**Национальный исследовательский ядерный университет «МИФИ»**

|  | **Институт интеллектуальных и кибернетических систем**  **Кафедра «Компьютерные системы и технологии»** |
| --- | --- |

Пояснительная записка

к проекту по курсу «Программирование

сетевых приложений»

на тему «Разработка веб-приложения для сети фитнес-центров»

Студент гр. Б20-513 Кучебо А.В /\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /

Студент гр. Б20-513 Федотова В.А /\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /

Руководитель Овчаренко Е. С. / \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /

Москва, 2023г.

АННОТАЦИЯ

Пояснительная записка 22 с., 13 рис., 2 табл., 13 ист.

Объектом разработки является приложение для фитнес-центров.

Целью работы является разработка серверной и клиентской части приложения, с помощью которого можно сгенерировать план питания, план тренировок и найти напарника для посещения.

В рамках анализа предметной области были рассмотрены приложения из сферы фитнес-центров.

Было произведено ручное и unit-тестирование проекта.

Серверная часть приложения реализована на языке программирования Golang. В качестве клиентской части выступает мобильное приложение, написанное на языке программирования JavaScript. Для работы с базой данных была выбрана СУБД PostgreSQL, который работает в Docker-контейнере.

Содержание

[**Введение** 4](#_heading=h.gjdgxs)

[**1.**](#_heading=h.30j0zll) **Теоретическая часть** 5

[**1.1.**](#_heading=h.1fob9te) **Краткий анализ существующих решений** 5

[**1.2.**](#_heading=h.3znysh7) **Постановка задачи** 5

[**1.3.**](#_heading=h.2et92p0) **Схема приложения** 6

[**1.4.**](#_heading=h.tyjcwt) **Формализация данных** 6

[**2.**](#_heading=h.sn4nvl1fo5vt) **Практическая часть** 8

[**2.1.**](#_heading=h.3dy6vkm) **Используемые технологии** 8

[**2.2.**](#_heading=h.1t3h5sf) **Описание таблиц базы данных** 9

[**2.3.**](#_heading=h.17dp8vu) **Уровень взаимодействия** 10

[**2.3.1.**](#_heading=h.3rdcrjn) **Пути для хендлеров** 11

[**2.3.2.**](#_heading=h.1ijnr6j75pdr) **Аутентификация пользователей** 12

[**2.4.**](#_heading=h.2jxsxqh) **Веб-приложение** 13

[**3.**](#_heading=h.4i7ojhp) **Тестирование** 20

[**Заключение**](#_heading=h.3as4poj) 21

[**Список используемых источников**](#_heading=h.1pxezwc) 22

# **Введение**

Современный образ жизни все чаще заставляет нас обратить внимание на заботу о здоровье и физической форме. В связи с этим веб-приложения для сети фитнес-центров становятся все более популярными, предоставляя пользователям возможность легкого доступа к информации о занятиях, питании и поиске единомышленников для совместных посещений.

Разработка веб-приложения, объединяющего функционал планирования питания, тренировок и поиска партнеров для посещения фитнес-центров, представляет собой актуальную задачу, направленную на удовлетворение потребностей современных любителей здорового образа жизни. Данное приложение поможет пользователям создавать персонализированные планы питания и тренировок, искать единомышленников для совместных посещений тренировок, а также получать доступ к полезной информации о своем здоровье и физической активности.

В данном работе рассмотрим основные точки, которые следует учесть при разработке такого веб-приложения, а также преимущества, которые оно предоставляет как администраторам фитнес-центров, так и конечным пользователям.

# **Теоретическая часть**

# **Краткий анализ существующих решений**

В настоящее время существуют большое количество сервисов для фитнесс-центров, к примеру «DDX» [1] или «XFIT» [2]. Однако они нацелены в основном на предоставление краткой информации об организации и на возможность быстрой оплаты членства в фитнесс-клубе.

Обычно они представляют собой сайт, но некоторые организции предоставляют пользователям мобильное приложение, в общем случае в которых есть следующие функции:

* возможность просмотра информации о подписке, тренерах, своей активности;
* возможность записи на групповые занятия;
* возможность управлением подпиской.

Из минусов сайта и приложения можно выделить:

* отсутствие возможности смены подписки.

# **Постановка задачи**

Исходя из полученных результатов были выделены следующие требования к сайту, которые позволяют расширить его функционал:

* возможность генерации планов питания;
* возможность генерации планов тренировок;
* возможность поиска напарников для совместных тренировок;
* В качестве клиентской части должен выступать сайт

Кроме вышеперечисленных функций на сайте также должна быть реализована аутентификация и авторизация пользователя.

# **Схема приложения**

Клиент-серверное приложение состоит из нескольких связанных частей. Схема взаимодействия между частями приложения представлена на рисунке 1.



Рисунок 1. Схема взаимодействия между частями приложения

Пользователь через сайт может совершать действия, которые будут обрабатываться сервером и сохраняться в базе данных.

# **Формализация данных**

База данных должна хранить данные о следующих сущностях, представленных в таблице 1.

Таблица 1. Сущности базы данных

| **Сущности** | **Сведения** |
| --- | --- |
| Пользователь | Имя, Фамилия, Отчество, Телефон, Email, Аватарка, Время создания профиля, Хешированный пароль |
| Контакты | Опциональная ссылка, Телеграмм, Инстаграмм (запрещен на территории РФ), ВК |
| План тренировок | План на день, Время создания плана на день, План на неделю. Время создания плана на неделю |
| Список плана тренировок | Связь с планом тренировок и пользователем |
| План питания | План на день, Время создания плана на день, План на неделю. Время создания плана на неделю |
| Список плана питания | Связь с планом питания и пользователем |
| Тренер | Имя, Фамилия, Отчество, Телефон, Email, Аватарка, Время создания профиля |
| Групповые тренировки | Название, Описание |
| Список тренер-групповые тренировки | Связь с групповыми тренировками и тренером |
| Заявка | Цель, сообщение, время создания |
| Заявка-пользователь | Связь заявки и пользователя |
| Заявка-групповые тренировки | Связь групповых тренировок и заявки |
| Цель | Тип цели |
| Пол | Пол |
| Фитнесс-центр | Название, Адрес, Телефон |
| Список фитнесс-центров | Связь заявки и фитнесс-центров |
| Время дня | Время дня |
| Статус | Статус заявки |

# **Практическая часть**

# **Используемые технологии**

В качестве языка программирования для разработки серверной части приложения был выбран язык программирования Golang [3], поскольку он создан для написания бекенда для микросервисов, а также его можно использовать и в монолитных приложениях. Данный язык упрощает работу с написанием серверной части, является простым и гибким.

В качестве базы данных используется PostgreSQL [4], как самая распространённая, надежная и быстрая СУБД с открытым исходным кодом.

Для развертывания приложения был выбран способ с использованием метода виртуализации с помощью Docker [5] – для развертывания отдельного сервиса (к примеру, базы данных) и Docker-compose [6] – для развертывания нескольких сервисов.

Для разработки пользовательского интерфейса было принято использовать языки HTML [7], CSS [8] и JavaScript [9]. Эти технологии позволяют создавать веб-страницы с помощью относительно небольшого объема кода и мощных инструментов. HTML используется для описания структуры контента страницы, CSS — для оформления и стилизации элементов, а JavaScript обеспечивает динамическое поведение страницы, интерактивность и обработку событий, что позволяет создавать более динамичные пользовательские интерфейсы и улучшать пользовательский опыт.

# **Описание таблиц базы данных**

Чтобы организовать хранение информации о пользователях, группах и питомцах используется база данных PostgreSQL. Структура базы данных представлена ниже.

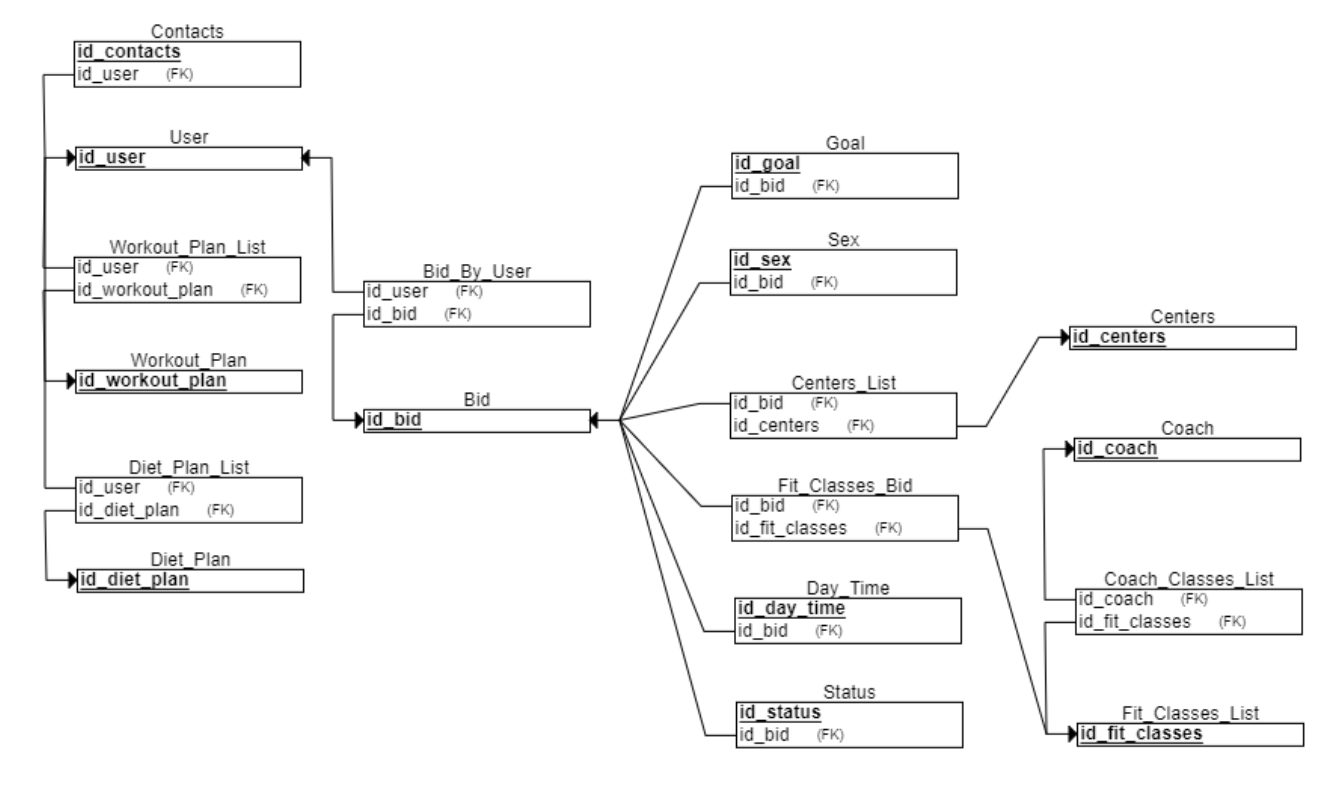


Рисунок 2. Структура базы данных

Сущность users – в ней хранятся id, имя, фамилия, отчество, номер телефона, email, ссылка на аватарку, время создания профиля и хешированный пароль пользователя.

Сущность contacts – в ней хранятся id, опциональная ссылка, ссылка на телеграмм, ссылка на инстаграмм (запрещен на территории РФ), и ссылка на ВК.

Сущность workout\_plan – в ней хранится id плана тренировок, план на день, время создания плана на день, план на неделю и время создания плана на неделю.

Сущность workout\_plan\_list – в ней хранится связь с планом тренировок и пользователем.

Сущность diet\_plan – в ней хранится id плана питания, план на день, время создания плана на день, план на неделю. время создания плана на неделю.

Сущность diet\_plan\_list – в ней хранится связь с планом питания и пользователем.

Сущность coach – в ней хранится id тренера, имя, фамилия, отчество, номер телефона, email, ссылка на аватарку и время создания профиля.

Сущность fit\_classes – в ней хранится id групповой тренировки, название и описание.

Сущность coach\_classes\_list – в ней хранится связь с групповыми тренировками и тренером.

Сущность bid – в ней хранится id заявки, цель, сообщение, время создания.

Сущность bid\_by\_user– в ней хранится связь заявки и пользователя.

Сущность fit\_classes\_bid – в ней хранится связь заявки и групповых тренировок.

Сущность goal – в ней хранится id цели, тип цели.

Сущность sex – в ней хранится id пола, тип цели..

Сущность centers – в ней хранится id центра, название, адрес, телефон.

Сущность centers\_list – в ней хранится связь заявки и фитнесс-центров.

Сущность day\_time – в ней хранится id времени дня, тип.

Сущность status – в ней хранится id статуса, тип статуса..

# **Уровень взаимодействия**

На этом уровне определяется способ взаимодействия сервиса с остальными пользователями сети. Было принято решение выбрать REST (Representation State Transfer) [10] – это набор правил и ограничений для проектирования сервиса, чтобы все системы легко обменивались данными и приложение можно было масштабировать.

Чтобы пользователи сети посылали запрос на сервис, нужны конечные точки, по запросу на которые сервис будет отдавать ответ. Для этой цели были созданы хендлеры, которые управляют запросами приходящие на сервер.

# **Пути для хендлеров**

Информацию о путях можно увидеть в таблице ниже.

Таблица 2. Конечных точек контроллера пользователей

| URL запроса | Функция конечной точки |
| --- | --- |
| /user/login | Аутентификация пользователя |
| /user/signup | Регистрация нового пользователя |
| /user/reset-password | Сброс пароля |
| /user/reset-email | Сброс почты |
| /user/logout | Выход пользователя |
| /status | Статус приложения (для дебага) |
| /about | О компании |
| /contacts | Контакты компании |
| /trainers | Список тренеров |
| /user/account | Аккаунт пользователя |
| /home | Домашняя страница |

# **2.3.2. Аутентификация пользователей**

В сервисе предусмотрена аутентификация пользователей и последующее взаимодействие используя JWT (JSON Web Token) [11]. Для этого был создан отдельный хендлер, который позволяет пользователю выполнять вход в аккаунт и выходить из него. После того, как пользователь посылает POST запрос на вход /user/login с логином и паролем в качестве параметров, генерируется JWT и отдаётся пользователю ответом. Пользователь должен использовать этот токен как поле Authentication в заголовке последующих запросов. После аутентификации пользователя, в контроллере возникает возможность получать id пользователя. Это преимущество даёт нам возможность не отправлять id пользователя при каждом запросе на сервер. Если пользователь захотел прекратить работу, он может выйти из аккаунта, послав POST запрос /user/logout. После этого токен будет считаться устаревшим. Получить доступ к сервису без аутентификации также не получится.

# **Веб-приложение**

Веб-приложение представляет собой многостраничный веб-сайт, который содержит основные страницы такие, как: информация о сети фитнес центров, информация о тренерах, информация о тарифах, контакты и личный кабинет.

Каждый посетитель сайта может просмотреть различную информацию о фитнес-центре FitZone. У пользователей есть возможность создать аккаунт, в котором представлена информация: ФИО, почта, номер телефона, поиска напарника, план питания и план тренировок. Чтобы зарегистрироваться нужно ввести почту и придумать пароль (а также повторить пароль, чтобы исключить человеческий фактор ошибки при вводе). Ниже представлена use-case диаграмма для аутентифицированных и не аутентифицированных пользователей.

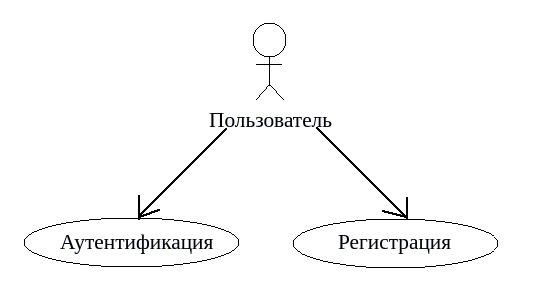


Рисунок 3. Use-case диаграмма пользователя, не вошедшего в аккаунт

На рисунках 4-5 представлен интерфейс поля аутентификации и регистрации.

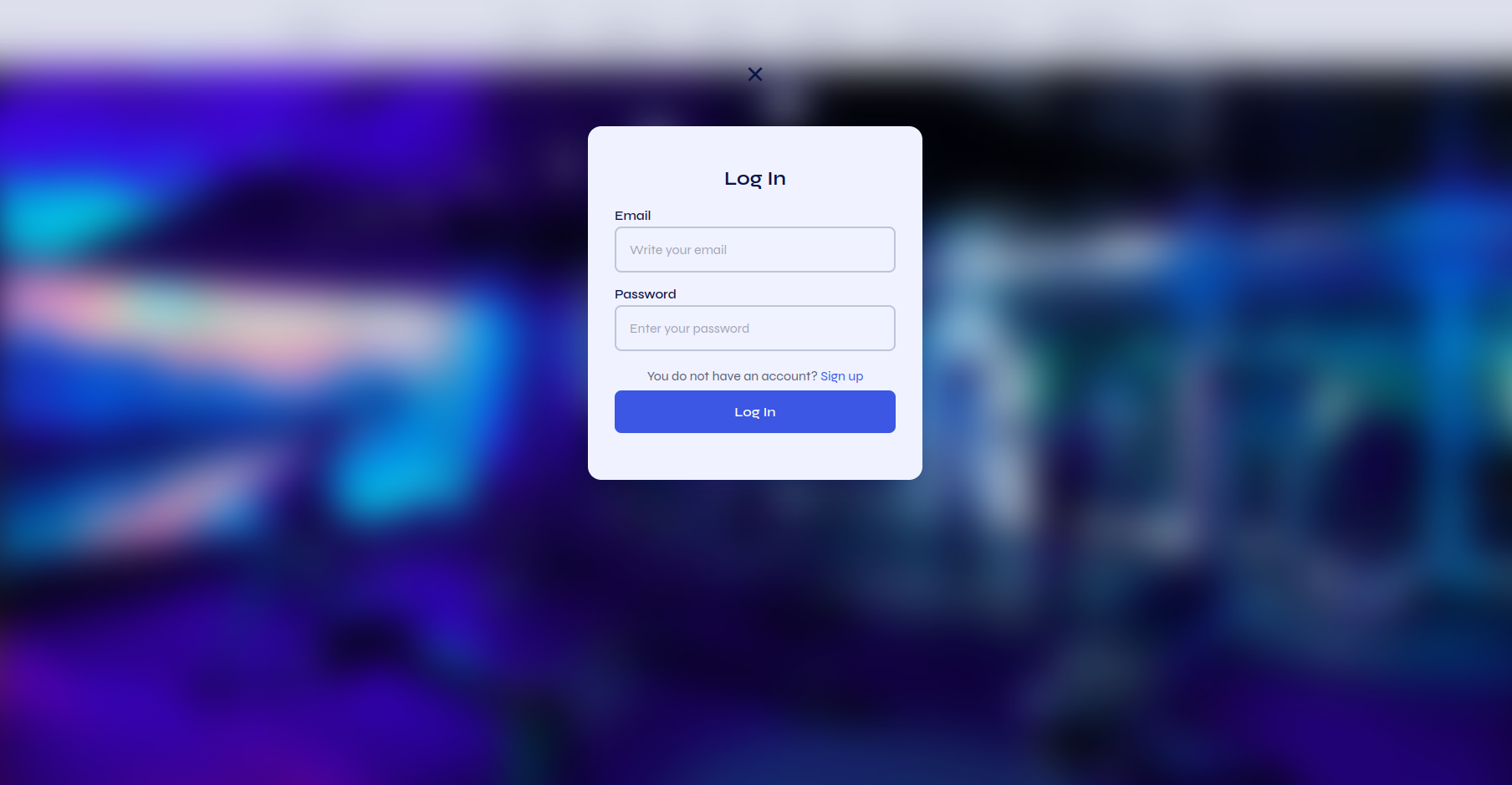


Рисунок 4. Экран входа

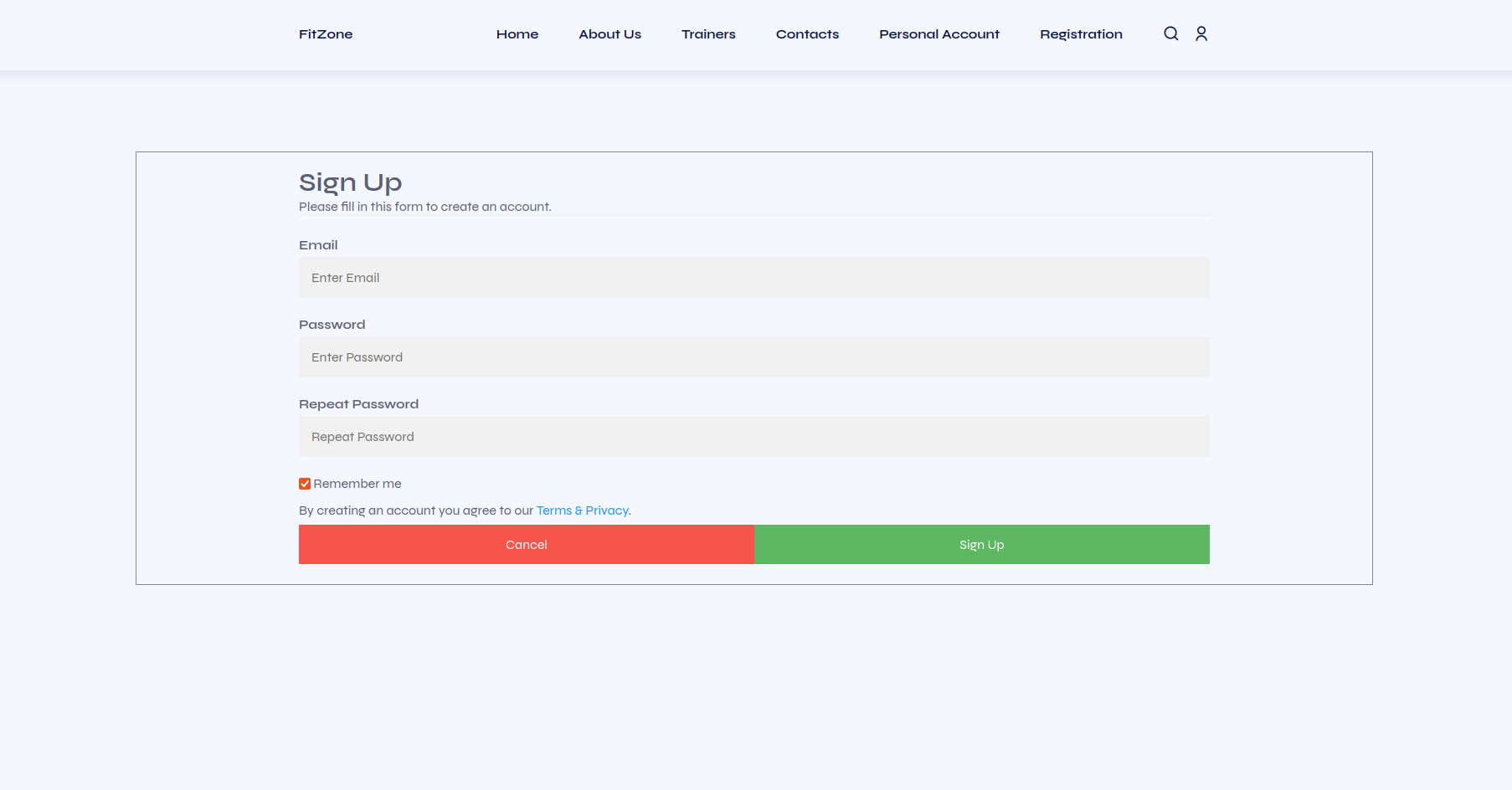


Рисунок 5. Экран регистрации

После того, как пользователь авторизовался или прошел регистрацию, он попадает в личный кабинет. Страница личного кабинета показана на рисунках 6-11. Все следующие блоки представлены на одной странице и чтобы их увидеть, надо промотать страницу вниз.

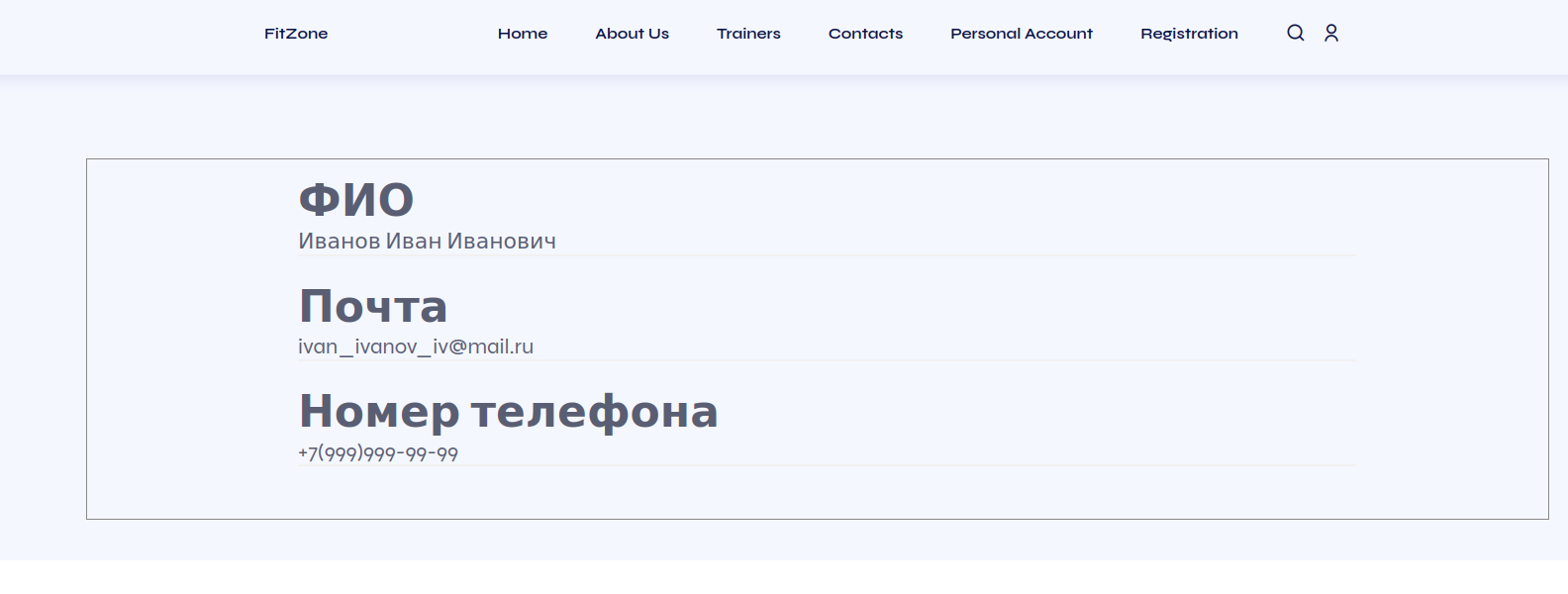


Рисунок 6. ФИО, почта и номер телефона зарегистрированного пользователя

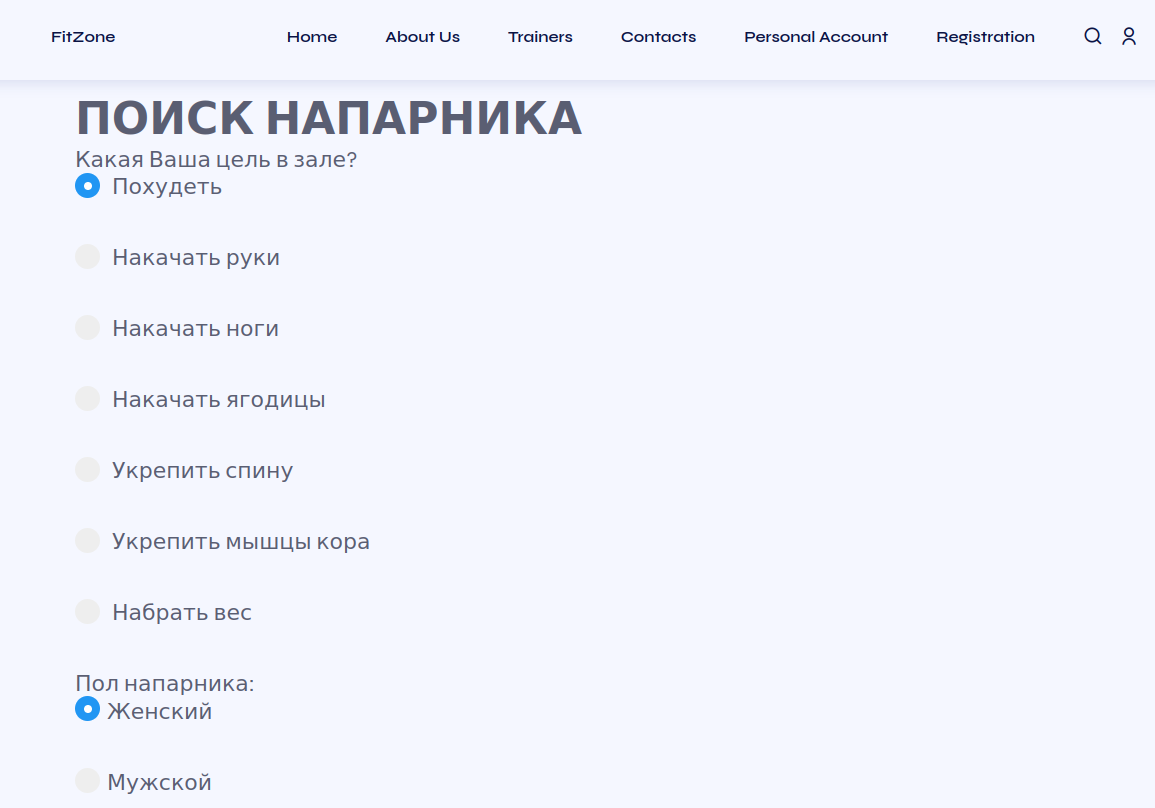


Рисунок 7. Поиск напарника для похода в зал. Для этого надо заполнить небольшой опрос

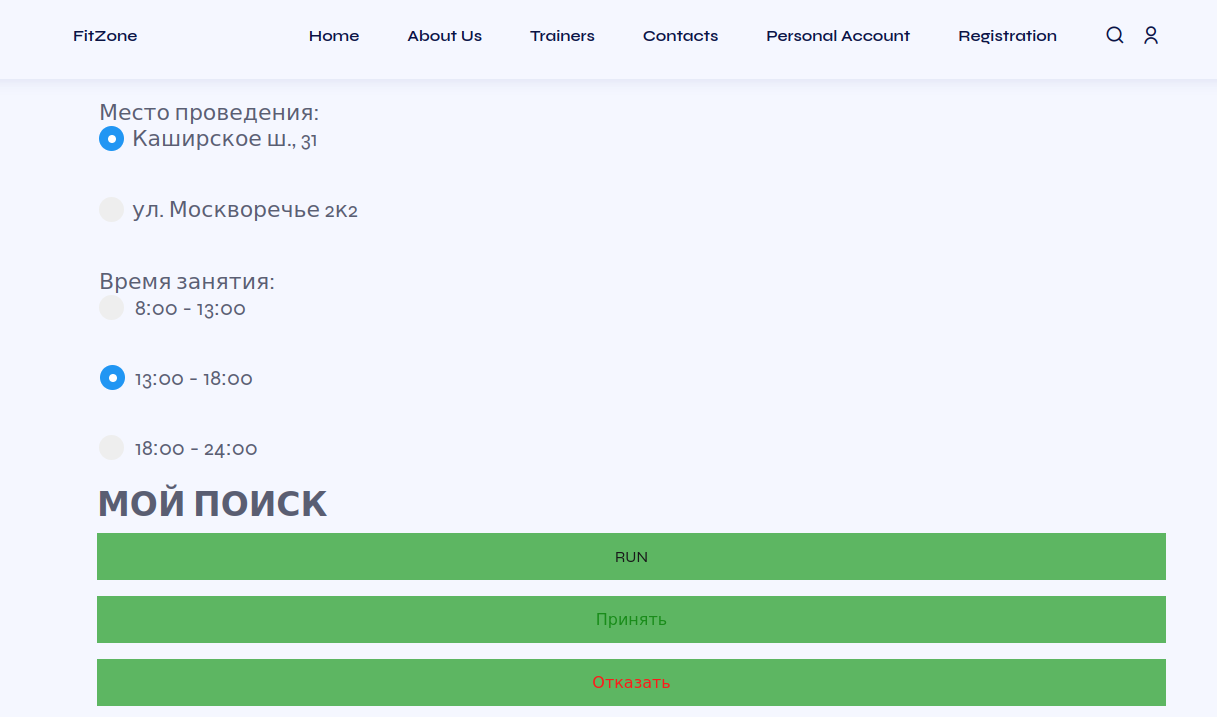


Рисунок 8. Продолжение опроса для поиска напарника

Если человек хочет найти напарника, то он нажимает кнопку “RUN”. После этого он нажимает на кнопку “Принять” и ему будет доступна почта данного человека (рис. 9). Если нажать на кнопку “Отказать”, то заявка человека удаляется из списка.

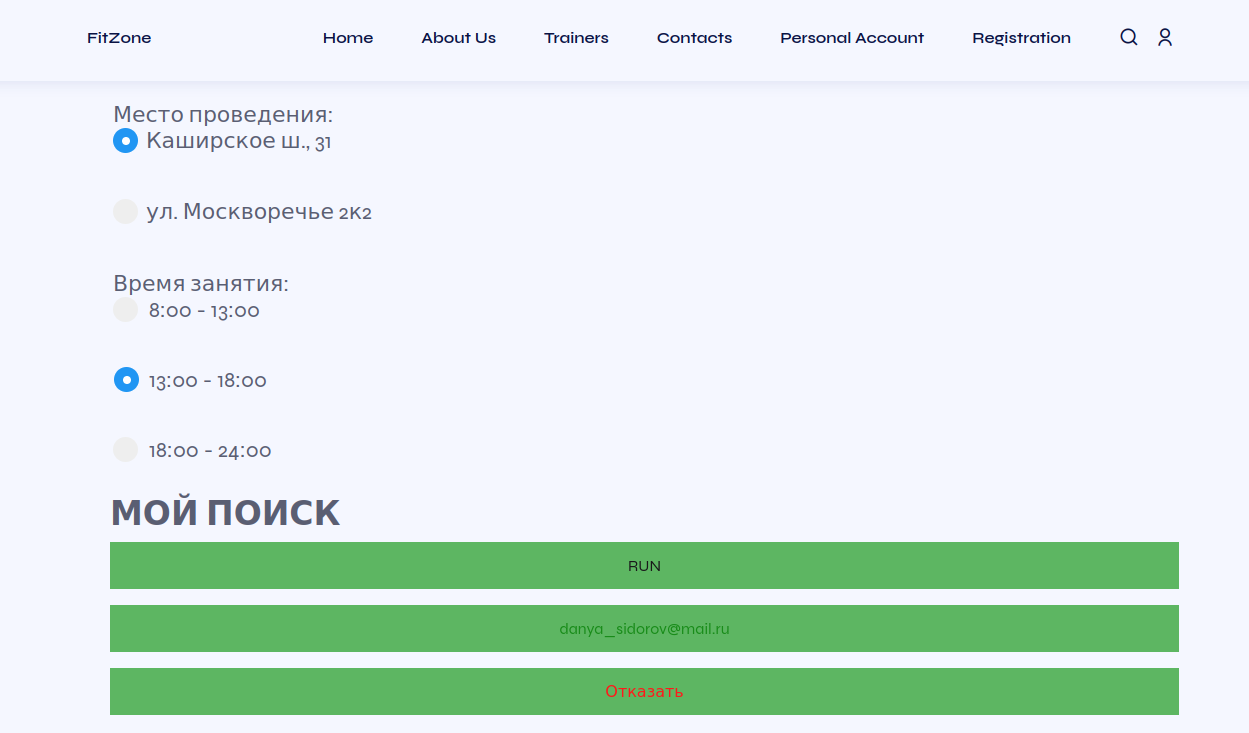


Рисунок 9. После нажатия кнопки “Принять” выводится почта человека

Для генерации питания было принято решение сделать его на неделю. При этом есть выбор, либо человек хочет похудеть, либо набрать массу. На рис. 10 представлен блок плана питания. Чтобы сгенерировать меню, надо нажать на кнопку “Открыть меню”. На рис. 11 представлено меню питания после нажатия кнопки.

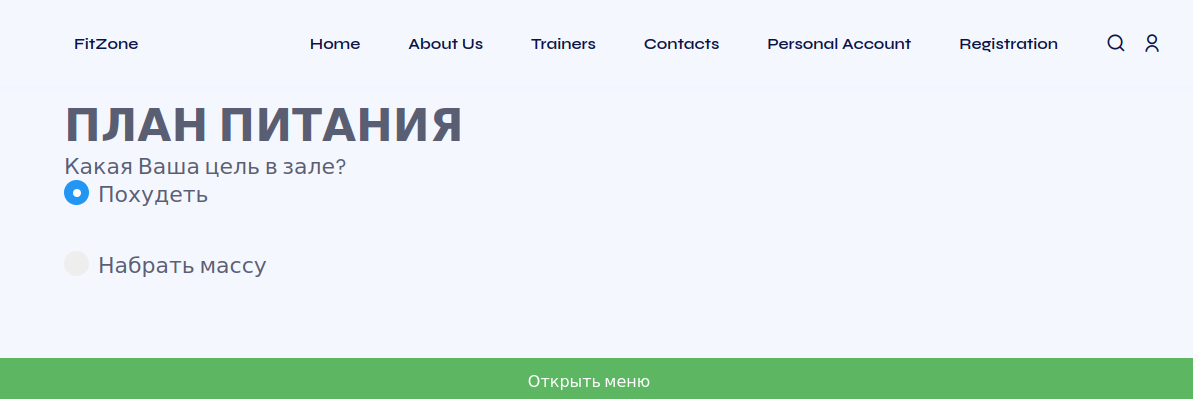


Рисунок 10. До нажатия кнопки “Открыть меню”

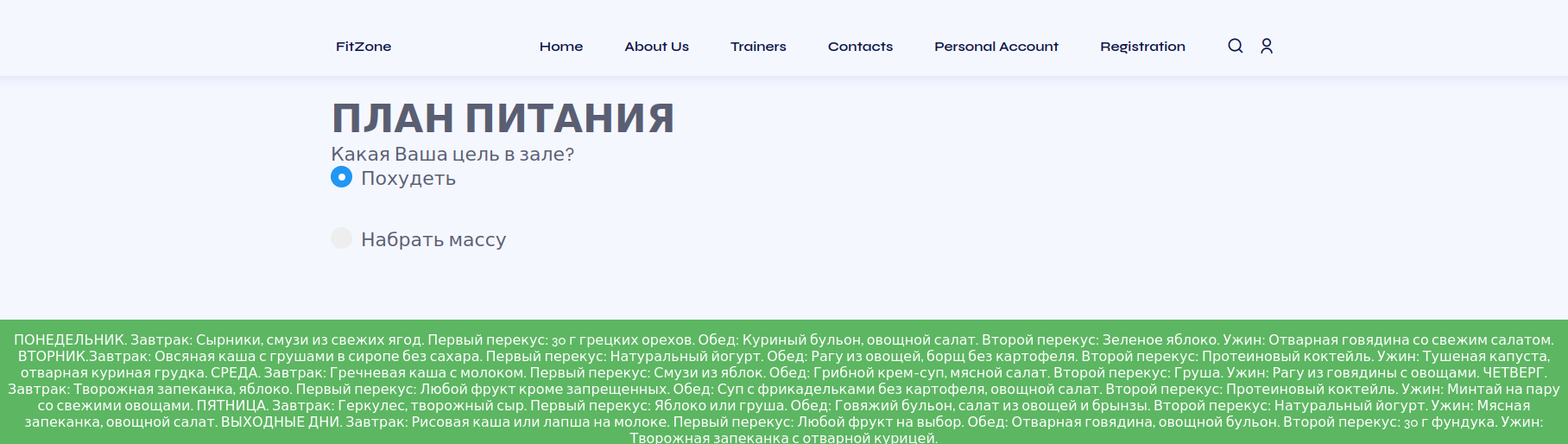


Рисунок 11. После нажатия кнопки “Открыть меню” (план питания сгенерирован)

На главной странице сайта фитнес-центра можно увидеть, как выглядит сам зал изнутри (рис. 12).

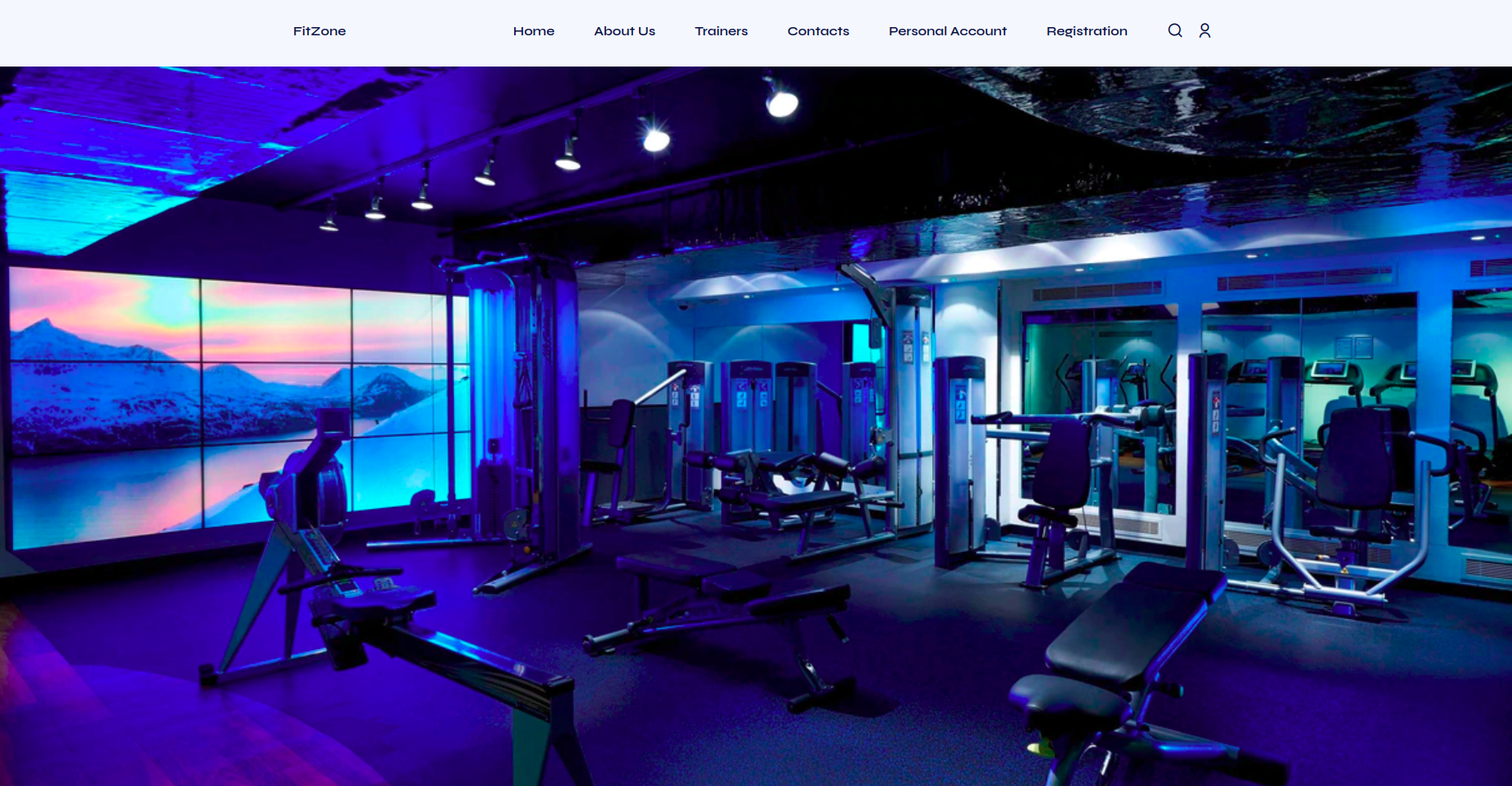


Рисунок 12. Главная страница веб-приложения FitZone

На странице “About Us” (рис. 13) представлена информация о фитнес-центре.



Рисунок 13. Информация о фитнес-центре FitZone

На странице “Contacts” представлена информация о расположении сети фитнес-центров (рис. 14).

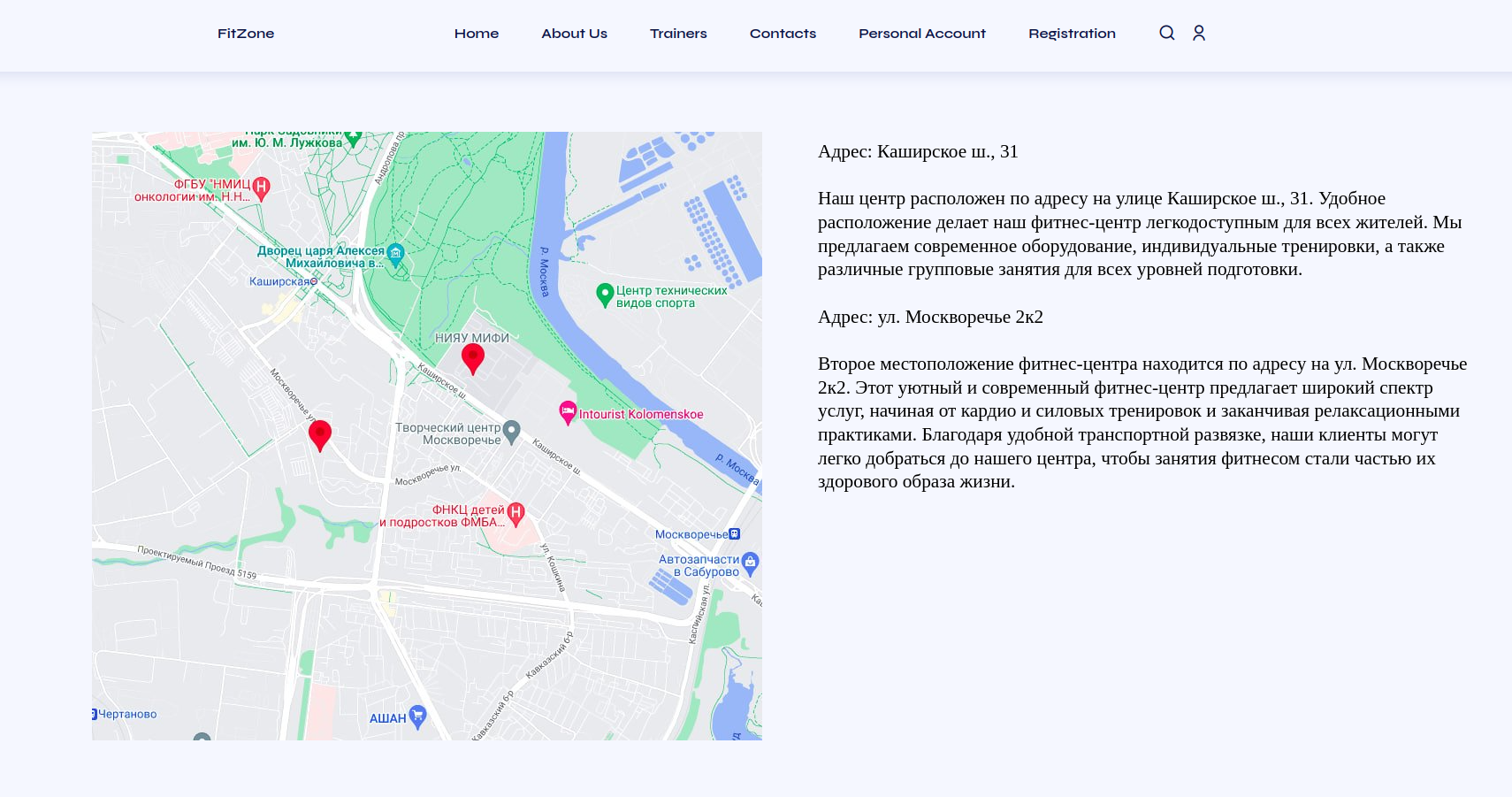


Рисунок 13. Адреса сети фитнес-центра FitZone

# **Тестирование**

При разработке серверной части приложения были проведены ручное тестирование. Онопроводилось с помощью инструментов Postman [12].

Postman — это популярный инструмент для тестирования веб-сервисов API. Он предоставляет пользовательский интерфейс, который облегчает отправку HTTP-запросов, проверку ответов и организацию данных, связанных с API-тестированием.

С помощью Postman разработчики могут создавать и отправлять разнообразные HTTP-запросы, такие как GET, POST, PUT и DELETE, для взаимодействия с API. Он также позволяет организовывать запросы в коллекции, добавлять переменные окружения, тестировать API-методы и автоматизировать тестовые сценарии.

Кроме того, Postman обеспечивает удобную среду для создания и отправки тестовых запросов, а также наглядный интерфейс для просмотра ответов, включая возможность работы с форматами данных, такими как JSON [13]. Этот инструмент стал важным компонентом процесса разработки веб-сервисов API благодаря своей гибкости, удобству использования и широким возможностям тестирования.

# 

# **Заключение**

В соответствии с техническим заданием было разработано приложение, которое состоит из серверной части, веб-приложения и базы данных.

В приложении представлены такие функции как:

* регистрация и вход в аккаунт пользователя;
* поиск напарника для тренировки;
* генерация плана питания на неделю;
* генерация плана тренировки.

Проведенный обзор существующих решений показал, что на данный момент веб-приложения, связанные с фитнесом, не всегда предоставляют пользователям удобный и интуитивно понятный интерфейс. В данной области основным клиентским инструментом часто является веб-сайт, который может быть менее удобным для пользователей по сравнению с другими доступными вариантами интерфейса.

Была спроектирована база данных, которая позволяет обращаться с информацией, которая нужна для функционирования приложения. Серверное приложение, которое позволяет производить действия по обработке данных и веб-приложение, которое является клиентской частью. С помощью него пользователь может получать информацию и вносить изменения в удобном виде.

Произведено ручное тестирование серверной и клиентской части.

Возможное дальнейшее развитие сервиса связано с улучшением интерфейса веб-приложения, добавление функционала по отслеживанию результатов занятий по сгенерированным тренировкам или плана питания.

# **Список используемых источников**

1. DDX [Электронный ресурс] 2023 URL: https://www.ddxfitness.ru/ (дата обращения 31.12.2023)
2. XFIT [Электронный ресурс] 2023 URL: https://www.xfit.ru/ (дата обращения 31.12.2023)
3. Golang [Электронный ресурс]. 2023 URL: https://go.dev/ (дата обращения 31.12.2023)
4. The PostgreSQL Global Development Group, PostgreSQL [Электронный ресурс]. 1996-2022 URL: https://www.postgresql.org/ (дата обращения: 17.12.2023)
5. Docker [Электронный ресурс] 2023 URL: https://www.docker.com/ (дата обращения 31.12.2023)
6. Docker-compose [Электронный ресурс]. 2023 URL: https://github.com/docker/compose (дата обращения: 31.12.2023)
7. HTML [Электронный ресурс]. 1998-2023 URL: https://developer.mozilla.org/en-US/docs/Web/HTML (дата обращения 13.12.2023)
8. CSS [Электронный ресурс]. 1998-2023 URL: https://developer.mozilla.org/en-US/docs/Web/CSS (дата обращения 13.12.2023)
9. JavaScript [Электронный ресурс]. 2005-2023 URL: https://devdocs.io/javascript/ (дата обращения 13.12.2023)
10. Lokesh Gupta. What is REST [Электронный ресурс]. 2022 URL: https://restfulapi.net/ (дата обращения 17.12.2023)
11. Okta. JWT [Электронный ресурс]. 2022 URL: https://jwt.io/ (дата обращения 17.12.2023)
12. Postman [Электронный ресурс]. 2023 URL: https://learning.postman.com/docs/introduction/overview/ (дата обращения 15.12.2023)
13. JSON [Электронный ресурс]. 2023 URL: https://www.json.org/json-en.html (дата обращения 15.12.2023)