ГУАП

КАФЕДРА №42

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

доцент, кандидат тех. наук		А.В. Бржезовский
должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К КУРСОВОЙ РАБОТЕ

Автоматизированная система спортивного клуба

по дисциплине: МЕТОДЫ И СРЕДСТВА ПРОЕКТИРОВАНИЯ ИНФОРМАЦИОННЫХ СИСТЕМ И ТЕХНОЛОГИЙ

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. № 4128		В.А.Тарапанов
	подпись, дата	инициалы, фамилия

Санкт-Петербург 2024

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Целью курсовой работы является разработка автоматизированной информационной системы спортивного клуба.

Основной задачей разработки данной системы является создание системы, благодаря которой спортсмены могут записываться на тренировки к интересующим их тренерам, а другие сотрудники могут администрировать все процессы, относящиеся к ним, в системе.

Перед разработкой системы необходимо составить перечень правил, требований, список будущих пользователей и их обязанности, расписать алгоритмы поведения пользователей.

Для разработки автоматизированной информационной системы HTML был выбран следующий стек технологий: HTML, CSS, JavaScript для создания веб-интерфейса и реализации интерактивных элементов, а также обработки логики на стороне клиента. База данных была реализована на MySQL.

1 Описание предметной области

1.1 Исходные данные

Спортивный клуб «Молот» сталкивается с проблемой регистрацией новых членов клуба, менеджеры часто не успевают отвечать на входящие звонки и тем самым теряют потенциальных клиентов.

Необходимо разработать онлайн-сервис для спортивной организации. Онлайн-сервис будут представлен в виде сайта, состоящего разделов: расписание групповых тренировок, где пользователь сможет ознакомиться с информацией о ближайших групповых занятиях, запись на индивидуальные занятия, где пользователь сможет выбрать тренера и тип тренировки, который ему нужен, приобретение спортивных пакетов клуба, где пользователь сможет выбрать абонемент, который соответствует заданным целям и финансовым возможностям , магазин для приобретения внутренней продукции клуба, где пользователь сможет приобрести товары: питания, для тренировочного процесса, товары с атрибутикой клуба, личный кабинет пользователя

1.2 Возможности бизнеса

1. **Увеличение числа клиентов**

- Упрощение процесса записи и покупки услуг привлечет больше новых клиентов, которые ценят удобство и скорость.
- Сокращение потерь потенциальных клиентов за счет автоматизации процесса записи на занятия.

2. **Повышение уровня обслуживания**

- Клиенты смогут самостоятельно выбирать тренировки, тренеров и абонементы в удобное для них время, что увеличит удовлетворенность услугами клуба.

3. **Дополнительные источники дохода**

- Продажа товаров в онлайн-магазине (спортивное питание, инвентарь, клубная атрибутика) позволит увеличить доход клуба.
- Введение различных спортивных пакетов с гибкими условиями откроет возможности для охвата аудитории с разным уровнем дохода.

4. **Оптимизация работы сотрудников**

- Снижение нагрузки на менеджеров за счет автоматизации записи и обработки платежей, что позволит им сосредоточиться на других важных задачах, например работе с текущими клиентами.
- Сокращение времени ожидания для клиентов, так как основную часть операций они смогут выполнять самостоятельно.

1.3 Бизнес-цели

BO-1 Автоматизация процессов. Сокращение времени обработки запросов клиентов

BO-2 Привлечение новых клиентов. Активное продвижение онлайн-сервиса через соцсети, email-рассылки и рекламу.

BO-3 Оптимизация работы сотрудников.Снижение нагрузки на менеджеров.

1.4 Критерии успеха

- Увеличение общей клиентской базы клуба на 20% за первый год.
- Сокращение времени ожидания записи на тренировки до 1-2 минут.
- Увеличение выручки от продаж абонементов и дополнительных услуг на 20% в течение первого года после запуска

1.5 Бизнес-риски

- Высокая конкурентность среди других спортивных клубов
- Отказ пользователей от возможности регистрации онлайн в связи с риском утери конфиденциальности данных.

- Проблемы с хостингом, взлом сайта, что может привести к потери базы данных
- Изменения в предпочтениях потребителей

2 Требования к системе

2.1 Определение требований пользователей

2.1.1 Определение списка пользователей

В таблице 1 представлены классы пользователей.

Роль	Описание
Спортсмен	В задачи пользователя спортивного клуба «Молот» входит выбор интересующих услуг, таких как групповые или индивидуальные тренировки, а также ознакомление с расписанием доступных занятий и тренеров. Пользователь может записываться на тренировки, выбирая дату, время и тренера через онлайн-сервис. Также в его задачи входит просмотр информации о спортивных пакетах и товарах, доступных в магазине клуба
Администратор	В задачи администратора клуба входит управление расписанием тренировок, добавление новых занятий и обновление времени существующих. Администратор следит за данными пользователей, проверяя и обновляя их профили. Администратор контролирует процесс оплаты абонементов, индивидуальных тренировок и покупок.
Тренер	В задачи тренера входит создание и управление расписанием своих тренировок, корректировка времени занятий при необходимости. Тренер управляет списком записанных клиентов, подтверждает записи и взаимодействует с клиентами для уточнения их предпочтений.
Продавец	В его обязанности входит управление онлайн-магазином клуба, добавление новых товаров, обновление цен и контроль за складскими остатками.

2.1.2 Варианты использования

На рисунке 1 представлена диаграмма вариантов использования.

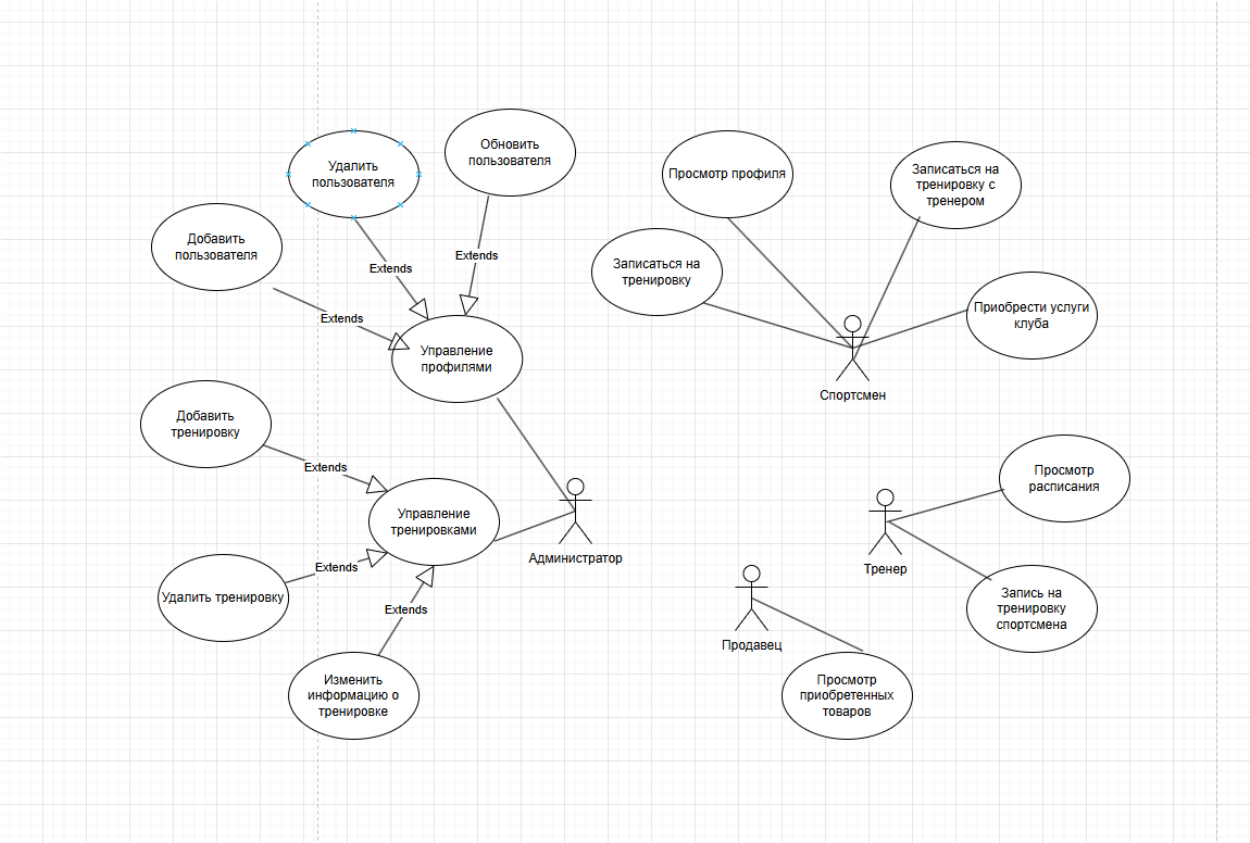

Рисунок 1 – диаграмма вариантов использования

2.1.3 Описание вариантов использования

Запись на тренировку с тренером

Предусловие: Клиент вошёл в систему

Триггер: Клиент нажал кнопку “Тренера”

1. Система отображает список доступных тренеров
2. Клиент выбирает тренера
3. Система предоставляет доступные тренировки по времени
4. Клиент вводит данные и записывается на тренировку
5. Система подтверждает запись и уведомляет клиента

   Расширение:

   4. Выбранное время для записи недоступно

      4а. Выберите другое время

Запись на тренировку

Предусловие: Клиент вошёл в систему

Триггер: Клиент нажал кнопку “Тренировки”

1. Система отображает список доступных тренировок
2. Клиент выбирает интересующую тренировку
3. Система предоставляет доступные слоты для записи на тренировку
4. Клиент заполняет данные и подтверждает запись
5. Система обрабатывает запись и уведомляет клиента об успешной записи

Расширение:

4.Ошибка при регистрации на тренировку

4а.Проверьте введённые данные

Удаление тренировки

Предусловие: Администратор вошёл в систему

Триггер: Администратор нажал кнопку “Тренировки”

1.Администратор нажимает кнопку “Посмотреть тренировки”

2.Система предоставляет список записей на тренировку

3. Администратор выбирает удаление

4.Система уведомляет об успешном удалении

Обновление тренировки

Предусловие: Администратор вошёл в систему

Триггер: Администратор нажал кнопку “Тренировки”

1.Администратор нажимает кнопку “Посмотреть тренировки”

2.Система предоставляет список записей на тренировку

3. Администратор выбирает обновление

4.Система отображает модальные окна для редактирования

5.Администратор вводит данные для обновления

6.Система уведомляет об успешном обновлении

Расширение:

5а.Выбранной категории не существует

5а1.Система предлагает выбор другой категории

5б. Выбранный день тренировки не соответствует категории

5б1.Система предлагает выбрать другой день

Оплата услуг клуба

Предусловие: Клиент вошёл в систему; Клиент выбрал услугу

Триггер: Клиент хочет оплатить услугу

1.Клиент переходит в режим оплаты

2.Система предоставляет доступные способы оплаты

3.Клиент выбирает предпочтительный способ оплаты

4.Система проводит оплату

5.Система предоставляет данные об оплате клиенту

Обновление профиля

Предусловие:Администратор вошёл в систему.Администратор выбрал пользователя.

Триггер: Администратор хочет обновить информацию о пользователе

1.Администратор выбирает спортсмена

2.Администратор нажимает кнопку “Обновить”

3.Администратор вводит новую информацию

4.Система уведомляет об успешном обновлении

2.2 Функциональные требования

2.2.1 Записаться на тренировку

Клиент может записаться на тренировку, выбрав при этом интересующую его тип и вид тренировки

Таблица 2 – Функциональные требования «Записаться на тренировку»

ЗаписатьсяНаТренировку	Клиент переходит в форму записи на тренировку
.УказатьИнформациюОСебе	Клиент указывает своё имя и контактные данные
.ВыбратьВидТренировки	Клиент выбирает нужный вид тренировки
.ВыбратьТипТренировку	Клиент выбирает нужный тип тренировку
.ВыбратьДеньТренировку	Клиент выбирает нужный день тренировки

2.2.2 Записаться на тренировку с тренером

Клиент может записаться на тренировку с тренером, выбрав при этом нужного специалиста, дату и время.

Таблица 3 – Функциональные требования «Записаться на тренировку с тренером»

ЗаписатьсяНаИндивидуальныеЗанятия	Клиент переходит в форму записи на тренировку
.УказатьИнформациюОСебе	Клиент указывает своё имя и контактные данные
.ВыбратьСпециалиста	Клиент выбирает специалиста из списка
.ВыбратьДату	Клиент выбирает дату тренировку
.ВыбратьВремя	Клиент выбирает время

2.2.3 Управление тренировками

Администратор может удалять и обновлять информацию о тренировках

Таблица 4 – Функциональные требования «Управление тренировками»

ПоказатьТренировки	Администратор открывает список тренировок
.УдалитьТренировку	Администратор может удалить тренировку
.ОбновитьТренировку	Администратор может обновить тренировку

2.2.4 Оплата услуг клуба

Клиент может выбрать услугу клуба, а затем перейти к оплате, выбрав предпочтительный способ оплаты.

Таблица 5 – Функциональные требования «Оплата услуг клуба»

ВыбратьУслугуКлуба	Клиент переходит в форму выбора услуги клба
.ПерейтиКОплате	Клиент переходит в форму оплаты услуги
.ВыборСпособОплаты	Клиент выбирает предпочтительный способ оплаты
.ПодтвердитьОплату	Клиент подтверждает оплату услуги
.ЗавершитьОплату	Система уведомляет клиента об успешной транзакции и предоставляет детали оплаты

2.2.5 Управление профилями

Администратор может добавлять, удалять и обновлять информацию о пользователях

Таблица 6 – Функциональные требования «Управление профилями»

ПоказатьПользователей	Администратор открывает список пользователей
.УдалитьПользователя	Администратор может удалить пользователя
.ОбновитьПользователя	Администратор может обновить пользователя
.Добавить пользователя	Администратор может добавить пользователя

3 Концептуальная модель данных

На рисунке 2 представлена концептуальная модель данных.

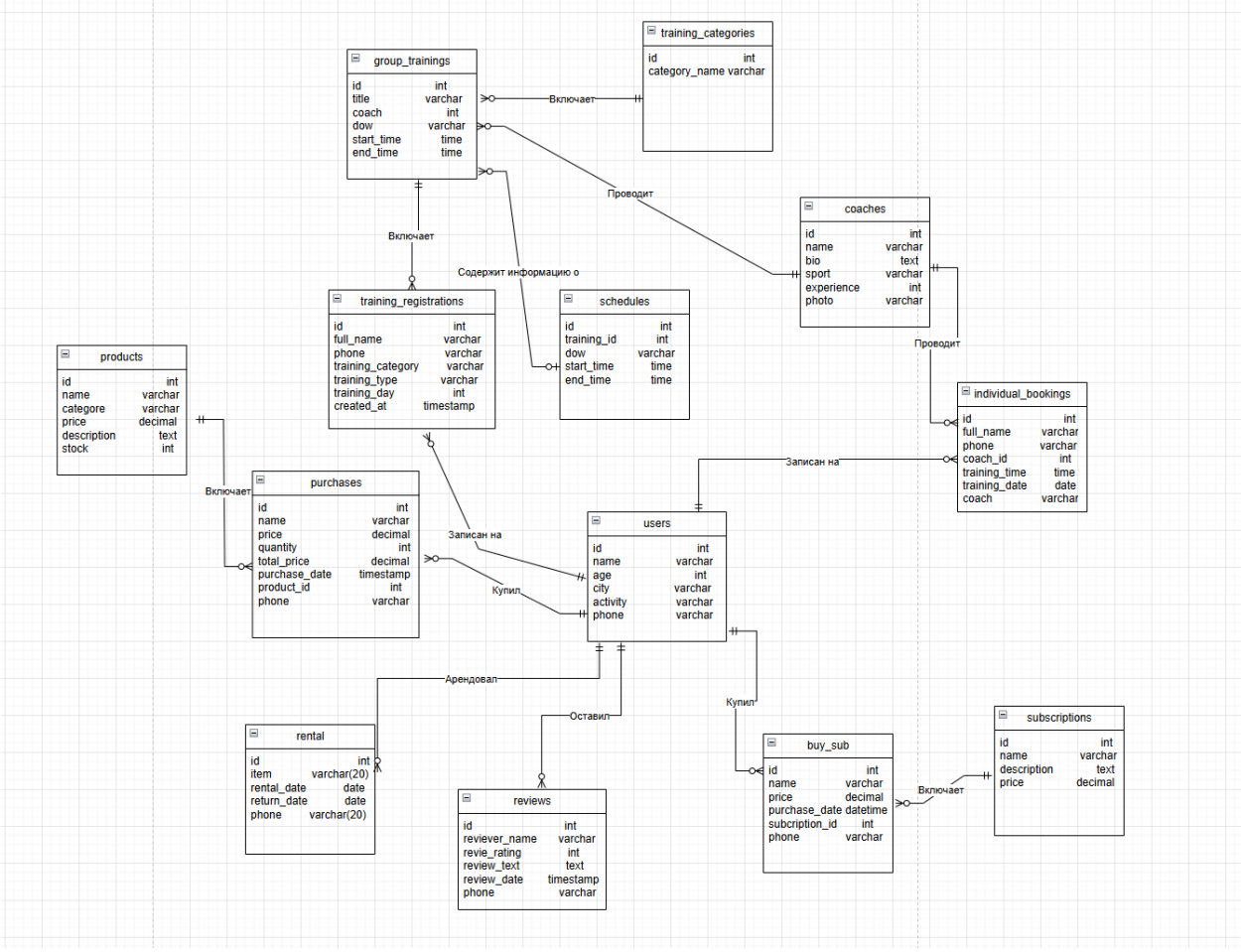

Рисунок 2 – концептуальная модель данных

1. Клиенты: Пользователи с уникальными данными, которые могут записываться на тренировки, арендовать оборудование, приобретать подписки.
2. Тренера: Ключевые фигуры, которые проводят занятия, как групповые, так и индивидуальные.
3. Регистрация на тренировки: процесс записи клиента на тренировки. Каждая запись содержит информацию о времени, типе тренировки.
4. Категории тренировок: содержит информацию о категориях тренировок, которые предлагаются в клубе.

5. Абонементы: предоставляют доступ клиентам клуба к услугам.
6. Покупка абонементов: предоставляет информацию о приобретенных абонементах пользователями.
7. Расписание: включает в себя информацию о доступных тренировках и времени
8. Отзывы: Клиенты могут оставлять отзывы о тренерах, клубе, а также ставить рейтинг.
9. Аренда: дополнительные услуги в зале, также аренда отдельных помещений
10. Покупки: информация о продуктах, приобретенных в магазине клуба
11. Продукты: непосредственно товары, соответствующие спортивной тематике
12. Индивидуальные тренировки: персонализированные занятия, проводимые с тренером, направленные на улучшение физической формы в индивидуальном порядке
13. Групповые занятия: занятия для группы клиентов, что позволяют тренерам работать с несколькими людьми одновременно

4 Физическая модель данных

На рисунке 3 представлена физическая модель базы данных. Код запроса SQL для создания базы данных, триггеров, хранимых процедур, индексов представлен в приложении А.

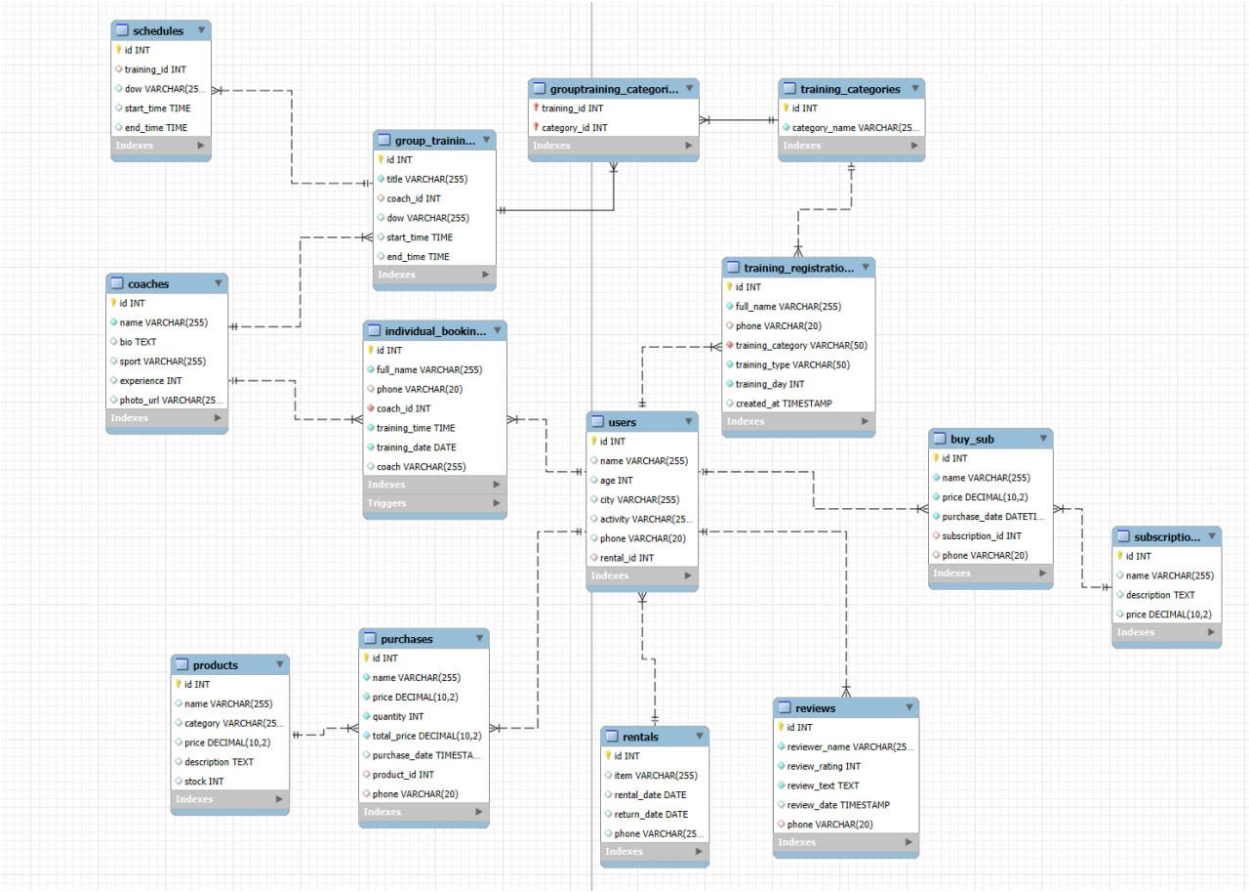

Рисунок 3 – физическая модель данных

4.1 Хранимые процедуры

1. Процедура add_review: эта процедура принимает четыре входных параметра — имя рецензента, рейтинг, текст отзыва и дату отзыва. Она добавляет новый отзыв в таблицу reviews, используя переданные значения. Процедура предназначена для вставки данных о новом отзыве в базу данных.

2.Процедура book_individual_training: процедура использует входные параметры для определения ID тренера и проверяет, доступно ли выбранное время для индивидуальной тренировки. Она выполняет поиск в таблице `individual_bookings` по ID тренера и времени тренировки. Если тренер

уже забронирован на это время, процедура уведомляет пользователя об этом. Это позволяет избежать дублирования бронирований.

3.Процедура register_for_training: эта процедура принимает параметр с названием категории тренировки и проверяет существование этой категории в таблице `Training_Categories`. В случае нахождения категории она регистрирует клиента на тренировку в соответствующую категорию. Если категория не найдена, процедура может выполнить дополнительные действия, например, уведомление о том, что такая категория тренировки не существует. Это позволяет корректно обрабатывать регистрацию клиентов на тренировки в зависимости от доступных категорий.

4.2 Триггеры

1.Триггер prevent_duplicate_booking: это триггер, который срабатывает до вставки записи в таблицу `individual_bookings`. Он предназначен для предотвращения дублирования записей, то есть если попытаться забронировать время с тренером, который уже занят, триггер проверяет это и не позволяет сделать вставку.

4.3 Индексы

1. Индекс idx_coach_name: индекс на поле name в таблице coaches помогает быстро искать тренеров по их имени.

2. Индекс idx_individual_booking: индекс на полях coach_id, training_time и training_date в таблице индивидуальных записей помогает ускорить поиск по этим полям.

5. Интерфейс пользователя

Пользователь входит в систему и перед ним открывается главная страница.

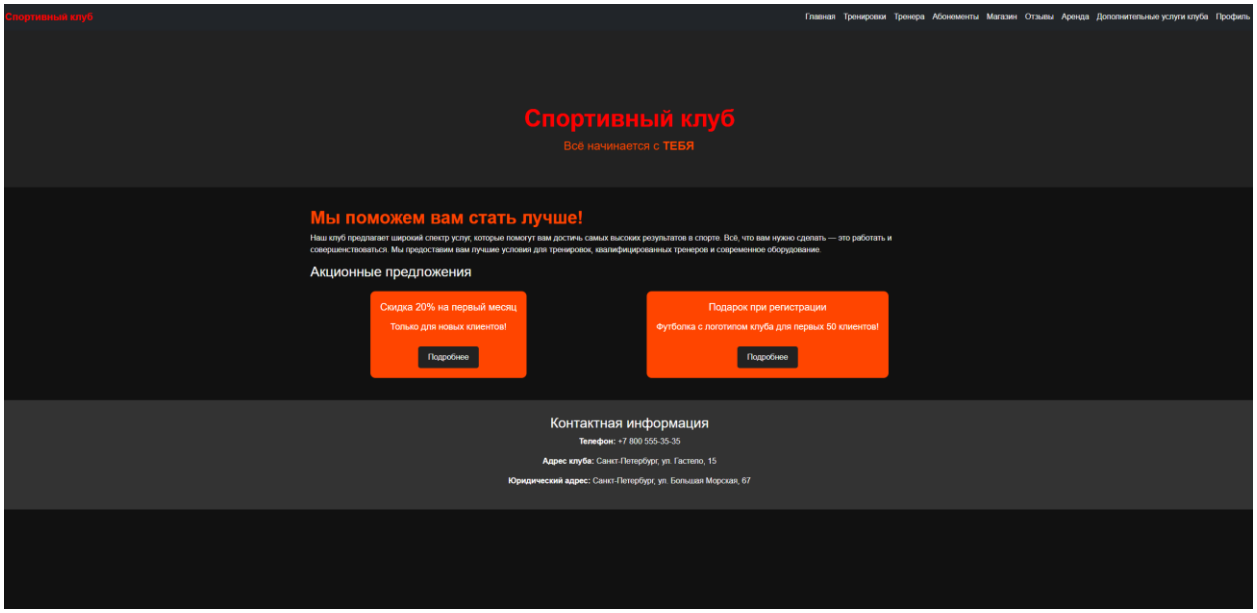

Рисунок 4 – Главная страница

Пользователь нажимает “Тренировки” на навигационной панель и переходит на страницу Тренировок.

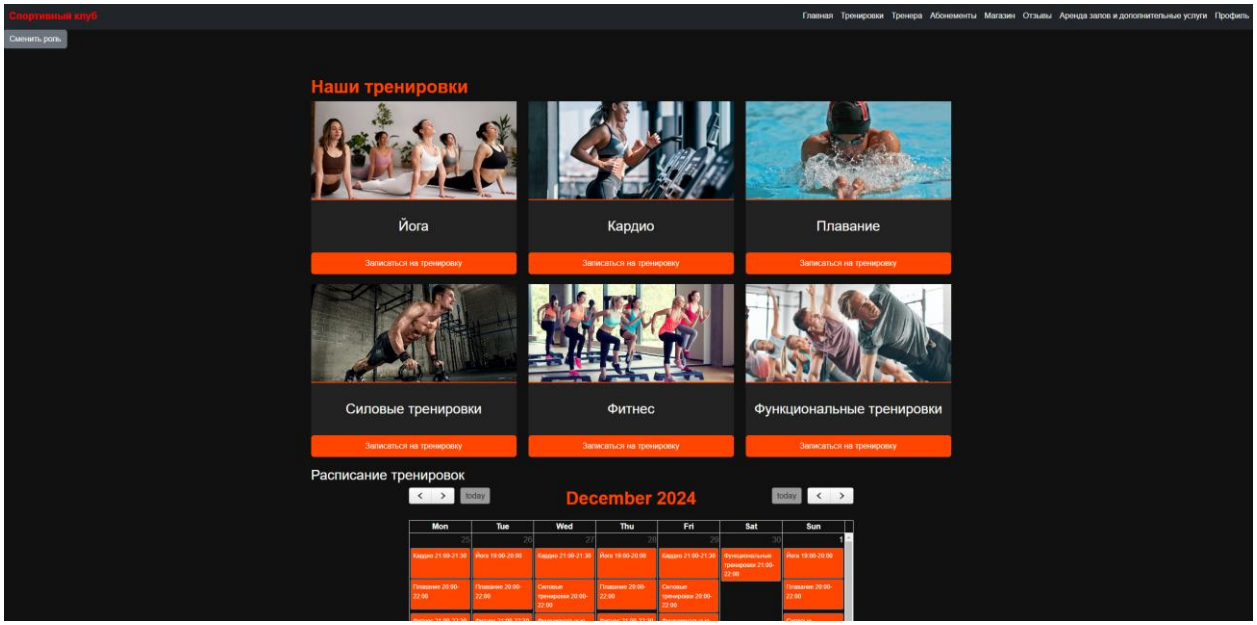

Рисунок 5 –Страница тренировок

Пользователь может выбрать интересующую его вид тренировки и перейти к записи на неё через форму.

Рисунок 6 – Форма для записи на тренировку

Администратор на этой же странице имеет доступ ко всем записям на тренировку и может посмотреть их.

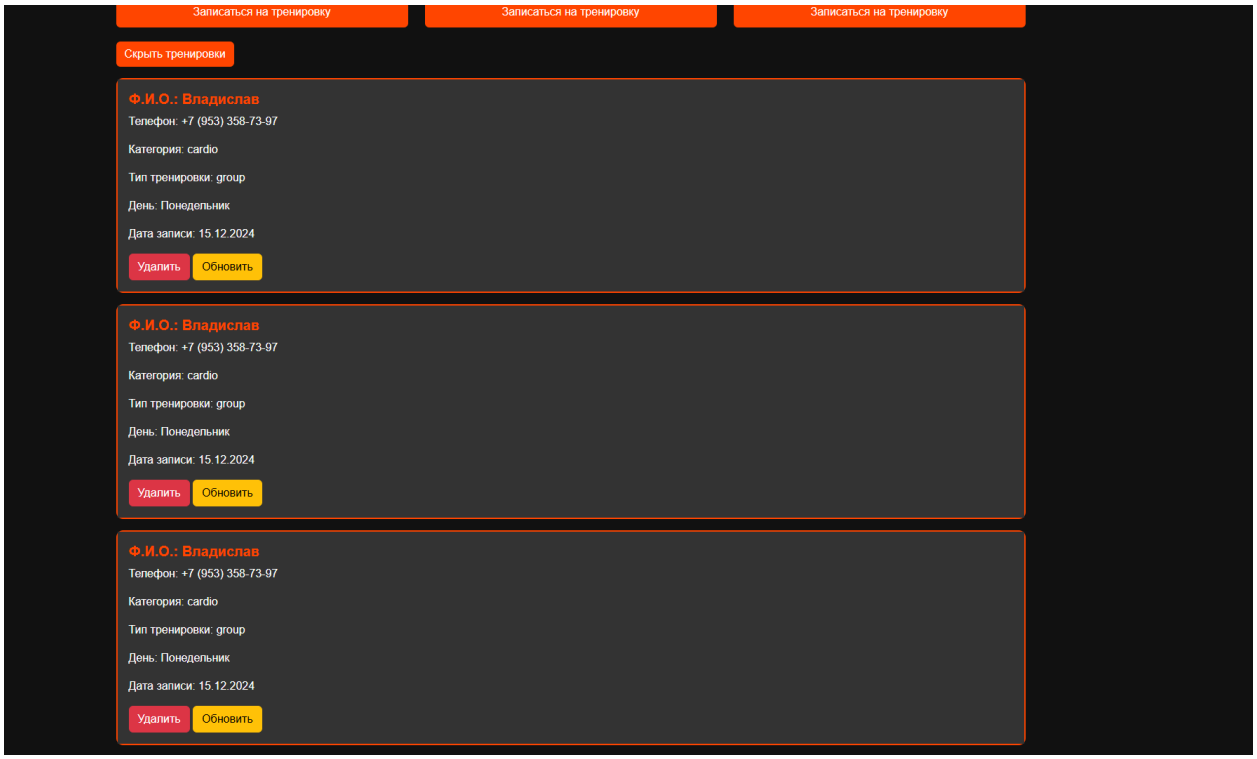

Рисунок 7 – Список с тренировками

Администратор может удалить тренировки.

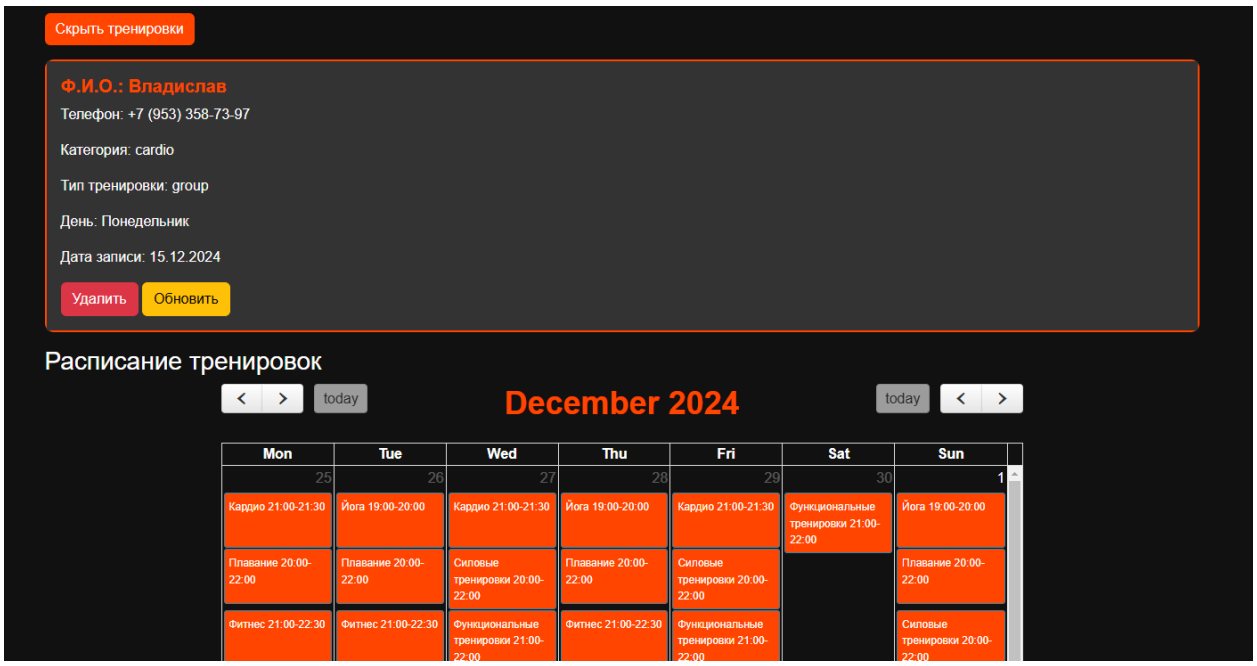

Рисунок 8 – Удаление тренировок

Администратор может обновить тренировку.

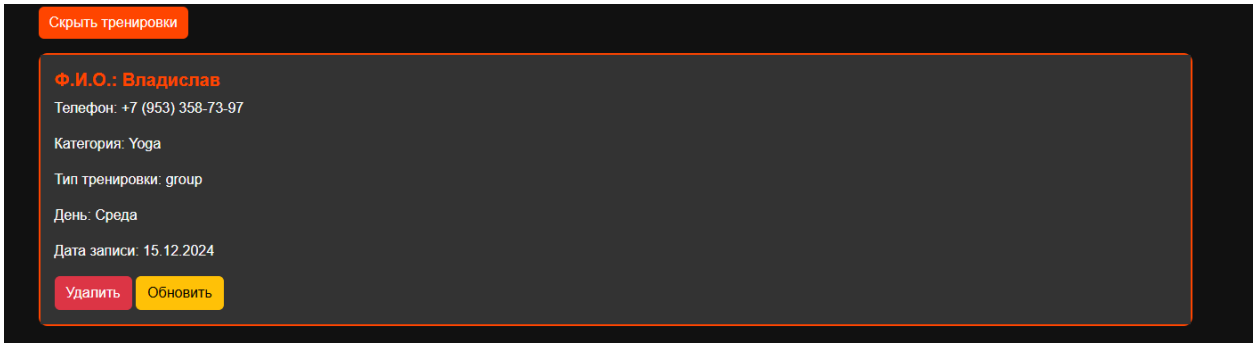

Рисунок 9 – Обновление тренировки

Также администратор может и добавить тренировку, заполнив регистрационную форму за спортсмена.

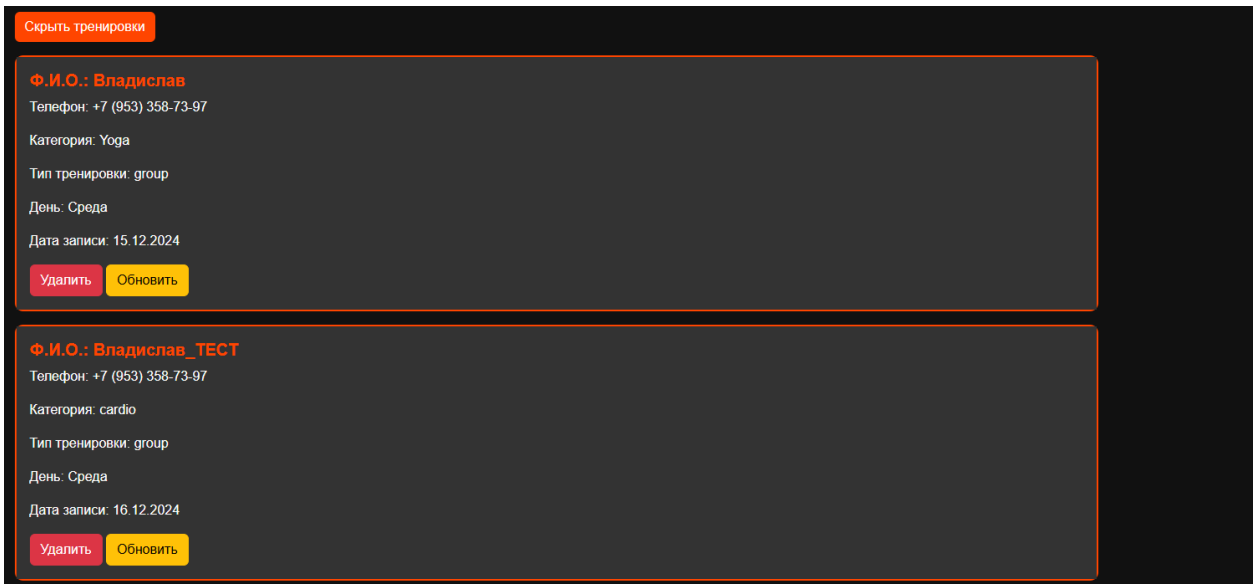

Рисунок 10-Добавление тренировки

Нажав на навигационной панели “Тренера”, пользователь переходит на страницу, содержащую информацию о тренерах.

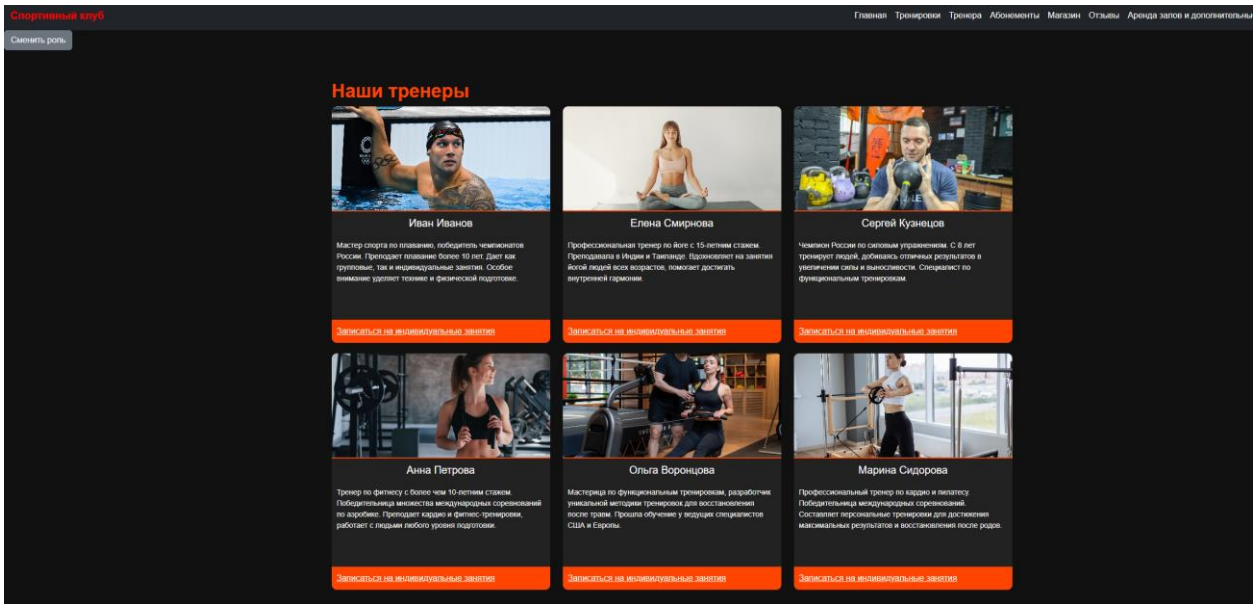

Рисунок 11 – Страница “Тренера”

Тренер может просматривать свои тренировки при нажатии на кнопку “Посмотреть тренировки”

Дата	Время	Тренер
28.12.2024	10:00 AM	ivan ivanov
22.12.2024	10:00 AM	ivan ivanov
27.12.2024	10:00 AM	ivan ivanov
27.12.2024	10:30 AM	ivan ivanov
27.12.2024	2:00 PM	ivan ivanov
27.12.2024	11:00 AM	ivan ivanov

Рисунок 12 – Тренировки тренера

Нажав на навигационной панели “Магазин”, пользователь переходит на страницу, содержащую информацию о спортивных предметах и питании.

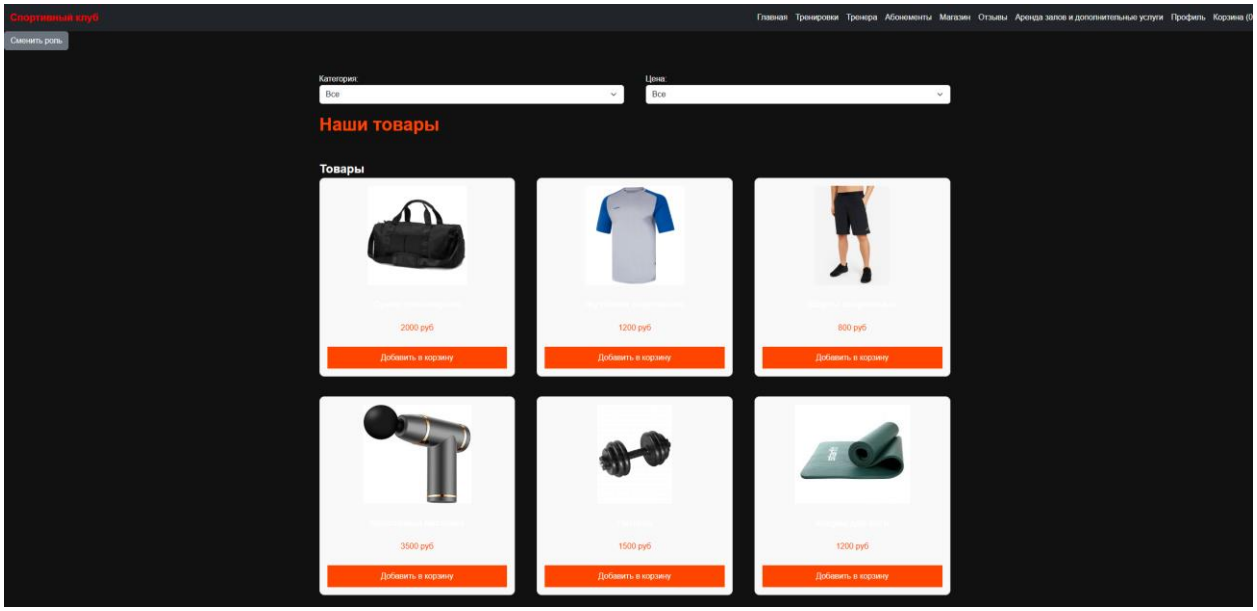

Рисунок 13 – Страница “Магазин”

При нажатии кнопки “Добавить в корзину” предмет добавляется в нее, а также на навигационной панели есть кнопка “Корзина “ для просмотра её содержимого.

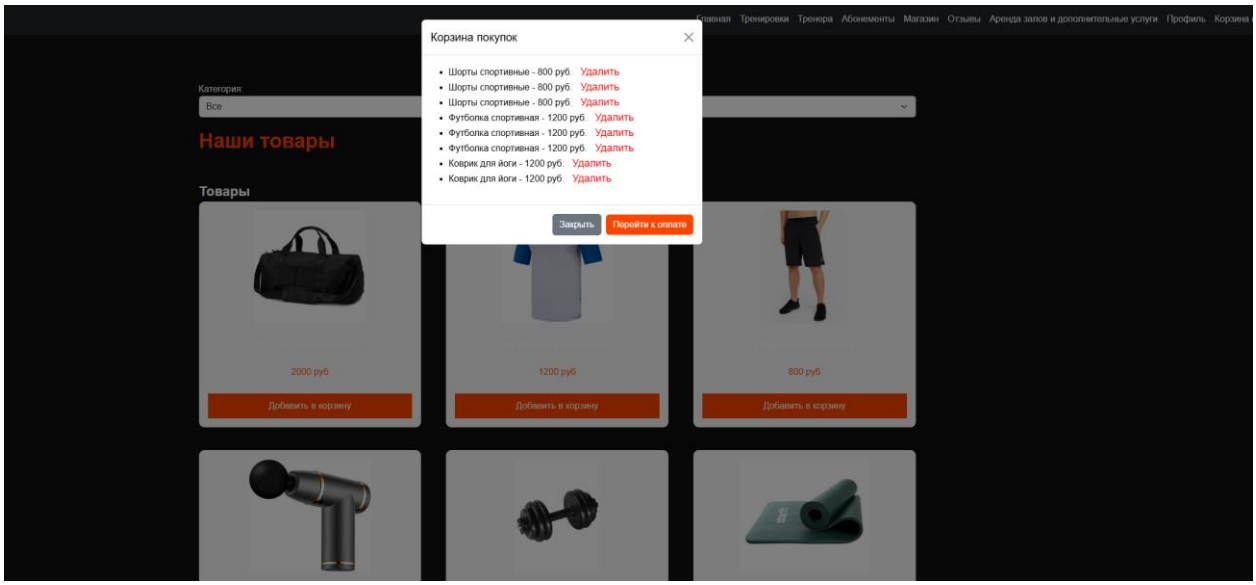

Рисунок 14 – Корзина

При нажатии “Перейти к оплате” перед пользователем появляется модальное окно с информацией. При последующем нажатии кнопки “Оплатить через СБП”. Пользователю выходят сообщения об успешной оплате.

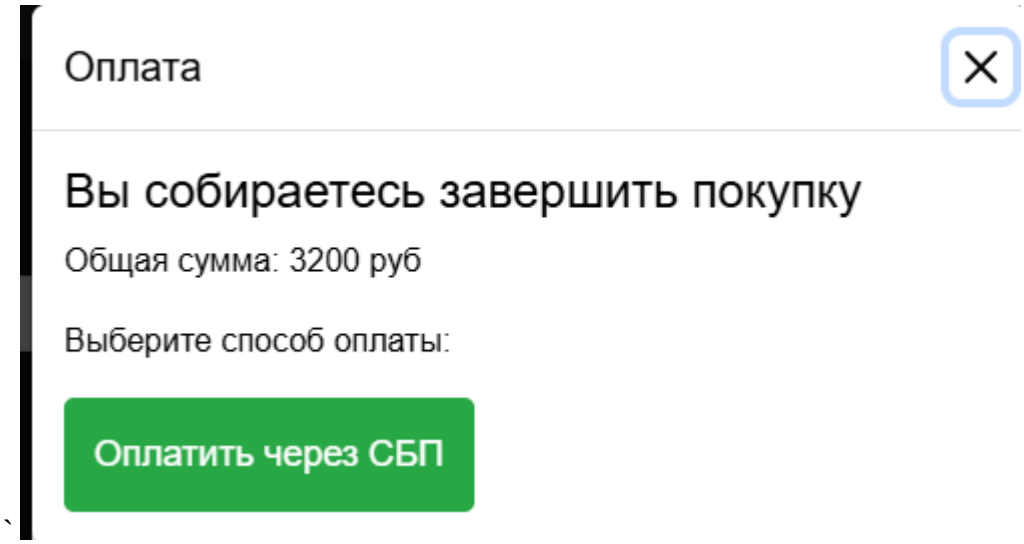

Рисунок 15 – Окно оплаты

Продавец может посмотреть список купленных товаров, нажав на кнопку “Посмотреть купленные товары”

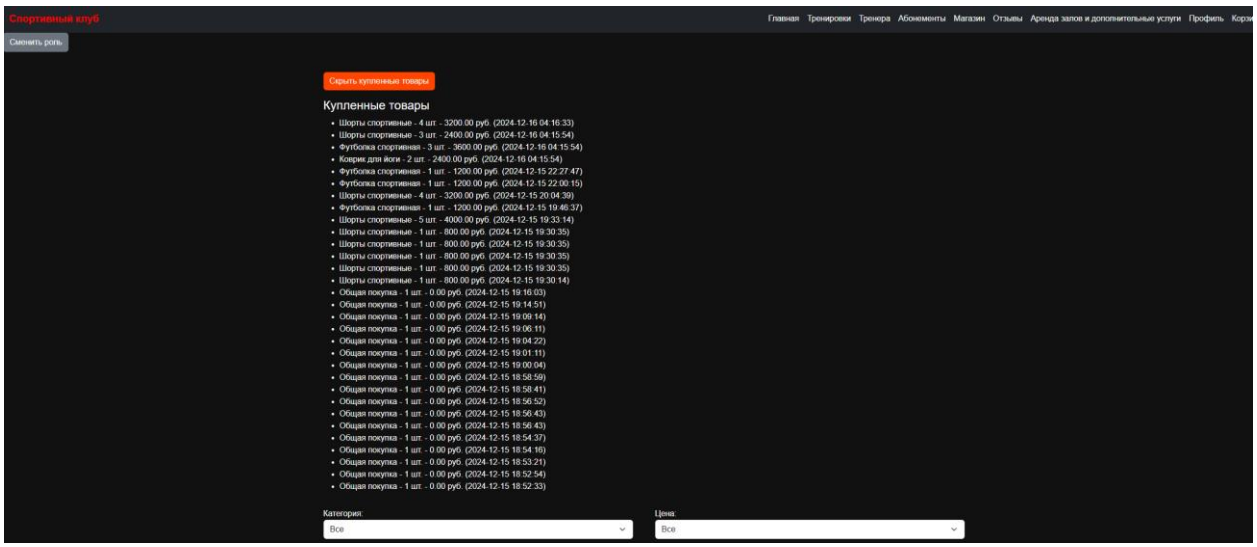

Рисунок 16 – Список купленных товаров

На рисунке 17 изображена страница “Профиль” , на которой администратор может добавить пользователя, обновить информацию о нём или удалить профиль.

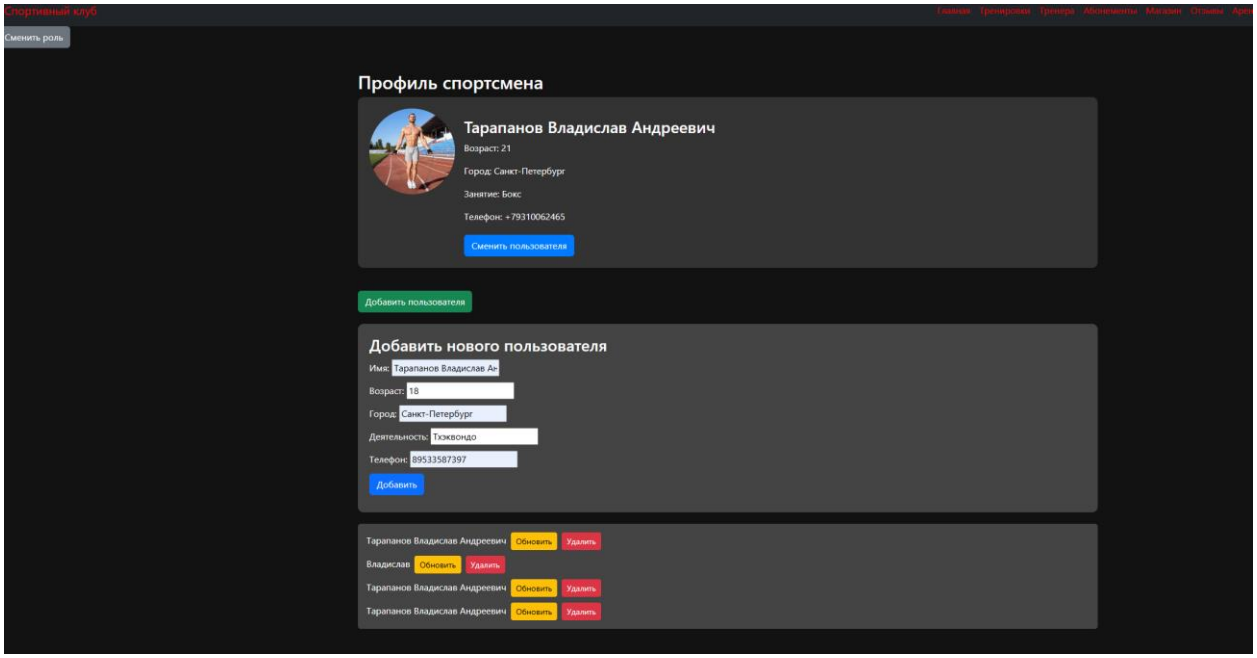

Рисунок 17 – Страница “Профиль”с функционалом администратора

ЗАКЛЮЧЕНИЕ

В ходе выполнения курсовой работы можно отметить, что цель разработки автоматизированной информационной системы спортивного клуба была успешна достигнута. Проект направлен на создание современного и функционального продукта, обеспечивающего удобный доступ к записям на тренировку и другим возможным взаимодействиям с системой. Ключевой особенностью системы является разработка интуитивно понятного интерфейса для пользователей, а также представителей клуба в лице: администратора, тренера, продавца. Перед началом разработки были четко определены правила и требования, а также были выделены роли пользователей с их обязанностями. Выбор технического стека технологий HTML, CSS, JavaScript для создания веб-интерфейса и реализации интерактивных элементов и базы на MySQL, обеспечивает надёжность и эффективность функционирования системы. В результате проведенной работы система готова к внедрению и использованию.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Дакетт, Дж. HTML и CSS: Дизайн и создание веб-сайтов / Дж. Дакетт. — Москва: Вильямс, 2014. — 480 с.
2. Снайдер, М. Node.js в примерах / М. Снайдер. — 2-е изд. — Москва: Вильямс, 2018. — 320 с.
3. Коберн А. Современные методы описания функциональных требований к системам/Пер. с англ. – М.: Издательство "Лори", 2012 – 264 с.: ил.
4. Ковальчук, А. JavaScript. Основы разработки динамических веб-приложений / А. Ковальчук. — Москва: Издательство Диалектика, 2016. — 392 с.
5. Мун, С. Node.js: Разработка масштабируемых серверных приложений / С. Мун. — Москва: Вильямс, 2016. — 512 с.

ПРИЛОЖЕНИЕ А

Скрипты для создания объектов БД

```
CREATE TABLE `buy_sub` (
  `id` int NOT NULL AUTO_INCREMENT,
  `name` varchar(255) NOT NULL,
  `price` decimal(10,2) NOT NULL,
  `purchase_date` datetime NOT NULL,
  `subscription_id` int DEFAULT NULL,
  `phone` varchar(20) DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `fk_subscription` (`subscription_id`),
  KEY `fk_phone_buy_sub_new` (`phone`),
  CONSTRAINT `fk_phone_buy_sub_new` FOREIGN KEY (`phone`) REFERENCES `users` (`phone`) ON DELETE SET NULL ON UPDATE CASCADE,
  CONSTRAINT `fk_subscription` FOREIGN KEY (`subscription_id`) REFERENCES `subscriptions` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=14 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE `coaches` (
  `id` int NOT NULL AUTO_INCREMENT,
  `name` varchar(255) NOT NULL,
  `bio` text,
  `sport` varchar(255) DEFAULT NULL,
  `experience` int DEFAULT NULL,
  `photo_url` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `idx_coach_name` (`name`)
) ENGINE=InnoDB AUTO_INCREMENT=7 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE `group_trainings` (
  `id` int NOT NULL AUTO_INCREMENT,
  `title` varchar(255) NOT NULL,
  `coach_id` int DEFAULT NULL,
  `dow` varchar(255) DEFAULT NULL,
  `start_time` time DEFAULT NULL,
```

```
  `end_time` time DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `coach_id` (`coach_id`),
  CONSTRAINT `group_trainings_ibfk_1` FOREIGN KEY (`coach_id`) REFERENCES `coaches` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=7 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE `grouptraining_categories` (
  `training_id` int NOT NULL,
  `category_id` int NOT NULL,
  PRIMARY KEY (`training_id`,`category_id`),
  KEY `category_id` (`category_id`),
  CONSTRAINT `grouptraining_categories_ibfk_1` FOREIGN KEY (`training_id`) REFERENCES
`group_trainings` (`id`),
  CONSTRAINT `grouptraining_categories_ibfk_2` FOREIGN KEY (`category_id`) REFERENCES
`training_categories` (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE `individual_bookings` (
  `id` int NOT NULL AUTO_INCREMENT,
  `full_name` varchar(255) NOT NULL,
  `phone` varchar(20) DEFAULT NULL,
  `coach_id` int NOT NULL,
  `training_time` time NOT NULL,
  `training_date` date NOT NULL,
  `coach` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `idx_individual_booking` (`coach_id`,`training_date`,`training_time`),
  KEY `fk_individual_bookings_users` (`phone`),
  CONSTRAINT `fk_individual_bookings_users` FOREIGN KEY (`phone`) REFERENCES `users` (`phone`) ON
DELETE SET NULL ON UPDATE CASCADE,
  CONSTRAINT `individual_bookings_ibfk_1` FOREIGN KEY (`coach_id`) REFERENCES `coaches` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=21 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
DELIMITER ;;
/*!50003 CREATE*/ /*!50017 DEFINER=`root`@`localhost`*/ /*!50003 TRIGGER `prevent_duplicate_booking`
BEFORE INSERT ON `individual_bookings` FOR EACH ROW BEGIN
    -- Проверяем, есть ли уже запись на это время с этим тренером
    DECLARE duplicate_count INT;
```

```
    SELECT COUNT(*) INTO duplicate_count
    FROM individual_bookings
    WHERE coach_id = NEW.coach_id
    AND training_date = NEW.training_date
    AND training_time = NEW.training_time;

    IF duplicate_count > 0 THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Это время уже занято. Пожалуйста, выберите другое время.';
    END IF;
END */;;

CREATE TABLE `products` (
  `id` int NOT NULL AUTO_INCREMENT,
  `name` varchar(20) DEFAULT NULL,
  `category` varchar(20) DEFAULT NULL,
  `price` decimal(10,2) DEFAULT NULL,
  `description` text,
  `stock` int DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=10 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE `purchases` (
  `id` int NOT NULL AUTO_INCREMENT,
  `name` varchar(20) NOT NULL,
  `price` decimal(10,2) NOT NULL,
  `quantity` int NOT NULL,
  `total_price` decimal(10,2) NOT NULL,
  `purchase_date` timestamp NULL DEFAULT CURRENT_TIMESTAMP,
  `product_id` int DEFAULT NULL,
  `phone` varchar(20) DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `fk_product` (`product_id`),
  KEY `fk_phone_purchases` (`phone`),
  CONSTRAINT `fk_phone_purchases` FOREIGN KEY (`phone`) REFERENCES `users` (`phone`),
```

```
  CONSTRAINT `fk_product` FOREIGN KEY (`product_id`) REFERENCES `products` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=32 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE `rentals` (
  `id` int NOT NULL AUTO_INCREMENT,
  `item` varchar(255) DEFAULT NULL,
  `rental_date` date DEFAULT NULL,
  `return_date` date DEFAULT NULL,
  `phone` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `fk_phone_rentals` (`phone`)
) ENGINE=InnoDB AUTO_INCREMENT=32 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE `reviews` (
  `id` int NOT NULL AUTO_INCREMENT,
  `reviewer_name` varchar(255) NOT NULL,
  `review_rating` int NOT NULL,
  `review_text` text NOT NULL,
  `review_date` timestamp NULL DEFAULT CURRENT_TIMESTAMP,
  `phone` varchar(20) DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `fk_phone_reviews` (`phone`),
  CONSTRAINT `fk_phone_reviews` FOREIGN KEY (`phone`) REFERENCES `users` (`phone`),
  CONSTRAINT `reviews_chk_1` CHECK ((`review_rating` between 1 and 5))
) ENGINE=InnoDB AUTO_INCREMENT=32 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE `schedules` (
  `id` int NOT NULL AUTO_INCREMENT,
  `training_id` int DEFAULT NULL,
  `dow` varchar(255) DEFAULT NULL,
  `start_time` time DEFAULT NULL,
  `end_time` time DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `training_id` (`training_id`),
  CONSTRAINT `schedules_ibfk_1` FOREIGN KEY (`training_id`) REFERENCES `group_trainings` (`id`)
```

```
) ENGINE=InnoDB AUTO_INCREMENT=24 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE `subscriptions` (
  `id` int NOT NULL AUTO_INCREMENT,
  `name` varchar(255) DEFAULT NULL,
  `description` text,
  `price` decimal(10,2) DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE `training_categories` (
  `id` int NOT NULL AUTO_INCREMENT,
  `category_name` varchar(255) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `category_name` (`category_name`)
) ENGINE=InnoDB AUTO_INCREMENT=8 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE `training_registrations` (
  `id` int NOT NULL AUTO_INCREMENT,
  `full_name` varchar(255) NOT NULL,
  `phone` varchar(20) DEFAULT NULL,
  `training_category` varchar(50) NOT NULL,
  `training_type` varchar(50) NOT NULL,
  `training_day` int NOT NULL,
  `created_at` timestamp NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  KEY `training_category` (`training_category`),
  KEY `fk_training_registrations_users` (`phone`),
  CONSTRAINT `fk_training_registrations_users` FOREIGN KEY (`phone`) REFERENCES `users` (`phone`) ON DELETE SET NULL ON UPDATE CASCADE,
  CONSTRAINT `training_registrations_ibfk_1` FOREIGN KEY (`training_category`) REFERENCES `training_categories` (`category_name`)
) ENGINE=InnoDB AUTO_INCREMENT=33 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE `users` (
  `id` int NOT NULL AUTO_INCREMENT,
```

```
  `name` varchar(255) DEFAULT NULL,
  `age` int DEFAULT NULL,
  `city` varchar(255) DEFAULT NULL,
  `activity` varchar(255) DEFAULT NULL,
  `phone` varchar(20) DEFAULT NULL,
  `rental_id` int DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `idx_phone_users` (`phone`),
  KEY `fk_rental_user` (`rental_id`),
  CONSTRAINT `fk_rental_user` FOREIGN KEY (`rental_id`) REFERENCES `rentals` (`id`) ON DELETE SET NULL ON UPDATE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=24 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

DELIMITER ;;
CREATE DEFINER=`root`@`localhost` PROCEDURE `add_review`(
    IN reviewerName VARCHAR(255),
    IN reviewRating INT,
    IN reviewText TEXT,
    IN reviewDate DATETIME
)
BEGIN
    INSERT INTO reviews (reviewer_name, review_rating, review_text, review_date)
    VALUES (reviewerName, reviewRating, reviewText, reviewDate);
END ;;

CREATE DEFINER=`root`@`localhost` PROCEDURE `book_individual_training`(
    IN fullName VARCHAR(255),
    IN phone VARCHAR(20),
    IN coach VARCHAR(255),
    IN trainingDate DATE,
    IN trainingTime TIME
)
BEGIN
    DECLARE coachId INT;
    -- Получаем ID тренера
    SELECT id INTO coachId FROM Coaches WHERE name COLLATE utf8mb4_unicode_ci = coach;
```

```
    -- Проверяем доступность времени
    IF EXISTS (
        SELECT 1 FROM individual_bookings
        WHERE coach_id = coachId AND training_date = trainingDate AND training_time = trainingTime
    ) THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Это время уже занято. Пожалуйста, выберите другое время.';
    ELSE
        -- Добавляем запись на занятие
        INSERT INTO individual_bookings (full_name, phone, coach_id, training_time, training_date, coach)
        VALUES (fullName, phone, coachId, trainingTime, trainingDate, coach);
    END IF;
END ;;

CREATE DEFINER=`root`@`localhost` PROCEDURE `register_for_training`(
    IN fullName VARCHAR(255),
    IN phone VARCHAR(20),
    IN trainingCategory VARCHAR(255),
    IN trainingType VARCHAR(255),
    IN trainingDay VARCHAR(20)
)
BEGIN
    DECLARE categoryId INT;
    -- Проверка существования категории
    SELECT id INTO categoryId FROM Training_Categories WHERE category_name = trainingCategory;

    -- Если категория не существует, добавляем её
    IF categoryId IS NULL THEN
        INSERT INTO Training_Categories (category_name) VALUES (trainingCategory);
        SELECT LAST_INSERT_ID() INTO categoryId;
    END IF;
    -- Добавляем запись на тренировку
    INSERT INTO training_registrations (full_name, phone, training_category, training_type, training_day)
    VALUES (fullName, phone, trainingCategory, trainingType, trainingDay);
END ;;
```

ПРИЛОЖЕНИЕ Б

Код программы

Additional_services.html

```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Дополнительные услуги</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/css/bootstrap.min.css" rel="stylesheet">
    <link href="css/styles.css" rel="stylesheet">
</head>
<body>

    <!-- Навигация -->
    <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
        <div class="container-fluid">
            <a class="navbar-brand" href="index.html">Спортивный клуб</a>
            <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Переключить
навигацию">
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbarNav">
                <ul class="navbar-nav ms-auto">
                    <li class="nav-item">
                        <a class="nav-link" href="index.html">Главная</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="trainings.html">Тренировки</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="coaches.html">Тренера</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="membership.html">Абонементы</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="shop.html">Магазин</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="reviews.html">Отзывы</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="rental.html">Аренда залов и дополнительные
услуги</a>
```

```
                </li>

                <li class="nav-item"><a class="nav-link" href="profile.html">Профиль</a>
                </li>

            </ul>
        </div>
    </div>
</nav>

<div class="container">
    <h2>Дополнительные услуги</h2>

    <div class="row">
        <div class="col-md-4">
            <div class="card">
                <img src="img\fitnes-bar.jpg" class="card-img-top" alt="Фитнес-бар">
                <div class="card-body">
                    <h5 class="card-title">Фитнес-бар</h5>
                    <p class="card-text">После тренировки можно расслабиться и насладиться
вкусными напитками: кофе, протеиновые коктейли, и другие напитки для восстановления сил.</p>
                </div>
            </div>
        </div>

        <div class="col-md-4">
            <div class="card">
                <img src="img\sauna.jpg" class="card-img-top" alt="Сауна">
                <div class="card-body">
                    <h5 class="card-title">Сауна</h5>
                    <p class="card-text">Для полного расслабления и восстановления
предлагается посещение сауны после тренировки. Отлично для улучшения кровообращения и снятия
напряжения.</p>
                </div>
            </div>
        </div>

        <div class="col-md-4">
            <div class="card">
                <img src="img\spa.jpg" class="card-img-top" alt="SPA">
                <div class="card-body">
                    <h5 class="card-title">SPA</h5>
                    <p class="card-text">Для тех, кто хочет получить максимум удовольствия
и пользы от тренировок, предлагается расслабляющие SPA-процедуры, чтобы восстановить силы и
улучшить самочувствие.</p>
                </div>
            </div>
        </div>
    </div>
```

```
    </div>

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/js/bootstrap.bundle.min.js"></script>
</body>
</html>
```

Coaches.html

```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Тренера</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/css/bootstrap.min.css" rel="stylesheet">
    <link href="css/styles.css" rel="stylesheet">
    <style>
        .coach-card {
            background-color: #222;
            color: white;
            border-radius: 10px;
            margin-bottom: 20px;
            display: flex;
            flex-direction: column;
            justify-content: space-between;
            height: 450px;
            overflow: hidden;
        }

        .coach-card img {
            width: 100%;
            height: 200px
            object-fit: cover
            border-bottom: 2px solid #FF4500;
        }

        .coach-card h3 {
            margin: 10px;
            text-align: center;
            font-size: 20px;
        }

        .coach-card p {
            padding: 10px;
            font-size: 14px;
            height: 120px;
            overflow: hidden;
            flex-grow: 1;
```

```
            text-overflow: ellipsis;
        }

        .btn-book {
            background-color: #FF4500;
            color: white;
            border: none;
            width: 100%;
            padding: 10px;
            cursor: pointer;
            font-size: 16px;
        }

        .btn-book:hover {
            background-color: #FF0000;
        }

        .form-container {
            background-color: #f8f9fa;
            padding: 20px;
            border-radius: 10px;
            box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
            margin-top: 40px;
        }

        .form-group {
            margin-top: 20px;
        }

        .btn-register {
            background-color: #FF4500;
            color: white;
            width: 100%;
            padding: 10px;
            border: none;
            cursor: pointer;
            margin-top: 20px;
        }

        .btn-register:hover {
            background-color: #FF0000;
        }

        .container {
            margin-top: 40px;
        }
        .form-container h4 {
            color: black;
        }
    </style>
```

```
</head>
<body>

    <!-- Навигация -->
    <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
        <div class="container-fluid">
            <a class="navbar-brand" href="index.html">Спортивный клуб</a>
            <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Переключить
навигацию">
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbarNav">
                <ul class="navbar-nav ms-auto">
                    <li class="nav-item">
                        <a class="nav-link" href="index.html">Главная</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="trainings.html">Тренировки</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="coaches.html">Тренера</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="membership.html">Абонементы</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="shop.html">Магазин</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="reviews.html">Отзывы</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="rental.html">Аренда залов и дополнительные
услуги</a>
                    </li>

                    <li class="nav-item">
                        <a class="nav-link" href="profile.html">Профиль</a>
                    </li>

                </ul>
            </div>
        </div>
    </nav>
    <button id="roleToggleBtn" class="btn btn-secondary mb-3">Сменить роль</button>
    <div class="container">
        <h2>Наши тренеры</h2>

        <div class="row">
```

```
            <!-- Карточка тренера 1 -->
            <div class="col-md-4">
                <div class="coach-card">
                    <img src="img/trener_plovets.jpg" alt="Тренер 1">
                    <h3>Иван Иванов</h3>
                    <p>Мастер спорта по плаванию, победитель чемпионатов России. Преподает плавание более 10 лет. Дает как групповые, так и индивидуальные занятия. Особое внимание уделяет технике и физической подготовке.</p>
                    <a href="#booking-form" class="btn-book">Записаться на индивидуальные занятия</a>
                    <a href="#" class="showTrainingsBtn btn-book" data-bs-toggle="modal" data-bs-target="#trainingModal" data-coach="ivan ivanov" style="display:none;">Посмотреть тренировки</a>
                </div>
            </div>
            <!-- Карточка тренера 2 -->
            <div class="col-md-4">
                <div class="coach-card">
                    <img src="img/trener_yoga.jpg" alt="Тренер 2">
                    <h3>Елена Смирнова</h3>
                    <p>Профессиональная тренер по йоге с 15-летним стажем. Преподавала в Индии и Таиланде. Вдохновляет на занятия йогой людей всех возрастов, помогает достигать внутренней гармонии.</p>
                    <a href="#booking-form" class="btn-book">Записаться на индивидуальные занятия</a>
                    <a href="#" class="showTrainingsBtn btn-book" data-bs-toggle="modal" data-bs-target="#trainingModal" data-coach="elena smirnova" style="display:none;">Посмотреть тренировки</a>
                </div>
            </div>
            <!-- Карточка тренера 3 -->
            <div class="col-md-4">
                <div class="coach-card">
                    <img src="img/trener_silovye.jpg" alt="Тренер 3">
                    <h3>Сергей Кузнецов</h3>
                    <p>Чемпион России по силовым упражнениям. С 8 лет тренирует людей, добиваясь отличных результатов в увеличении силы и выносливости. Специалист по функциональным тренировкам.</p>
                    <a href="#booking-form" class="btn-book">Записаться на индивидуальные занятия</a>
                    <a href="#" class="showTrainingsBtn btn-book" data-bs-toggle="modal" data-bs-target="#trainingModal" data-coach="sergey kuznetsov" style="display:none;">Посмотреть тренировки</a>
                </div>
            </div>
            <!-- Карточка тренера 4 -->
            <div class="col-md-4">
                <div class="coach-card">
                    <img src="img/trener_fitnes.jpg" alt="Тренер 4">
                    <h3>Анна Петрова</h3>
```

```
                    <p>Тренер по фитнесу с более чем 10-летним стажем. Победительница множества международных соревнований по аэробике. Преподает кардио и фитнес-тренировки, работает с людьми любого уровня подготовки.</p>
                    <a href="#booking-form" class="btn-book">Записаться на индивидуальные занятия</a>
                    <a href="#" class="showTrainingsBtn btn-book" data-bs-toggle="modal" data-bs-target="#trainingModal" data-coach="anna petrova" style="display:none;">Посмотреть тренировки</a>
                </div>
            </div>
            <!-- Карточка тренера 5 -->
            <div class="col-md-4">
                <div class="coach-card">
                    <img src="img/trener_funktsional.jpg" alt="Тренер 5">
                    <h3>Ольга Воронцова</h3>
                    <p>Мастерица по функциональным тренировкам, разработчик уникальной методики тренировок для восстановления после травм. Прошла обучение у ведущих специалистов США и Европы.</p>
                    <a href="#booking-form" class="btn-book">Записаться на индивидуальные занятия</a>
                    <a href="#" class="showTrainingsBtn btn-book" data-bs-toggle="modal" data-bs-target="#trainingModal" data-coach="olga vorontsova" style="display:none;">Посмотреть тренировки</a>
                </div>
            </div>
            <!-- Карточка тренера 6 -->
            <div class="col-md-4">
                <div class="coach-card">
                    <img src="img/trener_kardio.jpeg" alt="Тренер 6">
                    <h3>Марина Сидорова</h3>
                    <p>Профессиональный тренер по кардио и пилатесу. Победительница международных соревнований. Составляет персональные тренировки для достижения максимальных результатов и восстановления после родов.</p>
                    <a href="#booking-form" class="btn-book">Записаться на индивидуальные занятия</a>
                    <a href="#" class="showTrainingsBtn btn-book" data-bs-toggle="modal" data-bs-target="#trainingModal" data-coach="marina sidorova" style="display:none;">Посмотреть тренировки</a>
                </div>
            </div>
        </div>

<!-- Модальное окно для тренировки -->
<div class="modal fade" id="trainingModal" tabindex="-1" aria-labelledby="trainingModalLabel" aria-hidden="true">
    <div class="modal-dialog">
        <div class="modal-content">
            <div class="modal-header">
                <h5 class="modal-title" id="trainingModalLabel">Расписание тренировок</h5>
```

```
                <button type="button" class="btn-close" data-bs-dismiss="modal" aria-
label="Закрыть"></button>
            </div>
            <div class="modal-body">
                <table class="table table-striped">
                    <thead>
                        <tr>
                            <th>Дата</th>
                            <th>Время</th>
                            <th>Тренер</th>
                        </tr>
                    </thead>
                    <tbody id="trainingList">

                    </tbody>
                </table>
            </div>
        </div>
    </div>
</div>

        <!-- Форма для записи на индивидуальные занятия -->
        <div class="form-container" id="booking-form">
            <h4>Записаться на индивидуальное занятие</h4>
            <form action="/submit_booking" method="POST">
                <div class="form-group">
                    <label for="fullName">Ф.И.О.</label>
                    <input type="text" class="form-control" id="fullName" name="fullName"
placeholder="Введите ваше имя" required>
                </div>

                <div class="form-group">
                    <label for="phone">Номер телефона</label>
                    <input type="tel" class="form-control" id="phone" name="phone"
placeholder="Введите ваш номер телефона" required>
                </div>

                <div class="form-group">
                    <label for="coach">Выберите тренера</label>
                    <select class="form-control" id="coach" name="coach" required>
                        <option value="ivan ivanov">Иван Иванов</option>
                        <option value="elena smirnova">Елена Смирнова</option>
                        <option value="sergey kuznetsov">Сергей Кузнецов</option>
                        <option value="anna petrova">Анна Петрова</option>
                        <option value="olga vorontsova ">Ольга Воронцова</option>
                        <option value="marina sidorova">Марина Сидорова</option>
                    </select>
                </div>

                <div class="form-group">
```

```
                    <label for="date">Дата занятия</label>
                    <input type="date" class="form-control" id="date" name="date" required>
                </div>

                <div class="form-group">
                    <label for="time">Время занятия</label>
                    <select class="form-control" id="time" name="time" required></select>
                </div>

                <button type="submit" class="btn-register">Записаться</button>
            </form>
        </div>
    </div>

    <script>
        // Заполнение времени с интервалом 30 минут с 10:00 до 21:30
        const timeSelect = document.getElementById('time');
        const startTime = 10; // Начало времени (10:00)
        const endTime = 21; // Конец времени (21:30)


        function formatTime(hour, minute) {
            const h = hour < 10 ? '0' + hour : hour;
            const m = minute < 10 ? '0' + minute : minute;
            return `${h}:${m}`;
        }

        // Генерация времени с интервалом 30 минут
        for (let hour = startTime; hour <= endTime; hour++) {
            timeSelect.innerHTML += `<option value="${formatTime(hour, 0)}">${formatTime(hour,
0)}</option>`;
            timeSelect.innerHTML += `<option value="${formatTime(hour,
30)}">${formatTime(hour, 30)}</option>`;
        }

        // Обработка отправки формы
// Обработчик отправки формы
document.getElementById('booking-form').addEventListener('submit', function (e) {
    e.preventDefault(); // Останавливаем стандартное поведение формы (перезагрузку)

    const form = e.target;
    const formData = new FormData(form);

    const data = {
        fullName: formData.get('fullName'),
        phone: formData.get('phone'),
        coach: formData.get('coach'),
        date: formData.get('date'),
        time: formData.get('time')
    };
```

```
    fetch('http://localhost:3000/submit_booking', {
        method: 'POST',
        headers: {
            'Content-Type': 'application/json'
        },
        body: JSON.stringify(data)
    })
    .then(response => response.text())
    .then(data => {
        alert(data);
    .catch(error => {
        console.error('Ошибка:', error);
        alert('Произошла ошибка при записи на занятие');
    });
});

// Функция для форматирования времени в AM/PM
function formatTimeAMPM(time) {
    const [hours, minutes] = time.split(':');
    const hour = parseInt(hours);
    const period = hour >= 12 ? 'PM' : 'AM';
    const formattedHour = hour > 12 ? hour - 12 : hour;
    const formattedMinutes = minutes ? (minutes.length === 1 ? `0${minutes}` : minutes) :
'00';
    return `${formattedHour}:${formattedMinutes} ${period}`;
}

// Функция для обновления списка тренировок из базы данных
function updateTrainingList(coach) {
    const trainingList = document.getElementById('trainingList');
    trainingList.innerHTML = ''; // Очистить список

    // Отправляем запрос на сервер для получения тренировок
    fetch(`/trainings/${coach}`)
        .then(response => response.json())
        .then(trainings => {
            if (trainings.length === 0) {
                const row = document.createElement('tr');
                row.innerHTML = `<td colspan="3">Тренировки не найдены</td>`;
                trainingList.appendChild(row);
            } else {
                trainings.forEach(training => {
                    const row = document.createElement('tr');
                    const trainingDate = new Date(training.training_date);
                    const localDate = trainingDate.toLocaleDateString('ru-RU');

                    row.innerHTML = `
                        <td>${localDate}</td>
```

```
                        <td>${formatTimeAMPM(training.training_time)}</td> <!-- Преобразуем
время в AM/PM формат -->
                        <td>${training.coach}</td>
                    `;
                    trainingList.appendChild(row);
                });
            }
        })
        .catch(error => {
            const row = document.createElement('tr');
            row.innerHTML = `<td colspan="3">Ошибка загрузки данных</td>`;
            trainingList.appendChild(row);
        });
}


// Добавляем обработчик для кнопки "Посмотреть тренировки"
const trainingButtons = document.querySelectorAll('.btn-book');
trainingButtons.forEach(button => {
    button.addEventListener('click', () => {
        const coach = button.getAttribute('data-coach');
        updateTrainingList(coach);
    });
});

let currentRole = 'user'; // Начальная роль пользователя

// Переключение ролей
document.getElementById('roleToggleBtn').addEventListener('click', function() {
    // Переключаем между ролями
    if (currentRole === 'user') {
        currentRole = 'trainer';
        alert('Вы вошли как тренер');
    } else {
        currentRole = 'user';
        alert('Вы вошли как пользователь');
    }

    toggleRoleFeatures();
});

function toggleRoleFeatures() {
    const showTrainingsBtns = document.querySelectorAll('.showTrainingsBtn');

    // Показываем кнопки "Посмотреть тренировки" только для тренеров
    showTrainingsBtns.forEach(function(btn) {
        if (currentRole === 'trainer') {
            btn.style.display = 'block';
        } else {
            btn.style.display = 'none';
```

```
        }
    });
}


// Пример обработки модального окна
document.querySelector('.btn-book').addEventListener('click', function(event) {
    const coach = event.target.getAttribute('data-coach');
    console.log('Тренер:', coach); // Выводим имя тренера, можно передать в модальное окно
});

    </script>

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/js/bootstrap.bundle.min.js"></script>
</body>
</html>
```

Index.html

```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Спортивный клуб</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/css/bootstrap.min.css" rel="stylesheet">
    <link href="css/styles.css" rel="stylesheet">
    <style>
        body {
            background-color: #111;
            color: white;
            font-family: Arial, sans-serif;
        }

        .hero-section {
            position: relative;
            background-color: #222;
            color: white;
            text-align: center;
            padding: 150px 20px 50px;
            margin-bottom: 30px;
        }

        .hero-section h1 {
            font-size: 48px;
            font-weight: bold;
            color: red;
        }
```

```
.hero-section p {
    font-size: 24px;
    line-height: 1.5;
    font-weight: 300;
}

.highlight {
    color: #FF4500;
    font-weight: bold;
}

.welcome-text {
    font-size: 36px;
    color: #FF4500;
    animation: slideIn 2s forwards;
}

@keyframes slideIn {
    from {
        transform: translateX(-100%);
        opacity: 0;
    }
    to {
        transform: translateX(0);
        opacity: 1;
    }
}

.container {
    margin-top: 40px;
}

.contact-info {
    background-color: #333;
    color: white;
    padding: 30px;
    margin-top: 30px;
    text-align: center;
}

.offer-card {
    background-color: #FF4500;
    color: white;
    margin: 15px;
    padding: 20px;
    border-radius: 10px;
    text-align: center;
}

.offer-card h5 {
```

```
            margin-bottom: 15px;
        }

        .offer-card p {
            font-size: 18px;
        }

        .offer-card button {
            background-color: #222;
            color: white;
            border: none;
            padding: 10px 20px;
            cursor: pointer;
            border-radius: 5px;
            margin-top: 10px;
        }

        .offer-card button:hover {
            background-color: #FF0000;
        }
        .navbar-nav {
            margin-left: auto;
        }
    </style>
</head>
<body>

    <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
        <div class="container-fluid">
            <a class="navbar-brand" href="index.html">Спортивный клуб</a>
            <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Переключить
навигацию">
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbarNav">
                <ul class="navbar-nav ms-auto">
                    <li class="nav-item">
                        <a class="nav-link" href="index.html">Главная</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="trainings.html">Тренировки</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="coaches.html">Тренера</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="membership.html">Абонементы</a>
                    </li>
                    <li class="nav-item">
```

```
                    <a class="nav-link" href="shop.html">Магазин</a>
                </li>
                <li class="nav-item">
                    <a class="nav-link" href="reviews.html">Отзывы</a>
                </li>
                <li class="nav-item">
                    <a class="nav-link" href="rental.html">Аренда залов и дополнительные
услуги</a>
                </li>

                <li class="nav-item">
                    <a class="nav-link" href="profile.html">Профиль</a>
                </li>
            </ul>
        </div>
    </div>
</nav>

<!-- Главная страница -->
<header class="hero-section">
    <h1>Спортивный клуб</h1>
    <p class="welcome-text">Всё начинается с <span class="highlight">ТЕБЯ</span></p>
</header>

<div class="container">
    <section>
        <h2>Мы поможем вам стать лучше!</h2>
        <p>Наш клуб предлагает широкий спектр услуг, которые помогут вам достичь самых
высоких результатов в спорте. Всё, что вам нужно сделать — это работать и совершенствоваться.
Мы предоставим вам лучшие условия для тренировок, квалифицированных тренеров и современное
оборудование.</p>
    </section>

    <section>
        <h3>Акционные предложения</h3>
        <div class="d-flex justify-content-around">
            <div class="offer-card">
                <h5>Скидка 20% на первый месяц</h5>
                <p>Только для новых клиентов!</p>
                <button>Подробнее</button>
            </div>
            <div class="offer-card">
                <h5>Подарок при регистрации</h5>
                <p>Футболка с логотипом клуба для первых 50 клиентов!</p>
                <button>Подробнее</button>
            </div>
        </div>
    </section>
</div>
```

```
    <div class="contact-info">
        <h3>Контактная информация</h3>
        <p><strong>Телефон:</strong> +7 800 555-35-35</p>
        <p><strong>Адрес клуба:</strong> Санкт-Петербург, ул. Гастело, 15</p>
        <p><strong>Юридический адрес:</strong> Санкт-Петербург, ул. Большая Морская, 67</p>
    </div>

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/js/bootstrap.bundle.min.js"></script>
</body>
</html>
```

Membership.html

```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Оформление абонемента</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/css/bootstrap.min.css" rel="stylesheet">
    <style>
        body {
            background-color: #121212;
            color: white;
        }
        .navbar {
            background-color: #222;
        }
        .navbar-brand, .nav-link {
            color: #e60000 !important;
        }
        .nav-link:hover {
            color: #ff0000 !important;
        }
        .tariff-card {
            border: 1px solid #444;
            border-radius: 10px;
            padding: 15px;
            margin-bottom: 20px;
            text-align: center;
            background-color: #333;
            height: 100%;
            display: flex;
            flex-direction: column;
            justify-content: space-between;
        }
        .tariff-card img {
            width: 100%;
```

```
            height: 200px;
            object-fit: cover;
            border-radius: 10px;
            margin-bottom: 15px;
        }
        .tariff-card h5 {
            font-size: 24px;
            margin-bottom: 10px;
        }
        .tariff-card p {
            font-size: 16px;
            margin-bottom: 15px;
        }
        .tariff-card button {
            background-color: #e60000;
            border: none;
            font-size: 18px;
            width: 48%;
            margin-right: 4%;
            margin-bottom: 10px;
        }
        .tariff-card button:hover {
            background-color: #ff0000;
        }
        .tariff-card .btn:last-child {
            margin-right: 0;
        }
        .container {
            margin-top: 40px;
        }
        .list-group-item {
            background-color: #333;
            border: none;
            color: white;
        }
        .list-group-item input {
            margin-right: 10px;
        }
        .cart {
            background-color: #333;
            border-radius: 10px;
            padding: 20px;
            margin-top: 40px;
        }
        .cart-item {
            display: flex;
            justify-content: space-between;
            padding: 5px 0;
        }
        .cart-item span {
```

```
            font-weight: bold;
        }
        .cart-item button {
            background-color: #e60000;
            border: none;
            color: white;
            padding: 5px 10px;
            cursor: pointer;
        }
        .cart-item button:hover {
            background-color: #ff0000;
        }
        .payment-button {
            width: 100%;
            margin-bottom: 10px;
            font-size: 16px;
        }
        .modal-title {
        color: black !important;
        }
    </style>
</head>
<body>

    <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
        <div class="container-fluid">
            <a class="navbar-brand" href="index.html">Спортивный клуб</a>
            <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Переключить
навигацию">
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbarNav">
                <ul class="navbar-nav ms-auto">
                    <li class="nav-item">
                        <a class="nav-link" href="index.html">Главная</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="trainings.html">Тренировки</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="coaches.html">Тренера</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="membership.html">Абонементы</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="shop.html">Магазин</a>
                    </li>
                    <li class="nav-item">
```

```
                    <a class="nav-link" href="reviews.html">Отзывы</a>
                </li>
                <li class="nav-item">
                    <a class="nav-link" href="rental.html">Аренда</a>
                </li>
                <li class="nav-item">
                    <a class="nav-link" href="additional_services.html">Дополнительные
услуги клуба</a>
                </li>
                <li class="nav-item">
                    <a class="nav-link" href="profile.html">Профиль</a>
                </li>
                <li class="nav-item">
                    <a class="nav-link" href="#" data-bs-toggle="modal" data-bs-
target="#cartModal">
                        Корзина (<span id="cartCount">0</span>)
                    </a>
                </li>
            </ul>
        </div>
    </div>
</nav>
    <button id="roleToggleBtn" class="btn btn-secondary mb-3">Сменить роль</button>

<div class="container mt-4">
    <button id="viewPurchasesBtn" class="btn btn-warning togglePurchasesButton"
style="display: none;" onclick="viewPurchasedSubscriptions()">Посмотреть купленные
абонементы</button>
    <div id="subscriptionsListContainer" style="display: none;">
        <button class="btn btn-danger mb-3"
onclick="toggleSubscriptionsList()">Закрыть</button>
        <ul id="subscriptionsList" class="list-group mt-4"></ul> <!-- Список купленных
абонементов -->
    </div>
</div>


    <div class="container">
        <h2 class="mt-4">Оформление абонемента</h2>
        <p>Выберите абонемент, который вам подходит:</p>

        <div class="row">
            <!-- Абонемент на 1 месяц -->
            <div class="col-md-4 mb-4">
                <div class="tariff-card">
                    <img src="img/abonement1.jpg" alt="Абонемент 1 месяц">
                    <h5>Абонемент на 1 месяц</h5>
                    <p>Стоимость: 3000 руб.</p>
                    <p>Подходит для тех, кто хочет попробовать клуб или посещает редко.</p>
                    <div class="d-flex justify-content-between">
```

```
                    <button class="btn" onclick="addToCart('Абонемент на 1 месяц',
3000)">Выбрать</button>
                </div>
            </div>
        </div>

        <!-- Абонемент на 6 месяцев -->
        <div class="col-md-4 mb-4">
            <div class="tariff-card">
                <img src="img/abonement2.jpg" alt="Абонемент 6 месяцев">
                <h5>Абонемент на 6 месяцев</h5>
                <p>Стоимость: 15000 руб.</p>
                <p>Подходит для тех, кто планирует посещать клуб регулярно.</p>
                <div class="d-flex justify-content-between">
                    <button class="btn" onclick="addToCart('Абонемент на 6 месяцев',
15000)">Выбрать</button>
                </div>
            </div>
        </div>

        <div class="col-md-4 mb-4">
            <div class="tariff-card">
                <img src="img/abonement3.jpg" alt="Абонемент 12 месяцев">
                <h5>Абонемент на 12 месяцев</h5>
                <p>Стоимость: 27000 руб.</p>
                <p>Идеален для тех, кто серьезно настроен на тренировки.</p>
                <div class="d-flex justify-content-between">
                    <button class="btn" onclick="addToCart('Абонемент на 12 месяцев',
27000)">Выбрать</button>
                </div>
            </div>
        </div>
    </div>

    <div id="cartModal" class="modal fade" tabindex="-1" aria-labelledby="cartModalLabel"
aria-hidden="true">
        <div class="modal-dialog modal-lg">
            <div class="modal-content">
                <div class="modal-header">
                    <!-- Заголовок с чёрным цветом -->
                    <h5 class="modal-title" id="cartModalLabel" style="color: black;">Ваша
корзина</h5>
                    <button type="button" class="btn-close" data-bs-dismiss="modal" aria-
label="Close"></button>
                </div>
                <div class="modal-body">
                    <div id="cart">
                        <ul class="list-group">
                            <li class="list-group-item">В корзине нет товаров</li>
                        </ul>
```

```
                        </div>
                        <div class="mt-4">
                            <h4 style="color: black;">Итого: <span id="totalPrice">0</span>
руб.</h4>
                        </div>
                    </div>
                    <div class="modal-footer">
                        <button type="button" class="btn btn-secondary" data-bs-
dismiss="modal">Закрыть</button>
                        <button type="button" class="btn btn-primary"
onclick="proceedToPayment()">Перейти к оплате</button>
                    </div>
                </div>
            </div>
        </div>

        <!-- Модальное окно для завершения покупки -->
        <div class="modal fade" id="paymentModal" tabindex="-1" aria-
labelledby="paymentModalLabel" aria-hidden="true">
            <div class="modal-dialog">
                <div class="modal-content">
                    <div class="modal-header">
                        <h5 class="modal-title" id="paymentModalLabel">Оплата</h5>
                        <button type="button" class="btn-close" data-bs-dismiss="modal" aria-
label="Close"></button>
                    </div>
                    <div class="modal-body">
                        <h5 style="color: black;">Общая сумма: <span
id="paymentTotalPrice">0</span> руб.</h5>
                        <button class="btn btn-success payment-button"
onclick="payWithSBP()">Оплатить через СБП</button>
                        <button class="btn btn-danger payment-button"
onclick="completePayment()">Завершить покупку</button>
                    </div>
                </div>
            </div>
        </div>
    </div>

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/js/bootstrap.bundle.min.js"></script>
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>

    <script>
        let cartItems = [];

        // Добавление товара в корзину
        function addToCart(name, price) {
            const item = { name, price };
            cartItems.push(item);
```

```
        updateCartDisplay();
    }

    // Обновление отображения корзины
    function updateCartDisplay() {
        const cart = document.getElementById("cart");
        const cartCount = document.getElementById("cartCount");
        let total = 0;
        cart.innerHTML = '';

        cartItems.forEach(item => {
            total += item.price;
            const listItem = document.createElement("li");
            listItem.classList.add("list-group-item");
            listItem.textContent = `${item.name}: ${item.price} руб.`;
            cart.appendChild(listItem);
        });

        document.getElementById("totalPrice").textContent = total;
        cartCount.textContent = cartItems.length;
    }

    // Переход к оплате
    function proceedToPayment() {
        const total = cartItems.reduce((sum, item) => sum + item.price, 0);
        document.getElementById('paymentTotalPrice').textContent = total;
        const paymentModal = new bootstrap.Modal(document.getElementById('paymentModal'));
        paymentModal.show();
    }

    // Оплата через СБП
    function payWithSBP() {
        alert("Вы выбрали оплату через СБП. Пожалуйста, следуйте инструкциям на экране.");
        completePayment();
    }

    // Завершение покупки
    function completePayment() {
        alert('Покупка завершена! Спасибо за покупку.');

        // Отправка данных на сервер для сохранения в БД
        cartItems.forEach(item => {
            fetch('http://localhost:3000/purchase', {
                method: 'POST',
                headers: {
                    'Content-Type': 'application/json'
                },
                body: JSON.stringify({
                    name: item.name,
                    price: item.price
```

```
                    })
                })
                .then(response => {
                    if (response.ok) {
                        console.log('Покупка записана в базу данных');
                    } else {
                        console.error('Ошибка при записи в базу данных');
                    }
                })
                .catch(error => {
                    console.error('Ошибка при отправке данных на сервер:', error);
                });
            });

            // Очищаем корзину
            cartItems.length = 0;
            document.getElementById('cartCount').textContent = `0`;
            document.getElementById('cart').innerHTML = '<li class="list-group-item">В корзине
нет товаров</li>';
            document.getElementById('totalPrice').textContent = '0';
            const paymentModal =
bootstrap.Modal.getInstance(document.getElementById('paymentModal'));
            paymentModal.hide();  // Закрываем модальное окно оплаты
        }

        let currentRole = 'user';

        document.getElementById('roleToggleBtn').addEventListener('click', function () {
            if (currentRole === 'admin') {
                currentRole = 'user';
                alert('Вы вошли как пользователь');
            } else {
                currentRole = 'admin';
                alert('Вы вошли как администратор');
            }

            toggleRoleFeatures(); //
        });

        function toggleRoleFeatures() {
            const viewPurchasesBtn = document.getElementById('viewPurchasesBtn');


            if (currentRole === 'admin') {
                viewPurchasesBtn.style.display = 'block';
            } else {
                viewPurchasesBtn.style.display = 'none';
            }
        }
```

```
        function viewPurchasedSubscriptions() {
            fetch('http://localhost:3000/get-purchased-subscriptions') /
.json())
                .then(data => {
                    const subscriptionsList = document.getElementById('subscriptionsList');
                    subscriptionsList.innerHTML = '';

                    data.forEach(subscription => {
                        const listItem = document.createElement('li');
                        listItem.classList.add('list-group-item');
                        listItem.textContent = `Абонемент: ${subscription.name}, Цена:
${subscription.price} руб.`;
                        subscriptionsList.appendChild(listItem);
                    });
                })
                .catch(error => {
                    console.error('Ошибка при получении данных:', error);
                });
        }

function toggleSubscriptionsList() {
    const subscriptionsListContainer = document.getElementById('subscriptionsListContainer');
    const viewPurchasesBtn = document.getElementById('viewPurchasesBtn');

    // Если список скрыт, показываем его
    if (subscriptionsListContainer.style.display === 'none') {
        subscriptionsListContainer.style.display = 'block';
        viewPurchasesBtn.style.display = 'none';
    } else {
        subscriptionsListContainer.style.display = 'none';
        viewPurchasesBtn.style.display = 'block';
    }
}

function viewPurchasedSubscriptions() {
    fetch('http://localhost:3000/get-purchased-subscriptions') // Запрос к серверу на
получение данных
        .then(response => response.json())
        .then(data => {
            const subscriptionsList = document.getElementById('subscriptionsList');
            subscriptionsList.innerHTML = ''; // Очищаем список

            data.forEach(subscription => {
                const listItem = document.createElement('li');
                listItem.classList.add('list-group-item');
                listItem.textContent = `Абонемент: ${subscription.name}, Цена:
${subscription.price} руб.`;
                subscriptionsList.appendChild(listItem);
            });
```

```
            toggleSubscriptionsList();
        })
        .catch(error => {
            console.error('Ошибка при получении данных:', error);
        });
}
        toggleRoleFeatures();
    </script>
</body>
</html>
```

Profile.html

```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Профиль спортсмена</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/css/bootstrap.min.css" rel="stylesheet">
    <style>
        body {
            background-color: #121212;
            color: white;
        }
        .navbar {
            background-color: #222;
        }
        .navbar-brand, .nav-link {
            color: #e60000 !important;
        }
        .nav-link:hover {
            color: #ff0000 !important;
        }
        .profile-card {
            background-color: #333;
            border-radius: 10px;
            padding: 20px;
            margin-bottom: 40px;
        }
        .profile-card img {
            width: 150px;
            height: 150px;
            border-radius: 50%;
            object-fit: cover;
        }
        .profile-card h3 {
            margin-top: 15px;
```

```
        }
        .history-list {
            background-color: #444;
            padding: 10px;
            border-radius: 5px;
            margin-bottom: 20px;
        }
        .history-item {
            display: flex;
            justify-content: space-between;
            padding: 5px 0;
        }
        .history-item span {
            font-weight: bold;
        }
        .btn-change-user {
            background-color: #007bff;
            color: white;
            margin-top: 20px;
        }
        .btn-change-user:hover {
            background-color: #0056b3;
        }
        .user-list {
            background-color: #444;
            padding: 10px;
            border-radius: 5px;
            max-height: 200px;
            overflow-y: auto;
            margin-top: 20px;
        }
        .user-item {
            padding: 5px;
            cursor: pointer;
        }
        .user-item:hover {
            background-color: #555;
        }
        .add-user-form {
            background-color: #444;
            padding: 20px;
            border-radius: 10px;
            display: none;
            margin-top: 20px;
        }
        .add-user-form input, .add-user-form button {
            margin-bottom: 10px;
        }
    </style>
</head>
```

```
<body>
    <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
        <div class="container-fluid">
            <a class="navbar-brand" href="index.html">Спортивный клуб</a>
            <div class="collapse navbar-collapse" id="navbarNav">
                <ul class="navbar-nav ms-auto">
                    <li class="nav-item"><a class="nav-link"
href="index.html">Главная</a></li>
                    <li class="nav-item"><a class="nav-link"
href="trainings.html">Тренировки</a></li>
                    <li class="nav-item"><a class="nav-link"
href="coaches.html">Тренера</a></li>
                    <li class="nav-item"><a class="nav-link"
href="membership.html">Абонементы</a></li>
                    <li class="nav-item"><a class="nav-link" href="shop.html">Магазин</a></li>
                    <li class="nav-item"><a class="nav-link"
href="reviews.html">Отзывы</a></li>
                    <li class="nav-item"><a class="nav-link"
href="rental.html">Аренда</a></li>
                    <li class="nav-item"><a class="nav-link"
href="additional_services.html">Дополнительные услуги клуба</a></li>
                    <li class="nav-item"><a class="nav-link"
href="profile.html">Профиль</a></li>

                </ul>
            </div>
        </div>
    </nav>
    <button id="roleToggleBtn" class="btn btn-secondary mb-3">Сменить роль</button>
    <div class="container">
        <h2 class="mt-4">Профиль спортсмена</h2>

        <!-- Карточка профиля -->
        <div class="profile-card d-flex" id="profileCard">
            <img src="img/fotoatleta.jfif" alt="Фото спортсмена">
            <div class="ms-3">
                <h3 id="athleteName">Иван Иванов</h3>
                <p>Возраст: <span id="athleteAge">25 лет</span></p>
                <p>Город: <span id="athleteCity">Москва</span></p>
                <p>Занятие: <span id="athleteActivity">Фитнес, кардио, йога</span></p>
                <p>Телефон: <span id="athletePhone">+7 000 000 00 00</span></p>

                <button class="btn btn-change-user admin-only"
onclick="toggleUserList()">Сменить пользователя</button>
            </div>
        </div>

        <!-- Кнопка для добавления пользователя -->
        <button class="btn btn-success admin-only" onclick="toggleAddUserForm()">Добавить
пользователя</button>
```

```
<!-- Форма для добавления пользователя -->
<div class="add-user-form" id="addUserForm">
    <h3>Добавить нового пользователя</h3>
    <form id="addUserFormElement">
        <label for="name">Имя:</label>
        <input type="text" id="name" name="name" required><br>

        <label for="age">Возраст:</label>
        <input type="number" id="age" name="age" required><br>

        <label for="city">Город:</label>
        <input type="text" id="city" name="city" required><br>

        <label for="activity">Деятельность:</label>
        <input type="text" id="activity" name="activity" required><br>
        <label for="phone">Телефон:</label>
        <input type="tel" id="phone" name="phone" required><br>
        <button type="submit" class="btn btn-primary">Добавить</button>
    </form>
</div>
<!-- Форма для обновления пользователя -->
<div class="add-user-form" id="updateUserForm" style="display: none;">
    <h3>Обновить данные пользователя</h3>
    <form id="updateUserFormElement">
        <label for="updateName">Имя:</label>
        <input type="text" id="updateName" name="updateName" required><br>

        <label for="updateAge">Возраст:</label>
        <input type="number" id="updateAge" name="updateAge" required><br>

        <label for="updateCity">Город:</label>
        <input type="text" id="updateCity" name="updateCity" required><br>

        <label for="updateActivity">Деятельность:</label>
        <input type="text" id="updateActivity" name="updateActivity" required><br>
        <label for="updatePhone">Телефон:</label>
        <input type="tel" id="updatePhone" name="updatePhone" required><br>


        <button type="submit" class="btn btn-warning">Обновить</button>
    </form>
</div>


<!-- Список пользователей -->


<div class="user-list" id="userList" style="display: none;">
```

```
        </div>

    </div>

    <script>
        let currentRole = 'user';

        // Переключение ролей
        document.getElementById('roleToggleBtn').addEventListener('click', function () {
            if (currentRole === 'admin') {
                currentRole = 'user';
                alert('Вы вошли как пользователь');
            } else {
                currentRole = 'admin';
                alert('Вы вошли как администратор');
            }

            toggleRoleFeatures(); // Обновляем видимость кнопок
        });

        // Функция для управления видимостью элементов в зависимости от роли
        function toggleRoleFeatures() {
            const adminOnlyElements = document.querySelectorAll('.admin-only');
            adminOnlyElements.forEach(el => {
                el.style.display = currentRole === 'admin' ? 'block' : 'none';
            });
        }

        // Инициализация отображения
        toggleRoleFeatures();
        // Функция для переключения отображения списка пользователей
        function toggleUserList() {
            const userList = document.getElementById('userList');
            if (userList.style.display === 'none') {
                userList.style.display = 'block';
                loadUsers();
            } else {
                userList.style.display = 'none';
            }
        }

        // Функция для переключения отображения формы добавления пользователя
        function toggleAddUserForm() {
            const addUserForm = document.getElementById('addUserForm');
            if (addUserForm.style.display === 'none' || addUserForm.style.display === '') {
                addUserForm.style.display = 'block';
            } else {
                addUserForm.style.display = 'none';
            }
        }
```

```
        function loadUsers() {
            fetch('http://localhost:3000/api/users')
                .then(response => response.json())
                .then(data => {
                    const userList = document.getElementById('userList');
                    userList.innerHTML = '';


                    data.users.forEach(user => {
                        const div = document.createElement('div');
                        div.className = 'user-item';
                        div.textContent = user.name;
                        div.onclick = () => changeUser(user.id);
                        userList.appendChild(div);
                    });
                })
                .catch(error => console.error('Ошибка при загрузке пользователей:', error));
        }



        document.getElementById('addUserFormElement').addEventListener('submit', function
(event) {
    event.preventDefault();

    // Получаем данные из формы
    const name = document.getElementById('name').value;
    const age = document.getElementById('age').value;
    const city = document.getElementById('city').value;
    const activity = document.getElementById('activity').value;
    const phone = document.getElementById('phone').value;

    // Отправка данных на сервер
    fetch('http://localhost:3000/api/addUser', {
    method: 'POST',
    headers: {
        'Content-Type': 'application/json',
    },
    body: JSON.stringify({ name, age, city, activity, phone })
})
    .then(response => response.json())
    .then(data => {
        if (data.success) {
            alert('Пользователь успешно добавлен!');
            loadUsers();
            document.getElementById('addUserForm').style.display = 'none';
        } else {
            alert('Ошибка при добавлении пользователя!');
        }
```

```
    })
    .catch(error => console.error('Ошибка при добавлении пользователя:', error));
});



let currentUserId = null;

// Функция для переключения отображения списка пользователей
function toggleUserList() {
    const userList = document.getElementById('userList');
    if (userList.style.display === 'none') {
        userList.style.display = 'block';
        loadUsers();  // Загружаем список пользователей
    } else {
        userList.style.display = 'none';
    }
}

// Функция для переключения отображения формы добавления пользователя
function toggleAddUserForm() {
    const addUserForm = document.getElementById('addUserForm');
    if (addUserForm.style.display === 'none' || addUserForm.style.display === '') {
        addUserForm.style.display = 'block';
    } else {
        addUserForm.style.display = 'none';
    }
}

// Функция для загрузки списка пользователей с сервера
function loadUsers() {
fetch('http://localhost:3000/api/users')
.then(response => response.json())
.then(data => {
    const userList = document.getElementById('userList');
    userList.innerHTML = '';  // Очищаем предыдущий список

    // Заполняем список пользователей
    data.users.forEach(user => {
        const div = document.createElement('div');
        div.className = 'user-item';
        div.textContent = user.name;

        // Кнопка для обновления пользователя
        const updateButton = document.createElement('button');
        updateButton.textContent = 'Обновить';
        updateButton.className = 'btn btn-warning btn-sm ms-2';
        updateButton.onclick = () => openUpdateForm(user.id);

        // Кнопка для удаления пользователя
```

```
        const deleteButton = document.createElement('button');
        deleteButton.textContent = 'Удалить';
        deleteButton.className = 'btn btn-danger btn-sm ms-2';
        deleteButton.onclick = () => deleteUser(user.id);

        div.appendChild(updateButton);
        div.appendChild(deleteButton);

        div.onclick = () => {
            currentUserId = user.id;
            changeUser(user.id);
        };

        userList.appendChild(div);
    });
})
.catch(error => console.error('Ошибка при загрузке пользователей:', error));
}

function changeUser(userId) {
    fetch(`http://localhost:3000/api/user/${userId}`)
        .then(response => response.json())
        .then(data => {
            const user = data.user;
            document.getElementById('athleteName').textContent = user.name;
            document.getElementById('athleteAge').textContent = user.age;
            document.getElementById('athleteCity').textContent = user.city;
            document.getElementById('athleteActivity').textContent = user.activity;
            document.getElementById('athletePhone').textContent = user.phone;
            document.getElementById('userList').style.display = 'none';
        })
        .catch(error => console.error('Ошибка при загрузке данных пользователя:', error));
}


function openUpdateForm(userId) {
    fetch(`http://localhost:3000/api/user/${userId}`)
        .then(response => response.json())
        .then(data => {
            const user = data.user;
            document.getElementById('updateName').value = user.name;
            document.getElementById('updateAge').value = user.age;
            document.getElementById('updateCity').value = user.city;
            document.getElementById('updateActivity').value = user.activity;
            document.getElementById('athletePhone').textContent = user.phone;
            document.getElementById('updateUserForm').style.display = 'block';
            currentUserId = userId;
        })
        .catch(error => console.error('Ошибка при загрузке данных для обновления:', error));
}
```

```
function deleteUser(userId) {
    fetch(`http://localhost:3000/api/user/${userId}`, {
        method: 'DELETE'
    })
    .then(response => response.json())
    .then(data => {
        if (data.success) {
            alert('Пользователь успешно удален!');
            loadUsers
        } else {
            alert('Ошибка при удалении пользователя!');
        }
    })
    .catch(error => console.error('Ошибка при удалении пользователя:', error));
}
document.getElementById('updateUserFormElement').addEventListener('submit', function (event) {
    event.preventDefault();

    const name = document.getElementById('updateName').value;
    const age = document.getElementById('updateAge').value;
    const city = document.getElementById('updateCity').value;
    const activity = document.getElementById('updateActivity').value;
    const phone = document.getElementById('updatePhone').value;

    fetch(`http://localhost:3000/api/user/${currentUserId}`, {
        method: 'PUT',
        headers: {
            'Content-Type': 'application/json',
        },
        body: JSON.stringify({ name, age, city, activity, phone })
    })
    .then(response => response.json())
    .then(data => {
        if (data.success) {
            alert('Данные пользователя обновлены!');
            loadUsers();
            document.getElementById('updateUserForm').style.display = 'none';
        } else {
            alert('Ошибка при обновлении данных пользователя!');
        }
    })
    .catch(error => console.error('Ошибка при обновлении данных пользователя:', error));
});



    </script>
</body>
</html>
```

Rental.html

```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Аренда залов и дополнительные услуги</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/css/bootstrap.min.css" rel="stylesheet">
    <link href="css/styles.css" rel="stylesheet">
    <style>
        .rental-item img {
            width: 100%;
            height: auto;
        }
        .rental-item {
            display: flex;
            flex-direction: column;
            height: 100%;
        }
        .rental-item .card-body {
            display: flex;
            flex-direction: column;
            justify-content: space-between;
            height: 100%;
        }
        .rental-item .btn {
            margin-top: auto;
        }
        .row {
            margin-top: 20px;
        }
        .contact-form {
            display: none;
            margin-top: 20px;
        }
        .admin-controls {
            margin-top: 20px;
        }
        .admin-controls .btn {
            margin-right: 10px;
        }
    </style>
</head>
<body>

    <!-- Навигация -->
    <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
        <div class="container-fluid">
            <a class="navbar-brand" href="index.html">Спортивный клуб</a>
```

```
            <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Переключить
навигацию">
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbarNav">
                <ul class="navbar-nav ms-auto">
                    <li class="nav-item">
                        <a class="nav-link" href="index.html">Главная</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="trainings.html">Тренировки</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="coaches.html">Тренера</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="membership.html">Абонементы</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="shop.html">Магазин</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="reviews.html">Отзывы</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="rental.html">Аренда залов и дополнительные
услуги</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="profile.html">Профиль</a>
                    </li>
                </ul>
            </div>
        </div>
    </nav>
    <button class="btn btn-secondary mb-3" onclick="toggleRole()">Сменить роль</button>
    <div class="container">
        <h2 class="mt-5">Аренда инвентаря и залов</h2>

        <!-- Кнопки смены ролей и администрирования -->
        <div class="admin-controls">
            <button class="btn btn-info" onclick="viewAllBookings()" style="display:
none;">Посмотреть все бронирования</button>
            <div id="booking-list" class="mt-4"></div>
        </div>

        <!-- Аренда залов -->
        <h3 class="mt-4">Аренда залов</h3>
        <p>Выберите зал для аренды:</p>
```

```
        <div class="row">
            <!-- Зал 1 (Силовые тренировки) -->
            <div class="col-md-4">
                <div class="card rental-item">
                    <img src="img/zal_1.jpg" alt="Зал 1">
                    <div class="card-body">
                        <h5 class="card-title">Зал 1 - Силовые тренировки</h5>
                        <p>Время: 10:00 - 14:00</p>
                        <p>Описание: Зал для силовых тренировок, оснащён современными
тренажерами.</p>
                        <button class="btn btn-primary" onclick="showContactForm(this, 'Зал 1
- Силовые тренировки')">Забронировать</button>
                        <div class="contact-form">
                            <p>Для бронирования введите ваш телефон:</p>
                            <input type="tel" class="form-control" placeholder="Введите номер
телефона" required>
                            <button class="btn btn-success mt-2"
onclick="submitContactForm(this)">Отправить</button>
                        </div>
                    </div>
                </div>
            </div>
            <!-- Зал 2 (Фитнес) -->
            <div class="col-md-4">
                <div class="card rental-item">
                    <img src="img/zal_2.jpg" alt="Зал 2">
                    <div class="card-body">
                        <h5 class="card-title">Зал 2 - Фитнес</h5>
                        <p>Время: 14:00 - 18:00</p>
                        <p>Описание: Зал для фитнес тренировок, подходит для кардио и
групповых занятий.</p>
                        <button class="btn btn-primary" onclick="showContactForm(this, 'Зал 2
- Фитнес')">Забронировать</button>
                        <div class="contact-form">
                            <p>Для бронирования введите ваш телефон:</p>
                            <input type="tel" class="form-control" placeholder="Введите номер
телефона" required>
                            <button class="btn btn-success mt-2"
onclick="submitContactForm(this)">Отправить</button>
                        </div>
                    </div>
                </div>
            </div>
            <!-- Зал 3 (Боевые искусства) -->
            <div class="col-md-4">
                <div class="card rental-item">
                    <img src="img/zal_3.jpg" alt="Зал 3">
                    <div class="card-body">
                        <h5 class="card-title">Зал 3 - Боевые искусства</h5>
                        <p>Время: 18:00 - 22:00</p>
```

```
                        <p>Описание: Зал для тренировок по боевым искусствам, оборудован
необходимым инвентарём.</p>
                        <button class="btn btn-primary" onclick="showContactForm(this, 'Зал 3
- Боевые искусства')">Забронировать</button>
                        <div class="contact-form">
                            <p>Для бронирования введите ваш телефон:</p>
                            <input type="tel" class="form-control" placeholder="Введите номер
телефона" required>
                            <button class="btn btn-success mt-2"
onclick="submitContactForm(this)">Отправить</button>
                        </div>
                    </div>
                </div>
            </div>
        </div>

        <!-- Аренда инвентаря -->
        <h3 class="mt-5">Дополнительные услуги</h3>
        <p>Выберите доп. услугу:</p>
        <div class="row">
            <div class="col-md-4">
                <div class="card rental-item">
                    <img src="img/sauna.jpg" alt="Сауна">
                    <div class="card-body">
                        <h5 class="card-title">Сауна</h5>
                        <p>Для полного расслабления и восстановления предлагается посещение
сауны после тренировки. Отлично для улучшения кровообращения и снятия напряжения.</p>
                        <button class="btn btn-primary" onclick="showContactForm(this,
'Сауна')">Забронировать</button>
                        <div class="contact-form">
                            <p>Для бронирования введите ваш телефон:</p>
                            <input type="tel" class="form-control" placeholder="Введите номер
телефона" required>
                            <button class="btn btn-success mt-2"
onclick="submitContactForm(this)">Отправить</button>
                        </div>
                    </div>
                </div>
            </div>
            <div class="col-md-4">
                <div class="card rental-item">
                    <img src="img/spa.jpg" alt="СПА">
                    <div class="card-body">
                        <h5 class="card-title">SPA</h5>
                        <p>Для тех, кто хочет получить максимум удовольствия и пользы от
тренировок, предлагается расслабляющие SPA-процедуры, чтобы восстановить силы и улучшить
самочувствие.</p>
                        <button class="btn btn-primary" onclick="showContactForm(this,
'SPA')">Забронировать</button>
                        <div class="contact-form">
```

```
                        <p>Для бронирования введите ваш телефон:</p>
                        <input type="tel" class="form-control" placeholder="Введите номер
телефона" required>
                        <button class="btn btn-success mt-2"
onclick="submitContactForm(this)">Отправить</button>
                    </div>
                </div>
            </div>
        </div>


    </div>

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/js/bootstrap.bundle.min.js"></script>

    <script>
let isAdmin = false
let isBookingsVisible = false;

// Функция для переключения роли
function toggleRole() {
    isAdmin = !isAdmin;
    const bookingButton = document.querySelector('.admin-controls .btn-info');

    if (isAdmin) {
        alert('Вы перешли в роль администратора');
        bookingButton.style.display = 'inline-block';
    } else {
        alert('Вы перешли в роль пользователя');
        bookingButton.style.display = 'none';
    }
}

// Функция для отображения списка бронирований
function viewAllBookings() {
    if (isAdmin) {
        const bookingListContainer = document.getElementById('booking-list');

        // Если список скрыт, показываем его
        if (!isBookingsVisible) {
            // Отправляем запрос на сервер для получения всех бронирований
            fetch('http://localhost:3000/rentals')
                .then(response => response.json())
                .then(data => {
                    bookingListContainer.innerHTML = '';  // Очищаем контейнер

                    if (data.length === 0) {
                        bookingListContainer.innerHTML = '<p>Нет доступных бронирований.</p>';
```

```
                    } else {
                        // Отображаем все бронирования
                        data.forEach(booking => {
                            const bookingItem = document.createElement('div');
                            bookingItem.classList.add('mb-3');
                            bookingItem.innerHTML = `
                                <div>
                                    <h5>${booking.item}</h5>
                                    <p><strong>Дата аренды:</strong>
${booking.rental_date}</p>
                                    <p><strong>Дата возврата:</strong>
${booking.return_date}</p>
                                    <p><strong>Телефон:</strong> ${booking.phone}</p>
                                </div>
                                <hr>
                            `;
                            bookingListContainer.appendChild(bookingItem);
                        });
                    }

                    // Переключаем видимость списка
                    isBookingsVisible = true;
                    // Меняем текст на кнопке
                    document.querySelector('.admin-controls .btn-info').textContent = 'Скрыть
все бронирования';
                })
                .catch(error => {
                    console.error('Ошибка при получении данных:', error);
                    alert('Не удалось получить данные о бронированиях.');
                });
        } else {
            bookingListContainer.innerHTML = '';  // Очищаем контейнер
            isBookingsVisible = false;
            document.querySelector('.admin-controls .btn-info').textContent = 'Посмотреть все
бронирования';
        }
    } else {
        alert('Только администратор может просматривать все бронирования.');
    }
}

// Функция для отображения формы для связи
function showContactForm(button, item) {
    const contactForm = button.parentElement.querySelector('.contact-form');
    contactForm.style.display = 'block';
    button.style.display = 'none';
    contactForm.dataset.item = item;
}

// Функция для отправки формы
```

```
function submitContactForm(button) {
    const phoneInput = button.parentElement.querySelector('input[type="tel"]');
    const item = button.parentElement.dataset.item;

    if (phoneInput.value) {
        fetch('/submit_rental', {
            method: 'POST',
            headers: {
                'Content-Type': 'application/json',
            },
            body: JSON.stringify({
                item: item,
                rental_date: new Date().toISOString().split('T')[0],
                return_date: new Date(new Date().getTime() + 2 * 60 * 60 *
1000).toISOString().split('T')[0],
                phone: phoneInput.value,
            })
        })
        .then(response => response.json())
        .then(data => {
            if (data.success) {
                alert('Ваша заявка отправлена! Менеджер свяжется с вами.');
                button.parentElement.style.display = 'none';
                button.parentElement.parentElement.querySelector('.btn').style.display =
'block';
            } else {
                alert('Ошибка при отправке данных. Попробуйте снова.');
            }
        })
        .catch(error => {
            console.error('Error:', error);
            alert('Ошибка при отправке данных. Попробуйте снова.');
        });
    } else {
        alert('Пожалуйста, введите номер телефона.');
    }
}

    </script>
</body>
</html>
```

Reviews.html

```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Отзывы о зале и тренировках</title>
```

```
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/css/bootstrap.min.css" rel="stylesheet">
    <link href="css/styles.css" rel="stylesheet">
</head>
<body>

    <!-- Навигация -->
    <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
        <div class="container-fluid">
            <a class="navbar-brand" href="index.html">Спортивный клуб</a>
            <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Переключить
навигацию">
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbarNav">
                <ul class="navbar-nav ms-auto">
                    <li class="nav-item"><a class="nav-link"
href="index.html">Главная</a></li>
                    <li class="nav-item"><a class="nav-link"
href="trainings.html">Тренировки</a></li>
                    <li class="nav-item"><a class="nav-link"
href="coaches.html">Тренера</a></li>
                    <li class="nav-item"><a class="nav-link"
href="membership.html">Абонементы</a></li>
                    <li class="nav-item"><a class="nav-link" href="shop.html">Магазин</a></li>
                    <li class="nav-item"><a class="nav-link"
href="reviews.html">Отзывы</a></li>
                    <li class="nav-item"><a class="nav-link" href="rental.html">Аренда залов и
дополнительные услуги</a></li>
                    <li class="nav-item"><a class="nav-link"
href="profile.html">Профиль</a></li>
                </ul>
            </div>
        </div>
    </nav>

    <div class="container">
        <h2 class="mt-5">Отзывы о зале и тренировках</h2>

        <!-- Форма для добавления отзыва -->
        <div class="mt-4">
            <h4>Оставьте отзыв</h4>
            <form id="reviewForm">
                <div class="mb-3">
                    <label for="reviewerName" class="form-label">Ваше имя</label>
                    <input type="text" class="form-control" id="reviewerName"
placeholder="Введите ваше имя">
                </div>
                <div class="mb-3">
```

```
                    <label for="reviewRating" class="form-label">Оценка</label>
                    <select class="form-select" id="reviewRating">
                        <option value="5">★★★★★ Отлично</option>
                        <option value="4">★★★★☆ Хорошо</option>
                        <option value="3">★★★☆☆ Средне</option>
                        <option value="2">★★☆☆☆ Плохо</option>
                        <option value="1">★☆☆☆☆ Очень плохо</option>
                    </select>
                </div>
                <div class="mb-3">
                    <label for="reviewText" class="form-label">Ваш отзыв</label>
                    <textarea class="form-control" id="reviewText" rows="3"
placeholder="Опишите ваш опыт..."></textarea>
                </div>
                <button type="submit" class="btn btn-primary">Отправить отзыв</button>
            </form>
        </div>

        <!-- Список последних отзывов -->
        <div class="mt-5">
            <h4>Последние отзывы:</h4>
            <div id="reviewsList"></div>
        </div>
    </div>

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/js/bootstrap.bundle.min.js"></script>
    <script>
        document.addEventListener('DOMContentLoaded', () => {
            const reviewsList = document.getElementById('reviewsList');

            // Загружаем отзывы при загрузке страницы
            fetch('/reviews')
                .then(response => response.json())
                .then(data => {
                    reviewsList.innerHTML = '';
                    data.forEach(review => {
                        const reviewCard = document.createElement('div');
                        reviewCard.classList.add('card', 'mt-3');
                        reviewCard.innerHTML = `
                            <div class="card-body">
                                <h5 class="card-title">${review.reviewer_name}</h5>
                                <div class="rating">${'★'.repeat(review.review_rating)}</div>
                                <p class="card-text">${review.review_text}</p>
                                <small class="text-muted">${review.review_date}</small>
                            </div>
                        `;
                        reviewsList.appendChild(reviewCard);
                    });
                })
```

```
                .catch(err => console.error('Ошибка загрузки отзывов:', err));

            // Добавление нового отзыва
            document.getElementById('reviewForm').addEventListener('submit', function (event)
{
                event.preventDefault();

                const reviewerName = document.getElementById('reviewerName').value.trim();
                const reviewRating = document.getElementById('reviewRating').value;
                const reviewText = document.getElementById('reviewText').value.trim();

                fetch('/reviews', {
                    method: 'POST',
                    headers: { 'Content-Type': 'application/json' },
                    body: JSON.stringify({ reviewerName, reviewRating, reviewText })
                })
                    .then(response => {
                        if (!response.ok) throw new Error('Ошибка при добавлении отзыва');
                        return response.text();
                    })
                    .then(() => {
                        alert('Отзыв успешно добавлен!');
                        document.getElementById('reviewForm').reset();
                        location.reload();
                    })
                    .catch(err => console.error('Ошибка добавления отзыва:', err));
            });
        });
    </script>
</body>
</html>
```

Shop.html

```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Магазин спортивных товаров</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/css/bootstrap.min.css" rel="stylesheet">
    <link href="css/styles.css" rel="stylesheet">
    <style>
        body {
            font-family: Arial, sans-serif;
        }

        .product-card {
            border: 1px solid #ddd;
            border-radius: 10px;
```

```
            padding: 15px;
            text-align: center;
            background-color: #f9f9f9;
            margin-bottom: 20px;
            display: flex;
            flex-direction: column;
            justify-content: space-between;
            height: 400px;
        }

        .product-card img {
            width: 100%;
            max-height: 200px;
            object-fit: contain;
            border-radius: 5px;
        }

        .product-title {
            font-size: 18px;
            font-weight: bold;
            margin-top: 10px;
        }

        .product-price {
            font-size: 16px;
            color: #FF4500;
            margin-top: 5px;
        }

        .btn-buy {
            background-color: #FF4500;
            color: white;
            border: none;
            padding: 10px;
            width: 100%;
            cursor: pointer;
            margin-top: 10px;
        }

        .btn-buy:hover {
            background-color: #FF0000;
        }

        .row {
            display: flex;
            flex-wrap: wrap;
            gap: 20px;
        }

        .col-md-4 {
```

```
        flex: 1 1 calc(33% - 20px);
    }

    .filter-container {
        margin-top: 30px;
        margin-bottom: 20px;
    }

    .cart-container {
        margin-top: 30px;
        padding: 20px;
        background-color: #f9f9f9;
        border-radius: 10px;
    }

    .category-title {
        margin-top: 50px;
        font-size: 24px;
        font-weight: bold;
    }

    .remove-item {
        color: red;
        cursor: pointer;
        font-size: 18px;
        margin-left: 10px;
    }

    .remove-item:hover {
        text-decoration: underline;
    }

    .payment-container {
        margin-top: 20px;
        text-align: center;
    }

    .payment-button {
        background-color: #28a745;
        color: white;
        padding: 15px;
        font-size: 18px;
        border: none;
        border-radius: 5px;
        cursor: pointer;
    }

    .payment-button:hover {
        background-color: #218838;
    }
```

```
        .modal-body p {
            font-size: 16px;
        }
        .modal-title{
            color: black;
        }
        .modal-body{
            color: black;
        }

    </style>
</head>
<body>

    <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
        <div class="container-fluid">
            <a class="navbar-brand" href="index.html">Спортивный клуб</a>
            <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Переключить
навигацию">
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbarNav">
                <ul class="navbar-nav ms-auto">
                    <li class="nav-item">
                        <a class="nav-link" href="index.html">Главная</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="trainings.html">Тренировки</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="coaches.html">Тренера</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="membership.html">Абонементы</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="shop.html">Магазин</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="reviews.html">Отзывы</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="rental.html">Аренда залов и дополнительные
услуги</a>
                    </li>

                    <li class="nav-item">
                        <a class="nav-link" href="profile.html">Профиль</a>
```

```
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="#" data-bs-toggle="modal" data-bs-
target="#cartModal">
                            Корзина (<span id="cartCount">0</span>)
                        </a>
                    </li>

                </ul>
            </div>
        </div>
    </nav>
    <button id="roleToggleBtn" class="btn btn-secondary mb-3">Сменить роль</button>
    <div class="container">
    <div class="container mt-4">
        <button id="togglePurchasesButton" class="btn btn-primary">Посмотреть купленные
товары</button>
        <div id="purchasesContainer" class="mt-3" style="display: none;">
            <h4>Купленные товары</h4>
            <ul id="purchasesList"></ul>
        </div>
    </div>
    <div class="container filter-container">
        <div class="row">
            <div class="col-md-4">
                <label for="categoryFilter">Категория:</label>
                <select id="categoryFilter" class="form-select">
                    <option value="">Все</option>
                    <option value="Одежда">Одежда</option>
                    <option value="Спортивное питание">Спортивное питание</option>
                    <option value="Аксессуары">Аксессуары</option>
                    <option value="Инвентарь">Инвентарь</option>
                </select>
            </div>
            <div class="col-md-4">
                <label for="priceFilter">Цена:</label>
                <select id="priceFilter" class="form-select">
                    <option value="">Все</option>
                    <option value="1">До 1000 руб</option>
                    <option value="2">1000 - 3000 руб</option>
                    <option value="3">Более 3000 руб</option>
                </select>
            </div>
        </div>
    </div>

    <div class="container">
        <h2>Наши товары</h2>

        <div class="category-title">Товары</div>
```

```
        <div class="row" id="productList"></div>
    </div>

    <!-- Модальное окно корзины -->
    <div class="modal fade" id="cartModal" tabindex="-1" aria-labelledby="cartModalLabel"
aria-hidden="true">
        <div class="modal-dialog">
            <div class="modal-content">
                <div class="modal-header">
                    <h5 class="modal-title" id="cartModalLabel">Корзина покупок</h5>
                    <button type="button" class="btn-close" data-bs-dismiss="modal" aria-
label="Close"></button>
                </div>
                <div class="modal-body">
                    <ul id="cartItems">
                        <li>В корзине нет товаров</li>
                    </ul>
                </div>
                <div class="modal-footer">
                    <button type="button" class="btn btn-secondary" data-bs-
dismiss="modal">Закрыть</button>
                    <button type="button" class="btn btn-primary"
onclick="proceedToPayment()">Перейти к оплате</button>
                </div>
            </div>
        </div>
    </div>

    <!-- Модальное окно оплаты -->
    <div class="modal fade" id="paymentModal" tabindex="-1" aria-
labelledby="paymentModalLabel" aria-hidden="true">
        <div class="modal-dialog">
            <div class="modal-content">
                <div class="modal-header">
                    <h5 class="modal-title" id="paymentModalLabel">Оплата</h5>
                    <button type="button" class="btn-close" data-bs-dismiss="modal" aria-
label="Close"></button>
                </div>
                <div class="modal-body">
                    <h4>Вы собираетесь завершить покупку</h4>
                    <p>Общая сумма: <span id="totalPrice"></span> руб</p>
                    <p>Выберите способ оплаты:</p>
                    <button class="payment-button" onclick="payWithSBP()">Оплатить через
СБП</button>
                </div>
            </div>
        </div>
    </div>
```

```
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/js/bootstrap.bundle.min.js"></script>
    <script>
        const products = [
            { name: 'Сумка тренажёрная', category: 'Аксессуары', price: 2000, image:
'img/sumkatrenazhyornaya1.jpg' },
            { name: 'Футболка спортивная', category: 'Одежда', price: 1200, image:
'img/futbolka_1.jpg' },
            { name: 'Шорты спортивные', category: 'Одежда', price: 800, image:
'img/shorty_1.jfif' },
            { name: 'Массажный пистолет', category: 'Аксессуары', price: 3500, image:
'img/massazhnyy pistolet.jpg' },
            { name: 'Гантели', category: 'Инвентарь', price: 1500, image: 'img/ganteli_1.png'
},
            { name: 'Коврик для йоги', category: 'Инвентарь', price: 1200, image:
'img/kovrikyoga.jfif' },
            { name: 'Обувь спортивная', category: 'Одежда', price: 2500, image:
'img/obuv_1.jfif' },
            { name: 'Протеин', category: 'Спортивное питание', price: 1800, image:
'img/protein_1.jfif' },
            { name: 'Спортивная бутылка', category: 'Аксессуары', price: 500, image:
'img/sportivnayabutylka_1.jpg' }
        ];

        const cart = [];

        // Отображение товаров
        function displayProducts(productsToShow) {
            const productList = document.getElementById('productList');
            productList.innerHTML = '';
            productsToShow.forEach(product => {
                const div = document.createElement('div');
                div.classList.add('col-md-4');
                div.innerHTML = `
                    <div class="product-card">
                        <img src="${product.image}" alt="${product.name}">
                        <div class="product-title">${product.name}</div>
                        <div class="product-price">${product.price} руб</div>
                        <button class="btn-buy" onclick="addToCart('${product.name}',
${product.price})">Добавить в корзину</button>
                    </div>
                `;
                productList.appendChild(div);
            });
        }
        function addToCart(name, price) {
    cart.push({ name, price });
    document.getElementById('cartCount').textContent = cart.length;  // Обновляем количество
товаров в корзине
```

```
    const cartItems = document.getElementById('cartItems');
    cartItems.innerHTML = '';
    cart.forEach((item, index) => {
        const li = document.createElement('li');
        li.innerHTML = `${item.name} - ${item.price} руб. <span class="remove-item"
onclick="removeFromCart(${index})">Удалить</span>`;
        cartItems.appendChild(li);
    });
}


        // Удаление товара из корзины
        function removeFromCart(index) {
            cart.splice(index, 1);
            displayCart();
        }

        // Отображение корзины
        function displayCart() {
            document.querySelector('.nav-link.btn').textContent = `Корзина (${cart.length})`;

            const cartItems = document.getElementById('cartItems');
            cartItems.innerHTML = '';
            cart.forEach((item, index) => {
                const li = document.createElement('li');
                li.innerHTML = `${item.name} - ${item.price} руб. <span class="remove-item"
onclick="removeFromCart(${index})">Удалить</span>`;
                cartItems.appendChild(li);
            });
        }

        // Фильтрация товаров по категории и цене
        function filterProducts() {
            const categoryFilter = document.getElementById('categoryFilter').value;
            const priceFilter = document.getElementById('priceFilter').value;

            let filteredProducts = products;

            // Фильтрация по категории
            if (categoryFilter) {
                filteredProducts = filteredProducts.filter(product => product.category ===
categoryFilter);
            }

            // Фильтрация по цене
            if (priceFilter === "1") {
                filteredProducts = filteredProducts.filter(product => product.price <= 1000);
            } else if (priceFilter === "2") {
                filteredProducts = filteredProducts.filter(product => product.price > 1000 &&
product.price <= 3000);
```

```
            } else if (priceFilter === "3") {
                filteredProducts = filteredProducts.filter(product => product.price > 3000);
            }

            displayProducts(filteredProducts);
        }

        // Обработка оплаты
        function proceedToPayment() {
            let total = cart.reduce((sum, item) => sum + item.price, 0);
            document.getElementById('totalPrice').textContent = total;
            const paymentModal = new bootstrap.Modal(document.getElementById('paymentModal'));
            paymentModal.show();
        }

        function payWithSBP() {
    alert("Вы выбрали оплату через СБП. Пожалуйста, следуйте инструкциям на экране.");
    savePurchasesToDB(); // Сохраняем покупки в БД
}

        function completePayment() {
            // Покупка завершена
            alert('Покупка завершена! Спасибо за покупку.');

            // Очистка корзины
            cart.length = 0;

            // Обновление отображения количества товаров в корзине и списка
            document.getElementById('cartCount').textContent = '0';
            const cartItems = document.getElementById('cartItems');
            cartItems.innerHTML = '<li>В корзине нет товаров</li>';

            // Закрытие модального окна оплаты
            const paymentModal =
bootstrap.Modal.getInstance(document.getElementById('paymentModal'));
            paymentModal.hide();

        }

        function groupCartItems(cart) {
    const grouped = {};

    cart.forEach((item) => {
        if (grouped[item.name]) {
            grouped[item.name].quantity += 1;
            grouped[item.name].totalPrice += item.price;
        } else {
            grouped[item.name] = {
                name: item.name,
                price: item.price,
```

```
                quantity: 1,
                totalPrice: item.price,
            };
        }
    });

    return Object.values(grouped); // Возвращаем массив сгруппированных объектов
}
    function savePurchasesToDB() {
        const groupedCart = groupCartItems(cart);

        fetch('/save-purchase', {
            method: 'POST',
            headers: {
                'Content-Type': 'application/json',
            },
            body: JSON.stringify(groupedCart), // Отправляем сгруппированные данные корзины
        })
        .then(response => {
            if (response.ok) {
                alert('Покупки успешно сохранены в базу данных!');
                completePayment(); // Завершаем покупку
            } else {
                alert('Ошибка при сохранении покупок.');
            }
        })
        .catch(error => {
            console.error('Ошибка:', error);
            alert('Ошибка при сохранении покупок.');
        });
    }
function proceedToPayment() {
    let total = cart.reduce((sum, item) => sum + item.price, 0);
    document.getElementById('totalPrice').textContent = total;

    const paymentModal = new bootstrap.Modal(document.getElementById('paymentModal'));
    paymentModal.show();
}

        // Слушатели изменений в фильтрах
        document.getElementById('categoryFilter').addEventListener('change', filterProducts);
        document.getElementById('priceFilter').addEventListener('change', filterProducts);

        // Инициализация
        displayProducts(products); // Отображаем все товары
        filterProducts(); // Применяем начальную фильтрацию товаров

        let currentRole = 'user'; // Начальная роль пользователя

// Переключение ролей
```

```
document.getElementById('roleToggleBtn').addEventListener('click', function () {
    // Переключаем между ролями
    if (currentRole === 'shop') {
        currentRole = 'user';
        alert('Вы вошли как пользователь');
    } else {
        currentRole = 'shop';
        alert('Вы вошли как продавец');
    }

    toggleRoleFeatures(); // Обновляем видимость кнопок
});

// Управление доступом к кнопке "Посмотреть купленные товары"
function toggleRoleFeatures() {
    const showPurchasesButton = document.getElementById('togglePurchasesButton');

    if (currentRole === 'shop') {
        showPurchasesButton.style.display = 'block';
    } else {
        showPurchasesButton.style.display = 'none';
}

// Логика кнопки "Посмотреть купленные товары"
const togglePurchasesButton = document.getElementById('togglePurchasesButton');
const purchasesContainer = document.getElementById('purchasesContainer');
const purchasesList = document.getElementById('purchasesList');

togglePurchasesButton.addEventListener('click', () => {
    if (currentRole !== 'shop') {
        alert('Только продавец может смотреть купленные товары!');
        return;
    }

    if (purchasesContainer.style.display === 'none') {
        purchasesContainer.style.display = 'block';
        togglePurchasesButton.textContent = 'Скрыть купленные товары';
        loadPurchases();
    } else {
        purchasesContainer.style.display = 'none';
        togglePurchasesButton.textContent = 'Посмотреть купленные товары';
    }
});

function loadPurchases() {
    fetch('/get-purchases', { method: 'GET' })
        .then(response => {
            if (!response.ok) {
                throw new Error('Ошибка при загрузке купленных товаров');
            }
```

```
            return response.json();
        })
        .then(data => {
            purchasesList.innerHTML = '';
            if (data.length === 0) {
                purchasesList.innerHTML = '<li>Нет купленных товаров</li>';
                return;
            }
            data.forEach(purchase => {
                const li = document.createElement('li');
                li.textContent = `${purchase.name} - ${purchase.quantity} шт. -
${purchase.total_price} руб. (${purchase.purchase_date})`;
                purchasesList.appendChild(li);
            });
        })
        .catch(error => {
            console.error('Ошибка:', error);
            purchasesList.innerHTML = '<li>Ошибка при загрузке купленных товаров</li>';
        });
}

// Инициализация
toggleRoleFeatures();

    </script>
</body>
</html>
```

Trainings.html

```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Тренировки</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/css/bootstrap.min.css" rel="stylesheet">
    <link
href="https://cdnjs.cloudflare.com/ajax/libs/fullcalendar/3.2.0/fullcalendar.min.css"
rel="stylesheet">
    <link href="css/styles.css" rel="stylesheet">
    <style>
        .training-card {
            background-color: #222;
            color: white;
            border-radius: 10px;
            margin-bottom: 20px;
            display: flex;
            flex-direction: column;
            justify-content: space-between;
```

```
        height: 350px;
    }

    .training-card img {
        width: 100%;
        height: 200px;
        object-fit: cover;
        border-bottom: 2px solid #FF4500;
    }

    .training-card h3 {
        margin: 10px;
        text-align: center;
    }

    .training-card .btn {
        background-color: #FF4500;
        color: white;
        border: none;
        width: 100%;
        padding: 10px;
        cursor: pointer;
    }

    .training-card .btn:hover {
        background-color: #FF0000;
    }

    .container {
        margin-top: 40px;
    }

    #calendar {
        max-width: 900px;
        margin: 0 auto;
    }
    .form-container {
        background-color: #f8f9fa;
        padding: 20px;
        border-radius: 10px;
        box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
        margin-top: 40px;
    }

    .form-group {
        margin-top: 20px;
    }

    .btn-register {
        background-color: #FF4500;
```

```
            color: white;
            width: 100%;
            padding: 10px;
            border: none;
            cursor: pointer;
            margin-top: 20px;
        }

        .btn-register:hover {
            background-color: #FF0000;
        }

        .fc-day-grid-event {
            background-color: #FF4500 !important;
            color: white;
            min-height: 50px;
            padding: 5px;
            font-size: 14px;
            line-height: 1.3;
        }
        .fc-title {
            font-size: 12px;
            white-space: normal;
            word-wrap: break-word;
            text-align: center;

        /* Модальное окно */
        .modal-body p {
            font-size: 18px;
        }
        /* Заголовок формы */
        .form-container h4 {
            color: black;
        }
    </style>
</head>
<body>

    <!-- Навигация -->
    <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
        <div class="container-fluid">
            <a class="navbar-brand" href="index.html">Спортивный клуб</a>
            <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Переключить
навигацию">
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbarNav">
                <ul class="navbar-nav ms-auto">
                    <li class="nav-item">
```

```
                    <a class="nav-link" href="index.html">Главная</a>
                </li>
                <li class="nav-item">
                    <a class="nav-link" href="trainings.html">Тренировки</a>
                </li>
                <li class="nav-item">
                    <a class="nav-link" href="coaches.html">Тренера</a>
                </li>
                <li class="nav-item">
                    <a class="nav-link" href="membership.html">Абонементы</a>
                </li>
                <li class="nav-item">
                    <a class="nav-link" href="shop.html">Магазин</a>
                </li>
                <li class="nav-item">
                    <a class="nav-link" href="reviews.html">Отзывы</a>
                </li>
                <li class="nav-item">
                    <a class="nav-link" href="rental.html">Аренда залов и дополнительные
услуги</a>
                </li>

                <li class="nav-item">
                    <a class="nav-link" href="profile.html">Профиль</a>
                </li>

            </ul>
        </div>
    </div>
</nav>
<button id="roleToggleBtn" class="btn btn-secondary mb-3">Сменить роль</button>
<div class="container">
    <h2>Наши тренировки</h2>


    <div class="row">
        <!-- Карточка тренировки: Йога -->
        <div class="col-md-4">
            <div class="training-card">
                <img src="img/joga.jpg" alt="Йога">

                <h3>Йога</h3>
                <a href="#training-form" class="btn">Записаться на тренировку</a>
            </div>
        </div>
        <!-- Карточка тренировки: Кардио -->
        <div class="col-md-4">
            <div class="training-card">
                <img src="img/kardio.jpg" alt="Кардио">
                <h3>Кардио</h3>
```

```
                <a href="#training-form" class="btn">Записаться на тренировку</a>
            </div>
        </div>
        <!-- Карточка тренировки: Плавание -->
        <div class="col-md-4">
            <div class="training-card">
                <img src="img/plavanie.png" alt="Плавание">
                <h3>Плавание</h3>
                <a href="#training-form" class="btn">Записаться на тренировку</a>
            </div>
        </div>
        <!-- Карточка тренировки: Силовые -->
        <div class="col-md-4">
            <div class="training-card">
                <img src="img/silovye.jpg" alt="Силовые">
                <h3>Силовые тренировки</h3>
                <a href="#training-form" class="btn">Записаться на тренировку</a>
            </div>
        </div>
        <!-- Карточка тренировки: Фитнес -->
        <div class="col-md-4">
            <div class="training-card">
                <img src="img/fitnes.jpg" alt="Фитнес">
                <h3>Фитнес</h3>
                <a href="#training-form" class="btn">Записаться на тренировку</a>
            </div>
        </div>
        <!-- Карточка тренировки: Функциональные -->
        <div class="col-md-4">
            <div class="training-card">
                <img src="img/funktsionalnye.jpg" alt="Функциональные тренировки">
                <h3>Функциональные тренировки</h3>
                <a href="#training-form" class="btn">Записаться на тренировку</a>
            </div>
        </div>
    </div>
    <button id="showTrainingsBtn" class="btn btn-primary mb-3">Показать
тренировки</button>
    <div id="trainingsListContainer" style="display: none;"></div>

    <h3>Расписание тренировок </h3>
    <div id="calendar"></div>

    <!-- Форма для записи на тренировку -->
    <div class="form-container" id="training-form">

        <h4>Записаться на тренировки</h4>
        <form action="/submit_registration" method="POST">
            <div class="form-group">
                <label for="fullName">Ф.И.О.</label>
```

```
                    <input type="text" class="form-control" id="fullName" name="fullName"
placeholder="Введите ваше имя" required>
                </div>

                <div class="form-group">
                    <label for="phone">Телефон</label>
                    <input type="text" class="form-control" id="phone" name="phone"
placeholder="Введите ваш номер" required>
                </div>

                <div class="form-group">
                    <label for="trainingCategory">Выберите категорию тренировки</label>
                    <select class="form-control" id="trainingCategory" name="trainingCategory"
required>
                        <option value="cardio">Кардио</option>
                        <option value="strength">Силовые</option>
                        <option value="yoga">Йога</option>
                        <option value="functional">Функциональные</option>
                        <option value="swimming">Плавание</option>
                        <option value="fitness">Фитнес</option>
                    </select>
                </div>

                <div class="form-group">
                    <label for="trainingType">Тип тренировки</label>
                    <select class="form-control" id="trainingType" name="trainingType"
required>
                        <option value="group">Групповая</option>
                        <option value="individual">Индивидуальная</option>
                    </select>
                </div>

                <div class="form-group">
                    <label for="trainingDay">Выберите день тренировки</label>
                    <select class="form-control" id="trainingDay" name="trainingDay" required>

                    </select>
                </div>

                <button type="submit" class="btn-register">Записаться</button>
            </form>
        </div>
    </div>

    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/moment.js/2.29.1/moment.min.js"></script>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/fullcalendar/3.2.0/fullcalendar.min.js"></script>
```

```
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/js/bootstrap.bundle.min.js"></script>
    <script>
        $(document).ready(function() {
            $('#calendar').fullCalendar({
                firstDay: 1,
                events: [
                    {
                        title: 'Йога 19:00-20:00',
                        dow: [2, 4, 0], // Вторник, Четверг, Воскресенье
                    },
                    {
                        title: 'Кардио 21:00-21:30',
                        dow: [1, 3, 5], // Понедельник, Среда, Пятница
                    },
                    {
                        title: 'Силовые тренировки 20:00-22:00',
                        dow: [0, 3, 5], // Воскресенье, Среда, Пятница
                    },
                    {
                        title: 'Фитнес 21:00-22:30',
                        dow: [1, 2, 4], // Понедельник, Вторник, Четверг
                    },
                    {
                        title: 'Плавание 20:00-22:00',
                        dow: [1, 2, 4, 0], // Понедельник, Вторник, Четверг, Воскресенье
                    },
                    {
                        title: 'Функциональные тренировки 21:00-22:00',
                        dow: [0, 1, 2, 3, 4, 5, 6], // Все дни недели
                    }
                ],
                locale: 'ru',
                header: {
                    left: 'prev,next today',
                    center: 'title',
                },
            });
                const daysMap = {
                cardio: [1, 3, 5], // Понедельник, Среда, Пятница
                strength: [0, 3, 5], // Воскресенье, Среда, Пятница
                yoga: [2, 4, 0], // Вторник, Четверг, Воскресенье
                functional: [0, 1, 2, 3, 4, 5, 6], // Все дни недели
                swimming: [1, 2, 4, 0], // Понедельник, Вторник, Четверг, Воскресенье
                fitness: [1, 2, 4], // Понедельник, Вторник, Четверг
            };

            $('#trainingCategory').on('change', function() {
                let selectedCategory = $(this).val();
                let availableDays = daysMap[selectedCategory];
```

```
                console.log('Выбрана категория тренировки:', selectedCategory);
                console.log('Доступные дни:', availableDays);

                let dayOptions = ["Понедельник", "Вторник", "Среда", "Четверг", "Пятница",
"Суббота", "Воскресенье"];

                $('#trainingDay').empty();

                if (availableDays && availableDays.length > 0) {
                    availableDays.forEach(function(dayIndex) {
                        $('#trainingDay').append(`<option
value="${dayIndex}">${dayOptions[dayIndex]}</option>`);
                    });
                } else {

                    $('#trainingDay').append('<option disabled>Нет доступных дней</option>');
                }
            });


            $('#trainingCategory').trigger('change');
        });

                $('#trainingForm').on('submit', function(e) {
            e.preventDefault
            let formData = $(this).serialize

            $.ajax({
                url: '/register-training',
                type: 'POST',
                data: formData,
                success: function(response) {
                    alert(response);
                error: function() {
                    alert('Ошибка при записи на тренировку.');
                }
            });
        });
        document.getElementById('showTrainingsBtn').addEventListener('click', function() {
            fetch('/api/trainings')
                .then(response => response.json())
                .then(trainings => {
                    const container = document.getElementById('trainingsListContainer');
                    container.innerHTML = '';  // Очистка контейнера

                    const dayNames = ["Воскресенье", "Понедельник", "Вторник", "Среда",
"Четверг", "Пятница", "Суббота"];

                    trainings.forEach(training => {
```

```
                    const trainingElement = document.createElement('div');
                    trainingElement.classList.add('card', 'mb-3');
                    trainingElement.innerHTML = `
                        <div class="card-body">
                            <h5 class="card-title">Ф.И.О.: ${training.full_name}</h5>
                            <p class="card-text">Телефон: ${training.phone}</p>
                            <p class="card-text">Категория:
${training.training_category}</p>
                            <p class="card-text">Тип тренировки:
${training.training_type}</p>
                            <p class="card-text">День:
${dayNames[training.training_day]}</p>
                            <p class="card-text">Дата записи: ${new
Date(training.created_at).toLocaleDateString()}</p>
                            <button class="btn btn-danger delete-btn" data-
id="${training.id}">Удалить</button>
                            <button class="btn btn-warning update-btn" data-
id="${training.id}">Обновить</button>
                        </div>
                    `;
                    container.appendChild(trainingElement);
                });

                document.querySelectorAll('.delete-btn').forEach(button => {
                    button.addEventListener('click', function() {
                        const trainingId = this.getAttribute('data-id');
                        deleteTraining(trainingId);
                    });
                });

                document.querySelectorAll('.update-btn').forEach(button => {
                    button.addEventListener('click', function() {
                        const trainingId = this.getAttribute('data-id');
                        updateTraining(trainingId);
                    });
                });
            });
        });
        function deleteTraining(trainingId) {
            fetch(`/api/trainings/${trainingId}`, { method: 'DELETE' })
                .then(response => response.json())
                .then(data => {
                    alert('Тренировка удалена');
                    document.getElementById('showTrainingsBtn').click();
                })
                .catch(error => console.error('Ошибка удаления тренировки:', error));
        }

        // Обновление тренировки
        function updateTraining(trainingId) {
```

```
    const newCategory = prompt("Введите новую категорию тренировки:");
    const newDay = prompt("Введите новый номер дня недели (0 - воскресенье, 6 - суббота):");

    if (!newCategory || !newDay) {
        alert("Категория и день тренировки не могут быть пустыми.");
        return;
    }

    fetch(`/api/trainings/${trainingId}`, {
        method: 'PUT',
        headers: {
            'Content-Type': 'application/json',
        },
        body: JSON.stringify({
            training_category: newCategory,
            training_day: parseInt(newDay, 10),
        }),
    })
        .then((response) => {
            if (!response.ok) {
                throw new Error('Ошибка обновления тренировки');
            }
            return response.json();
        })
        .then((data) => {
            alert(data.message || 'Тренировка успешно обновлена');
            document.getElementById('showTrainingsBtn').click();
        .catch((error) => {
            console.error('Ошибка обновления тренировки:', error);
            alert('Не удалось обновить тренировку.');
        });
}
let isAdmin = false;

document.getElementById('roleToggleBtn').addEventListener('click', function() {
    isAdmin = !isAdmin;
    alert(isAdmin ? 'Вы вошли как администратор' : 'Вы вошли как пользователь');
    toggleAdminFeatures();
});

    function toggleAdminFeatures() {
        const showTrainingsBtn = document.getElementById('showTrainingsBtn');
        showTrainingsBtn.style.display = isAdmin ? 'block' : 'none';
    document.getElementById('showTrainingsBtn').addEventListener('click', function() {
        const container = document.getElementById('trainingsListContainer');
        if (container.style.display === 'none' || container.style.display === '') {
            container.style.display = 'block';
            this.textContent = 'Скрыть тренировки';
        } else {
            container.style.display = 'none'
```

```
            this.textContent = 'Показать тренировки';
        }
    });

toggleAdminFeatures();
    </script>

</body>
</html>
```

Server.js

```
const express = require('express');
const mysql = require('mysql2');
const bodyParser = require('body-parser');
const path = require('path');
const app = express();
const port = 3000;
const db = mysql.createConnection({
    host: 'localhost',
    user: 'root',
    password: 'root',
    database: 'sportsclub'
});

db.connect((err) => {
    if (err) {
        console.error('Ошибка подключения к базе данных: ', err);
        return;
    }
    console.log('Подключено к базе данных');
});

app.use(bodyParser.urlencoded({ extended: false }));
app.use(bodyParser.json());

app.use(express.static(path.join(__dirname)));


app.get('/', (req, res) => {
    res.sendFile(path.join(__dirname, 'index.html'));

app.post('/submit_registration', (req, res) => {
    const { fullName, phone, trainingCategory, trainingType, trainingDay } = req.body;

    db.query('CALL register_for_training(?, ?, ?, ?, ?)',
        [fullName, phone, trainingCategory, trainingType, trainingDay],
        (err, results) => {
            if (err) {
                console.error('Ошибка при регистрации на тренировку, Проверьте введённые
данные: ', err);
```

```
                res.status(500).send('Ошибка при регистрации на тренировку, Проверьте
введённые данные');
                return;
            }
            res.send('Запись на тренировку успешна!');
        }
    );
});

app.get('/api/trainings', (req, res) => {
    const query = 'SELECT id, full_name, phone, training_category, training_type,
training_day, created_at FROM training_registrations';
    db.query(query, (err, results) => {
        if (err) {
            console.error('Ошибка получения тренировок:', err);
            res.status(500).json({ error: 'Ошибка получения тренировок' });
            return;
        }
        res.json(results);
    });
});

app.delete('/api/trainings/:id', (req, res) => {
    const { id } = req.params;
    const query = 'DELETE FROM training_registrations WHERE id = ?';
    db.query(query, [id], (err, results) => {
        if (err) {
            res.status(500).json({ error: 'Ошибка удаления тренировки' });
            return;
        }
        res.json({ message: 'Тренировка удалена' });
    });
});

app.put('/api/trainings/:id', (req, res) => {
    const { id } = req.params;
    const { training_day, training_category } = req.body

    if (!training_day || !training_category) {
        return res.status(400).json({ error: 'Необходимо указать training_day и
training_category' });
    }

    const query = 'UPDATE training_registrations SET training_day = ?, training_category = ?
WHERE id = ?';
    db.query(query, [training_day, training_category, id], (err, result) => {
        if (err) {
            console.error('Ошибка обновления тренировки:', err);
            return res.status(500).json({ error: 'Ошибка обновления тренировки' });
        }
```

```
        if (result.affectedRows === 0) {
            return res.status(404).json({ error: 'Тренировка с указанным ID не найдена' });
        }

        res.json({ message: 'Тренировка успешно обновлена' });
    });
});



app.post('/submit_booking', (req, res) => {
    const { fullName, phone, coach, date, time } = req.body;

    const trimmedCoachName = coach.trim();

    db.query('CALL book_individual_training(?, ?, ?, ?, ?)',
        [fullName, phone, trimmedCoachName, date, time],
        (err, results) => {
            if (err) {
                console.error('Ошибка при бронировании занятия.Выберите другое время для
записи: ', err);
                res.status(500).send('Ошибка при бронировании занятия.Выберите другое время
для записи');
                return;
            }
            res.status(200).send('Вы успешно записались на занятие!');
        }
    );
});

app.get('/trainings/:coach', (req, res) => {
    const coach = req.params.coach;

    const query = `SELECT training_date, training_time, coach FROM individual_bookings WHERE
coach = ?`;

    db.execute(query, [coach], (err, results) => {
      if (err) {
        return res.status(500).json({ error: 'Ошибка при получении данных' });
      }

      res.json(results);
    });
  });



//АБОНЕМЕНТЫ
```

```
app.post('/purchase', (req, res) => {
    const { name, price } = req.body;
    const purchaseDate = new Date();

    const query = 'INSERT INTO buy_sub (name, price, purchase_date) VALUES (?, ?, ?)';
    db.query(query, [name, price, purchaseDate], (err, result) => {
        if (err) {
            console.error('Ошибка при записи в базу данных:', err);
            return res.status(500).send('Ошибка при записи в базу данных');
        }
        res.status(200).send('Покупка успешно добавлена');
    });
});
app.get('/get-purchased-subscriptions', (req, res) => {
    const query = 'SELECT * FROM buy_sub';

    db.query(query, (err, results) => {
        if (err) {
            console.error('Ошибка при извлечении данных:', err);
            return res.status(500).send('Ошибка при извлечении данных');
        }
        res.json(results);
    });
});


//МАГАЗ
// Маршрут для записи покупок
app.post('/save-purchase', (req, res) => {
    const purchases = req.body;

    if (!Array.isArray(purchases) || purchases.length === 0) {
        return res.status(400).send('Нет данных для записи');
    }

    const sql = `
        INSERT INTO purchases (name, price, quantity, total_price, purchase_date)
        VALUES (?, ?, ?, ?, NOW())
    `;

    purchases.forEach((item) => {
        db.query(sql, [item.name, item.price, item.quantity, item.totalPrice], (err) => {
            if (err) {
                console.error('Ошибка при добавлении данных:', err);
                return res.status(500).send('Ошибка при добавлении данных');
            }
        });
    });

    res.status(200).send('Покупки успешно сохранены');
```

```
});

app.get('/get-purchases', (req, res) => {
    const sql = `
        SELECT name, price, quantity, total_price,
        DATE_FORMAT(purchase_date, '%Y-%m-%d %H:%i:%s') AS purchase_date
        FROM purchases
        ORDER BY purchase_date DESC
    `;

    db.query(sql, (err, results) => {
        if (err) {
            console.error('Ошибка при получении данных:', err);
            return res.status(500).send('Ошибка при получении данных');
        }

        res.json(results);
    });
});

//АРЕНДА
app.post('/submit_rental', (req, res) => {
    console.log(req.body);  // Log the request body
    const { item, rental_date, return_date, phone } = req.body;

    const query = 'INSERT INTO rentals (item, rental_date, return_date, phone) VALUES (?, ?,
?, ?)';
    db.query(query, [item, rental_date, return_date, phone], (err, result) => {
        if (err) {
            console.error('Ошибка при записи в базу данных:', err);
            return res.status(500).json({ success: false, message: 'Ошибка при записи в базу
данных' });
        }
        console.log('Data inserted:', result);
        res.status(200).json({ success: true, message: 'Данные успешно сохранены' });
    });
});
app.get('/rentals', (req, res) => {
    const query = 'SELECT * FROM rentals';
    db.query(query, (err, results) => {
        if (err) {
            console.error('Ошибка при выполнении запроса: ' + err.stack);
            res.status(500).json({ error: 'Не удалось получить данные' });
        } else {
            res.json(results);
        }
    });
});
```

```
app.get('/reviews', (req, res) => {
    const query = 'SELECT * FROM reviews ORDER BY review_date DESC';
    db.query(query, (err, result) => {
        if (err) {
            console.error('Ошибка при получении отзывов: ' + err.stack);
            return res.status(500).send('Ошибка сервера');
        }
        res.json(result);
    });
});

app.post('/reviews', (req, res) => {
    const { reviewerName, reviewRating, reviewText } = req.body;
    const reviewDate = new Date().toISOString().slice(0, 19).replace('T', ' '

    const query = 'INSERT INTO reviews (reviewer_name, review_rating, review_text,
review_date) VALUES (?, ?, ?, ?)';
    db.query(query, [reviewerName, reviewRating, reviewText, reviewDate], (err, result) => {
        if (err) {
            console.error('Ошибка при добавлении отзыва: ' + err.stack);
            return res.status(500).send('Ошибка сервера');
        }
        res.status(200).send('Отзыв успешно добавлен');
    });
});




app.get('/api/users', (req, res) => {
    db.query('SELECT * FROM users', (err, result) => {
        if (err) {
            return res.status(500).json({ error: 'Failed to retrieve users' });
        }
        res.json({ users: result });
    });
});

app.get('/api/user/:id', (req, res) => {
    const userId = req.params.id;
    db.query('SELECT * FROM users WHERE id = ?', [userId], (err, result) => {
        if (err) {
            return res.status(500).json({ error: 'Failed to retrieve user data' });
        }
        if (result.length === 0) {
            return res.status(404).json({ error: 'User not found' });
        }
        res.json({ user: result[0] });
    });
});
```

```
app.get('/api/user/:name', (req, res) => {
    const userName = req.params.name;
    const query = 'SELECT * FROM users WHERE name = ?';
    db.query(query, [userName], (err, results) => {
        if (err) {
            console.error('Ошибка при получении данных пользователя: ', err);
            return res.status(500).json({ error: 'Ошибка при получении данных пользователя'
});
        }

        if (results.length === 0) {
            return res.status(404).json({ error: 'Пользователь не найден' });
        }

        const user = results[0];
        res.json({
            user: {
                name: user.name,
                age: user.age,
                city: user.city,
                activity: user.activity
            }
        });
    });
});

app.post('/api/addUser', (req, res) => {
    const { name, age, city, activity } = req.body;

    const query = `INSERT INTO users (name, age, city, activity) VALUES (?, ?, ?, ?)`;

    db.query(query, [name, age, city, activity], (err, result) => {
        if (err) {
            console.error('Ошибка при добавлении пользователя: ', err);
            return res.status(500).send('Ошибка при добавлении пользователя');
        }
        res.json({ success: true, message: 'Пользователь успешно добавлен!' });
    });
});

app.put('/api/user/:id', (req, res) => {
    const userId = req.params.id;
    const { name, age, city, activity } = req.body;

    console.log(`Обновление пользователя: ${userId}, ${name}, ${age}, ${city}, ${activity}`);

    const query = `UPDATE users SET name = ?, age = ?, city = ?, activity = ? WHERE id = ?`;
```

```
    db.query(query, [name, age, city, activity, userId], (err, result) => {
        if (err) {
            console.error('Ошибка при обновлении пользователя: ', err);
            return res.status(500).json({ error: 'Ошибка при обновлении пользователя' });
        }

        if (result.affectedRows === 0) {
            return res.status(404).json({ error: 'Пользователь не найден' });
        }

        res.json({ success: true, message: 'Пользователь успешно обновлен!' });
    });
});

app.delete('/api/user/:id', (req, res) => {
    const userId = req.params.id;

    const query = `DELETE FROM users WHERE id = ?`;

    db.query(query, [userId], (err, result) => {
        if (err) {
            console.error('Ошибка при удалении пользователя: ', err);
            return res.status(500).json({ error: 'Ошибка при удалении пользователя' });
        }

        if (result.affectedRows === 0) {
            return res.status(404).json({ error: 'Пользователь не найден' });
        }

        res.json({ success: true, message: 'Пользователь успешно удален!' });
    });
});

// Запуск сервера
app.listen(port, () => {
    console.log(`Сервер работает на http://localhost:${port}`);
});
```