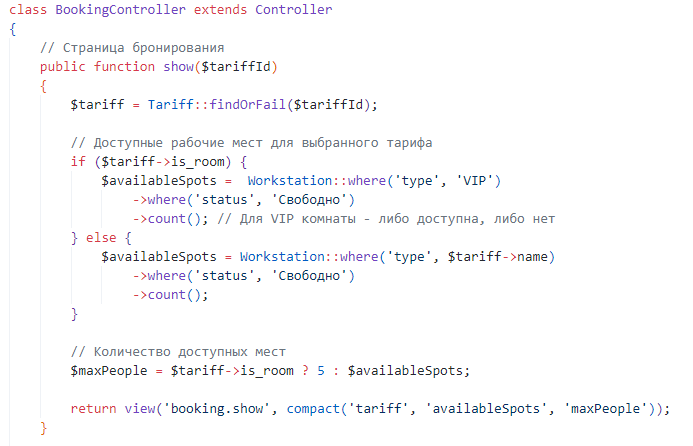


Рисунок 67 – Метод show в BookingController

Одним из ключевых методов данного контроллера является store – он отвечает за сохранения бронирования от пользователя. В нем мы сначала проводим валидацию данных, затем определяем тип тарифа и рассчитываем цену. Если это общий зал, цена рассчитывается как произведение количества часов на количество человек. Если это VIP-комната – только по часам. Далее проходим проверку на нецензурную лексику, затем создаем объект бронирования и сохраняем все в базу данных, после этого происходит выборка свободных рабочих мест, и их дальнейшая привязка к брони и изменение статуса на «Занято», в соответствии с рисунками 68, 69.



Рисунок 68 – Метод store в BookingController



Рисунок 69 – Метод store в BookingController

Заключительным методом является confirmation. Он отвечает за логику отображения информации на представлении «Бронирование подтверждено». В нем мы находим созданное бронирование, проходим проверку, принадлежит ли данное бронирование текущему пользователю и возвращаем представление, в соответствии с рисунком 70.

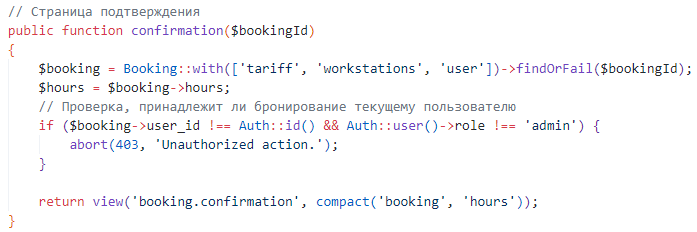


Рисунок 70 – Метод confirmation в BookingController

Контроллер ProfileController отвечает за отображение личного кабинета, обновление личных данных пользователя и пароля. В методе index, внутри которого подсчитывается общее количество бронирований, а также суммарное время, проведенное пользователем в клубе. Также в этом представлении отображается фиксированное количество бронирований на вкладке «История бронирований» , в соответствии с рисунком 71.

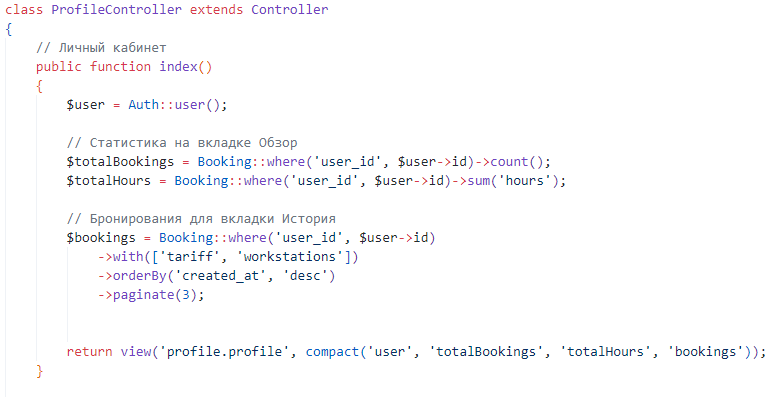


Рисунок 71 – Метод index в ProfileController

Метод update выполняет валидацию и сохраняет новые значения личных данных, которые ввел пользователь на странице личного кабинета, в соответствии с рисунком 72.

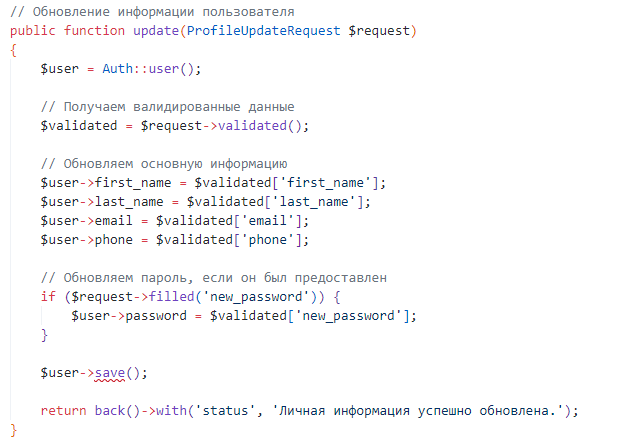


Рисунок 72 – Метод update в ProfileController

Метод updatePassword необходим для изменения пароля. Здесь также получаем валидированные данные, это текущий пароль пользователя, и новый пароль, который придумает пользователь, и при успешном обновлении пароля, нам высвечивается блок с информацией, что «Пароль успешно изменен», в соответствии с рисунком 73.

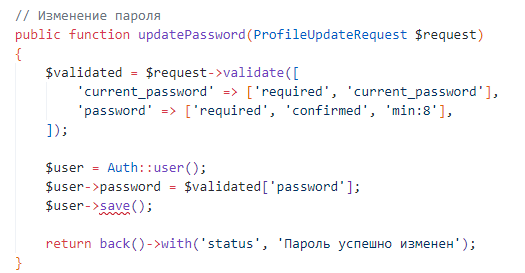


Рисунок 73 – метод updatePassword в ProfileController

На рисунке 74 был реализован AdminMiddleware, который ограничивает доступ к административной панели. В нем проверяется, авторизован ли пользователь и соответствует ли его роль значению «admin».

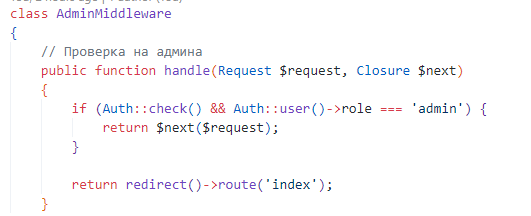


Рисунок 74 – посредник AdminMiddleware

Доступ к представлениям реализован через маршруты в файле web.php. Все маршруты сгруппированы по доступу: публичные – доступны всем, а защищенные –только авторизованным пользователям с помощью middleware auth. А маршруты для администраторов дополнительно защищены AdminMiddleware, в соответствии с рисунком 75. В приложении также используются различные HTTP-методы: GET для получения данных и отображения страниц, POST для создания новых записей, PUT для обновления существующих данных и DELETE для удаления записей.



Рисунок 75 – маршруты в web.php

Заключительным этапом является подключение к базе данных, оно реализовано с помощью файла «.env», в котором указываются все параметры подключения: тип СУБД, имя БД, логин, пароль и адрес сервера, в соответствии с рисунком 76.

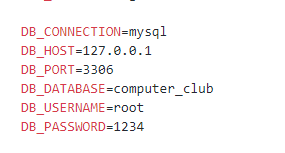


Рисунок 76 – Подключение к базе данных

# **Приложение А**

**Министерство образования Иркутской области**

Государственное бюджетное профессиональное

образовательное учреждение Иркутской области

«Иркутский авиационный техникум»

(ГБПОУИО «ИАТ»)

**ТЕХНИЧЕСКОЕ ЗАДАНИЕ**

**ВЕБ-ПРИЛОЖЕНИЕ «КОМПЬЮТЕРНЫЙ КЛУБ»**

|  |  |  |
| --- | --- | --- |
| Руководитель: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | (Н.Р. Карпова) |
|  | (дата, подпись) |  |
| Студент: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | (Н.А. Курашов) |
|  | (дата, подпись) |  |

Иркутск 2025

**1. Введение**

**1.1.** **Общие сведения**

Документ представляет собой техническое задание на создание веб-приложения для компьютерного клуба. Приложение будет предназначено для управления и автоматизации процессов бронирования рабочих мест, а также взаимодействия с клиентами.

**1.2 Цели и задачи**

Цель проекта: разработка веб-приложения «Компьютерный клуб», который обеспечит автоматизацию процессов бронирования рабочих мест, управления тарифами и взаимодействия с клиентами.

Задачи:

* Разработка пользовательского интерфейса для пользователей
* Реализация функционала бронирования рабочих мест, управление тарифами и панели администратора

**2. Основания для разработки**

**2.1.** **Нормативные документы**

Документ основывается на следующих нормативных документах:

* Методические указания КП МДК.09.01 2024 ВЕБ.
* приказ №325-у от 24.01.2025 г. Об утверждении тем курсовых проектов.
* ГОСТ Р 52872-2019 «Интернет-ресурсы и другая информация, представленная в электронно-цифровой форме».
* ГОСТ Р ИСО/МЭК 25010-2015 «Требования и оценка качества систем и программного обеспечения (SQuaRE). Модели качества систем и программных продуктов»

**2.2.** **Проектные документы**

Проектные документы включают:

* Пояснительную записку
* Руководство пользования

**3. Назначение системы**

**3.1. Общее описание**

Веб-приложение «Компьютерный клуб» предназначено для автоматизации процессов управления компьютерным клубом. Для администратора организации необходимы такие функции, как управление списком рабочих мест, настройка тарифов, управление бронированием. Для пользователя – просмотр доступных рабочих мест и их характеристик, бронирование рабочего места на выбранное время, оплата услуг.

**3.2.** **Преимущества и новизна**

Веб-приложение будет предоставлять:

* Удобный интерфейс.
* Выбор из нескольких тарифов.

**4. Требования к веб-приложению**

**4.1.** **Функциональные требования**

* Регистрация и авторизация
* Возможность регистрации новых пользователей
* Вход в систему
* Личный кабинет
* Персональные данные
* История бронирований
* Смена личных данных
* Смена пароля
* Админ-панель (если пользователь является администратором)
* Раздел «Приветствие»
* Раздел «Тарифы»
* Описание цен (в зависимости от тарифа)
* Раздел «Комплектующие»
* Описание комплектующих (в зависимости от тарифа)
* Раздел «Оформление заказа»
* Выбор количества часов
* Выбор количества человек
* Возможность оставить комментарий
* Раздел «Бронирование подтверждено»
* Детали бронирования
* Информация о тарифе
* Информация о клиенте
* Оплата
* Панель администратора
* Управление пользователями
* Управление тарифами
* Управление рабочими местами
* Управление бронированием

**4.2** **Технические требования**

* Производительность:
  + Время отклика системы не более 2 секунд.
  + Обработка до 500 задач одновременно.
* Надежность:
  + Доступность системы не менее 99,5% в год.

**4.3 Эксплуатационные требования**

* Удобство использования:
  + Удобный пользовательский интерфейс.

**5 Требования к техническому обеспечению**

**5.1 Оборудование**

Минимальные системные требования для рабочей станции:

1) Процессор: Intel Core i3 (или выше) / AMD Ryzen 5 (или выше);

2) Оперативная память: 8 Гб;

3) Жёсткий диск: 128 Гб;

Минимальные системные требования для сервера:

1) Процессор: Intel Core i5 (или выше) / AMD Ryzen 5 (или выше);

2) Оперативная память: 8 Гб;

3) Жёсткий диск: 128 Гб;

**5.2. Сетевые требования**

* Сеть: Доступ в Интернет со скоростью не менее 10 Мбит/с.
* Сетевые протоколы: Поддержка TCP/IP, HTTPS.

**6 Требования к программному обеспечению**

**6.1 Программные компоненты**

* Операционная система: Windows 10 и выше, Linux, MacOs.
* Базы данных: MySQL версии не ниже 10.0.
* Программное обеспечение: PHP 7.4 или выше

**6.2 Интерфейсы**

* Интерфейс пользователя: Веб-интерфейс с поддержкой браузеров Chrome, Firefox, Edge, Opera, Yandex.

**7 Организационно-технические требования**

**7.1 Этапы разработки**

В таблице 1 представлены плановые сроки начала и окончания работы по созданию веб-приложения

Таблица 1 – Плановые сроки по созданию веб-приложения

|  |  |
| --- | --- |
| **Наименование этапов курсового проекта** | **Сроки** |
| Предпроектное исследование предметной области (выбор темы, постановка цели, задач, описание области применения, исследование предметной области) | 30.01.25 - 09.02.25 |
| Разработка технического задания (выбор архитектуры программного обеспечения, выбор типа пользовательского интерфейса, выбор языка и среды программирования) | 09.02.25 - 19.02.25 |
| Проектирование программного обеспечения.  (разработка структурной схемы ПО, проектирование базы данных) | 19.02.25 - 04.03.25 |
| Оформление пунктов пояснительной записки (введение, предпроектное исследование, техническое задание и проектирование веб-приложения) | 04.03.25 - 11.03.25 |
| Разработка (программирование) и отладка программного продукта.  Результат- готовый программный продукт. | 11.03.25 - 12.05.25 |
| Составление программной документации (оформление ПЗ, руководство пользователя и презентации) | 13.05.25 - 13.06.25 |