МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»

Высшая школа электроники и компьютерных наук
Кафедра «Электронные вычислительные машины»

РАБОТА ПРОВЕРЕНА
Рецензент
_____
«___»_____ 2019 г.

ДОПУСТИТЬ К ЗАЩИТЕ
Заведующий кафедрой ЭВМ
_____ Г.И. Радченко
«___»_____ 2019 г.

Разработка веб-приложения для организации занятий воркаутом

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ

Руководитель работы,
к.т.н., доцент каф. ЭВМ
_____Е.С. Ярош
«___»_____2019 г.

Автор работы,
студент группы КЭ-452
_____А.Д. Сиротюк
«___»_____2019 г.

Нормоконтролёр,
ст. преп. каф. ЭВМ
_____ В.В. Лурье
«___»_____2019 г.

Челябинск-2019

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Высшая школа электроники и компьютерных наук
Кафедра «Электронные вычислительные машины»

УТВЕРЖДАЮ
Заведующий кафедрой ЭВМ
_____ Г.И. Радченко
«___»_____2019 г.

ЗАДАНИЕ
на выпускную квалификационную работу бакалавра

студенту группы КЭ-452
Сиротюку Александру Дмитриевичу
обучающемуся по направлению
09.03.01 «Информатика и вычислительная техника»

1. **Тема работы: «**Разработка веб-приложения для организации занятий воркаутом» утверждена приказом по университету от 25 апреля 2019 г. № 899

2. **Срок сдачи студентом законченной работы:** 6 июня 2019 г.

3. **Исходные данные к работе:**

   Обеспечить основной функционал приложения:

   1. Организация Workout-зарубов и Workout-челленджей.
   2. Размещение обучающего контента и мотивационных материалов.
   3. Размещение информации о спортивных объектах.
   4. Организация спортивных соревнований по Workout.

5. Возможность ведения личной статистики.
6. Размещение книг для саморазвития.

Предусмотреть наличие ролей гостя, активного пользователя, модератора, администратора, создателя соревнований.

Обеспечить возможность оценки обучающего контента, спортивных объектов и книг для саморазвития.

Обеспечить регистрацию с подтверждением почты и смену пароля с отправкой письма на почту пользователя.

Среда и средства реализации – по выбору автора.

4. **Перечень подлежащих разработке вопросов:**

- анализ рынка приложений по тематике работы;
- детализация набора требований к приложению;
- выбор среды и средств реализации;
- проектирование архитектуры приложения;
- организация базы данных;
- создание серверной и клиентской частей приложения;
- тестирование разработанного программного обеспечения.

5. **Дата выдачи задания:** 1 декабря 2018 г.

Руководитель работы _____/*Е.С. Ярош*/

Студент _____/*А.Д .Сиротюк* /

КАЛЕНДАРНЫЙ ПЛАН

Этап	Срок сдачи	Подпись руководителя
Введение и обзор литературы	15 января 2019 года	
Разработка инфологической модели	2 февраля 2019 года	
Разработка датологической модели	22 февраля 2019 года	
Проектирование базы данных	10 марта 2019 года	
Разработка серверной части веб-приложения	5 апреля 2019 года	
Разработка внешнего оформления веб-приложения	25 апреля 2019 года	
Тестирование веб-приложения	7 мая 2019 года	
Оформление пояснительной записки	25 мая 2019 года	
Приемка приложения при комиссии высшего учебного заведения	6 июня 2019 года	

Руководитель работы _____/*Е.С. Ярош*/

Студент _____/*А.Д .Сиротюк* /

Аннотация

А.Д. Сиротюк. Веб-приложение для занятия воркаутом. – Челябинск: ФГАОУ ВО «ЮУрГУ (НИУ)», ВШЭКН; 2019, 163 с., 70 ил., библиогр. список – 13 наим.

В рамках выпускной квалификационной работы создается веб-приложение для занятия воркаутом. Street workout - это уличная гимнастика, которая может быть отнесена к любительскому виду спорта и представляет собой спортивную субкультуру. Включает в себя выполнение различных упражнений на уличных спортплощадках и дома.

Целями проекта являются: пропаганда здорового образа жизни, мотивация людей на самообразование и самосовершеноствование.

Новизна в плане функционала:

1. Система Workout-челленджей: любой пользователь может сделать упражнение, записать на видео и выложить на сайт. Другие спортсмены могут повторить упражнение, записать на видео и так же выложить на сайт.
2. Организация Workout-соревнований и их статистика: создание соревнований, ведение истории соревнований и статистики каждого спортсмена, принимающего участие в соревнованиях.
3. Система Workout-зарубов: пользователь может вызвать на соревнование 1 на 1 другого пользователя, предложив свои 2 упражнения. Другой пользователь может отказать, либо принять и предложить свои 2 упражнения.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Спортивное движение Street Workout позволяет без существенных материальных затрат вести здоровый образ жизни, воспитывает в людях целеустремленность, мотивирует их на самообразование и самосовершенствование.

Однако процесс развития этого движения тормозит отсутствие информационной инфраструктуры, позволяющей интегрировать систему тренировок, места их проведения, организацию различного вида соревнований, обеспечение информацией по здоровому образу жизни и саморазвитию в рамках избранного направления. Ликвидации этой проблемы посвящена данная работа.

В одном приложении сосредоточена вся необходимая информация (от системы тренировок, обучающих видео, индивидуальной статистики спортсмена до рекомендаций по правильному питанию, статей про здоровый образ жизни). Налажена система коммуникации (в том числе личной) между спортсменами различного уровня (обмен опытом, обучение элементам, проведение совместных тренировок и так далее). На карте можно посмотреть расположение площадок для занятий и тренажерных залов, а также их оценки и отзывы. Поддерживается система «челленджей», организация соревнований и их статистика, система «зарубов 1 на 1» и др.

Решаются также социальные проблемы:

1. Недостаточная коммуникабельность. Системы соревнований побуждают людей к общению. Общий спортивный интерес является важным психологическим фактором, влияющим на установление взаимодействия и конструктивных отношений.

2. Нездоровый образ жизни. Проект направлен на побуждение людей вести правильный образ жизни: правильно питаться, быть упорным и целеустремленным, регулярно тренироваться, заниматься саморазвитием, не иметь вредных привычек.
3. Низкая самооценка. Занятие спортом помогает повысить уровень уверенности в себе.

Стадии технологического процесса разработки:

1. Разработка инфологической модели с использованием программы PowerDesigner.
2. Разработка датологической модели с использованием программы PowerDesigner.
3. Проектирование базы данных с помощью СУБД MySQL.
4. Разработка серверной части веб-приложения (backend) с применением языка программирования PHP и фрэймворка Yii2.
5. Разработка внешнего оформления веб-приложения (frontend) с применением языка разметки HTML, таблицы каскадных стилей CSS, языка программирования JavaScript, библиотеки jQuery, фрэймворка Bootstrap.

Для разработки будут использоваться следующие языки программирования и технологии:

1. Frontend: HTML, CSS, JavaScript, JQuery, Bootstrap.
2. Backend: PHP, Yii2, MySQL.

1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1. ОБЗОР АНАЛОГОВ

Ниже перечислены аналогичные продукты на рынке:

1. workout.su

Разработчик ресурса: Антон Кучумов.

Ссылка на страницу с контактными данными разработчика ресурса и описанием веб-приложения: https://workout.su/info

Преимущества и недостатки веб-приложения приведены в таблице 1.1.

Таблица 1.1 – Преимущества и недостатки веб-приложения workout.su:

Преимущества	Недостатки
Наличие множества статей по воркауту, форума, проекта «100 – дневный воркаут», собственного магазина, карты площадок	Отсутствие системы статистики спортсменов, неоптимальное расположение видеоматериала, существует мобильное приложение только с функционалом проекта «100 дневный воркаут», отсутствие статей о книгах по самосовершенствованию

2. streetworkouts.ru

Разработчик ресурса неизвестен.

Преимущества и недостатки веб-приложения приведены в таблице 1.2.

Таблица 1.2 – Преимущества и недостатки веб-приложения streetworkouts.ru:

Преимущества	Недостатки
Наличие множества статей по воркауту, форума, собственного магазина	Отсутствие системы статистики спортсменов, неоптимальное расположение видеоматериала, отсутствие статей о книгах по самосовершенствованию, карты площадок

3. Мобильное приложение «100 – дневка»

Разработчик ресурса: Антон Кучумов.

Ссылка на страницу с контактными данными разработчика ресурса и описанием веб-приложения: https://workout.su/info

Преимущества и недостатки веб-приложения приведены в таблице 1.3.

Таблица 1.3 – Преимущества и недостатки мобильного приложения «100 – дневка»:

Преимущества	Недостатки
Реализован проект «100 – дневный воркаут»	Отсутствие системы статистики спортсменов, обучающего материала, статей о книгах по самосовершенствованию, собственного магазина, карты площадок, форума

4. Мобильное приложение «WorkOut: площадки и тренировки»

Разработчик ресурса: Антон Кучумов.

Ссылка на страницу с контактными данными разработчика ресурса и описанием веб-приложения: https://workout.su/info

Преимущества и недостатки веб-приложения приведены в таблице 1.4.

Таблица 1.4 – Преимущества и недостатки мобильного приложения «WorkOut: площадки и тренировки»:

Преимущества	Недостатки
Реализована база уличных площадок	Без регистрации нельзя использовать приложение, отсутствие системы статистики спортсменов, обучающего видеоматериала, статей о книгах по самосовершенствованию, собственного магазина, карты площадок, форума

5. Мобильное приложение «StayFit»

Разработчик ресурса: Меретский Андрей.

Ссылка на страницу с контактными данными разработчика ресурса и описанием мобильного приложения: https://play.google.com/store/apps/details?id=com.nau.streetworkoutrankmanager

Преимущества и недостатки веб-приложения приведены в таблице 1.5.

Таблица 1.5 – Преимущества и недостатки мобильного приложения «StayFit»:

Преимущества	Недостатки
Наличие множества статей по воркауту, системы статистики спортсменов, системы тренировок	Отсутствие обучающего материала, статей о книгах по самосовершенствованию, собственного магазина, карты площадок, форума

6. voitenko-books.ru

Разработчик ресурса: Игорь Войтенко.

Ссылка на страницу с контактными данными разработчика ресурса и описанием веб-приложения: http://voitenko-books.ru/about

Преимущества и недостатки веб-приложения приведены в таблице 1.6.

Таблица 1.6 – Преимущества и недостатки веб-приложения voitenko-books.ru:

Преимущества	Недостатки
Наличие статей о книгах по самосовершенствованию, наличие собственного магазина	Отсутствие системы статистики спортсменов, обучающего материала, статей о книгах по самосовершенствованию, собственного магазина, карты площадок, форума

1.2. АНАЛИЗ ОСНОВНЫХ ТЕХНОЛОГИЧЕСКИХ РЕШЕНИЙ

1.2.1 Backend

1.2.1.1Выбор языка программирования

Для разработки веб-приложения можно использовать следующие языки: Python, Java, Ruby, C, C++, PHP. Чтобы сразу отсечь нецелесообразные варианты, здесь и далее прежде всего были проанализированы недостатки перечисленных решений.

Недостатки языка C++ и C:

- очень высокий порог вхождения;
- очень низкий уровень абстракции;
- очень низкая скорость разработки веб-приложений.

Недостатки языка Java:

- высокий порог вхождения;
- более медленная скорость разработки из-за низкого уровня абстракции.

Недостатки языка Ruby:

- высокий порог вхождения;
- недостаток информационных ресурсов;
- медленно развивается.

Недостатки языка Python:

- плохая поддержка многопоточности;
- отсутствие коммерческой поддержки средств разработки;
- недостаток средств для работы с базами данных.

Для создания веб-приложения выбран язык программирования PHP, так как данный интерпретируемый язык программирования высокого уровня ориентирован именно для решения задач веб-разработки. Главным фактором языка PHP является практичность. PHP предоставляет программисту средства для быстрого и эффективного решения поставленных задач. Практический характер PHP обусловлен четырьмя важными характеристиками:

- традиционностью;
- простотой;
- эффективностью;
- безопасностью.

Остальные преимущества языка PHP [1, 13]:

- является свободным программным обеспечением, распространяемым под особой лицензией;
- несложен в освоении;
- имеет большое сообщество пользователей и разработчиков;
- имеет развитую поддержку баз данных;
- есть возможность использования в изолированной среде;
- предлагает нативные средства организации веб-сессий, программный интерфейс расширений;
- возможно развертывание почти на любом веб-сервере;
- поддерживается многими аппаратными платформами и операционными системами;
- предоставляет в распоряжение разработчиков и администраторов гибкие и эффективные средства безопасности.

Для упрощения разработки можно использовать либо CMS, либо фреймворк.

Недостатки использования CMS [2]:

- ограниченная функциональность. Большинство CMS удовлетворительно решают не более двух задач, на которые они рассчитаны, что ограничивает круг их использования;
- низкая производительность. У всех CMS ниже производительность, чем у аналогичных решений, разработанных на базе фреймворков;
- избыточность функциональности отдельных модулей. Обычно взгляды разработчиков и пользователей на необходимую функциональность модуля кардинально различаются. Это беда всех универсальных решений. Поэтому часто используется лишь незначительная часть возможностей;
- необходимость изучать сложную систему шаблонов. Дизайнеру придется разобраться с правилами создания шаблонов и их возможностями, которые могут заживо похоронить не одну задумку.

Преимущества использования фреймворков [2]:

- высокая производительность кода. По скорости работы могут уступать только веб-приложениям, полностью написанным на PHP;
- безопасность. Фреймворки пишутся одними опытными программистами для других программистов и тщательно тестируются всем сообществом. Это позволяет вовремя найти недочеты кода с точки зрения безопасности и устранить ошибки;
- гибкость. Фреймворки позволяют решать практически любые задачи. Имеется возможность использования готовых классов и библиотек, разработанных другими программистами.

Фреймворки являются отличным выбором для программистов, желающих разрабатывать сложные уникальные проекты с быстрым кодом. CMS подойдут как для новичков, впервые столкнувшихся с задачей сделать свое веб-приложение, так и для более опытных разработчиков при необходимости быстро создать стандартное веб-приложение.

Таким образом, в данной работе будет использоваться фреймворк.

1.2.1.2 Выбор фреймворка

Существует множество фреймворков для создания веб-приложений. К ним относятся: Laravel, Yii2, Symfony, Codeigniter и другие.

Недостатки фреймворка Codeigniter [3]:

- полное отсутствие механизмов авторизации, аутентификации и проверки прав;
- отсутствие ORM (механизма доступа к таблицам базы данных);
- плохое кэширование;
- нет генератора кода.

Недостатки фреймворка Laravel [4]:

- сложная документация, в основном написанная на английском языке;
- плохая поддержка пользователей;
- нарушение обратной совместимости между различными версиями фреймворка;
- нелогичное расположение файлов и каталогов.

Недостатки фреймворка Symphony [5]:

- небольшое сообщество разработчиков;
- перенасыщенность разного рода сущностей;
- интегрированный аннотационный синтаксис.

Для разработки проекта выбран PHP-фреймворк Yii2. Yii – это универсальный фреймворк, который может быть задействован во всех типах веб-приложений. Благодаря его компонентной структуре и отличной поддержке кэширования, фреймворк особенно подходит для разработки таких крупных проектов, как порталы, форумы, CMS, магазины или RESTful-приложения.

Особенности фреймворка Yii2 [6]:

- для организации кода Yii использует максимально интуитивную MVC (Model-View-Controller) архитектуру;
- Yii придерживается философии простого и элегантного кода, не пытаясь усложнять дизайн только ради следования каким-либо шаблонам проектирования;
- Yii является full-stack фреймворком и включает в себя проверенные и хорошо зарекомендовавшие себя возможности, такие как ActiveRecord для реляционных и NoSQL баз данных, поддержку REST API, многоуровневое кэширование и другие;
- Yii отлично расширяем. Можно настроить или заменить практически любую часть основного кода. Используя архитектуру расширений, легко делиться кодом или использовать код сообщества;
- одна из главных сторон Yii – хорошая производительность;
- наличие простого и понятного генератора кода.

1.2.1.3 Выбор системы управления базой данных

Данные можно хранить в файлах или базах данных (БД). Особенности организации данных в БД по сравнению с файловыми системами [7]:

- базы данных обеспечивают использование одних и тех же данных в различных приложениях;

- БД сводят к минимуму дублирование данных, прибегая к дублированию только для ускорения доступа к данным или для обеспечения восстановления БД при ее разрушении;
- одна из важных черт БД – независимость данных от особенностей прикладных программ, которые их используют, а также возможность создания этих программ в такой форме, что изменение особенностей хранения, логической структуры или значений данных не требует изменения программ их обработки;
- возможность изменения физических особенностей хранения данных без изменения их логической структуры.

Существует множество систем управления базами данных, которые можно использовать для веб-приложений, например MySQL, Microsoft SQL Server, PostgreSQL, MongoDB.

Недостатки Microsoft SQL Server [8]:

- является тяжеловесной;
- даже при тщательной настройке производительности СУБД может занять все доступные ресурсы;
- есть проблемы с использованием службы интеграции для импорта файлов;

Microsoft SQL Server идеально подходит для крупных организаций, которые уже используют ряд продуктов Microsoft.

Недостатки PostgreSQL [8]:

- неразборчивая документация;
- сложная конфигурация;
- скорость работы может падать во время проведения пакетных операций или выполнения запросов чтения.

PostgreSQL идеально подходит для проектов с ограниченным бюджетом, но квалифицированными специалистами.

Недостатки MongoDB [8]:

- SQL не используется в качестве языка запросов;
- программа установки может занять много времени;

MongoDB подходит для проектов с разнородными данными, которые тяжело поддаются классификации. Для внедрения потребуются высококлассные специалисты.

Преимущества MySQL [8]:

- распространяется бесплатно;
- понятная документация;
- простая установка;
- поддерживает набор пользовательских интерфейсов;
- является распространенной на хостингах;
- хорошая поддержка.

MySQL идеально подходит для проектов, которым требуется надежный инструмент управления базами данных, но бесплатный.

Таким образом, при разработке веб-приложения будет использоваться СУБД MySQL.

1.2.2 Frontend

1.2.2.1Выбор средств разработки

Для разработки клиентской части используются следующие средства:

- язык разметки HTML;
- таблицы каскадных стилей CSS;

- язык программирования JavaScript с библиотекой jQuery;
- фреймворк Bootstrap 4, использующий современные наработки в области CSS и HTML.

1.2.2.2 jQuery

jQuery – это библиотека JavaScript, основанная на принципе «пиши меньше, делай больше». Это инструмент, который упрощает написание общих задач JavaScript. Кроме того, jQuery обладает кроссбраузерной совместимостью, а значит, можно быть уверенным в том, что любой современный браузер корректно отобразит вывод программы [9].

1.2.2.3 Bootstrap 4

Главным элементом Bootstrap является адаптивная сетка. В общем-то, без нее фреймворк утратил бы практически всю свою ценность, потому именно благодаря сетке можно быстро создавать адаптивные шаблоны. При этом можно вообще не быть знакомым с медиа-запросами, они не нужны, потому что фреймворк берет на себя реализацию адаптивности, нужно лишь задать блокам правильные классы [10].

1.2.3 Карты

На рынке картографических и справочных сервисов можно выделить три основных конкурента:

- Яндекс.Карты;
- 2ГИС;
- Google Maps.

Сравнение карт по конкурирующим критериям приведено в таблице 1.7.

Таблица 1.7 – Сравнение карт по конкурирующим критериям [11]:

Критерий	**Яндекс.Карты**	**2ГИС**	**Google Maps**
Покрытие	Лучшее покрытие России, уступает Google в покрытии мира	Уступает конкурентам в покрытии как в России, так и в других странах	Лучшее покрытие всего мира
Детализация	Хорошая детализация России, достаточная в мире	Одна из лучших детализаций в городах присутствия	Хорошая детализация по всему миру. На карте России могут отсутствовать крупные города. В плане отображения невнятная детализация. Объекты хорошо видны только при достаточно сильном приближении

Продолжение таблицы 1.7

Критерий	**Яндекс.Карты**	**2ГИС**	**Google Maps**
Детализация на уровне здания	Нет	Небольшие склады, кафедры университетов	Крупные торговые центры
Возможность загрузки и использования офлайн	Да. Большой размер данных	Да	Да. Большой размер данных
Редактирование карт	Сервис «Народная карта» (web); Сообщение об ошибках	Сообщение об ошибках	Сообщение об ошибках
Вариант выбора отображения ландшафта	Карта, спутник, народная карта	Карта	Карта, спутник, Велокарта, общественный транспорт
Отображение пробок в крупных городах	Да. Отображение доп. информации о дорожной обстановке	Не все города	Не все города. Интеграция с сервисом Waze
Возможность общения между пользователями	«Разговорчики»	Нет	Нет

Продолжение таблицы 1.7

Критерий	**Яндекс.Карты**	**2ГИС**	**Google Maps**
Обзорные фотографии улиц (Streetview)	Яндекс Панорамы	Нет	Google Streetview
Поиск универсальный	Да. Интеллектуальный поиск	Да	Да. Интеллектуальный поиск
Голосовой ввод (на русском)	Да	Нет	Да
Режим 3D	(*)Одинаковая высота зданий	Да	Да
Ночной режим	Да	Нет	Да
Построение маршрута	Автомобиль, общественный транспорт. Строит с учетом пробок. Требует интернет для построения	Автомобиль, общественный транспорт. Возможность отдельно выбрать вариант «Метро». Не требует интернета для построения маршрута	Автомобиль, общественный транспорт, пешеходный маршрут. Возможность выбрать только один из видов транспорта или вариант пешком. Строит с учетом пробок

Продолжение таблицы 1.7

Критерий	**Яндекс.Карты**	**2ГИС**	**Google Maps**
Справочная информация	Подробная информация об организациях	Подробная информация об организациях. Ежемесячные обновления	Хуже других знает российские организации
Актуализация справочной информации	Нет информации	Обновления каждый месяц	Нет информации
Возможность оставить отзывы и оценить организацию	Оценка. Развивается сервис Яндекс.Город	Интеграция с сервисом Фламп	Отзыв и оценка
Интерфейс и юзабилити	Современный интерфейс. Осуществление большинства функций возможно в два шага	Интерфейс iPhone версии не адаптирован для iOS 7	Современный интерфейс. Некоторые функции не до конца понятны на интуитивном уровне

Продолжение таблицы 1.7

Критерий	**Яндекс.Карты**	**2ГИС**	**Google Maps**
Стоимость	Бесплатная, если количество запросов в сутки не превышает 25 тысяч.	Бесплатная при использовании не в коммерческих целях	Бесплатная, если количество запросов в месяц не превышает 28 тысяч

Также из достоинств 2ГИС карт стоит отметить понятную и грамотно составленную документацию и компактный код при использовании API в веб-приложении.

Таким образом, в веб-приложении будут использоваться 2ГИС карты.

1.3. ВЫВОД

Исследование рынка приложений для занятий воркаутом показало, что ни одно из них не является полнофункциональным. Следовательно, создание приложения, свободного от выявленных недостатков, является актуальной задачей. Для ее решения выбраны следующие технологические решения: для backend-язык программирования PHP, фреймворк Yii2, СУБД MySQL, для frontend-язык разметки HTML,таблицы каскадных стилей CSS, язык программирования JavaScript с библиотекой jQuery, фреймворк Bootstrap 4.

2. ОПРЕДЕЛЕНИЕ ТРЕБОВАНИЙ

2.1. ФУНКЦИОНАЛЬНЫЕ ТРЕБОВАНИЯ

2.1.1 Основные требования к функционалу системы

1. Вся необходимая информация для занятий воркаутом и фитнесом (тренажерный зал) должна находиться в одном приложении (от системы тренировок и программ до правильного питания, статей про здоровый образ жизни).
2. Наличие обучающих видео для спортсменов и мотивационных материалов.
3. Поддержка системы статистики каждого спортсмена (названия элементов, которые спортсмен умеет делать, какой вес жмет от груди и так далее).
4. Система «челленджей». Например, вызов «подтягивания за 2 минуты», «100 на 100» (выполнить жим 100 кг от груди 100 раз) и так далее. Рейтинг, результаты «челленджей» с учетом приватности.
5. Отметка нахождения площадок и тренажерных залов на карте, а также наличие оценок и отзывов.
6. Организация соревнований и их статистика.
7. Наличие системы коммуникаций между спортсменами различного уровня (обмен опытом, обучение элементам, проведение совместных тренировок и так далее).
8. Система «зарубов 1 на 1».
9. Пропаганда книг для самосовершенствования. Система отзывов и оценок.
10. Система модерирования приложения.
11. Система администрирования приложения.

2.1.2 Требования к функционалу системы модерирования приложения

1. Разрешить проведение челленджа.
2. Разрешить проведение соревнования.
3. Разрешить участие в челлендже.
4. Удалить отзыв о спортивном объекте.
5. Удалить отзыв о книге.
6. Разрешить опубликовать статью в разделе «Здоровое питание».
7. Разрешить опубликовать статью в разделе «Обучающие статьи».
8. Разрешить опубликовать спортивный объект на карте.

2.1.3 Требования к функционалу системы администрирования приложения

1. Администрирование челленджей:
    a. Редактировать данные о челлендже.
    b. Удалить челлендж.
2. Администрирование книг:
    a. Редактировать информацию о книге.
    b. Добавить книгу.
    c. Удалить книгу.
    d. Удалить отзыв о книге.
3. Администрирование соревнований:
    a. Редактировать данные о соревновании.
    b. Удалить соревнование.
    c. Добавить участника на соревнование.
    d. Исключить участника из соревнования.

4. Администрирование списка элементов воркаута:
    a. Добавить элемент воркаута.
    b. Редактировать информацию об элементе воркаута.
    c. Удалить элемент воркаута.
5. Администрирование новостей:
    a. Добавить новость.
    b. Редактировать новость.
    c. Удалить новость.
6. Администрирование статей:
    a. Добавить статью.
    b. Удалить статью.
    c. Редактировать статью.
    d. Добавить категорию статей.
    e. Удалить категорию статей.
    f. Редактировать название категории статей.
7. Администрирование спортивных площадок/тренажерных залов на карте:
    a. Добавить спортивную площадку/тренажерный зал на карту.
    b. Удалить спортивную площадку/тренажерный зал на карте.
    c. Удалить отзыв о спортивной площадке/тренажерном зале на карте.

2.2. НЕФУНКЦИОНАЛЬНЫЕ ТРЕБОВАНИЯ

2.2.1 Требования к пользователям

Каждый пользователь, работающий в системе, обязан:

- зарегистрироваться в системе и заполнить следующие обязательные поля в регистрационной форме:

a) имя пользователя (необязательно настоящее);

b) e-mail пользователя;

c) пароль для входа в систему;

d) повторный пароль для входа в систему;

- подтвердить свою личность через электронную почту, чтобы доказать, что он не является спам-ботом.

В системе должны быть выделены следующие типы пользователей:

- спортсмен;
- создатель соревнований;
- модератор;
- администратор.

Спортсмен должен иметь возможность воспользоваться любым функционалом системы, за исключением функций модерирования и администрирования.

Модератор должен иметь возможность воспользоваться функционалом модерирования.

Администратор должен иметь возможность воспользоваться функционалом администрирования.

2.2.2 Требования к системе безопасности

Система должна обладать следующими средствами обеспечения безопасности:

- пользователь при входе в систему обязан указывать логин и пароль;
- пользователь при регистрации в системе обязать указывать пароль два раза (для подтверждения);
- каждый пользователь должен иметь возможность скрыть определенную информацию о себе или, наоборот, показать;
- пароль должен храниться в зашифрованном виде;

- важные конфиденциальные данные должны передаваться в защищенном виде.

2.2.3 Требования к системе уведомлений

Приложение должно обладать следующим минимальным набором уведомлений:

- предложение о «зарубе»;
- получение сообщения от другого пользователя;
- предложение об участии в соревновании;
- предложение об участии в челлендже.

2.2.4 Требования к лингвистическому обеспечению

Приложение должно быть доступно на русском языке.

2.2.5 Требования к персоналу

Пользователи приложения должны обладать базовыми навыками владения компьютером.

3. ПРОЕКТИРОВАНИЕ

3.1. АРХИТЕКТУРА ПРЕДЛАГАЕМОГО РЕШЕНИЯ

Для разработки веб-приложения использовался фреймворк Yii2. Для организации кода Yii2 использует MVC (Model-View-Controller) архитектуру. Идея, которая лежит в основе конструкционного шаблона MVC, следующая: нужно чётко разделять ответственность за различное функционирование в приложении. Приложение разделяется на три основных компонента, каждый из которых отвечает за различные задачи. Схема архитектурного паттерна MVC представлена на рисунке 3.1 [12].

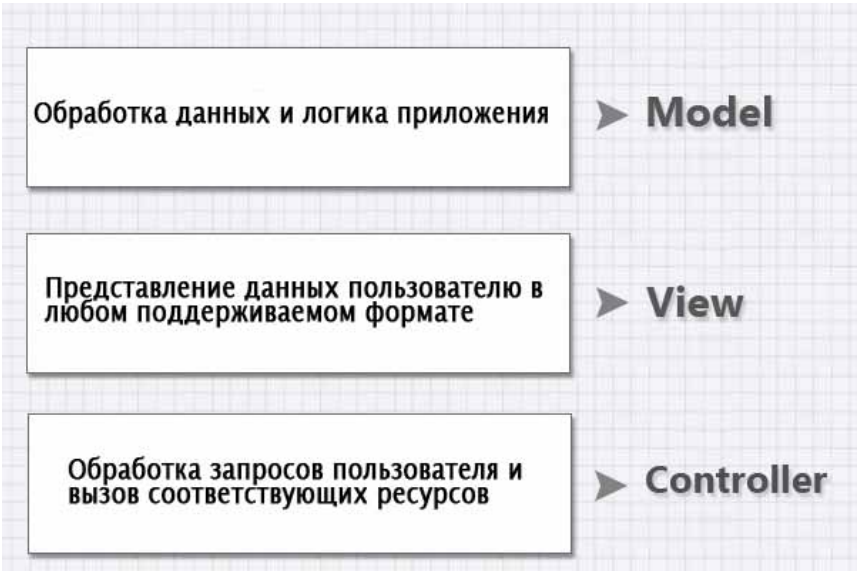

Рисунок 3.1 – Схема архитектурного паттерна MVC

Контроллер управляет запросами пользователя (получаемыми в виде запросов HTTP GET или POST, когда пользователь нажимает на элементы интерфейса для выполнения различных действий). Его основная функция — вызывать и координировать действие необходимых ресурсов и объектов, нужных для выполнения действий, задаваемых пользователем. Обычно

контроллер вызывает соответствующую модель для задачи и выбирает подходящий вид.

Модель - это данные и правила, используемые для работы с данными, которые представляют концепцию управления приложением. В любом приложении вся структура моделируется как данные, которые обрабатываются определённым образом. Модель даёт контроллеру представление данных, которые запросил пользователь (сообщение, страницу книги, фотоальбом и тому подобное). Модель данных будет одинаковой вне зависимости от того, как мы хотим представлять их пользователю. Поэтому выбирается любой доступный вид для отображения данных.

Модель содержит наиболее важную часть логики приложения, которая лежит в основе решения задачи. Контроллер содержит в основном организационную логику для самого приложения.

Вид обеспечивает различные способы представления данных, которые получены из модели. Он может быть шаблоном, который заполняется данными. Может быть несколько различных видов, и контроллер выбирает, какой подходит наилучшим образом для текущей ситуации.

Веб-приложение обычно состоит из набора контроллеров, моделей и видов. Контроллер может быть устроен как основной, который получает все запросы и вызывает другие контроллеры для выполнения действий в зависимости от ситуации [12].

Таблица с контроллерами веб-приложения (кроме контроллеров модуля модерации) приведена в таблице 3.1.

Таблица 3.1 – Контроллеры веб-приложения (кроме контроллеров модуля модерации):

Наименование	**Назначение**	**Количество action, шт.**	**Взаимодействие с другими компонентами**	
			View	**Model**
Article	Работа со статьями	6	• index; • view; • creating; • index-food; • editing	• Article; • ArticleCreating-Form; • ImageUploading; • ArticleEditingForm
Book	Работа с книгами	4	• index; • view	• Book
Challenge	Работа с челленджами	4	• index; • creating; • view; • taking-part	• Challenge-CreatingForm; • Challenge-Participation; • Challenge; • Challenge-ParticipantForm
Competition	Работа с соревнованиями	6	• index; • view; • view-parti-cipants; • creating; • completed	• Competition; • Competition-CreatingForm

Продолжение таблицы 3.1

Наименование	**Назначение**	**Количество action, шт.**	**Взаимодействие с другими компонентами**	
			View	**Model**
Element	Работа с элементами воркаута	5	• index; • 1lvl; • 2lvl; • 3lvl	• Element
Fight	Работа с «зарубами»	3	• upcoming-fights; • completed-fights; • creating	• Exercise; • Fight-CreatingForm; • Fight
Map	Работа с картой	4	• index; • creating; • view	• SportsFacility; • Sports-Facility-Comment; • Sports-Facility-CreatingForm; • Sports-FacilityMark; • Comment-Form;

Продолжение таблицы 3.1

Наименование	**Назначение**	**Количество action, шт.**	**Взаимодействие с другими компонентами**	
			View	**Model**
Site	Работа с пользователями	18	• index; • registration; • authorization; • profile; • edit-profile; • book-cart; • window-elements; • view-user; • window-fight-Offers; • verification; • changing-password; • mail-changing-password	• FightOffer-Exercise; • Image-Uploading; • EditProfile-Form; • FightOffer; • Registration-Form; • User; • Article; • AuthorizationForm; • Changing-Password-Form

Продолжение таблицы 3.1

Наименование	**Назначение**	**Количество action, шт.**	**Взаимодействие с другими компонентами**	
			View	**Model**
Feedback	Работа с обратной связью	1	• index	• Feedback-Form
Protocol	Работа с результатами соревнований	4	• _form; • index; • update; • view	• UserCompeti-tion

Таблица с контроллерами веб-приложения (модуль модерации) приведена в таблице 3.2.

Таблица 3.2 – Контроллеры веб-приложения (модуль модерации):

Наименование	**Назначение**	**Количество action, шт.**	**Взаимодействие с другими компонентами**	
			View	**Model**
AppModeration	Ограничение доступа ко всем контроллерам модуля модерации	0	Отсутствует	Отсутствует
Default	Стартовая страница модуля модерации	1	• index	Отсутствует

Продолжение таблицы 3.2

Наименование	**Назначение**	**Количество action, шт.**	**Взаимодействие с другими компонентами**	
			View	**Model**
Article	Проверка статей	4	• _form; • index; • update; • view	• Article
Challenge	Проверка Workout-челленджей	4	• _form; • index; • update; • view	• Challenge
Competition	Проверка соревнований	4	• _form; • index; • update; • view	• Competition
Map	Проверка спортивных объектов на карте	4	• _form; • index; • update; • view	• SportsFacility
Participation	Проверка участников Workout-челленджей	4	• _form; • index; • update; • view	• Participation

Продолжение таблицы 3.2

Наименование	**Назначение**	**Количество action, шт.**	**Взаимодействие с другими компонентами**	
			View	**Model**
Facility	Проверка отзывов о спортивных объектах	33	• index; • view	• BookComment
Book	Проверка отзывов о книгах	3	• index; • view	• Sports-FacilityComment

Ниже представлены модели веб-приложения «Street Workout» (кроме моделей модуля модерации):

Таблица с моделями веб-приложения (кроме моделей модуля модерации) приведена в таблице 3.3.

Таблица 3.3 – Модели веб-приложения (кроме моделей модуля модерации):

Наименование	**Назначение**
Article	Статьи (бизнес-логика, active record)
ArticleCreatingForm	Статьи (обработка формы)
AuthorizationForm	Обработка формы авторизации
Book	Книги (бизнес-логика, active record)
BookComment	Комментарии к книгам (бизнес-логика, active record)

Продолжение таблицы 3.3

Наименование	**Назначение**
Challenge	Челленджи (бизнес-логика, active record)
ChallengeCreatingForm	Обработка формы создания челленджа
ChallengeParticipantForm	Обработка формы принятия участия в челлендже
ChallengeParticipation	Участники челленджа (бизнес-логика, active record)
CommentForm	Обработка комментариев к книгам
Competition	Соренования (бизнес-логика, active record)
CompetitionCreatingForm	Обработка формы создания соревнования
EditProfileForm	Обработка формы редактирования профиля пользователя
Element	Элементы воркаута (бизнес-логика, active record)
Exercise	Упражнения воркаута (бизнес-логика, active record)
ExerciseFight	Связь между упражнениями и «зарубами» (бизнес-логика, active record)
Fight	«Зарубы» (бизнес-логика, active record)
FightCreatingForm	Обработка формы предложения «зарубов» пользователю

Продолжение таблицы 3.3

Наименование	**Назначение**
FightOffer	Предложения «зарубов» пользователям (бизнес-логика, active record)
FightOfferExercise	Предложения упражнений «зарубов» пользователям (бизнес-логика, active record)
ImageUploading	Загрузка картинок
RegistrationForm	Обработка регистрационной формы
User	Пользователи (бизнес-логика, active record)
UserFight	Связь между пользователями и «зарубами» (бизнес-логика, active record)
FeedbackForm	Обработка формы обратной связи
SportsFacility	Спортивные объекты (бизнес-логика, active record)
SportsFacilityComment	Отзывы к спортивным объектам (бизнес-логика, active record)
SportsFacilityCreatingForm	Обработка формы создания спортивного объекта
SportsFacilityMark	Оценки спортивных объектов (бизнес-логика, active record)
ArticleEditingForm	Обработка формы редактирование статьи

Продолжение таблицы 3.3

Наименование	**Назначение**
ChangingPasswordForm	Отправка на почту письма с инструкцией для смены пароля и обработка формы смены пароля
UserCompetition	Связь между пользователями и соревнованиями (бизнес-логика, active record)

Таблица с моделями веб-приложения (модуль модерации) приведена в таблице 3.4.

Таблица 3.4 – Модели веб-приложения (модуль модерации):

Наименование	**Назначение**
Article	Статьи (бизнес-логика, active record)
Challenge	Челледжи (бизнес-логика, active record)
Competition	Workout-соревнования (бизнес-логика, active record)
Participation	Участники Workout-челледжей (бизнес-логика, active record)
SportsFacility	Спортивные объекты (бизнес-логика, active record)
SportsFacilityComment	Отзывы о спортивных объектах (бизнес-логика, active record)

Продолжение таблицы 3.4

Наименование	**Назначение**
BookComment	Отзывы о книгах (бизнес-логика, active record)

Таблица с видами веб-приложения (кроме видов модуля модерации) приведена в таблице 3.5.

Таблица 3.5 – Виды веб-приложения (кроме видов модуля модерации):

Привязка к контроллеру	**Наименование**	**Назначение**
Article	creating	Отображение формы создания статьи
	index	Список обучающих статей
	index-food	Список статей о здоровом образе жизни
	view	Просмотр статьи
	editing	Редактирование статьи
Book	index	Список книг
	view	Просмотр книги
Challenge	creating	Отображение формы создания челленджа
	index	Список челленджей
	taking-part	Отображение формы для принятия участия в челлендже
	view	Просмотр челленджа

Продолжение таблицы 3.5

Привязка к контроллеру	**Наименование**	**Назначение**
Competition	completed	Список завершенных челленджей
	creating	Отображение формы создания челленджа
	index	Список челленджей
	view	Просмотр челленджа
Competition	view-participants	Список участников челленджа
Element	1lvl	Список элементов воркаута первого уровня сложности
	2lvl	Список элементов воркаута второго уровня сложности
	3lvl	Список элементов воркаута третьего уровня сложности
	index	Список уровней элементов воркаута
Fight	completed-fights	Список завершенных «заруб»
	creating	Отображение формы предложения о «зарубе»
	upcoming-fights	Список предстоящих «заруб»
Site	authorization	Отображение формы авторизации
	book-cart	Список прочитанных книг пользователем

Продолжение таблицы 3.5

Привязка к контроллеру	**Наименование**	**Назначение**
Site	edit-profile	Отображение формы редактирования профиля
	index	Отображение главной страницы веб-приложения
	profile	Отображение профиля пользователя
	registration	Отображение формы регистрации
	rules	Отображение правил сообщества
	view-user	Просмотр профиля пользователя
	window-elements	Список изученных элементов пользователем
	window-fightOffers	Список предложений пользователю о «зарубах»
	window-ownFightOffers	Список предложений пользователя о «зарубах»
	changing-password	Отображение формы смены пароля
	mail-changing-password	Отображения информации об отправке инструкции для смены пароля на почту

Продолжение таблицы 3.5

Привязка к контроллеру	**Наименование**	**Назначение**
Site	verification	Подтверждение верификации пользователя
Map	index	Отображение карты с площадками и тренажерными залами
Protocol	_form	Отображение формы редактирования результатов соревнований
	view	Просмотр результатов соревнования
	update	Отображение страницы редактирования результатов соревнований
	index	Отображение результатов соревнований
Map	creating	Отображение формы создания спортивного объекта
	view	Просмотр спортивного объекта
Feedback	index	Отображение формы обратной связи

Таблица с видами веб-приложения (модуль модерации) приведена в таблице 3.6.

Таблица 3.6 – Виды веб-приложения (модуль модерации):

Привязка к контроллеру	**Наименование**	**Назначение**
Article	_form	Отображение формы редактирования поля проверки статьи
	index	Список непроверенных статей
	update	Отображение страницы редактирования поля проверки статьи
	view	Просмотр непроверенной статьи
Challenge	_form	Отображение формы редактирования поля проверки Workout-челленджа
	index	Список непроверенных Workout-челленджей
	update	Отображение страницы редактирования поля проверки Workout-челленджа
	view	Просмотр непроверенного Workout-челленджа
Competition	_form	Отображение формы редактирования поля проверки соревнования

Продолжение таблицы 3.6

Привязка к контроллеру	**Наименование**	**Назначение**
Competition	index	Список непроверенных соревнований
	update	Отображение страницы редактирования поля проверки соревнования
	view	Просмотр непроверенного соревнования
Map	_form	Отображение формы редактирования поля проверки спортивного объекта
	index	Список непроверенных спортивных объектов
	update	Отображение страницы редактирования поля проверки спортивного объекта
	view	Просмотр непроверенного спортивного объекта
Participation	_form	Отображение формы редактирования поля проверки участника Workout-челленджа
	index	Список непроверенных Workout-челленджей

Продолжение таблицы 3.6

Привязка к контроллеру	**Наименование**	**Назначение**
Participation	update	Отображение страницы редактирования поля проверки участника Workout-челленджа
	view	Просмотр непроверенного Workout-челленджа
Facility	index	Отображение отзывов о спортивных площадках
	view	Просмотр отзыва о спортивной площадке
Book	index	Отображение отзывов о книгах
	view	Просмотр отзыва о книге

Таблица с почтовыми файлами вида приведена в таблице 3.7.

Таблица 3.7 – почтовые файлы вида:

Название	**Назначение**
verification	Верификация пользователя
changing-password	Смена пароля пользователя

Таблица с итогами разработки компонентов архитектуры веб-приложения приведена в таблице 3.8.

Таблица 3.8 – Итоги разработки архитектуры веб-приложения:

Количество контроллеров, шт	**Количество моделей, шт**	**Количество видов, шт**	**Количество почтовых файлов вида, шт**
19	39	69	2

3.2. ОПИСАНИЕ ДАННЫХ

Схема базы данных с использованием UML-диаграмм представлена на рисунке 3.2.

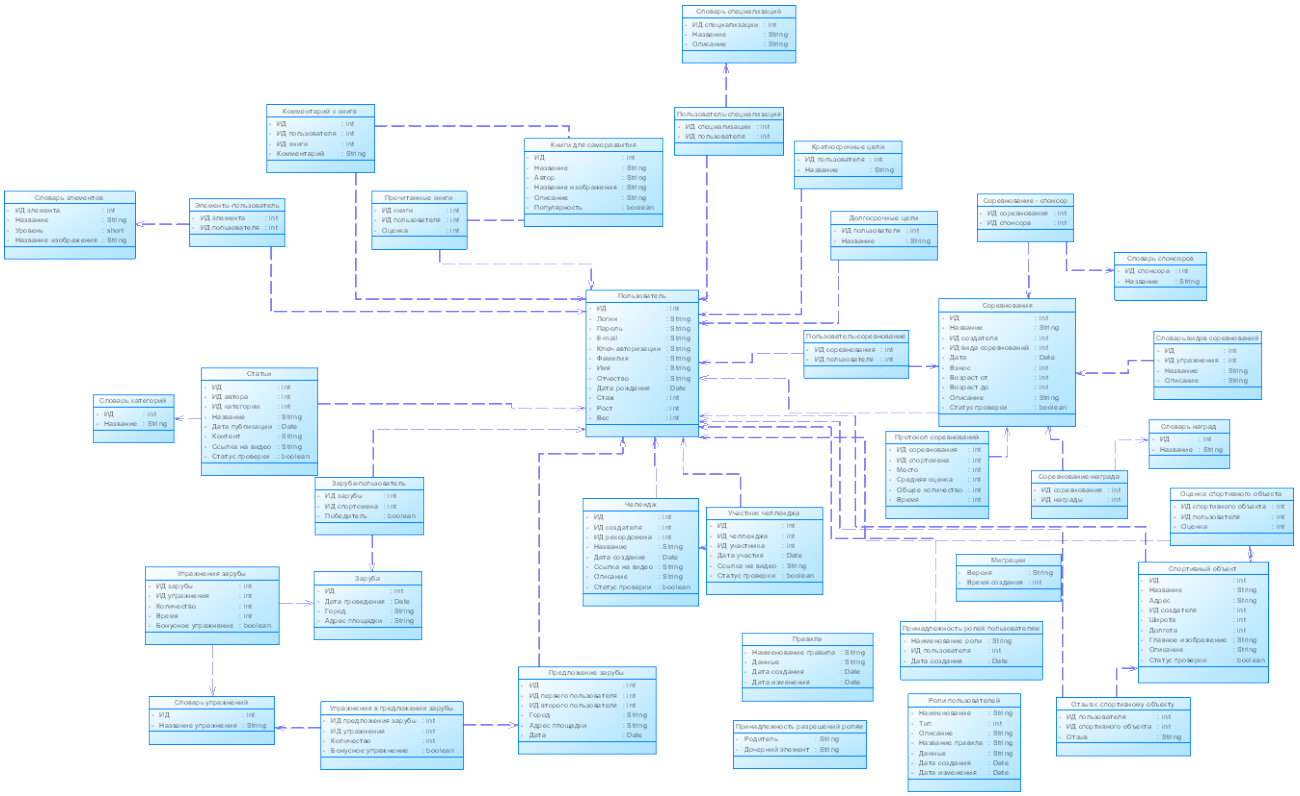

Рисунок 3.2 – Схема базы данных с использованием UML-диаграмм

Ниже представлены таблицы базы данных:

Таблица со списком таблиц базы данных приведена в таблице 3.9.

Таблица 3.9 – Таблицы базы данных:

Наименование	**Назначение**
article	Статьи
auth_assignment	Принадлежность ролей пользователям
auth_item	Роли пользователей
auth_assignment_child	Принадлежность разрешений ролям пользователей
auth_rule	Правила ролей пользователей
award	Награды соревнований
book	Книги
book_comment	Комментарии к книгам
category	Категории статей
challenge	Workout-челленджи
challenge_participation	Участники Workout-челленджей
competition	Соревнования
competition_award	Связывает таблицы «competition» и «award»
competition_protocol	Связывает таблицы «protocol» и «competition»
competition_sponsor	Связывает таблицы «sponsor» и «competition»
element	Workout-элементы
exercise	Workout-упражнения
exercise_fight	Связывает таблицы «exercise» и «fight»
fight	Workout-зарубы

Продолжение таблицы 3.9

Наименование	**Назначение**
fight_offer	Предложения пользователям о Workout-зарубах
fight_offer_exercise	Связывает таблицы «fight_offer» и «exercise»
long-term_goal	Долгосрочные цели
mark_sports_facility	Оценки спортивных объектов
migration	Миграции
news	Новости
short-term_goal	Краткосрочные цели
specialization	Специализации спортсменов
sponsor	Спонсоры соревнований
sports_facility	Спортивные объекты
sports_facility_comment	Отзывы о спортивных объектах
type_competition	Типы соревнований
user	Пользователи
user_book	Связывает таблицы «user» и «book»
user_competition	Связывает таблицы «competition» и «user»
user_element	Связывает таблицы «user» и «element»
user_fight	Связывает таблицы «user» и «fight»
user_specialization	Связывает таблицы «user» и «specialization»

Таблица с описанием полей таблицы «article» приведена в таблице 3.10.

Таблица 3.10 – Описание полей таблицы «article»:

Наименование	**Тип данных**	**Назначение**
id_article	integer	Идентификатор статьи (первичный ключ)
id_author	integer	Идентификатор автора статьи (внешний ключ)
id_category	integer	Идентификатор категории статьи (внешний ключ)
name_article	varchar	Название статьи
date_publication	datetime	Дата публикации статьи
annotation	text	Аннотация статьи
content	text	Текст статьи
main_image	varchar	Имя главное изображения статьи
video_link	varchar	Ссылка на видео
is_checked	enum	Проверка статьи модератором
is_advertising	enum	Вид статьи

Таблица с описанием полей таблицы «auth_assignment» приведена в таблице 3.11.

Таблица 3.11 – Описание полей таблицы «auth_assignment»:

Наименование	Тип данных	Назначение
item_name	varchar	Название роли пользователя (первичный ключ)
user_id	integer	Идентификатор пользователя (внешний ключ)
created_at	integer	Дата присвоения роли пользователю

Таблица с описанием полей таблицы «auth_item» приведена в таблице 3.12.

Таблица 3.12 – Описание полей таблицы «auth_item»:

Наименование	Тип данных	Назначение
name	varchar	Название роли пользователя (первичный ключ)
type	small integer	Тип роли пользователя (внешний ключ)
description	text	Описание роли пользователя
rule_name	varchar	Название правила (внешний ключ)
data	blob	Данные

Продолжение таблицы 3.12

Наименование	Тип данных	Назначение
created_at	integer	Дата создания роли пользователя
updated_at	integer	Дата изменения роли пользователя

Таблица с описанием полей таблицы «auth_item_child» приведена в таблице 3.13.

Таблица 3.13 – Описание полей таблицы «auth_item_child»:

Наименование	Тип данных	Назначение
parent	varchar	Название роли пользователя (первичный ключ)
child	varchar	Название разрешения роли пользователя (первичный и внешний ключ)

Таблица с описанием полей таблицы «auth_rule» приведена в таблице 3.14.

Таблица 3.14 – Описание полей таблицы «auth_rule»:

Наименование	Тип данных	Назначение
name	varchar	Название правила роли пользователя (первичный ключ)
data	blob	Данные

Продолжение таблицы 3.14

Наименование	Тип данных	Назначение
created_at	integer	Дата создания правила роли пользователя
updated_at	integer	Дата изменения правила роли пользователя

Таблица с описанием полей таблицы «award» приведена в таблице 3.15.

Таблица 3.15 – Описание полей таблицы «award»:

Наименование	Тип данных	Назначение
id	integer	Идентификатор награды (первичный ключ)
name	varchar	Название награды

Таблица с описанием полей таблицы «book» приведена в таблице 3.16.

Таблица 3.16 – Описание полей таблицы «book»:

Наименование	Тип данных	Назначение
id	integer	Идентификатор книги (первичный ключ)
name	varchar	Название книги
author	varchar	Автор книги
description	text	Описание книги
image	varchar	Имя изображения книги
popularity	enum	Популярность книги

Таблица с описанием полей таблицы «book_comment» приведена в таблице 3.17.

Таблица 3.17 – Описание полей таблицы «book_comment»:

Наименование	Тип данных	Назначение
id	integer	Идентификатор комментария к книге (первичный ключ)
id_user	integer	Идентификатор пользователя (внешний ключ)
id_book	integer	Идентификатор книги (внешний ключ)
content	text	Комментарий к книге

Таблица с описанием полей таблицы «category» приведена в таблице 3.18.

Таблица 3.18– Описание полей таблицы «category»:

Наименование	Тип данных	Назначение
id	integer	Идентификатор категории статьи (первичный ключ)
name_category	varchar	Название категории статьи

Таблица с описанием полей таблицы «challenge» приведена в таблице 3.19.

Таблица 3.19 – Описание полей таблицы «challenge»:

Наименование	Тип данных	Назначение
id_challenge	integer	Идентификатор Workout-челленджа (первичный ключ)
name_challenge	varchar	Название Workout-челленджа
date_creation	date	Дата создания Workout-челленджа
id_creator	integer	Идентификатор создателя Workout-челленджа (внешний ключ)
video_reference	varchar	Ссылка на видео
description	text	Описание Workout-челленджа
is_checked	enum	Проверка Workout-челленджа модератором

Таблица с описанием полей таблицы «challenge_participation» приведена в таблице 3.20.

Таблица 3.20 – Описание полей таблицы «challenge_participation»:

Наименование	Тип данных	Назначение
id	integer	Идентификатор участия Workout-челленджа (первичный ключ)

Продолжение таблицы 3.20

Наименование	Тип данных	Назначение
id_challenge	integer	Идентификатор Workout-челленджа (внешний ключ)
id_participant	integer	Идентификатор участника Workout-челленджа (внешний ключ)
date_participation	date	Дата участия спортсмена в Workout-челлендже
video_reference	varchar	Ссылка на виде
is_checked_participation	enum	Проверка Workout-челленджа модератором

Таблица с описанием полей таблицы «competition» приведена в таблице 3.21.

Таблица 3.21 – Описание полей таблицы «competition»:

Наименование	Тип данных	Назначение
id_competition	integer	Идентификатор соревнования (первичный ключ)
competition_name	varchar	Название соревнования
city	varchar	Название города, в котором будут проходить соревнования

Продолжение таблицы 3.21

Наименование	Тип данных	Назначение
id_creator	integer	Идентификатор создателя соревнования (внешний ключ)
id_type_competition	integer	Идентификатор типа соревнования
date	date	Дата проведения соревнования
contribution	float	Размер денежного взноса
age_from	integer	Нижняя граница разрешенного возраста спортсмена для принятия участия в соревновании
age_to	integer	Верхняя граница разрешенного возраста спортсмена для принятия участия в соревновании
description	text	Описание соревнования
is_checked	enum	Проверка соревнования модератором

Таблица с описанием полей таблицы «competition_award» приведена в таблице 3.22.

Таблица 3.22 – Описание полей таблицы «competition_award»:

Наименование	Тип данных	Назначение
id_competition	integer	Идентификатор соревнования (первичный и внешний ключ)
id_award	integer	Идентификатор награды (внешний ключ)

Таблица с описанием полей таблицы «competition_protocol» приведена в таблице 3.23.

Таблица 3.23 – Описание полей таблицы «competition_protocol»:

Наименование	Тип данных	Назначение
id_competition	integer	Идентификатор соревнования (первичный и внешний ключ)
id_user	integer	Идентификатор соревнования (первичный и внешний ключ)
place	integer	Место, занятое спортсменом на соревнованиях
average_mark	tiny integer	Средняя оценка
total_amount	integer	Итоговое количество

Продолжение таблицы 3.23

Наименование	Тип данных	Назначение
time	time	Время

Таблица с описанием полей таблицы «competition_sponsor» приведена в таблице 3.24.

Таблица 3.24 – Описание полей таблицы «competition_ sponsor»:

Наименование	Тип данных	Назначение
id_competition	integer	Идентификатор соревнования (первичный и внешний ключ)
id_sponsor	integer	Идентификатор спонсора (первичный и внешний ключ)

Таблица с описанием полей таблицы «element» приведена в таблице 3.25.

Таблица 3.25 – Описание полей таблицы «element»:

Наименование	Тип данных	Назначение
id	integer	Идентификатор Workout-элемента (первичный ключ)
name	varchar	Название Workout-элемента
level	enum	Уровень сложности элемента

Продолжение таблицы 3.25

Наименование	Тип данных	Назначение
image	varchar	Имя изображения элемента

Таблица с описанием полей таблицы «exercise» приведена в таблице 3.26.

Таблица 3.26 – Описание полей таблицы «exercise»:

Наименование	Тип данных	Назначение
id	integer	Идентификатор упражнения (первичный ключ)
name_exercise	varchar	Название упражнения

Таблица с описанием полей таблицы «exercise_fight» приведена в таблице 3.27.

Таблица 3.27 – Описание полей таблицы «exercise_fight»:

Наименование	Тип данных	Назначение
id_fight	integer	Идентификатор Workout-зарубы (первичный и внешний ключ)
id_exercise	integer	Идентификатор упражнения (первичный и внешний ключ)
amount	integer	Количество сделанных повторов упражнения спортсменом

Продолжение таблицы 3.27

Наименование	Тип данных	Назначение
time	integer	Время, за которое было сделано упражнения спортсменом.
bonus_exercise	enum	Вид упражнения

Таблица с описанием полей таблицы «fight» приведена в таблице 3.28.

Таблица 3.28 – Описание полей таблицы «fight»:

Наименование	Тип данных	Назначение
id	integer	Идентификатор Workout-зарубы (первичный ключ)
city	varchar	Название города, в котором будет проходить Workout-заруба
playground_address	varchar	Адрес площадки, на которой будет проходить Workout-заруба
date_fight	date	Дата проведения Workout-зарубы

Таблица с описанием полей таблицы «fight_offer» приведена в таблице 3.29.

Таблица 3.29 – Описание полей таблицы «fight_offer»:

Наименование	Тип данных	Назначение
id_offer	integer	Идентификатор предложения Workout-зарубы (первичный ключ)
id_first_user	integer	Идентификатор первого участника Workout-зарубы
id_second_user	integer	Идентификатор второго участника Workout-зарубы
city	varchar	Город, в котором будет проведение Workout-зарубы
playground_address	varchar	Адрес площадки, на которой будет проведение Workout-зарубы
date_fight	date	Дата Workout-зарубы

Таблица с описанием полей таблицы «fight_offer_exercise» приведена в таблице 3.30.

Таблица 3.30 – Описание полей таблицы «fight_offer_exercise»:

Наименование	Тип данных	Назначение
id_fight_offer	integer	Идентификатор предложения Workout-зарубы (первичный и внешний ключ)
id_exercise	integer	Идентификатор упражнения Workout-зарубы
amount	integer	Количество повторов упражнения Workout-зарубы
bonus_exercise	enum	Вид упражнения Workout-зарубы

Таблица с описанием полей таблицы «long-term_goal» приведена в таблице 3.31.

Таблица 3.31 – Описание полей таблицы «long-term_goal»:

Наименование	Тип данных	Назначение
id_user	integer	Идентификатор пользователя (внешний ключ)
name	varchar	Долгосрочная цель

Таблица с описанием полей таблицы «mark_sports_facility» приведена в таблице 3.32.

Таблица 3.32 – Описание полей таблицы «mark_sports_facility»:

Наименование	Тип данных	Назначение
id	integer	Идентификатор оценки спортивного объекта (первичный ключ)
id_facility	integer	Идентификатор спортивного объекта
id_user	integer	Идентификатор пользователя
value	tiny integer	Оценка спортивного объекта

Таблица с описанием полей таблицы «migration» приведена в таблице 3.33.

Таблица 3.33 – Описание полей таблицы «migration»:

Наименование	Тип данных	Назначение
version	varchar	Версия миграции (первичный ключ)
apply_time	integer	Дата создания миграции

Таблица с описанием полей таблицы «news» приведена в таблице 3.34.

Таблица 3.34 – Описание полей таблицы «news»:

Наименование	Тип данных	Назначение
id	integer	Идентификатор новости (первичный ключ)
id_author	integer	Идентификатор автора новости

Продолжение таблицы 3.34

Наименование	Тип данных	Назначение
name	varchar	Название новости
date_publication	date	Дата публикации новости
content	text	Текст новости
main_image	varchar	Имя главного изображения новости

Таблица с описанием полей таблицы «short-term_goal» приведена в таблице 3.35.

Таблица 3.35 – Описание полей таблицы «short-term_goal»:

Наименование	Тип данных	Назначение
id_user	integer	Идентификатор пользователя (первичный и внешний ключ)
name	varchar	Долгосрочная цель

Таблица с описанием полей таблицы «specialization» приведена в таблице 3.36.

Таблица 3.36 – Описание полей таблицы «specialization»:

Наименование	Тип данных	Назначение
id	integer	Идентификатор специализации (первичный ключ)
name	varchar	Название специализации
description	text	Описание специализации

Таблица с описанием полей таблицы «sponsor» приведена в таблице 3.37.

Таблица 3.37 – Описание полей таблицы «sponsor»:

Наименование	Тип данных	Назначение
id	integer	Идентификатор спонсора (первичный ключ)
name	varchar	Название спонсора

Таблица с описанием полей таблицы «sports_facility» приведена в таблице 3.38.

Таблица 3.38 – Описание полей таблицы «sports_facility»:

Наименование	Тип данных	Назначение
id_facility	integer	Идентификатор спортивного объекта (первичный ключ)
name_facility	varchar	Название спортивного объекта
address	varchar	Адрес спортивного объекта
id_author	integer	Идентификатор создателя спортивного объекта (внешний ключ)
latitude	float	Широта спортивного объекта на карте
longitude	float	Долгота спортивного объекта на карте
main_image	varchar	Имя главного изображения спортивного объекта

Продолжение таблицы 3.38

Наименование	Тип данных	Назначение
description	varchar	Описание спортивного объекта
is_checked	enum	Статус проверки спортивного объекта модератором

Таблица с описанием полей таблицы «sports_facility_comment» приведена в таблице 3.39.

Таблица 3.39 – Описание полей таблицы «sports_facility_comment»:

Наименование	Тип данных	Назначение
id	integer	Идентификатор отзыва о спортивном объекте (первичный ключ)
id_user	integer	Идентификатор пользователя, написавшего отзыв о спортивном объекте (внешний ключ)
id_facility	integer	Идентификатор спортивного объекта (внешний ключ)
content	text	Отзыв о спортивном объекте

Таблица с описанием полей таблицы «type_competition» приведена в таблице 3.40.

Таблица 3.40 – Описание полей таблицы «type_competition»:

Наименование	Тип данных	Назначение
id	integer	Идентификатор отзыва о типе соревнований (первичный ключ)
type_name	varchar	Название типа соревнований
description	text	Описание типа соревнований

Таблица с описанием полей таблицы «user» приведена в таблице 3.41.

Таблица 3.41 – Описание полей таблицы «user»:

Наименование	Тип данных	Назначение
id	integer	Идентификатор пользователя (первичный ключ)
login	varchar	Логин пользователя
password	varchar	Пароль пользователя
image_avatar	varchar	Имя изображения, представляющего пользователя
auth_key	varchar	Ключ авторизации
email	varchar	E-mail пользователя
verified	enum	Статус верификации пользователя
token	varchar	Токен для верификации пользователя

Продолжение таблицы 3.41

Наименование	**Тип данных**	**Назначение**
last_name	varchar	Фамилия пользователя
name	varchar	Имя пользователя
patronymic	varchar	Отчество пользователя
date_birth	date	Дата рождения пользователя
experience	integer	Стаж тренировок пользователя
growth	integer	Рост пользователя
weight	integer	Вес пользователя
max_pulling_up	integer	Максимально возможное количество подтягиваний спортсмена
max_push_up_bars	integer	Максимально возможное количество отжиманий спортсмена на брусьях
max_push_up	integer	Максимально возможное количество отжиманий спортсмена от пола
max_bench_press	integer	Максимально возможный вес штанги при жиме лежа спортсмена
reference_youtube	varchar	Ссылка на аккаунт в Youtube

Продолжение таблицы 3.41

Наименование	Тип данных	Назначение
reference_vk	varchar	Ссылка на аккаунт в VKontakte

Таблица с описанием полей таблицы «user_book» приведена в таблице 3.42.

Таблица 3.42 – Описание полей таблицы «user_book»:

Наименование	Тип данных	Назначение
id_user	integer	Идентификатор пользователя (первичный и внешний ключ)
id_book	integer	Идентификатор книги (первичный и внешний ключ)
mark_book	integer	Оценка книги

Таблица с описанием полей таблицы «user_competition» приведена в таблице 3.43.

Таблица 3.43 – Описание полей таблицы «user_competition»:

Наименование	Тип данных	Назначение
id_user	integer	Идентификатор пользователя (первичный и внешний ключ)
id_competition	integer	Идентификатор соревнования (первичный и внешний ключ)

Таблица с описанием полей таблицы «user_element» приведена в таблице 3.44.

Таблица 3.44 – Описание полей таблицы «user_element»:

Наименование	Тип данных	Назначение
id_user	integer	Идентификатор пользователя (первичный и внешний ключ)
id_element	integer	Идентификатор соревнования (первичный и внешний ключ)

Таблица с описанием полей таблицы «user_fight» приведена в таблице 3.45.

Таблица 3.45 – Описание полей таблицы «user_fight»:

Наименование	Тип данных	Назначение
id_user	integer	Идентификатор пользователя (первичный и внешний ключ)
id_fight	integer	Идентификатор Workout-зарубы (первичный и внешний ключ)
is_winner	enum	Результат Workout-зарубы

Таблица с описанием полей таблицы «user_ specialization» приведена в таблице 3.46.

Таблица 3.46 – Описание полей таблицы «user_ specialization»:

Наименование	**Тип данных**	**Назначение**
id_user	integer	Идентификатор пользователя (первичный и внешний ключ)
id_specialization	integer	Идентификатор специализации (первичный и внешний ключ)

Таблица с итогами разработки базы данных приведена в таблице 3.47.

Таблица 3.47 – Итоги разработки базы данных:

Количество таблиц, шт	**Количество полей таблиц, шт**
38	69

4. РЕАЛИЗАЦИЯ

4.1. ФОРМЫ ВВОДА ВЕБ-ПРИЛОЖЕНИЯ

В веб-приложении реализованы следующие *формы ввода*:

1. Форма регистрации пользователя.

В форме ввода отслеживается необходимость заполнения всех полей (в том числе отметки о принятии правил сообщества), уникальность логина и адреса электронной почты, совпадение паролей при первичном и повторном заполнении. Также имеются минимальные и максимальные ограничения количества символов логина и пароля. Адрес электронной почты должен иметь определенный вид. Скриншот формы регистрации пользователя представлен на рисунке 4.1.

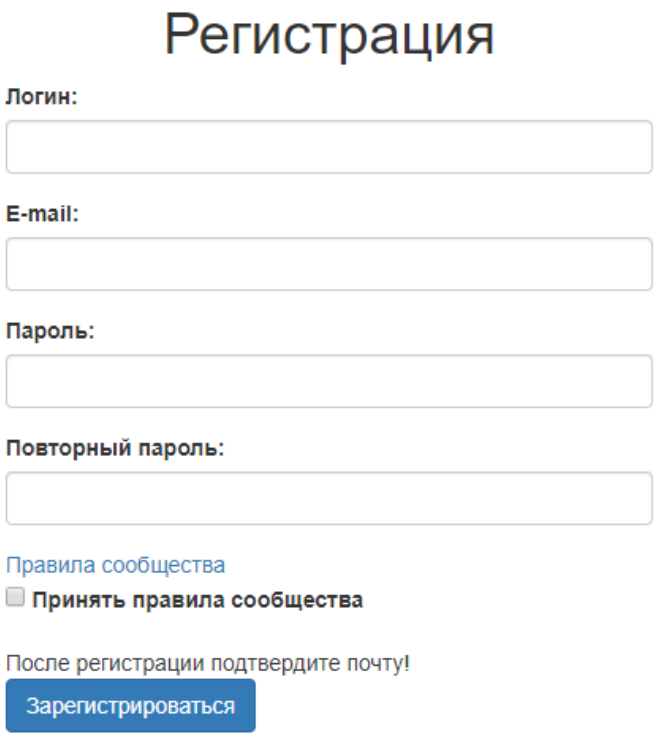

Рисунок 4.1 – Форма регистрации пользователя

2. Форма авторизации пользователя.

В форме ввода отслеживается необходимость заполнения всех полей, проверяется корректность вводимых данных. Скриншот формы авторизации пользователя представлен на рисунке 4.2.

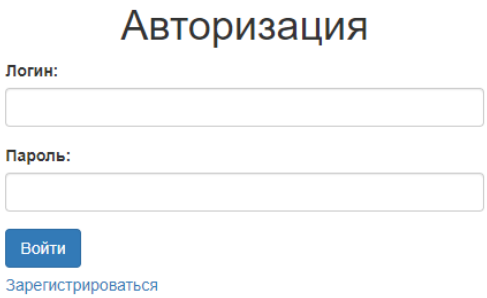

Рисунок 4.2 – Форма авторизации пользователя

3. Форма редактирования профиля пользователя.

В форме ввода не отслеживается необходимость заполнения всех полей, имеются минимальные и максимальные ограничения количества символов фамилии, имени, отчества, стажа, роста и веса. Дата рождения должна иметь определенный вид. Вводимый тип данных значений стажа, роста и веса – целочисленный, имеются максимальное и минимальное ограничения значений чисел. Отправляемый файл обязательно должен быть картинкой. Скриншот формы редактирования профиля пользователя представлен на рисунке 4.3.

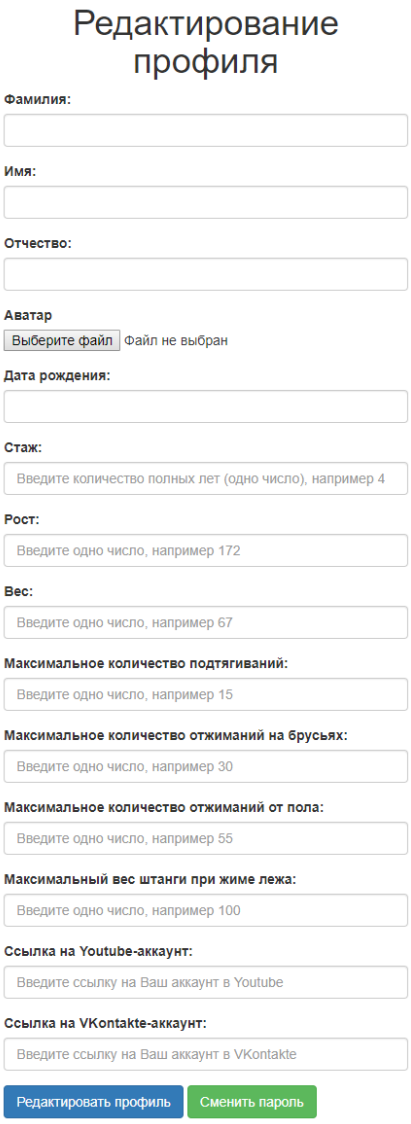

Рисунок 4.3 – Форма редактирования профиля пользователя

4. Форма обратной связи.

В форме ввода отслеживается необходимость заполнения всех полей. Имеются минимальные и максимальные ограничения количества символов имени и отзыва. Адрес электронной почты должен иметь определенный вид. Скриншот формы обратной связи представлен на рисунке 4.4.

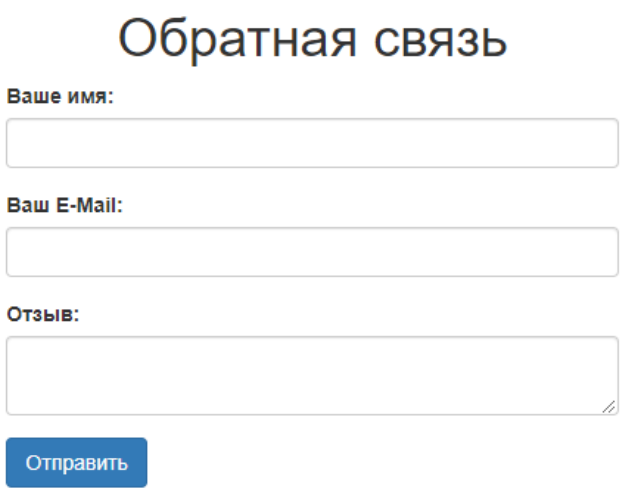

Рисунок 4.4 – Форма обратной связи

5. Форма создания соревнования.

В форме ввода отслеживается необходимость заполнения всех полей, имеются минимальные и максимальные ограничения количества символов названия соревнования, города. Значения возрастов должны иметь целочисленный тип данных, есть ограничение минимального и максимального значения числа. Дата должна иметь определенный вид. Скриншот формы создания соревнования представлен на рисунке 4.5.

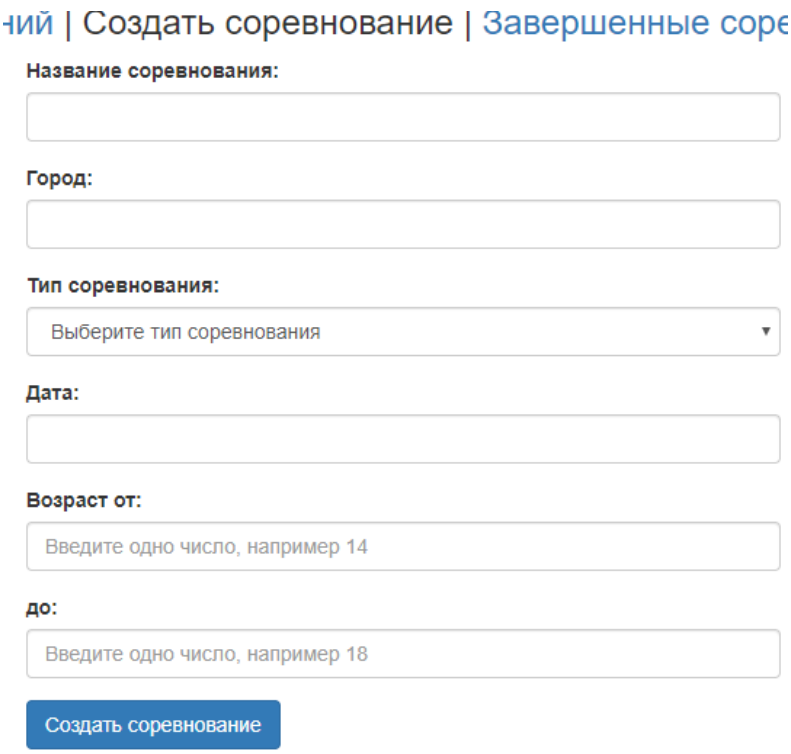

Рисунок 4.5 – Форма создания соревнования

6. Форма вызова пользователя на Workout-зарубу.

В форме ввода отслеживается необходимость заполнения всех полей, имеются минимальные и максимальные ограничения количества символов названия города, адреса площадки. Дата «зарубы» должна иметь определенный вид. Скриншот формы вызова пользователя на Workout-зарубу представлен на рисунке 4.6.

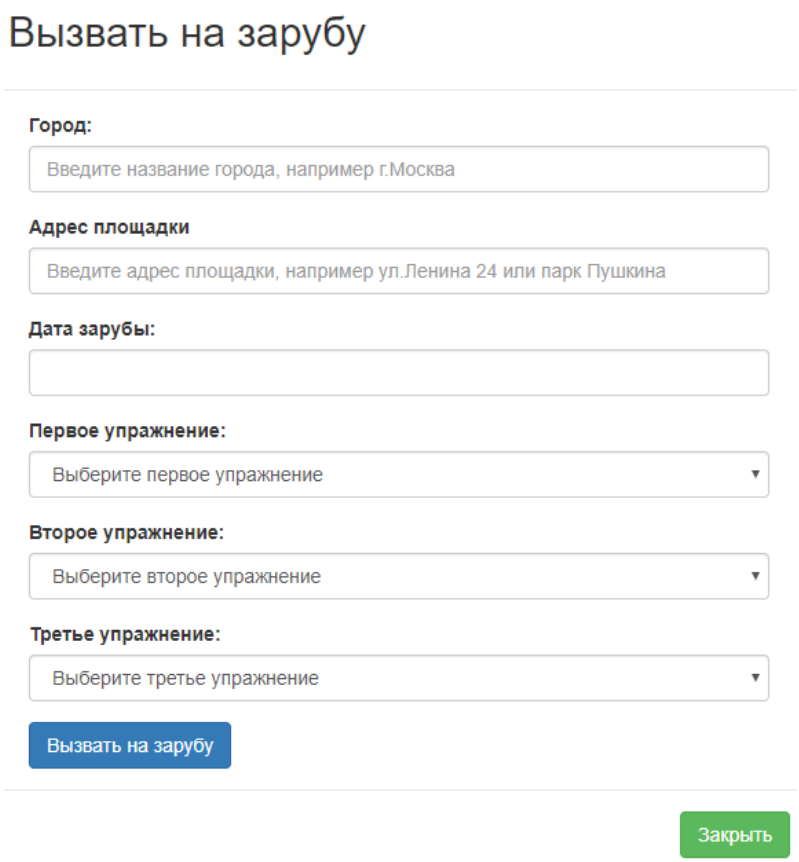

Рисунок 4.6 – Форма вызова пользователя на Workout-зарубу

7. Форма создания Workout-челленджа.

В форме ввода отслеживается необходимость заполнения всех полей, имеются минимальные и максимальные ограничения количества символов названия челленджа, ссылки на видео, описании челленджа. Скриншот формы создания Workout-челленджа представлен на рисунке 4.7.

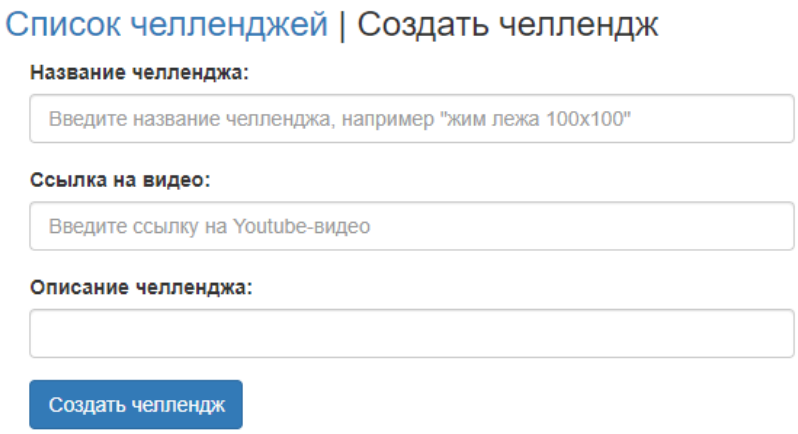

Рисунок 4.7 – Форма создания Workout-челленджа

Форма принятия участия пользователя в Workout-челлендже является аналогичной (см. рисунок 4.7).

8. Форма создания статьи.

В форме ввода отслеживается необходимость заполнения всех полей (кроме поля «рекламная статья»), имеются минимальные и максимальные ограничения количества символов названия статьи, аннотации и контента. Файл, загружаемый пользователем, должен быть изображением. Скриншоты формы создания статьи представлены на рисунке 4.8 и рисунке 4.9.

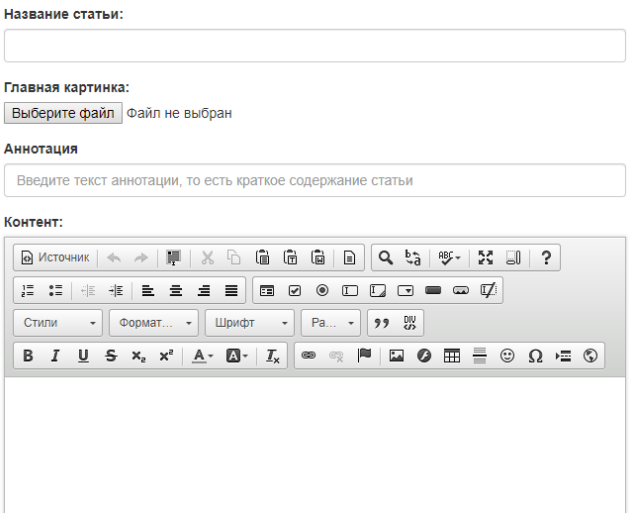

Рисунок 4.8 – Форма создания статьи

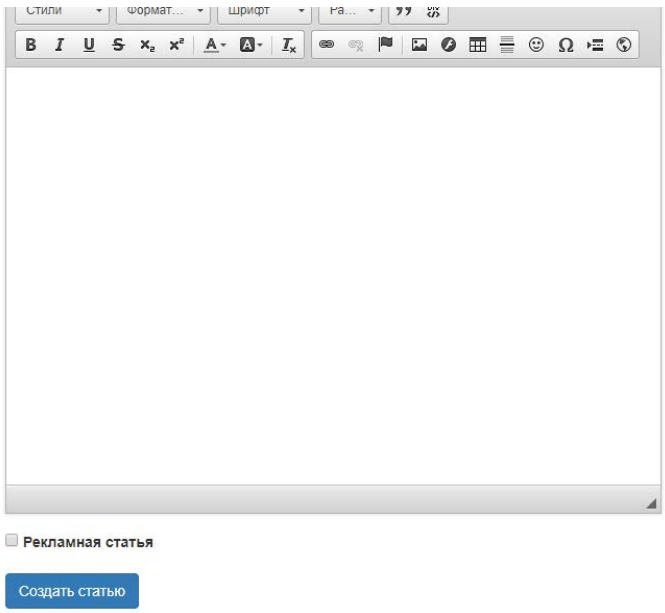

Рисунок 4.9 – Форма создания статьи

9. Форма создания спортивного объекта.

В форме ввода отслеживается необходимость заполнения всех полей (кроме поля «описание»), имеются минимальные и максимальные ограничения количества символов названия и адреса спортивного объекта. Также имеются максимальные ограничения количества символов описания спортивного объекта. Файл, загружаемый пользователем, должен быть изображением. Тип данных значений широты и долготы – вещественный. Скриншот формы создания спортивного объекта представлен на рисунке 4.10.

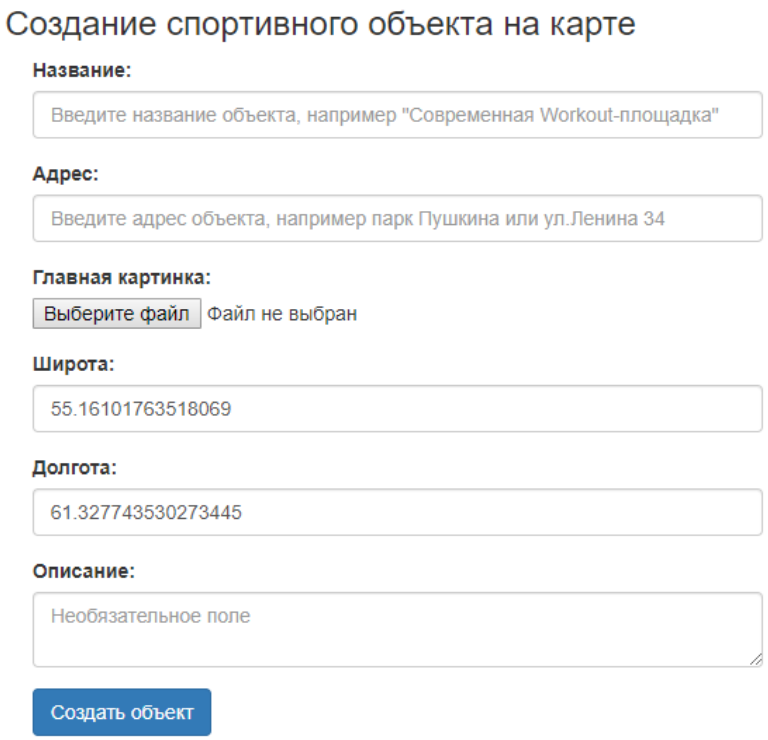

Рисунок 4.10 – Форма создания спортивного объекта

10. Форма создания спортивного объекта.

В форме ввода не отслеживается необходимость заполнения всех полей, имеются минимальные и максимальные ограничения количества символов названия статьи, аннотации и контента. Файл, загружаемый пользователем, должен быть изображением. Скриншот формы редактирования статьи представлен на рисунке 4.11.

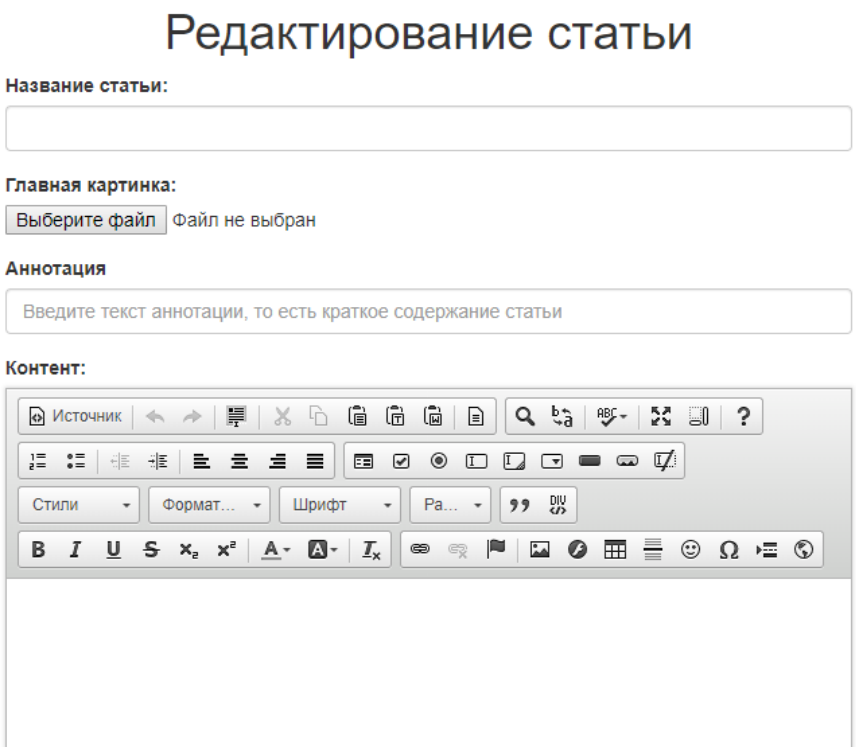

Рисунок 4.11 – Форма редактирования статьи

11. Форма смены пароля.

В форме ввода отслеживается необходимость заполнения всех полей, корректность вводимых данных логина и текущего пароля, совпадение новых паролей при первичном и повторном заполнении. Также имеются минимальные и максимальные ограничения количества символов нового пароля. Скриншот формы смены пароля представлен на рисунке 4.12.

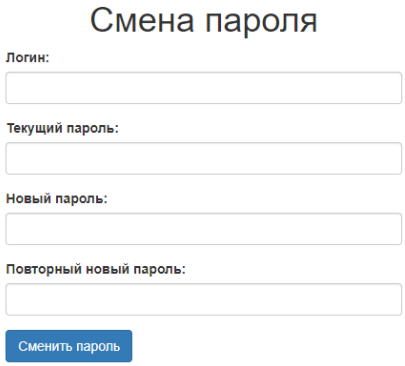

Рисунок 4.12 – Форма смены пароля

12. Форма объявления результатов соревнований.

В форме ввода отслеживается необходимость заполнения всех полей. Вводимые данные в поля формы «ИД пользователя», «ИД соревнований», «место» должны иметь целочисленный тип данных. Скриншот формы объявления результатов соревнований представлен на рисунке 4.13.

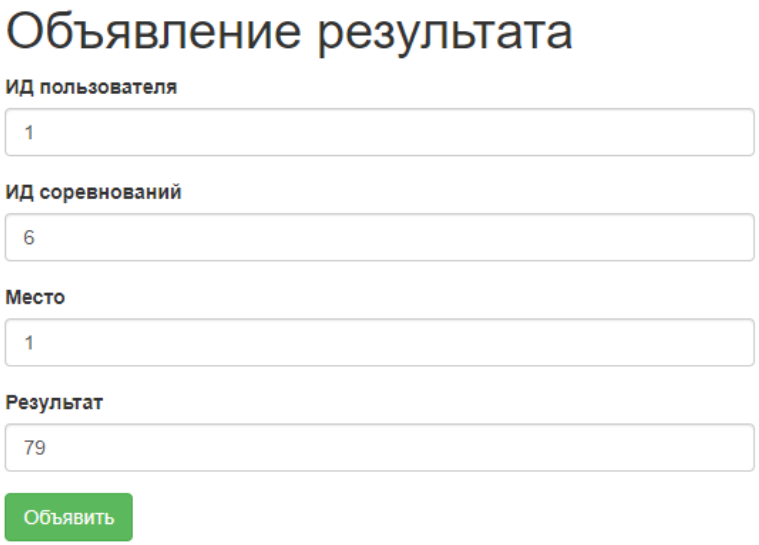

Рисунок 4.13 – Форма объявления результатов соревнований

13. Форма проверки статьи (модерация).

Скриншот формы проверки статьи (модерация) представлен на рисунке 4.14.

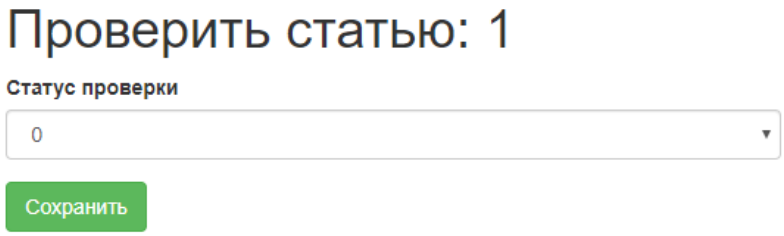

Рисунок 4.14 – Форма проверки статьи (модерация)

Проверенная модератором статья будет иметь статус «1». Аналогичным образом (см. рисунок 4.14) происходят проверки модератором других разделов веб-приложения.

14. Форма написания отзыва о книге.

В форме ввода отслеживается необходимость заполнения всех полей, имеются минимальные и максимальные ограничения количества символов отзыва. Скриншот формы написания отзыва о книге представлен на рисунке 4.15.

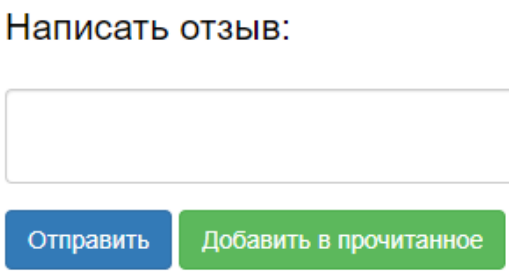

Рисунок 4.15 – Форма формы написания отзыва о книге

15. Форма написания отзыва о спортивном объекте.

В форме ввода отслеживается необходимость заполнения всех полей, имеются минимальные и максимальные ограничения количества символов отзыва. Скриншот формы написания отзыва о спортивном объекте представлен на рисунке 4.16.

Рисунок 4.16 – Форма формы написания отзыва о спортивном объекте

4.2. СТРАНИЦЫ ВЕБ-ПРИЛОЖЕНИЯ

Ниже представлены *страницы веб-приложения (внешний интерфейс)*.

1. Главная страница.

Скриншоты главной страницы веб-приложения представлены на рисунках 4.17 – 4.22.

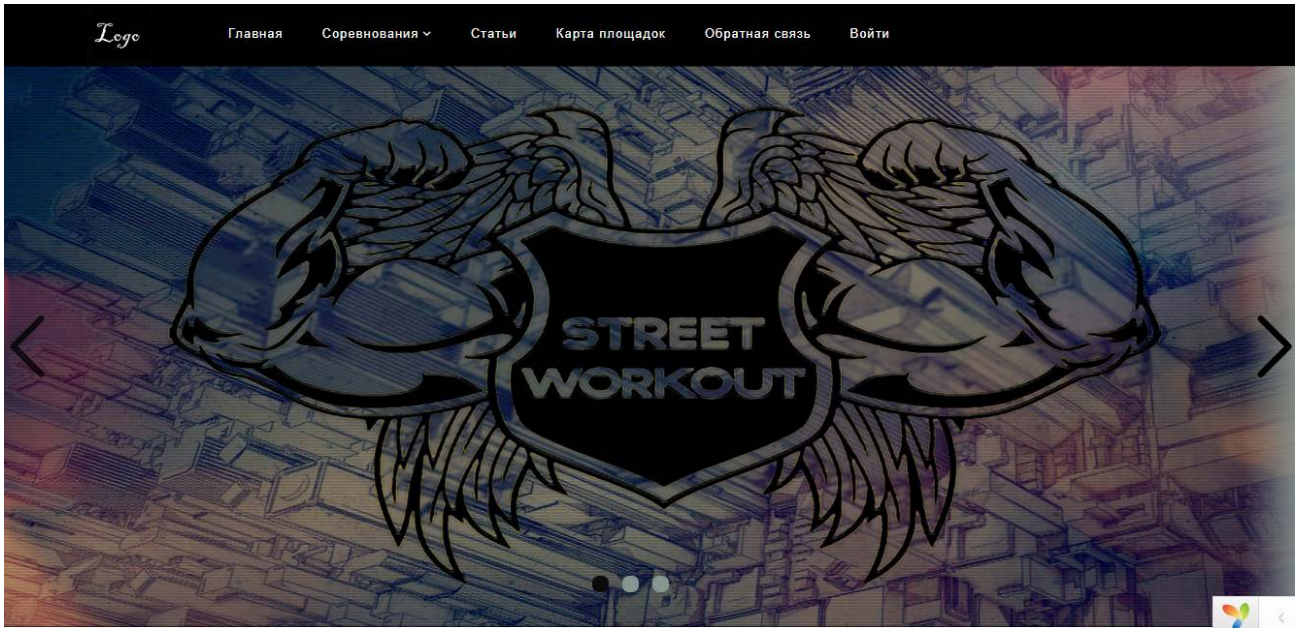

Рисунок 4.17 – Главное меню и слайдер главной страницы, слайд 1

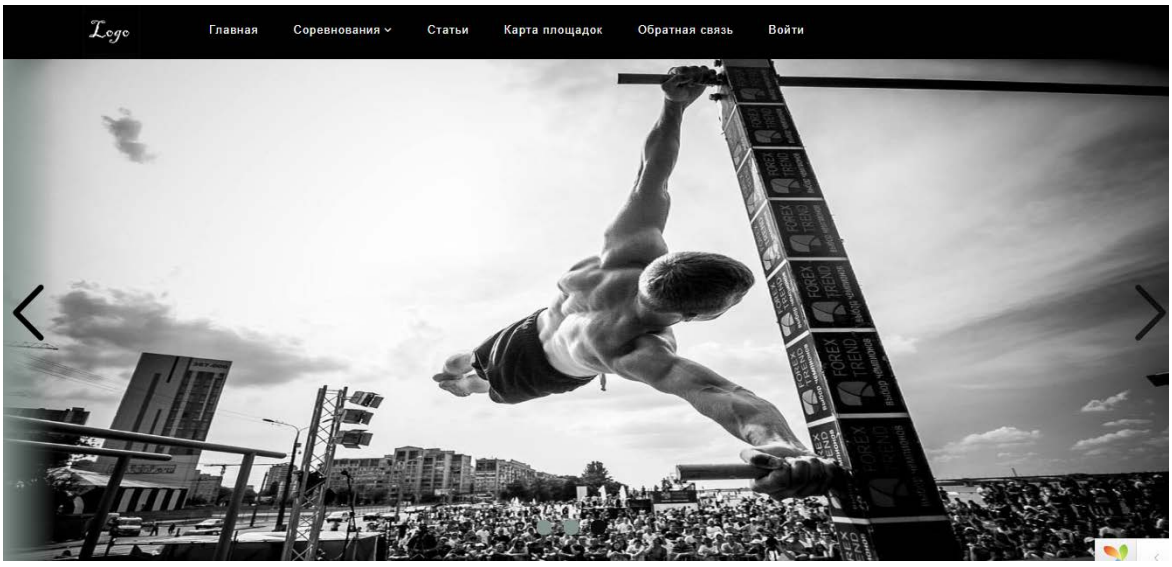

Рисунок 4.18 – Главное меню и слайдер главной страницы, слайд 2

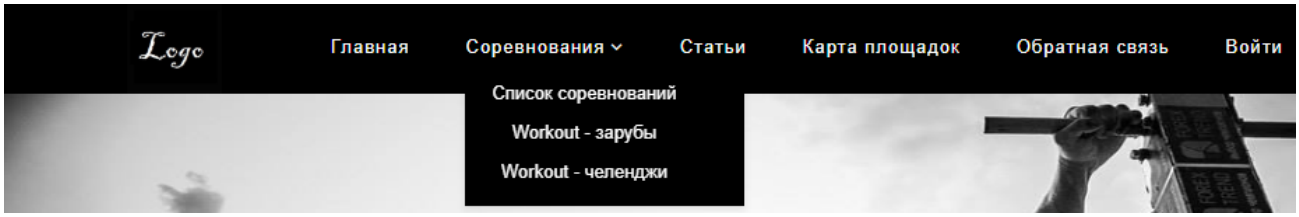

Рисунок 4.19 – Главное меню главной страницы, раздел соревнований

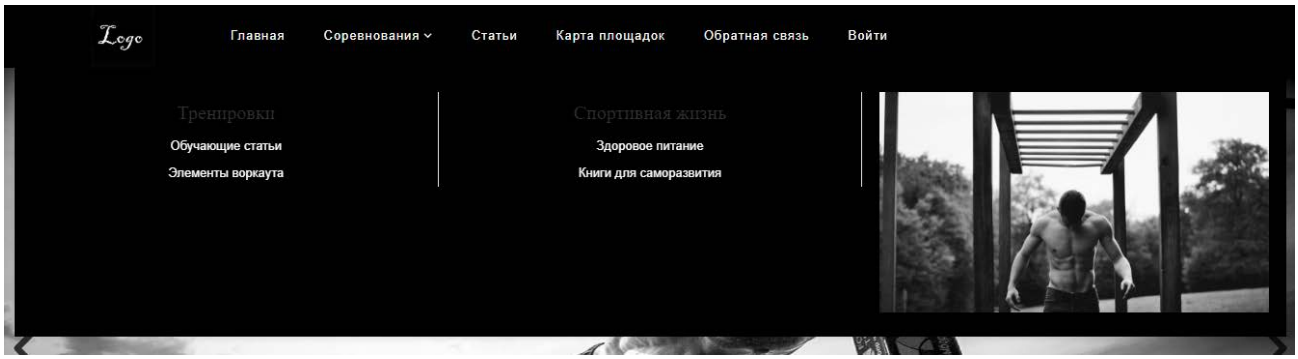

Рисунок 4.20 – Главное меню главной страницы, раздел статей

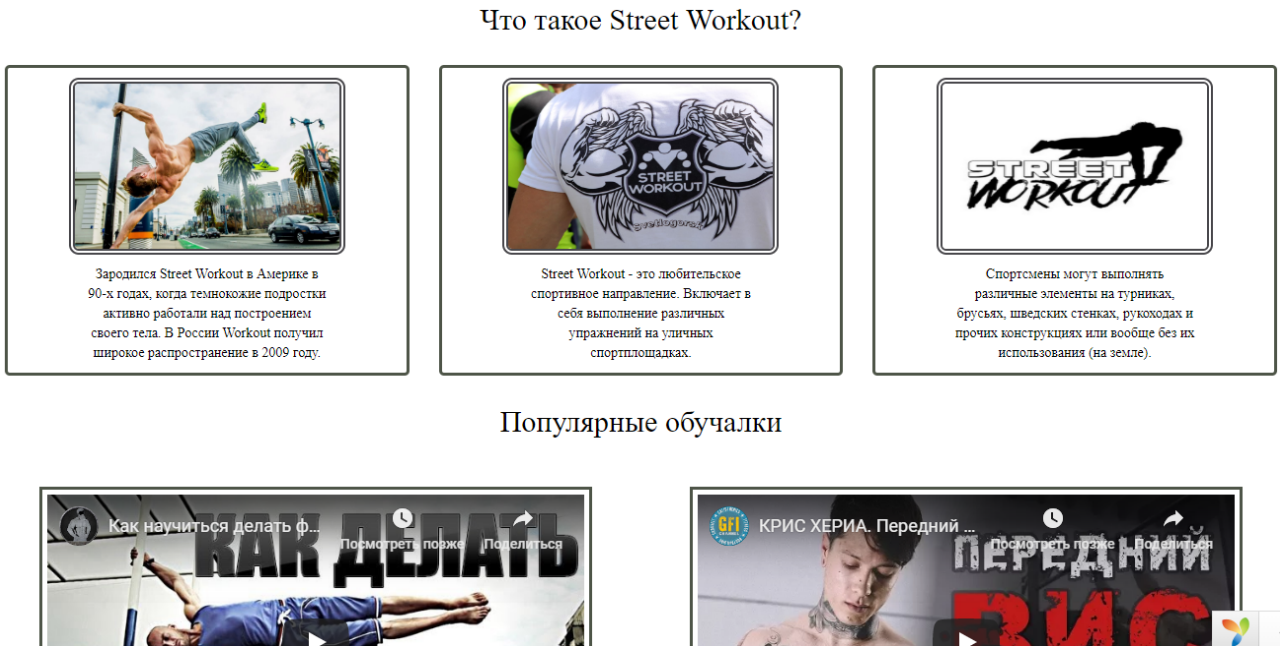

Рисунок 4.21– Главная страница веб-приложения, общие сведения о предметной области

Рисунок 4.22– Главная страница веб-приложения, обучающие материалы

2. Статьи.

Скриншоты списка обучающих статей представлены на рисунках 4.23, 4.24.

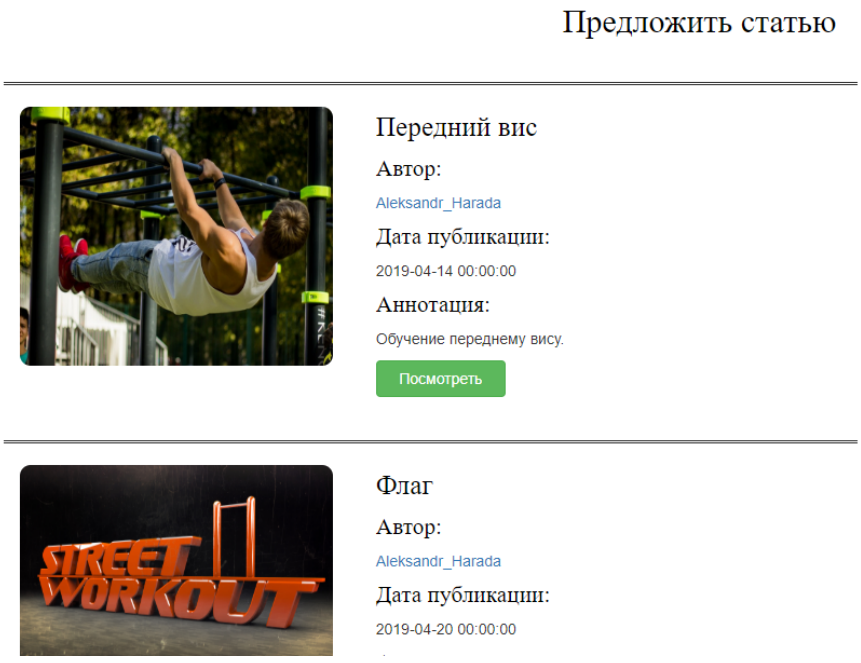

Рисунок 4.23 – Список обучающих статей, часть 1

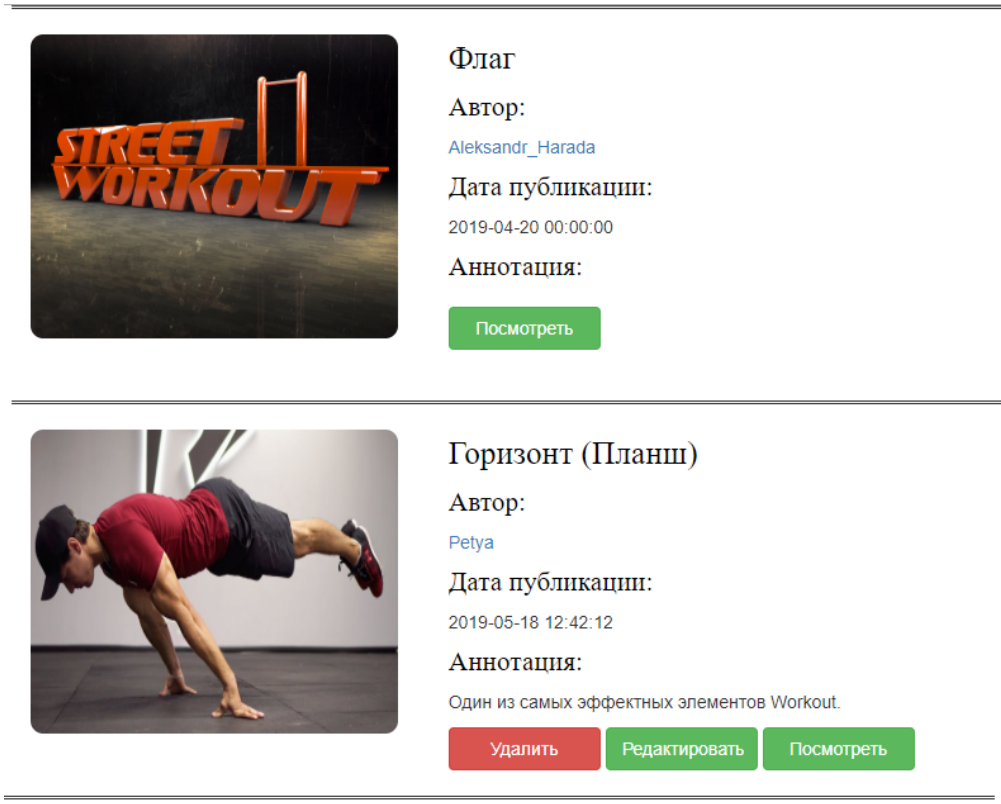

Рисунок 4.24 – Список обучающих статей, часть 2

Скриншот страницы просмотра обучающей статьи представлен на рисунке 4.25.

Рисунок 4.25 – Страница просмотра обучающей статьи

Скриншот списка статей о правильном питании представлен на рисунке 4.26.

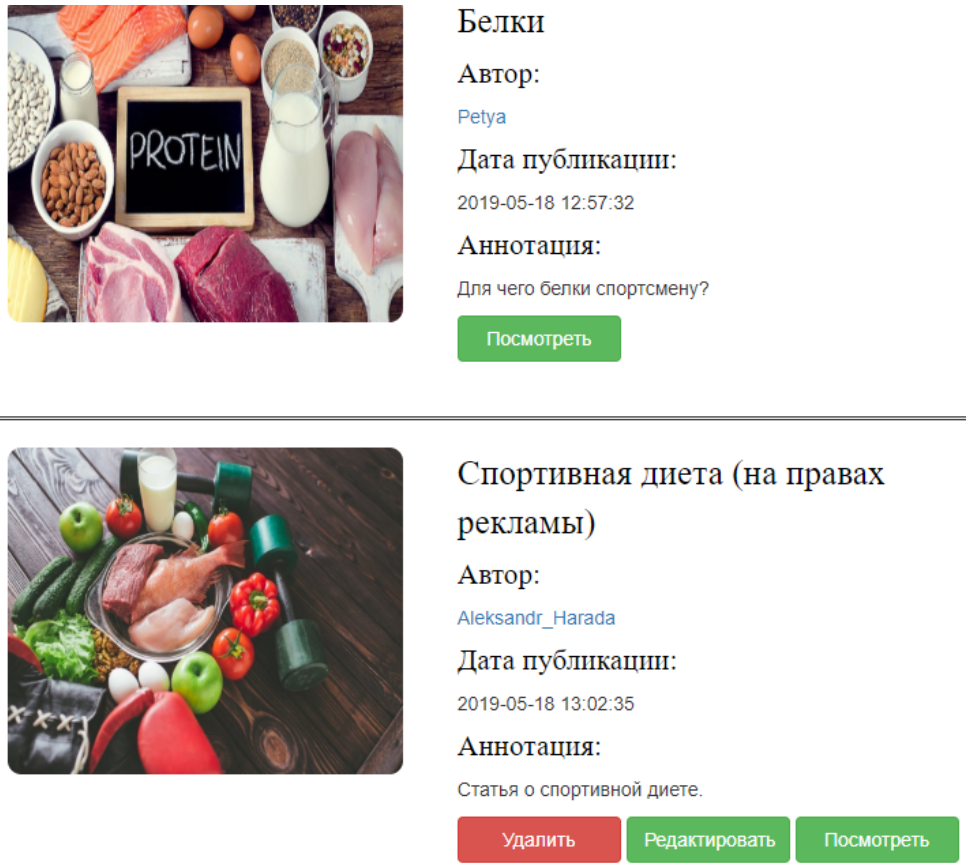

Рисунок 4.26 – Список статей о правильном питании

3. Книги.

Скриншот списка книг представлен на рисунке 4.27.

Рисунок 4.27 – Список книг

Скриншот страницы просмотра книги представлен на рисунке 4.28.

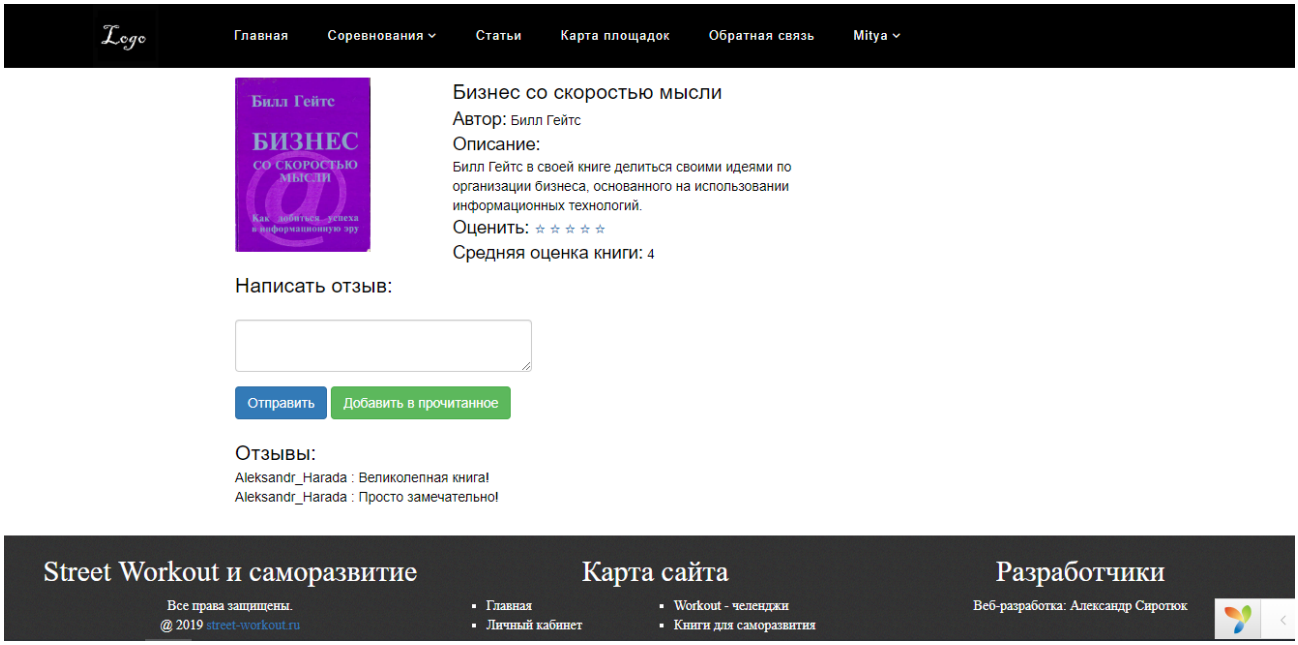

Рисунок 4.28 – Страница просмотра книги

4. Workout-элементы.

Скриншот списка уровней сложности Workout-элементов представлен на рисунке 4.29.

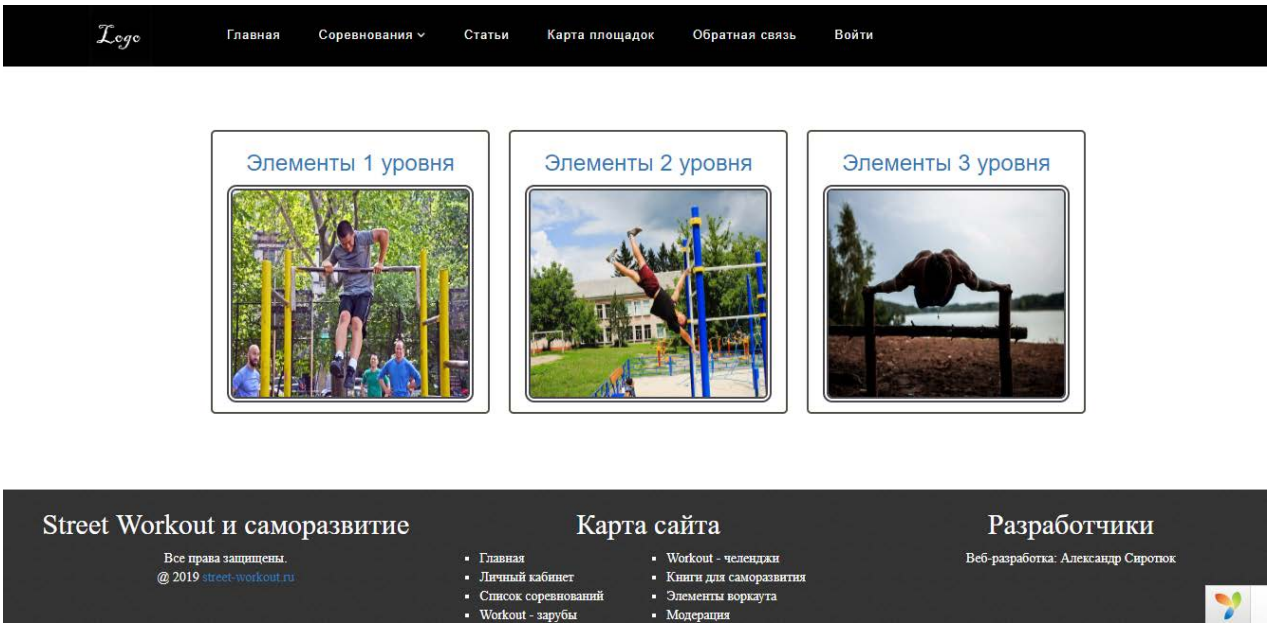

Рисунок 4.29 – Список уровней сложности Workout-элементов

Скриншот списка Workout-элементов первого уровня сложности представлен на рисунке 4.30.

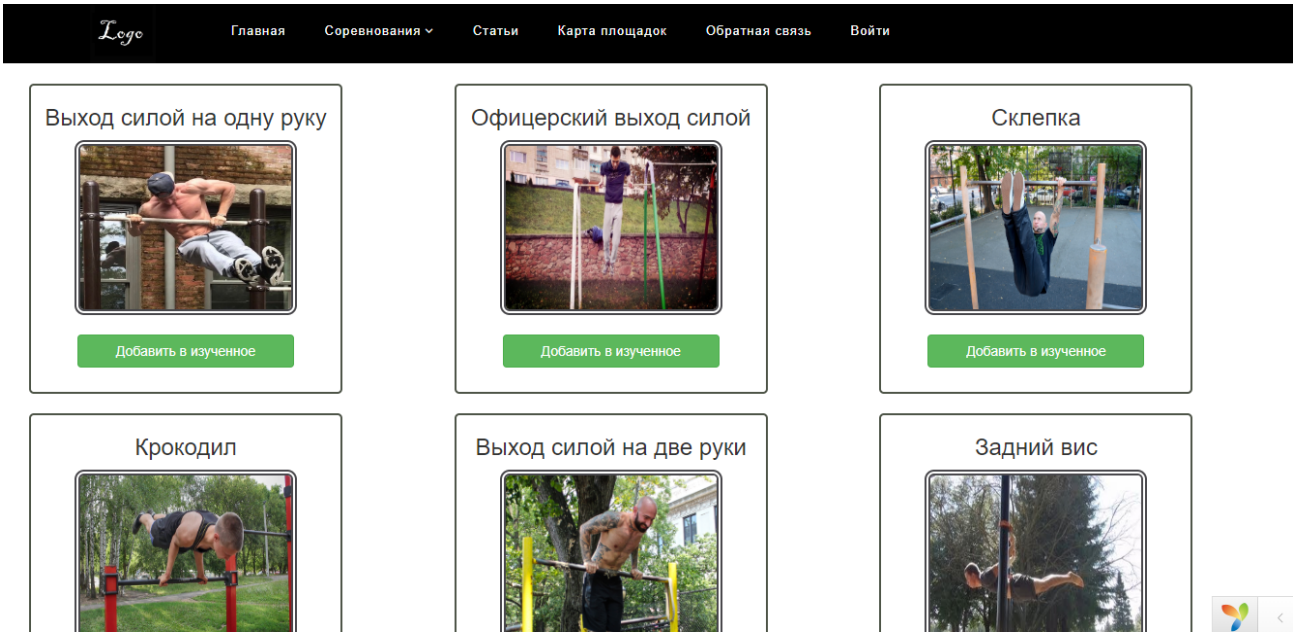

Рисунок 4.30 – Список Workout-элементов первого уровня сложности

5. Спортивные объекты.

Скриншот карты с нанесенными на нее метками спортивных объектов представлен на рисунке 4.31 (метки обведены красным карандашом).

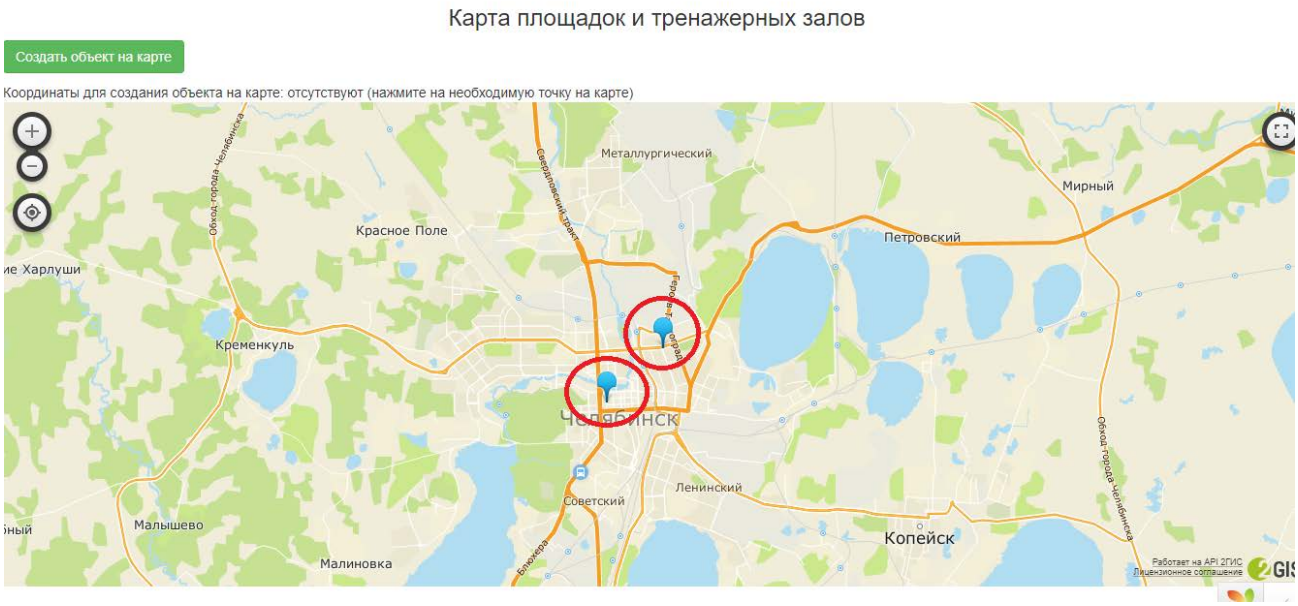

Рисунок 4.31 – Карта с нанесенными на нее метками спортивных объектов

Скриншот карты с раскрытой меткой спортивного объекта представлен на рисунке 4.32.

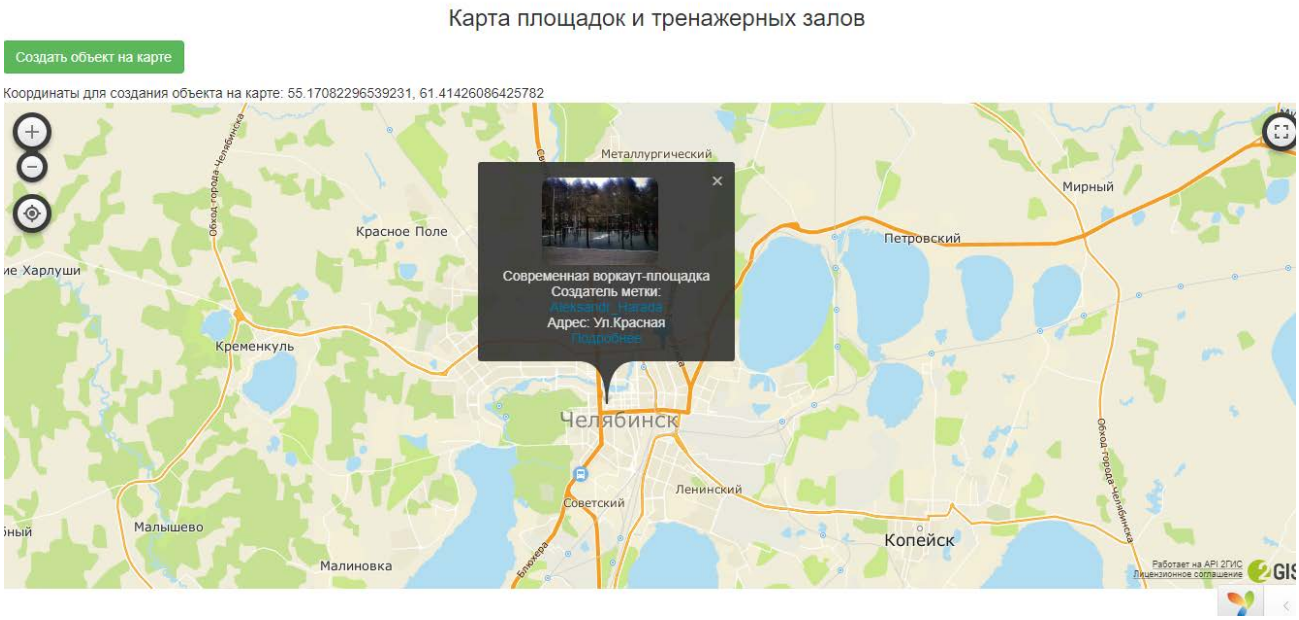

Рисунок 4.32 – Карта с раскрытой меткой спортивного объекта

Скриншот просмотра страницы спортивного объекта представлен на рисунке 4.33.

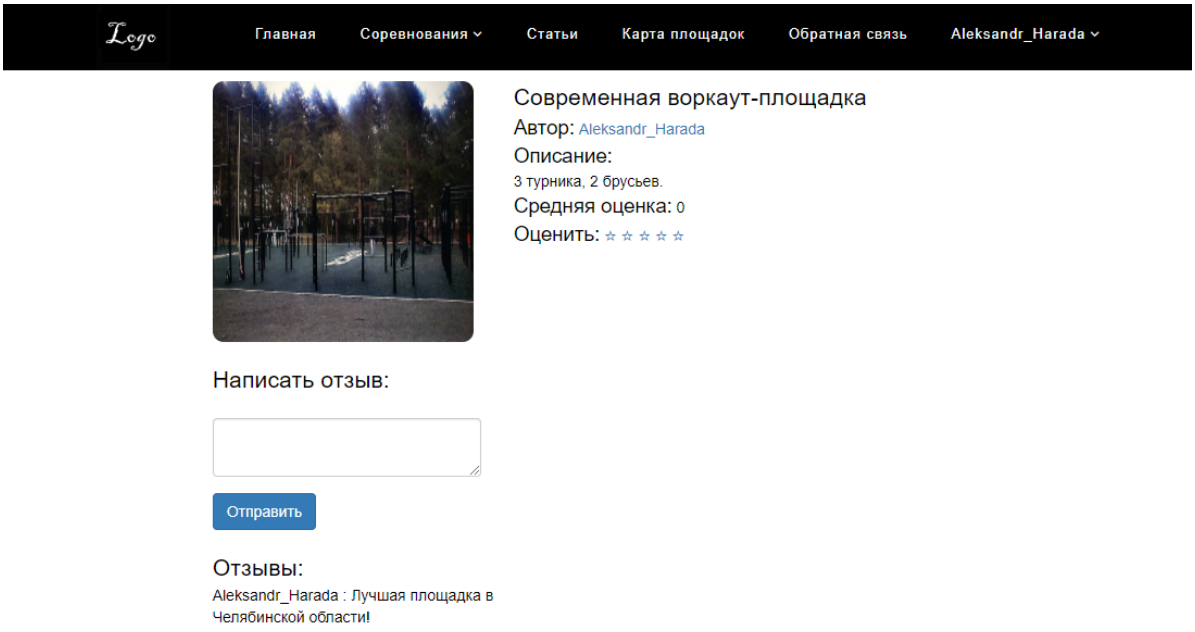

Рисунок 4.33 – Просмотр страницы спортивного объекта

6. Workout-соревнования.

Скриншот списка незавершенных Workout-соревнований представлен на рисунке 4.34.

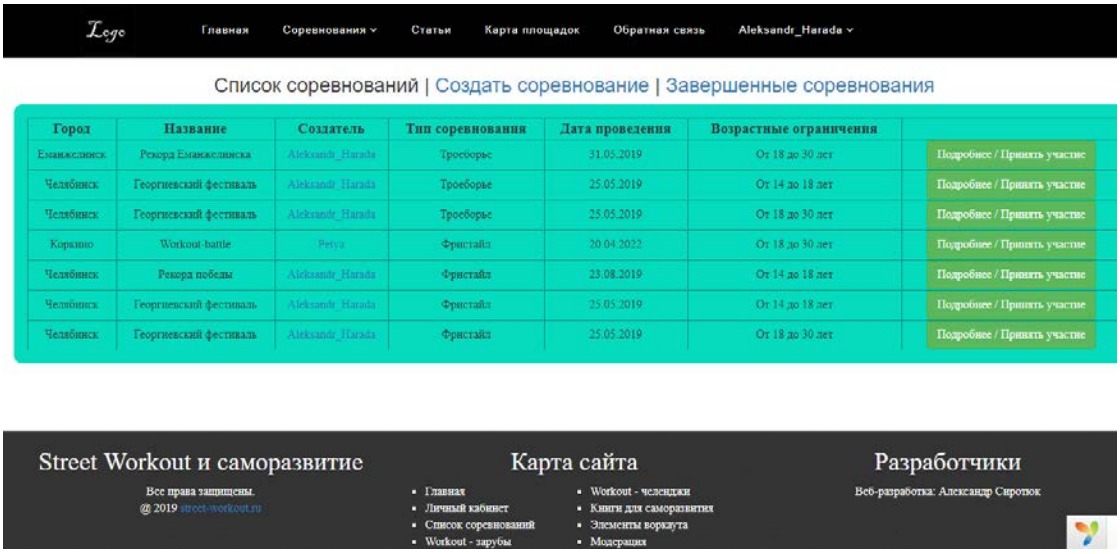

Рисунок 4.34 – Список незавершенных Workout-соревнований

Скриншот списка завершенных Workout-соревнований представлен на рисунке 4.35

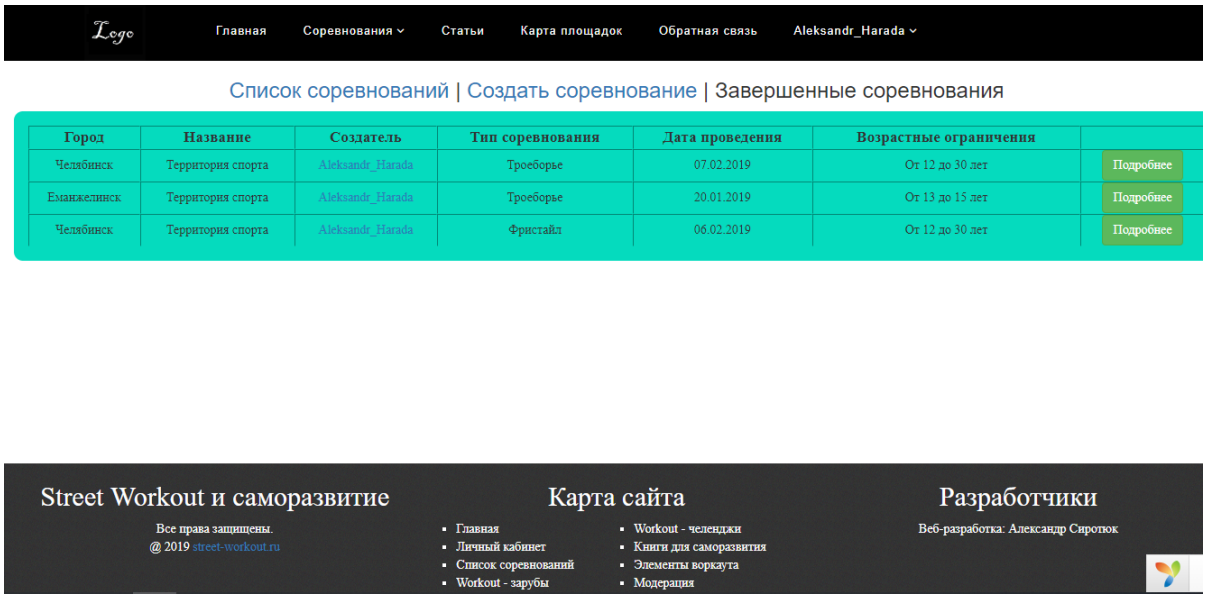

Рисунок 4.35 – Список завершенных Workout-соревнований веб-приложения

Скриншот просмотра информации о Workout-соревновании представлен на рисунке 4.36

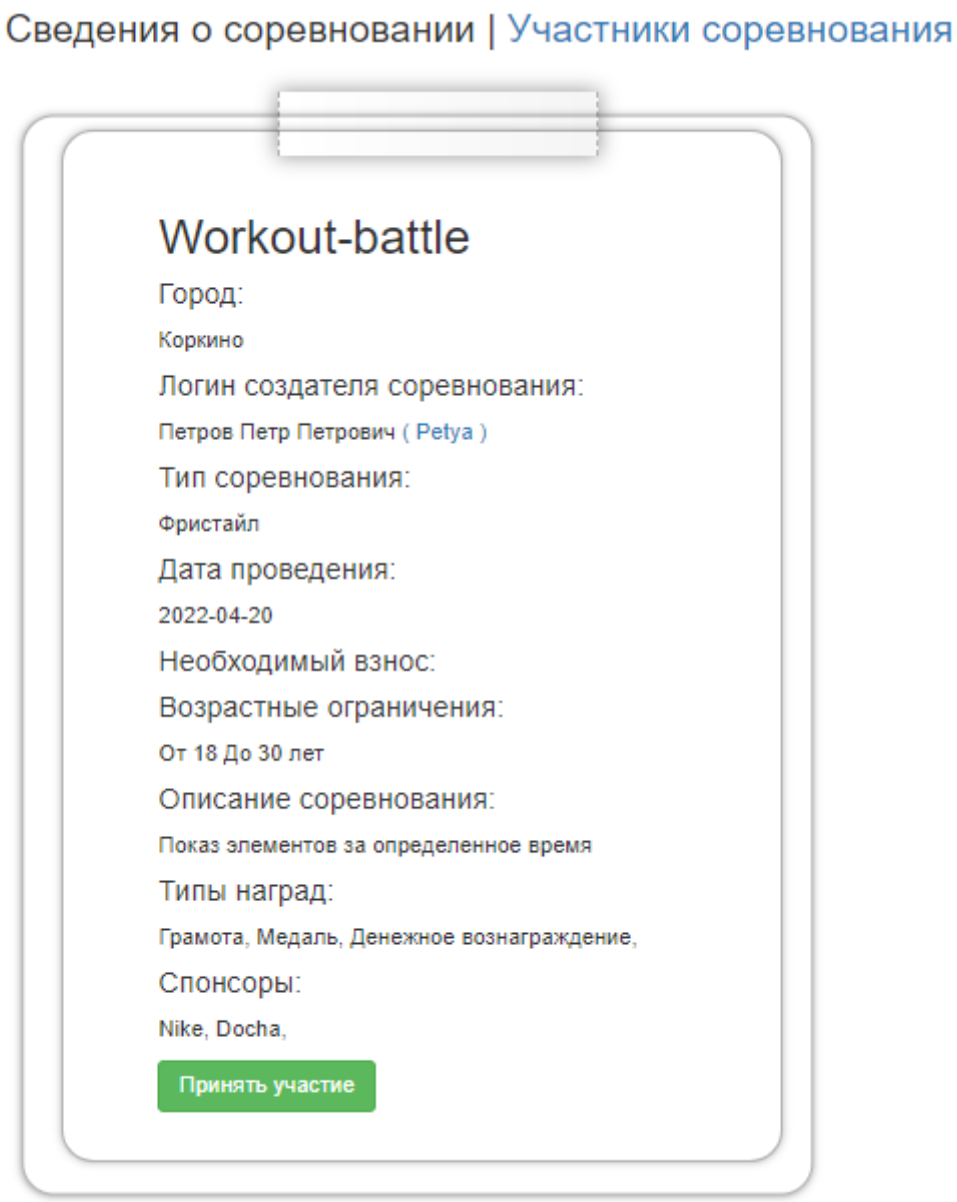

Рисунок 4.36 – Просмотр информации о Workout-соревновании

Скриншот просмотра списка участников Workout-соревнования представлен на рисунке 4.37.

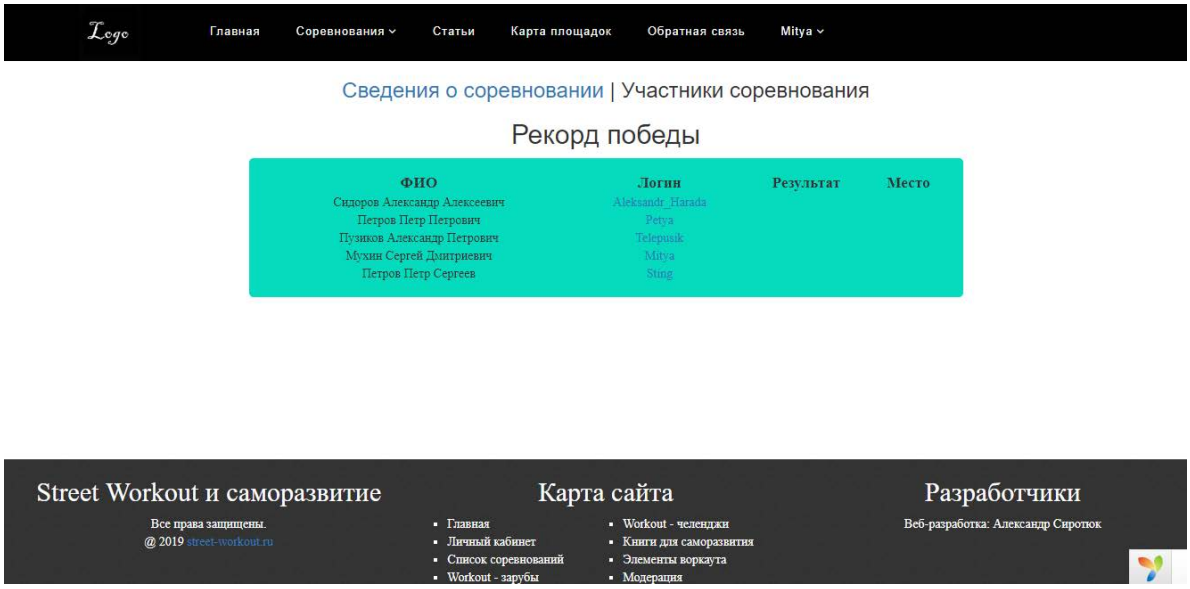

Рисунок 4.37 – Просмотр списка участников Workout-соревнования

Скриншот просмотра результатов Workout-соревнования представлен на рисунке 4.38.

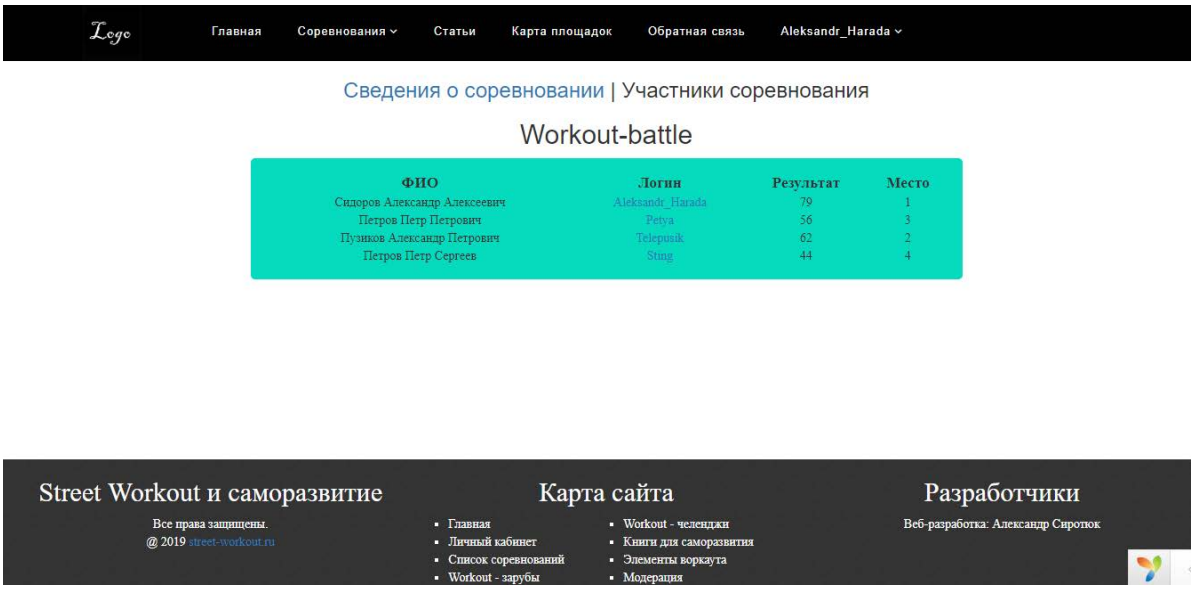

Рисунок 4.38 – Просмотр результатов Workout-соревнования

7. Workout-челленджи.

Скриншот списка Workout-челленджей представлен на рисунке 4.39

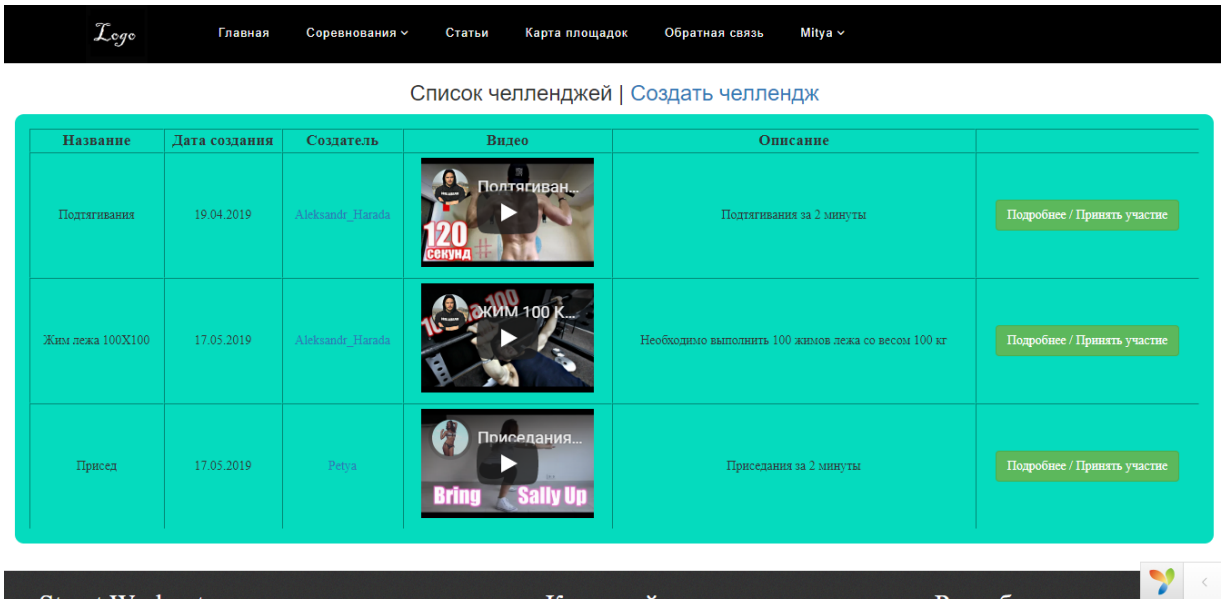

Рисунок 4.39 – Список Workout-челленджей

Скриншот страницы просмотра Workout-челленджа представлен на рисунке 4.40.

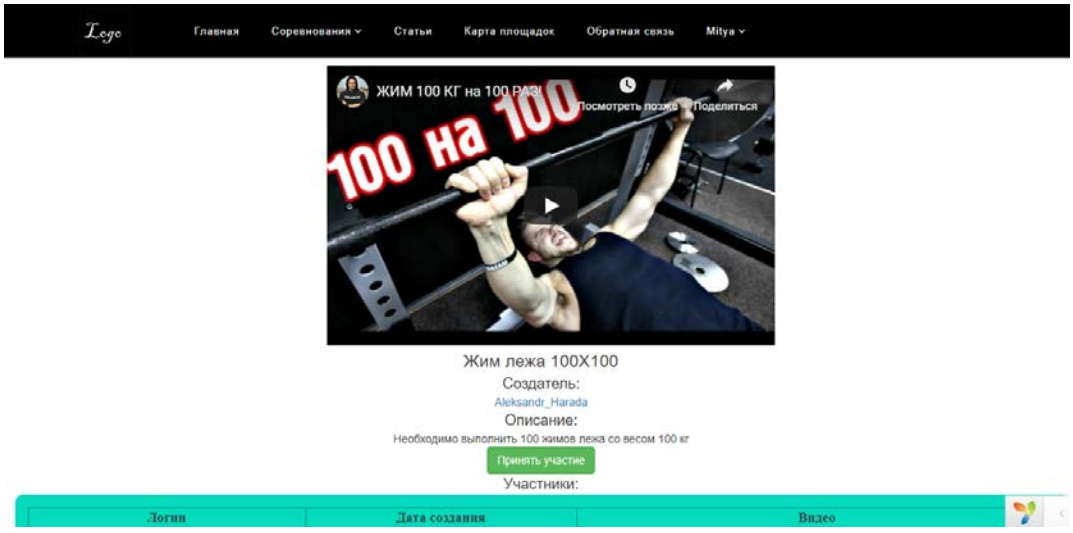

Рисунок 4.40 – Страница просмотра Workout-челленджа

Скриншот списка участников Workout-челленджа представлен на рисунке 4.41.

Рисунок 4.41 – Список участников Workout-челленджа

8. Workout-зарубы.

   Скриншот списка незавершенных Workout-заруб представлен на рисунке 4.42.

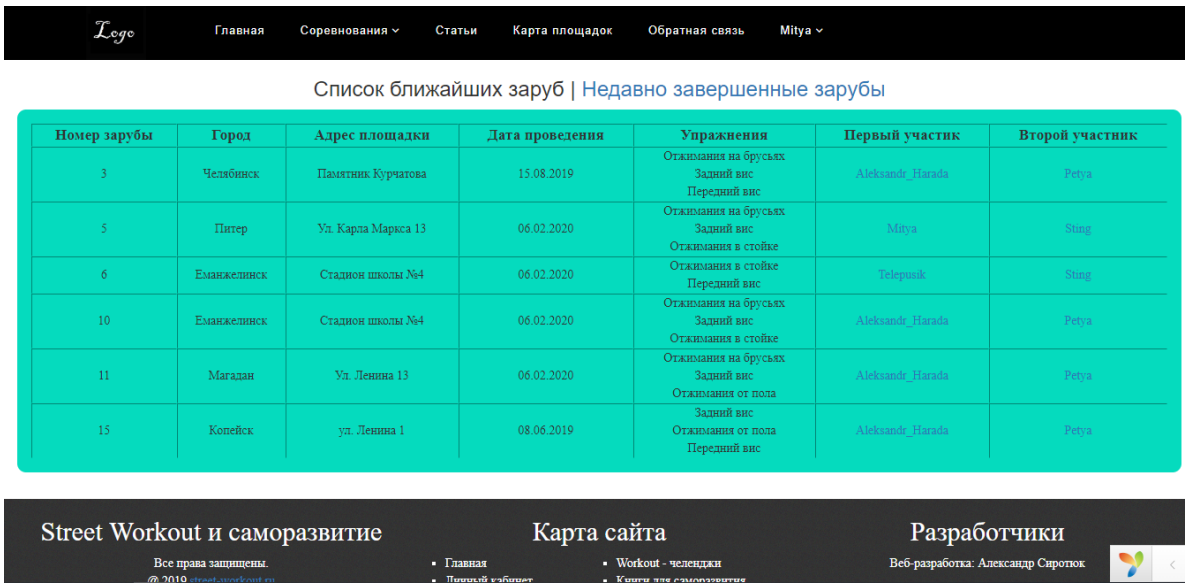

Рисунок 4.42 – Список незавершенных Workout-заруб

Скриншот списка завершенных Workout-заруб представлен на рисунке 4.43.

Рисунок 4.43 – Список завершенных Workout-заруб

9. Профиль пользователя.

Скриншот профиля пользователя представлен на рисунке 4.44.

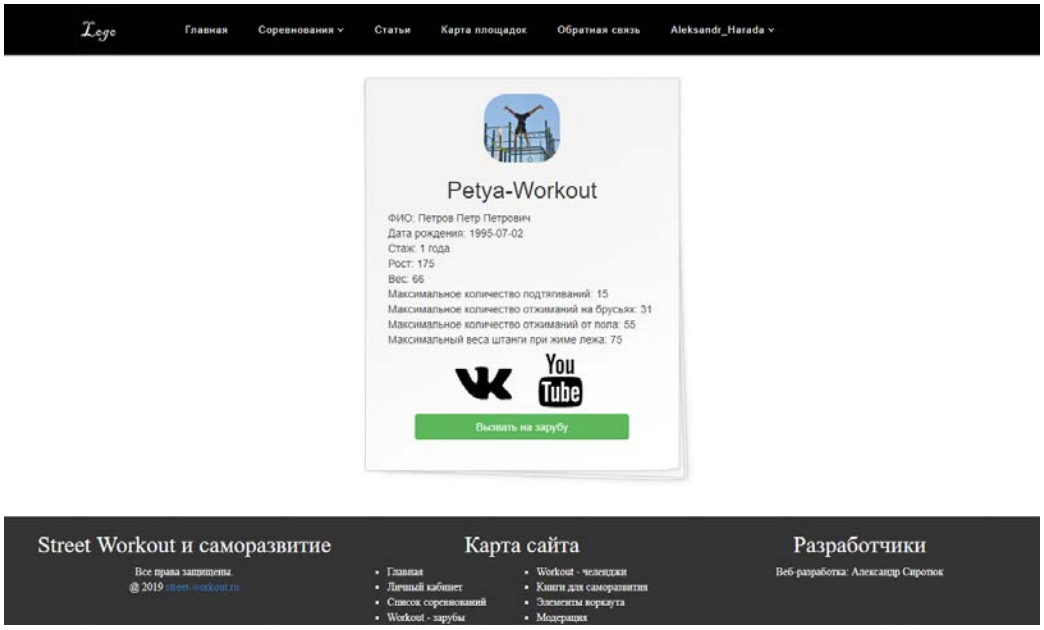

Рисунок 4.44 – Профиль пользователя

10. Личный кабинет пользователя.

Скриншот личного кабинета пользователя представлен на рисунке 4.45.

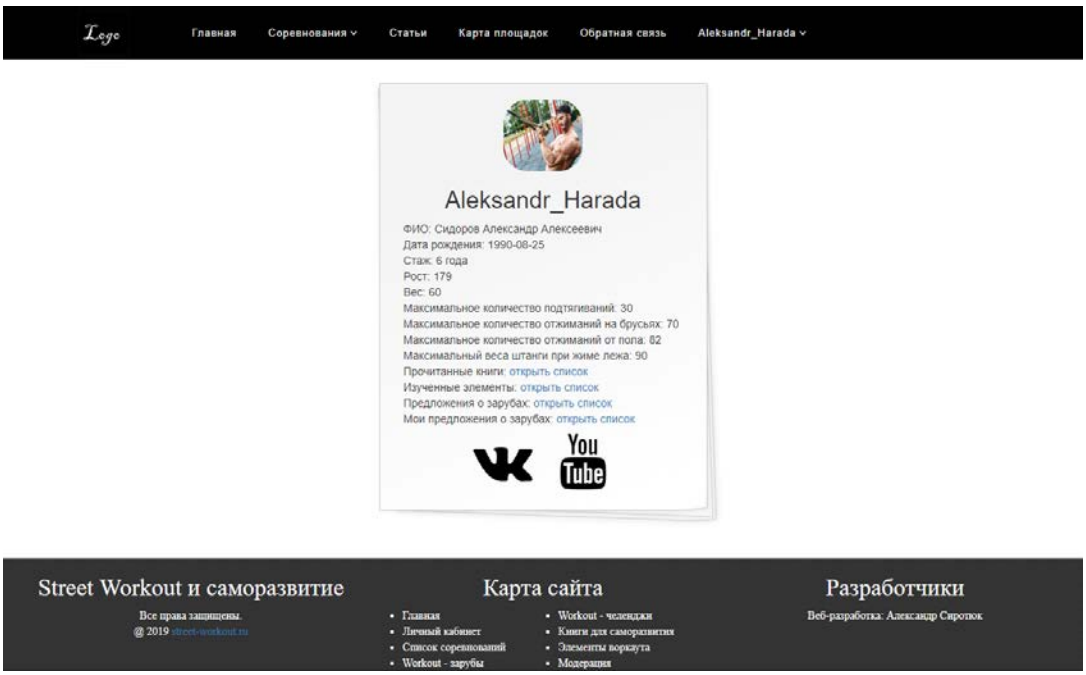

Рисунок 4.45 – Личного кабинет пользователя

Скриншот модального окна со списком прочитанных пользователем книг представлен на рисунке 4.46.

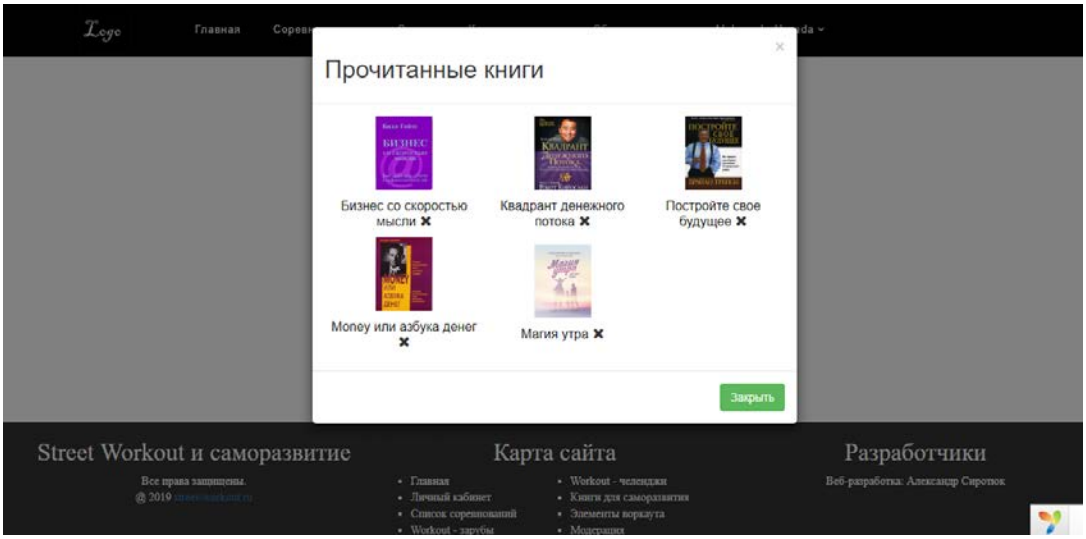

Рисунок 4.46 – Модальное окно со списком прочитанных пользователем книг

Скриншот модального окна со списком изученных пользователем элементов представлен на рисунке 4.47.

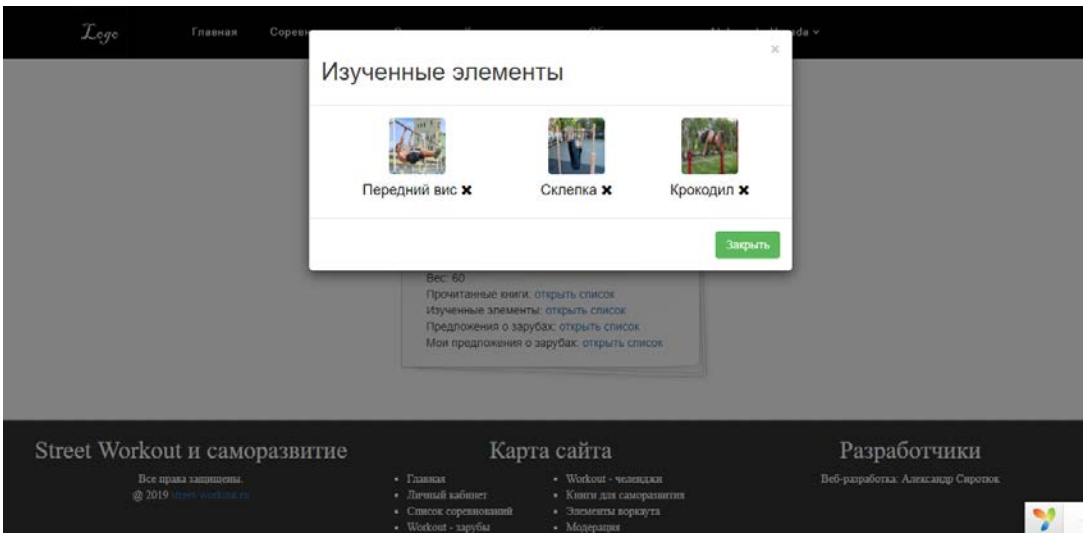

Рисунок 4.47 – Модальное окно со списком изученных пользователем элементов

Скриншот модального окна со списком полученных пользователем предложений о Workout-зарубе представлен на рисунке 4.48.

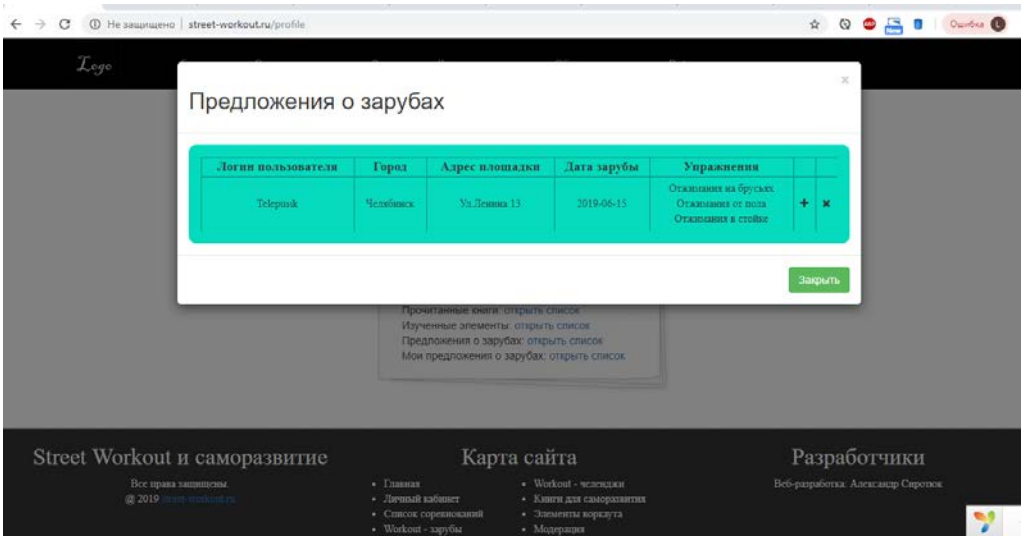

Рисунок 4.48 – Модальное окно со списком полученных пользователем предложений о Workout-зарубе

Скриншот модального окна со списком предложений пользователя другим пользователям о Workout-зарубе представлен на рисунке 4.49.

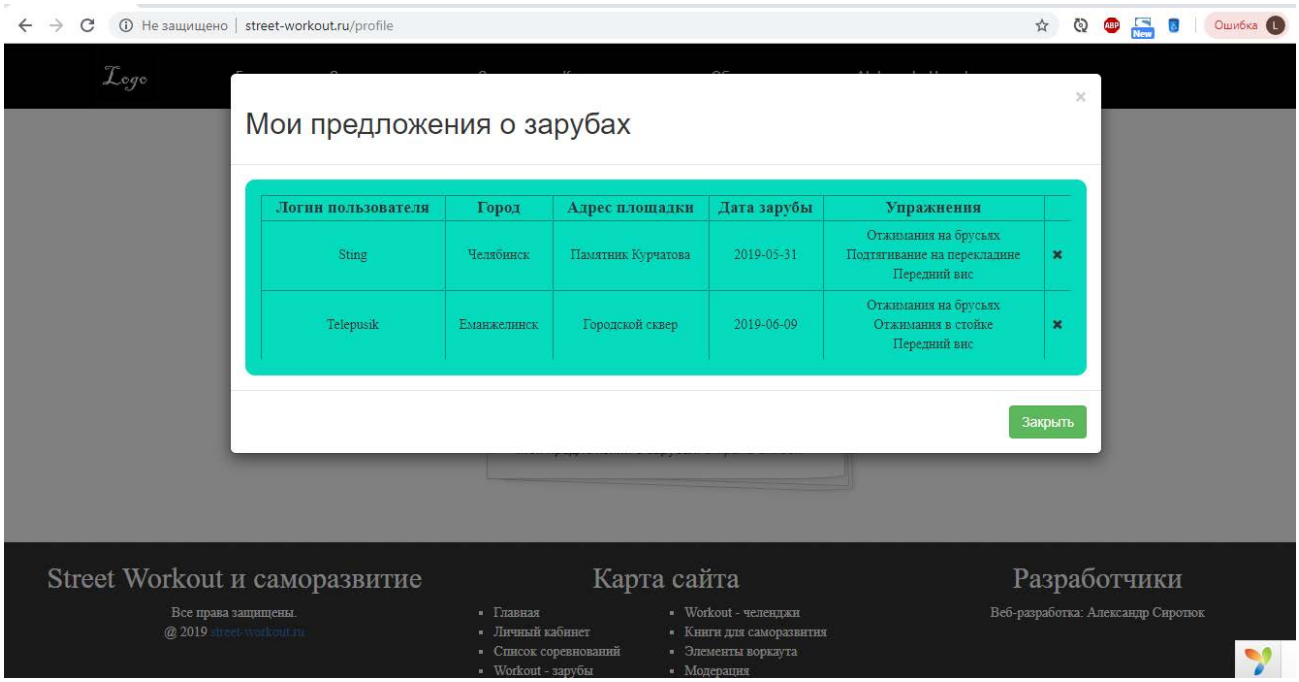

Рисунок 4.49 – Модального окно со списком предложений пользователя другим пользователям о Workout-зарубе

11. Обеспечение межпользовательской коммуникации.

Межпользовательская коммуникация осуществляется за счет возможности предложения Workout-заруб друг другу и опубликование ссылок на свои аккаунты в социальных сетях.

12. Модерирование веб-приложения.

Скриншот раздела модерирования веб-приложения представлен на рисунке 4.50.

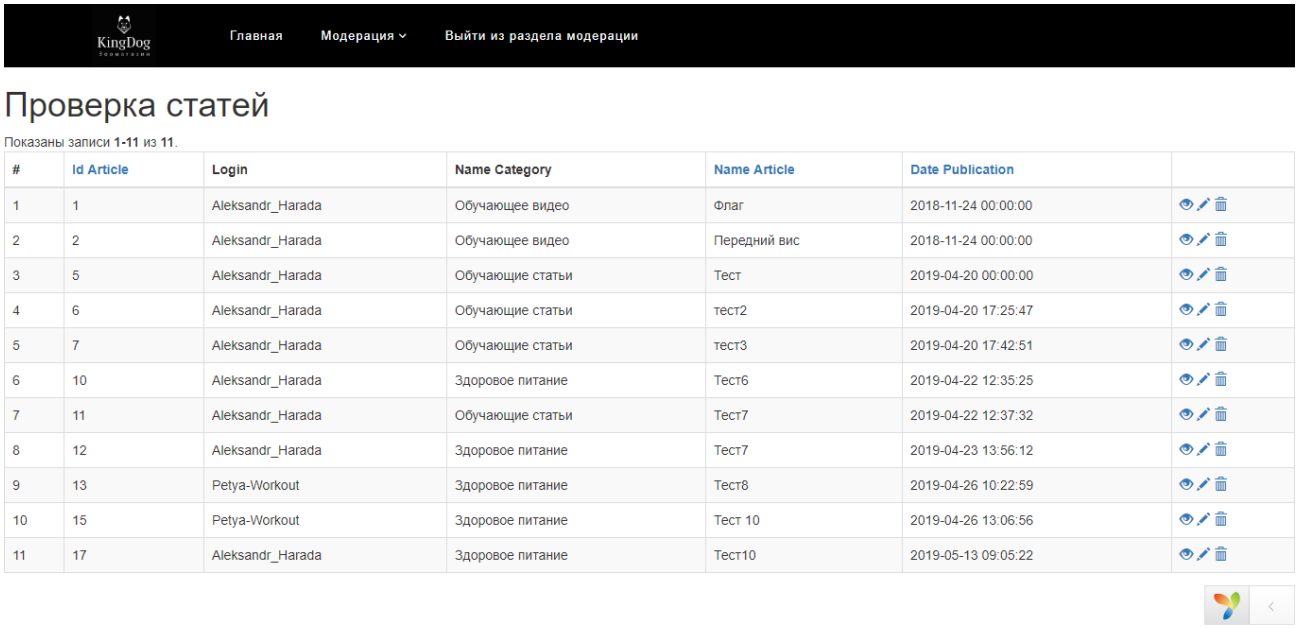

Рисунок 4.50 – Раздел модерирования веб-приложения

5. ТЕСТИРОВАНИЕ

5.1. ПРОВЕДЕНИЕ ПРОЦЕДУРЫ ТЕСТИРОВАНИЯ

Так как разработка проекта велась исключительно в ходе дипломного проектирования, на данный момент было проведено только альфа-тестирование. Ниже приведен перечень модульных тестов и показана работа программного обеспечения при выполнении этих тестов.

1. Тестирование регистрации и авторизации пользователей.

Скриншоты проверки (валидации) данных, введенных пользователем при его регистрации, представлены на рисунках 5.1, 5.2, 5.3.

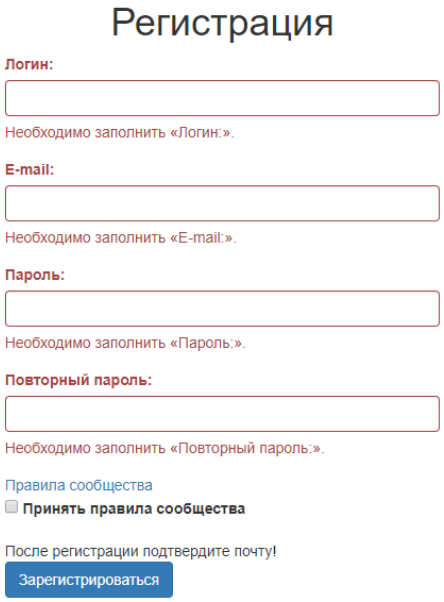

Рисунок 5.1 – Проверка (валидация) данных, введенных пользователем при регистрации, не заполненные обязательные поля

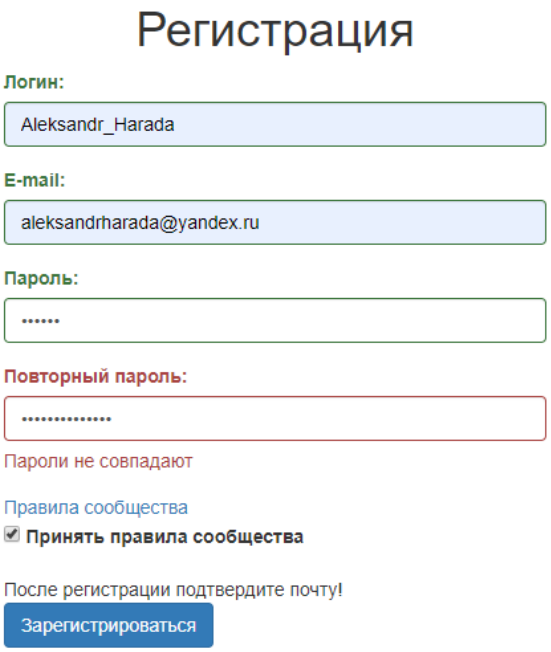

Рисунок 5.2 – Проверка (валидация) данных, введенных пользователем при регистрации, несовпадение пароля

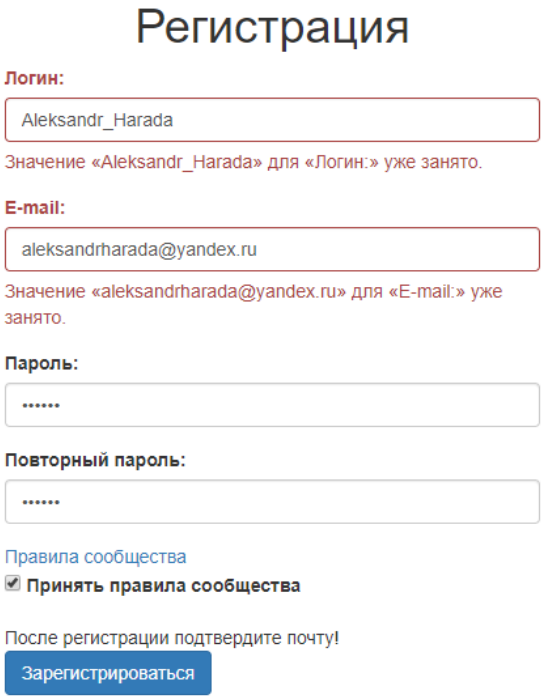

Рисунок 5.3 – Проверка (валидация) данных, введенных пользователем при регистрации, дублирование регистрационных данных

Скриншоты проверки (валидации) данных, введенных пользователем при авторизации, представлены на рисунках 5.4, 5.5.

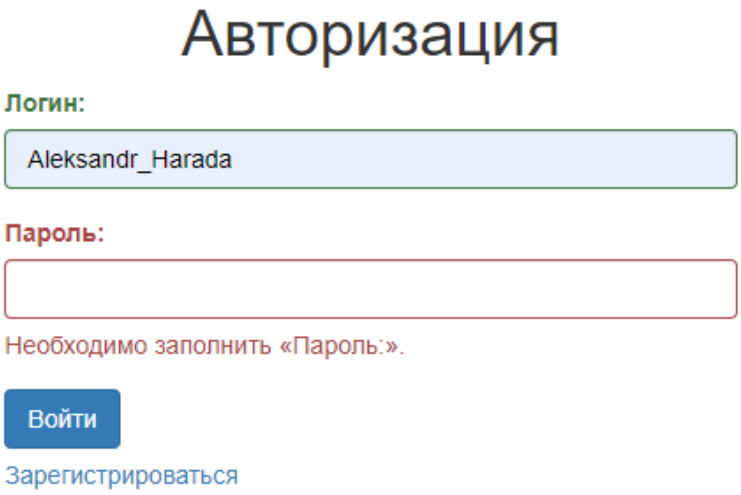

Рисунок 5.4 – Проверка (валидация) данных, введенных пользователем при авторизации, не введен пароль

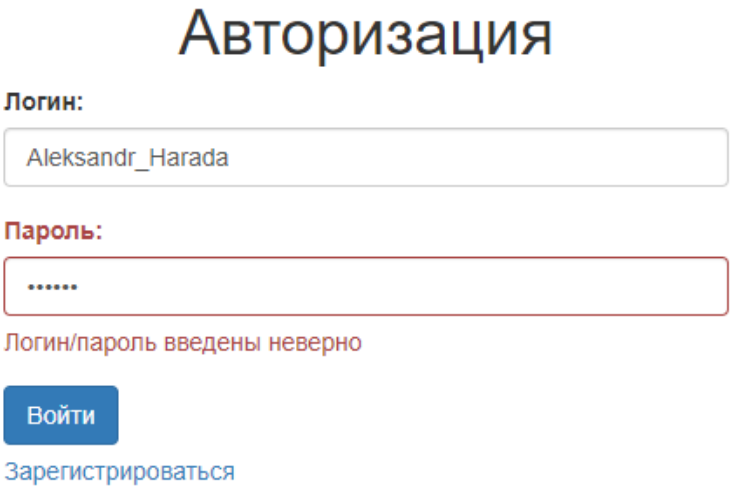

Рисунок 5.5 – Проверка (валидация) данных, введенных пользователем при авторизации, неверные данные

Скриншот сообщения, полученного на указанный пользователем при регистрации адрес электронной почты для его верификации, представлен на рисунке 5.6.

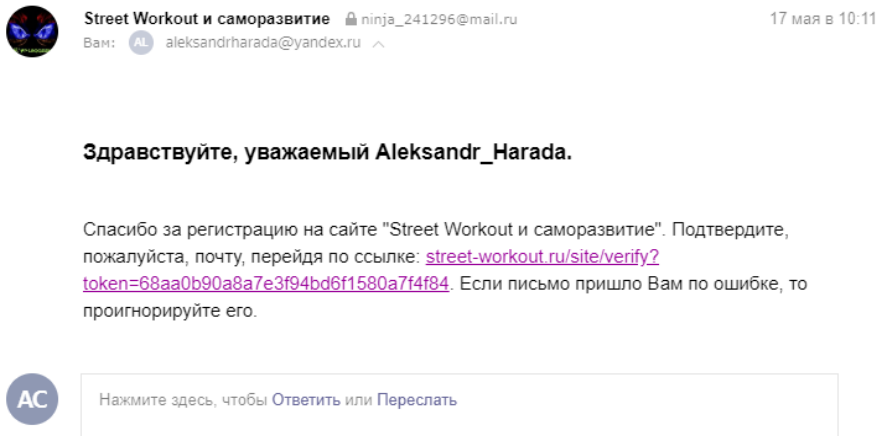

Рисунок 5.6 – Верификация пользователя

2. Тестирование редактирования личного профиля.

Скриншот проверки (валидации) данных, введенных пользователем при редактировании личного профиля, представлен на рисунке 5.6.

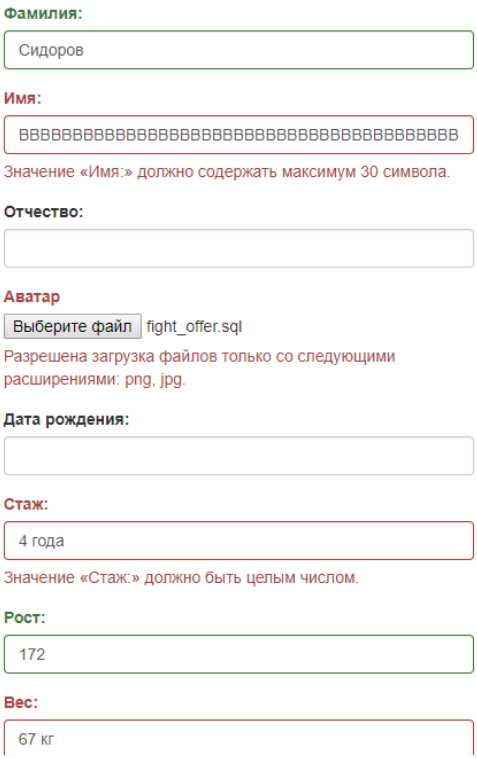

Рисунок 5.6 – Проверка (валидация) данных, введенных пользователем при редактировании личного профиля

Скриншот личного профиля после редактирования пользователем представлен на рисунке 5.7.

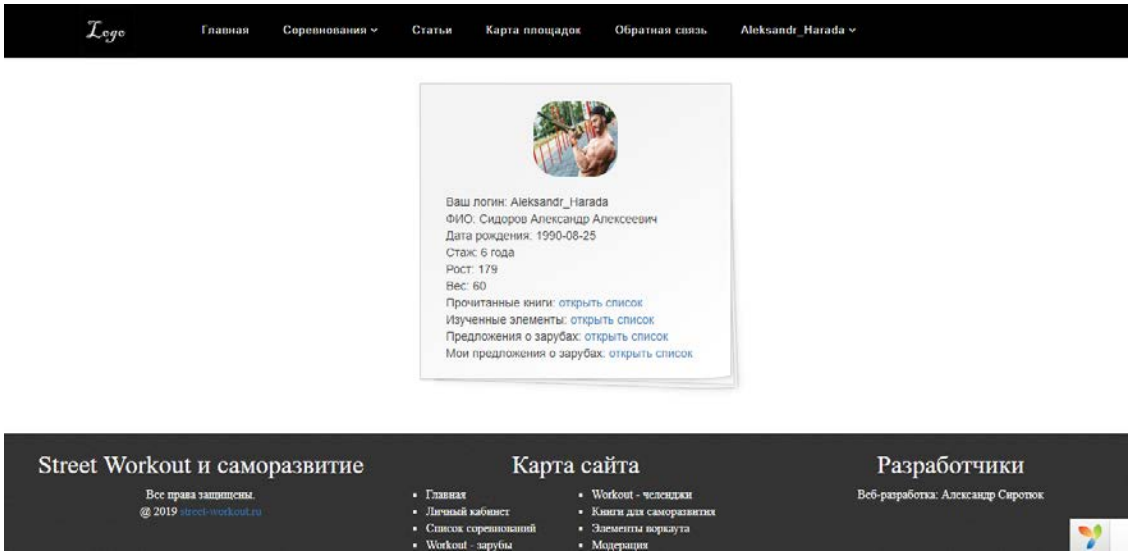

Рисунок 5.7 – Личный профиль после редактирования пользователем

3. Тестирование процедуры смены пароля.

Скриншот письма, полученного пользователем веб-приложения при процедуре смены пароля, представлен на рисунке 5.8.

Рисунок 5.8 – Письмо, полученное пользователем веб-приложения при процедуре смены пароля

При переходе по ссылке у пользователя открывается страница веб-приложения с формой ввода смены пароля (см. рисунок 4.12). Скриншот проверки (валидации) данных, введенных пользователем при смене пароля, представлен на рисунках 5.9, 5.10.

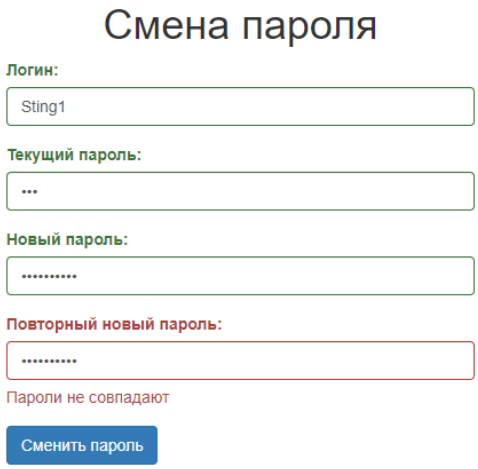

Рисунок 5.9 – Проверка (валидация) данных, введенных пользователем при смене пароля, несовпадение пароля

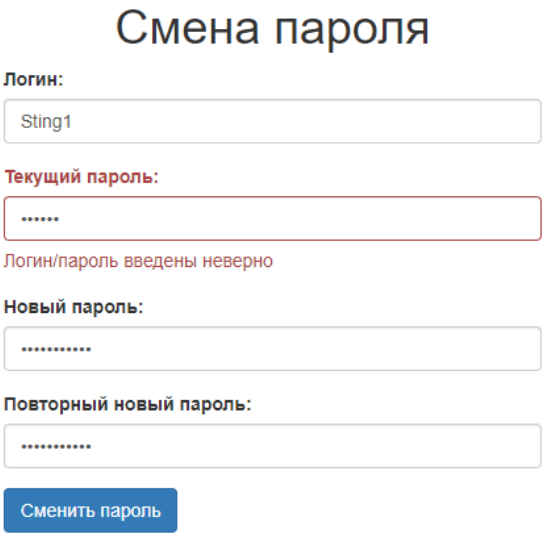

Рисунок 5.9 – Проверка (валидация) данных, введенных пользователем при смене пароля, ошибка ввода пароля

4. Тестирование обратной связи.

Скриншот проверки (валидации) данных, введенных пользователем при заполнении формы обратной связи, представлен на рисунках 5.10, 5.11:

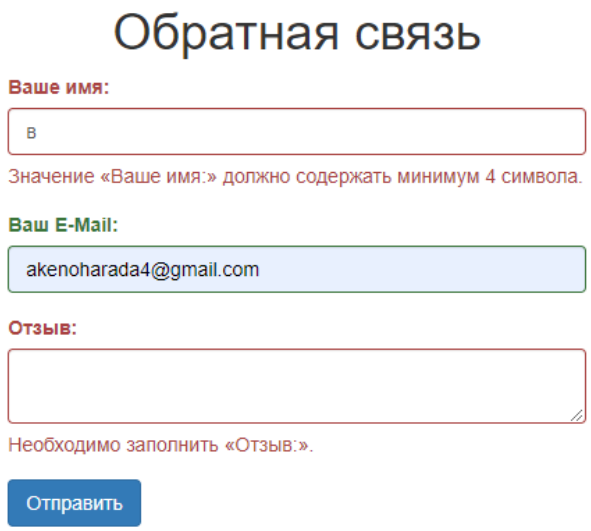

Рисунок 5.10 – Проверка (валидация) данных, введенных пользователем при заполнении формы обратной связи, вариант 1

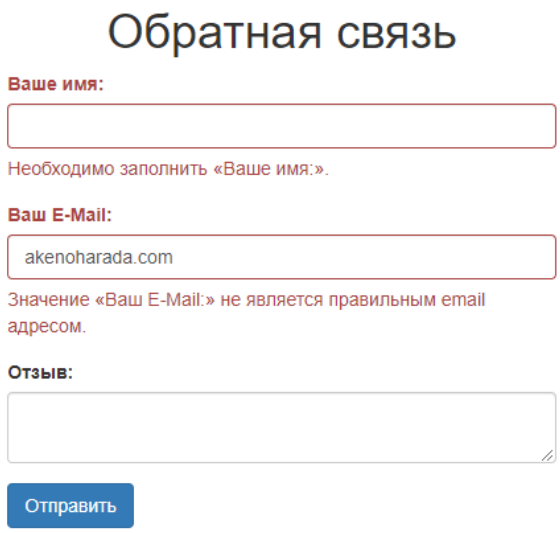

Рисунок 5.11 – Проверка (валидация) данных, введенных пользователем при заполнении формы обратной связи, вариант 2

Скриншот письма, полученного администратором от пользователя веб-приложения, представлен на рисунке 5.12.

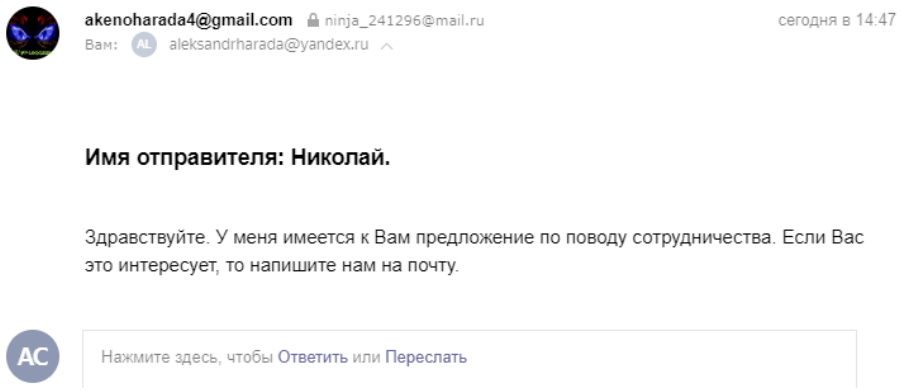

Рисунок 5.12 – Письмо, полученное администратором от пользователя веб-приложения

5. Тестирование создания и редактирования статьи.

Скриншот проверки (валидации) данных, введенных пользователем при создании статьи, представлен на рисунке 5.13.

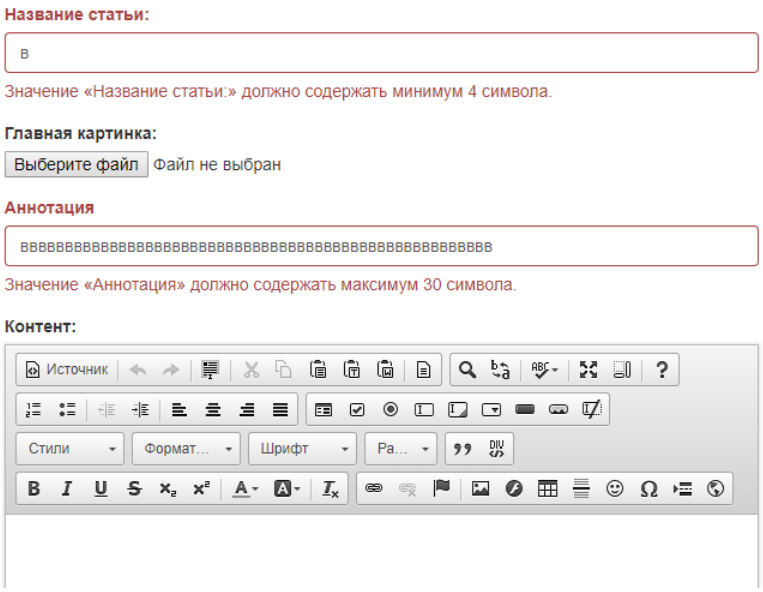

Рисунок 5.13 – Проверка (валидация) данных, введенных пользователем при создании статьи

Проверка данных, введенных пользователем, аналогична и для редактирования статьи (см. рисунок 5.13). Примеры созданных пользователями статей показаны на рисунках 4.24, 4.25, 4.26.

6. Тестирование создания соревнования.

Скриншот ограничения доступа пользователям, не имеющим права на создание соревнования, представлен на рисунке 5.14. Создавать соревнования могут только пользователи, роли которых «создатель соревнований», «модератор» или «администратор».

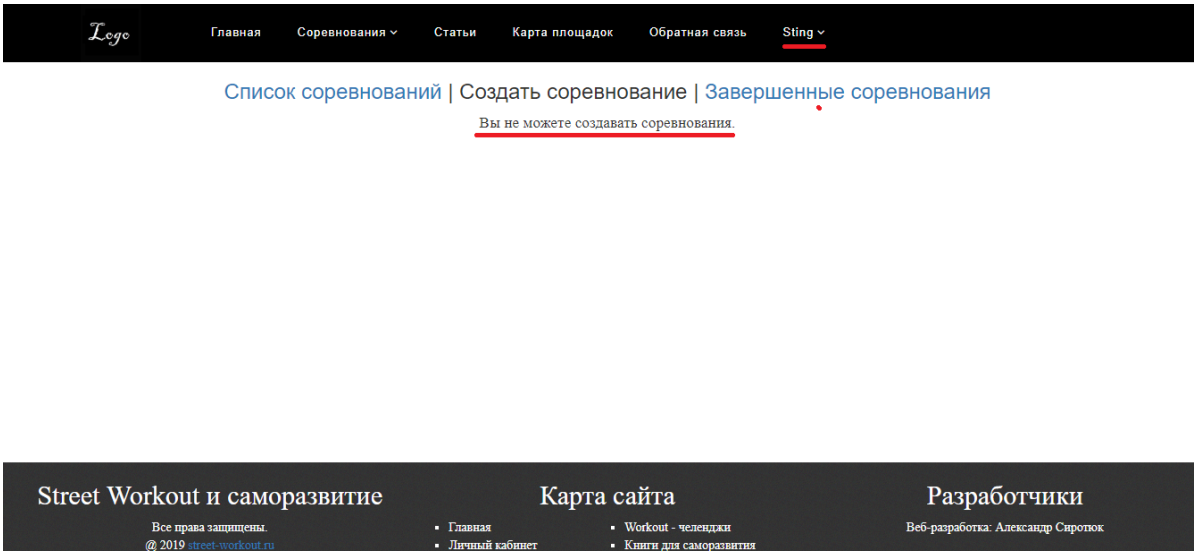

Рисунок 5.14 – Ограничение доступа пользователям, не имеющим права на создание соревнований

Скриншот проверки (валидации) данных, введенных пользователем при создании соревнования, представлен на рисунке 5.15.

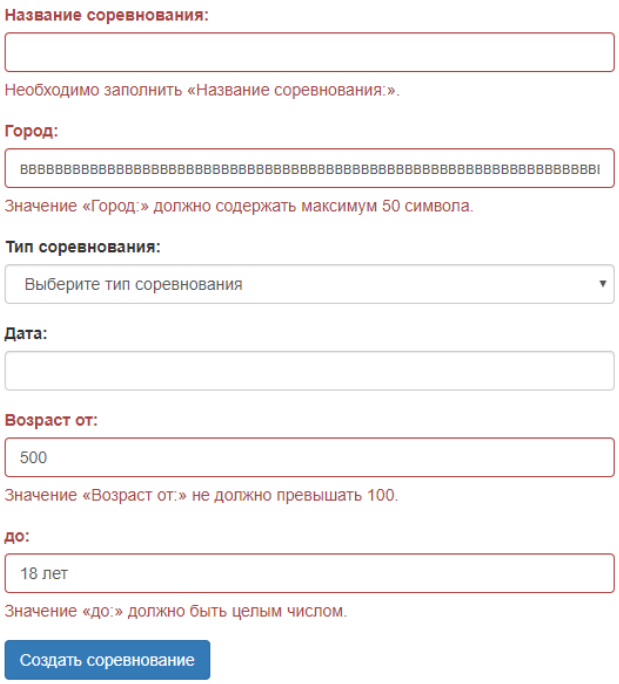

Рисунок 5.15 – Проверка (валидация) данных, введенных пользователем при создании соревнования

Примеры созданных пользователями соревнований можно посмотреть на рисунках 4.34, 4.35. Форму ввода объявления результатов соревнований можно посмотреть на рисунке 4.13. Пример оглашенных результатов соревнований можно посмотреть на рисунке 4.38

7. Тестирование создания Workout-челленджа и принятия участия в нем.

Скриншот проверки (валидации) данных, введенных пользователем при создании Workout-челленджа, представлен на рисунке 5.16.

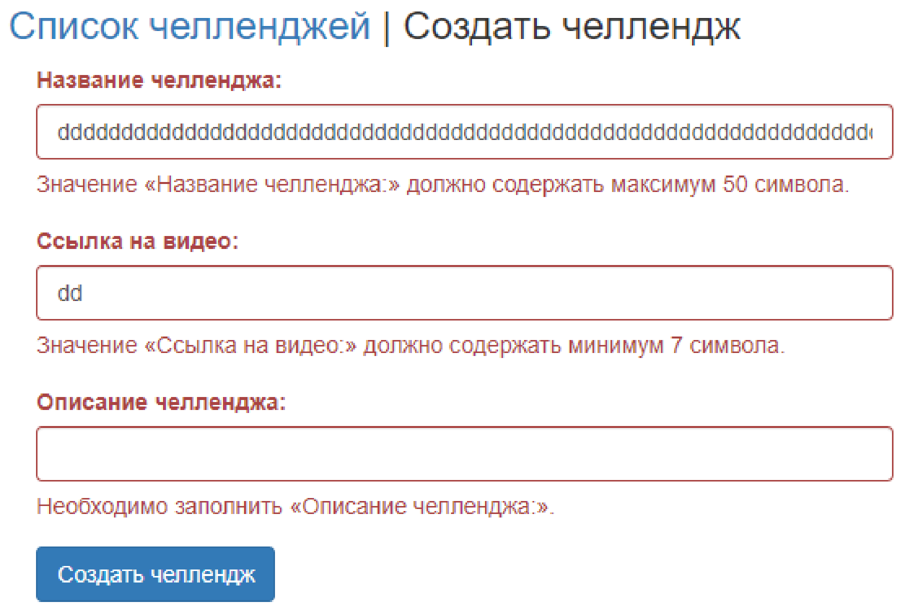

Рисунок 5.16 – Проверка (валидация) данных, введенных пользователем при создании Workout-челленджа

Проверка (валидация) данных, введенных пользователем при принятии участия в Workout-челлендже, является аналогичной (см. рисунок 5.16).

Пример созданных Workout-челленджей показан на рисунке 4.39. Пример списка участников Workout-челленджа показан на рисунке 4.41.

8. Тестирование предложения пользователю о Workout-зарубе и принятия предложения о Workout-зарубе от другого пользователя.

Скриншот проверки (валидации) данных, введенных пользователем при предложении Workout-зарубы другому пользователю, представлен на рисунке 5.17.

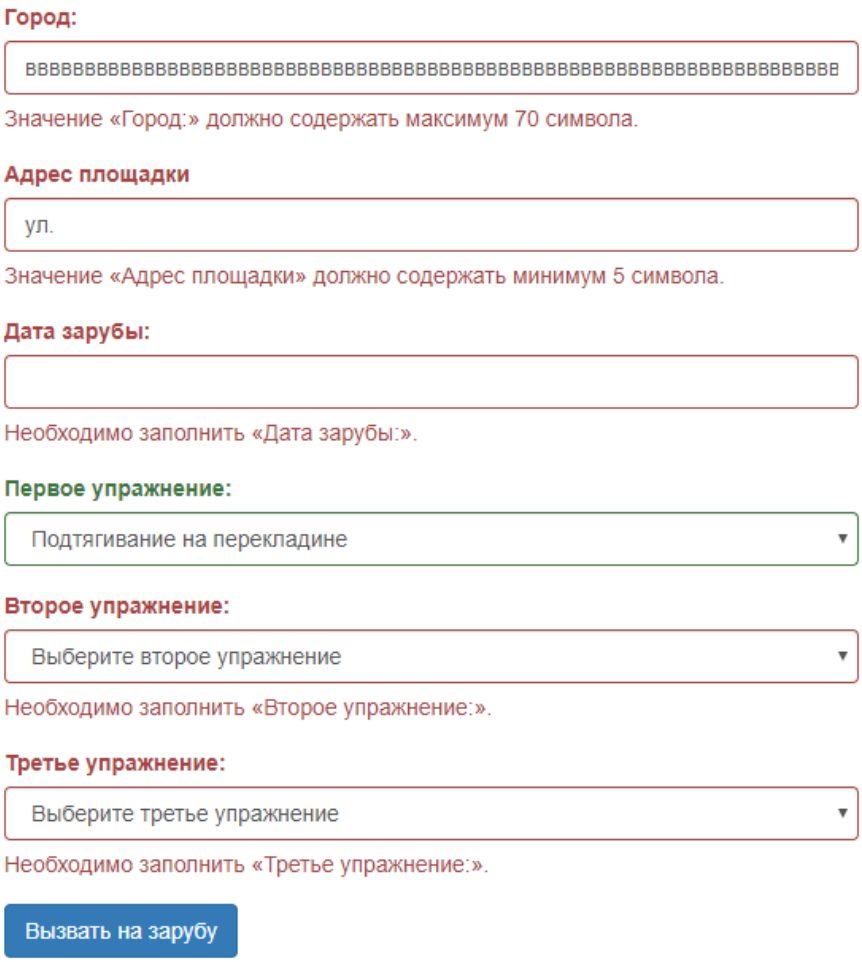

Рисунок 5.17 – Проверка (валидация) данных, введенных пользователем при предложении Workout-зарубы другому пользователю

Пример списка предложений пользователя о Workout-зарубе другим пользователям показан на рисунке 4.49. Пример списка полученных пользователем предложений о Workout-зарубах показан на рисунке 4.48. Примеры списка предстоящих и завершенных Workout-заруб показаны на рисунках 4.42, 4.43.

9. Тестирование создания спортивного объекта.

Скриншот проверки (валидации) данных, введенных пользователем при создании спортивного объекта, представлен на рисунке 5.18.

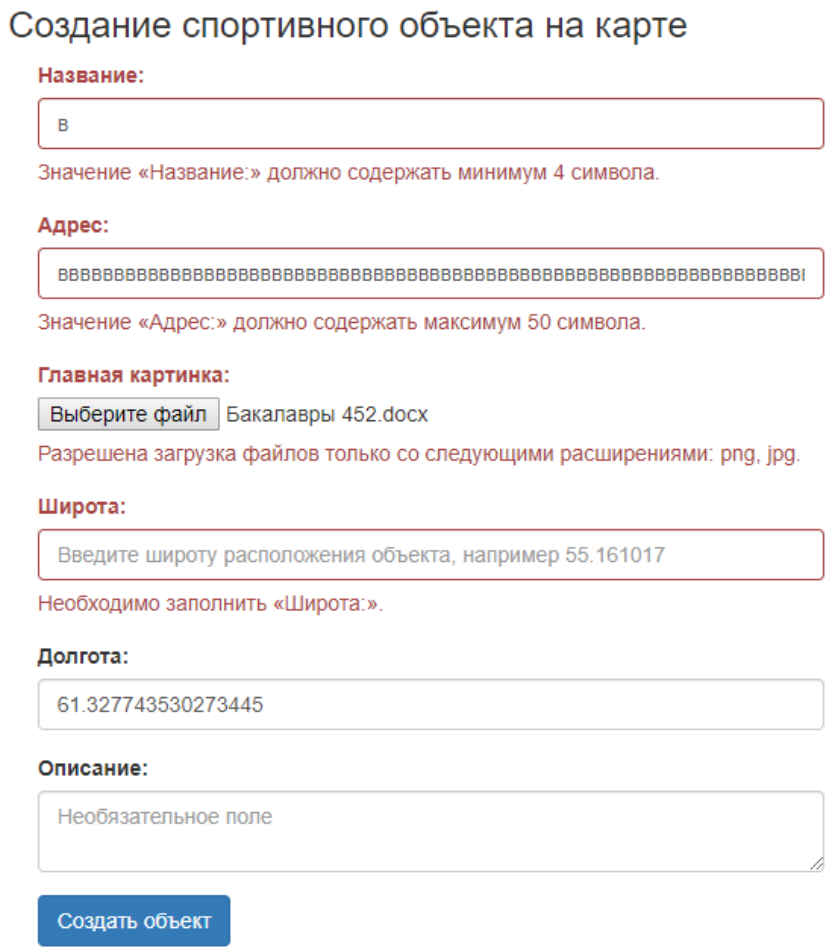

Рисунок 5.18 – Проверка (валидация) данных, введенных пользователем при создании спортивного объекта

Примеры созданных спортивных объектов можно посмотреть на рисунке 4.31.

10. Тестирование написания отзыва к книге.

Примеры написанных отзывов к книге можно посмотреть на рисунке 4.28.

11. Тестирование написания отзыва о спортивной площадке.

Пример написанного отзыва о спортивной площадке можно посмотреть на рисунке 4.33.

12. Тестирование добавления книги в «прочитанное».

Пример модального окна со списком прочитанных книг пользователем веб-приложения можно посмотреть на рисунке 4.46.

13. Тестирование добавления Workout-элемента в «изученное».

Пример модального окна со списком изученных Workout-элементов пользователем веб-приложения можно посмотреть на рисунке 4.47.

ЗАКЛЮЧЕНИЕ

В ходе дипломного проектирования было выполнено следующее:

1. Проведен анализ предметной области и рынка родственных разработок.
2. Проанализированы и выбраны средства для разработки веб-приложения.
3. Определены функциональные и нефункциональные требования к приложению.
4. Спроектирована архитектура веб-приложения.
5. Разработана база данных.
6. Разработано и протестировано программное обеспечение, решающее поставленную задачу.

В веб-приложении были реализованы следующие функции:

1. Наличие обучающих видеоматериалов для спортсменов и мотивационных материалов.
2. Поддержка системы статистики каждого спортсмена.
3. Система челленджей.
4. Отметка на карте местоположения площадок и тренажерных залов с системой оценок и отзывов.
5. Организация соревнований и их статистика.
6. Система «Workout-зарубов 1 на 1».
7. Система коммуникаций между спортсменами различного уровня.
8. Пропаганда книг для самосовершенствования с системой оценок и отзывов.
9. Система модерирование приложения.

В ближайшее время будет разработана система администрирования, после чего система будет готова к опытной эксплуатации.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Веб-сайт «vuzlit.ru». [Электронный ресурс]. URL: https://vuzlit.ru/949050/metody_sredstva_razrabotki. (Дата обращения: январь, февраль 2019 года).
2. Веб-сайт «Simple Solutions». [Электронный ресурс]. URL: http://www.sisols.ru/articles-353/. (Дата обращения: январь, февраль 2019 года).
3. Веб-сайт «Tarlyun Blog». [Электронный ресурс]. URL: http://tarlyun.com/blog/2012/07/19/nedostatki-code-igniter/. (Дата обращения: февраль 2019 года).
4. Веб-сайт «fokit». [Электронный ресурс]. URL: https://fokit.ru/frejmvork-laravel/. (Дата обращения: февраль 2019 года).
5. Веб-сайт «mkdev». [Электронный ресурс]. URL: https://mkdev.me/posts/top-5-php-freymvorkov-laravel-vs-yii-vs-zend-vs-phalcon-vs-symfony-plyusy-i-minusy. (Дата обращения: февраль 2019 года).
6. Веб-сайт «Yii framework». [Электронный ресурс]. URL: https://www.yiiframework.com/. (Дата обращения: март 2019 года).
7. Веб-сайт «studbooks.net». [Электронный ресурс]. URL: https://studbooks.net/2215550/informatika/bazy_dannyh_subd_prilozheniy. (Дата обращения: апрель 2019 года).
8. Веб-сайт «drach.pro». [Электронный ресурс]. URL: http://drach.pro/blog/hi-tech/item/145-. (Дата обращения: апрель 2019 года).
9. Веб-сайт «habr». [Электронный ресурс]. URL: https://habr.com/ru/company/piter/blog/308134/. (Дата обращения: апрель 2019 года).

10. Веб-сайт «erfa.ru». [Электронный ресурс]. URL: https://erfa.ru/bootstrap-osnovy-bootstrap-prostaya-adaptivnaya-verstka-dlya-novichkov-bystrye-floaty-i.html. Дата обращения: апрель 2019 года).
11. Веб-сайт «habr». [Электронный ресурс]. URL: https://habr.com/ru/post/242015/. (Дата обращения: апрель 2019 года).
12. Веб-сайт «ruseller.com». [Электронный ресурс]. URL: https://ruseller.com/lessons.php?id=666. (Дата обращения: апрель 2019 года).
13. Котеров, Д.В., Симдянов, И.В. PHP 7. – Санкт-Петербург: БХВ-Петербург, 2017. – 1088 с.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД КОНТРОЛЛЕРА ARTICLE

```
<?php
/**
 * Created by PhpStorm.
 * User: Aleks
 * Date: 21.01.2019
 * Time: 19:00
 */

namespace app\controllers;

use app\models\ArticleEditingForm;
use app\models\ImageUploading;
use Yii;
use app\models\ArticleCreatingForm;
use yii\web\Controller;
use app\models\Article;
use yii\web\UploadedFile;

class ArticleController extends Controller
{
    public function actionIndex()
    {
        $articleModel = new Article();
        $articles = $articleModel->getArticles();
        return $this->render('index', compact('articles'));
    }

    public function actionView($id)
    {
        $articleModel = new Article();
        $article = $articleModel->getArticleById($id);
        return $this->render('view', compact('article'));
    }

    public function actionCreate($categoryId)
    {
        $creatingForm = new ArticleCreatingForm();
        $imageUploadingModel = new ImageUploading();

        if (isset($_POST['ArticleCreatingForm'])) {
            $creatingForm->attributes = Yii::$app->request-
>post('ArticleCreatingForm');

            $typeImage = 'article';
            $file = UploadedFile::getInstance($creatingForm, 'main_image');
            $nameImage = $imageUploadingModel->uploadImage($file, $typeImage);
            $creatingForm->main_image = $nameImage;

            if ($creatingForm->validate() && $creatingForm-
>createArticle($categoryId)) {
                if ($categoryId == 2) {
                    return $this->redirect('/articles');
```

```
    }
                if ($categoryId == 4) {
                    return $this->redirect('/articles/healthy-food');
                }
            }
        }
        return $this->render('creating', compact('creatingForm'));
    }

    public function actionIndexFood()
    {
        $articleModel = new Article();
        $articles = $articleModel->getFoodArticles();
        return $this->render('index-food', compact('articles'));
    }

    public function actionDelete($id, $typeArticle)
    {
        $article = Article::findOne($id);
        if ($article['id_author'] == Yii::$app->user->identity['id']) {
            $article->delete();
        }
        if ($typeArticle == 'tutorials') {
            $articleModel = new Article();
            $articles = $articleModel->getArticles();
            return $this->render('index', compact('articles'));
        }
        if ($typeArticle == 'healthyFood') {
            $articleModel = new Article();
            $articles = $articleModel->getFoodArticles();
            return $this->render('index-food', compact('articles'));
        }
        return true;
    }

    public function actionEdit($id)
    {
        $article = Article::findOne($id);
        $articleEditingForm = new ArticleEditingForm();

        if ($article['id_author'] == Yii::$app->user->identity['id']) {
            if (isset($_POST['ArticleEditingForm'])) {
                $articleEditingForm->attributes = Yii::$app->request-
>post('ArticleEditingForm');
                if ($articleEditingForm->validate() && $articleEditingForm-
>editArticle($article)) {
                    $url = '/article/view?id=' . $id;
                    $this->redirect($url);

    }
            }
            return $this->render('editing', compact('articleEditingForm'));
        } else {
            return $this->goHome();
        }
    }
```

ПРИЛОЖЕНИЕ Б

ИСХОДНЫЙ КОД КОНТРОЛЛЕРА BOOK

```
<?php
/**
 * Created by PhpStorm.
 * User: Aleks
 * Date: 08.11.2018
 * Time: 13:22
 */

namespace app\controllers;

use app\models\Book;
use app\models\CommentForm;
use HttpException;
use yii\base\BaseObject;
use yii\web\Controller;
use Yii;

class BookController extends Controller
{
    public function actionIndex()
    {
        $book = Book::find()->asArray()->all();
        return $this->render('index', compact('book'));
    }

    public function actionView($id)
    {
        $book = Book::findOne($id);
        if (empty($book))
            throw new HttpException('404', 'Данной книги не существует');
        $averageMark = $book->getMarkBook($id);

        $commentForm = new CommentForm();
        if (isset($_POST['CommentForm'])) {


$commentForm->attributes = Yii::$app->request->post('CommentForm');
            if ($commentForm->validate()) {
                $commentForm->writeBookComment($id);
            }
        }
        $commentBook = $book->getBookComment($id);
        return  $this->render('view',  compact('book',  'averageMark',  'commentForm',
'commentBook'));
    }

    public function actionAdd($id)
    {
        $book = new Book();
        $book->addBook($id);
        return true;
    }
```

```
public function actionRate($id, $mark)
    {
        $book = new Book();
        $book->rateBook($id, $mark);
        return true;
    }
}
```

ПРИЛОЖЕНИЕ В

ИСХОДНЫЙ КОД КОНТРОЛЛЕРА CHALLENGE

```
<?php

namespace app\controllers;

use app\models\Challenge;
use app\models\ChallengeCreatingForm;
use app\models\ChallengeParticipantForm;
use app\models\ChallengeParticipation;
use Yii;
use yii\web\Controller;

class ChallengeController extends Controller
{
    public function actionIndex()
    {
        $challengeModel = new Challenge();
        $challenges = $challengeModel->getChallenges();
        return $this->render('index', compact('challenges'));
    }

    public function actionCreating()
    Продолжение приложения В
{
        $creatingForm = new ChallengeCreatingForm();
        if (isset($_POST['ChallengeCreatingForm'])) {
            $creatingForm->attributes              =              Yii::$app->request-
>post('ChallengeCreatingForm');
            if ($creatingForm->validate() && $creatingForm->signUp()) {
                return $this->redirect('/challenges');
            }
        }
        return $this->render('creating', compact('creatingForm'));
}

    public function actionView($id)
    {
        $challengeModel = new Challenge();
        $challengeParticipationModel = new ChallengeParticipation();

        $challenge = $challengeModel->getChallengeById($id);
        $challengeParticipations            =            $challengeParticipationModel-
>getChallengeById($id);

    Окончание приложения В
return $this->render('view', compact('challenge', 'challengeParticipations'));
    }

    public function actionTakingPart($id)
    {
        $this->layout = false;
        $participantForm = new ChallengeParticipantForm();
        if (isset($_POST['ChallengeParticipantForm'])) {
```

```
                $participantForm->attributes            =            Yii::$app->request-
>post('ChallengeParticipantForm');
                if ($participantForm->validate() && $participantForm->takePart($id)) {
                    $url = '/challenges/view?id=' . $id;
                    return $this->redirect($url);
                }
            }
            return $this->render('taking-part', compact('participantForm'));
        }
    }
```

ПРИЛОЖЕНИЕ Г

ИСХОДНЫЙ КОД КОНТРОЛЛЕРА COMPETITION

```
<?php

namespace app\controllers;

use app\models\TypeCompetition;
use yii\web\Controller;
use app\models\Competition;
use app\models\CompetitionCreatingForm;
use Yii;
use yii\web\ForbiddenHttpException;

class CompetitionController extends Controller
{
    public function actionIndex()
    {
        $competition_model = new Competition();
        $competitions = $competition_model->getBrieflyCompetitions();
        return $this->render('index', compact('competitions'));
    }

    public function actionView($id)
    {
        $competition_model = new Competition();
        $competition = $competition_model->getCompetitionById($id);
        $awards = $competition_model->getAwardsById($id);
        $sponsors = $competition_model->getSponsorsById($id);
        return  $this->render('view',  compact('competition',  'awards',  'sponsors',
'participants'));
    }

    public function actionViewParticipants($id)
   {
        $competition_model = new Competition();
        $competition = $competition_model->getCompetitionById($id);
        $participants = $competition_model->getParticipantsById($id);
        return       $this->render('view-participants',       compact('participants',
'competition'));
    }

    public function actionCreating()
    {
        $flagProhibition = false;
        if (Yii::$app->user->can('createCompetition')) {
            $typesCompetitionsQuery  =  TypeCompetition::find()->select(['type_name'])-
>asArray()->all();
            for ($i = 0; $i < count($typesCompetitionsQuery); $i++) {
                $types_competitions[$i              +              1]              =
$typesCompetitionsQuery[$i]['type_name'];
            }
            $creatingForm = new CompetitionCreatingForm();
            if (isset($_POST['CompetitionCreatingForm'])) {
                $creatingForm->attributes            =            Yii::$app->request-
>post('CompetitionCreatingForm');
```

```
        if ($creatingForm->validate() && $creatingForm->signUp()) {
                    return $this->redirect('/competitions');
                }
            }
            return $this->render('creating', compact('creatingForm',
                'flagProhibition', 'types_competitions'));
        } else {

$flagProhibition = true;
            return $this->render('creating', compact('flagProhibition'));
        }
    }

    public function actionCompleted()
    {
        $competition_model = new Competition();
        $competitions = $competition_model->getBrieflyCompetitions();
        return $this->render('completed', compact('competitions'));
    }

    public function actionAddParticipant($id_competition)
    {
        $competition_model = new Competition();
        $competition_model->addParticipant($id_competition);
        return true;
    }
}
```

ПРИЛОЖЕНИЕ Д

ИСХОДНЫЙ КОД КОНТРОЛЛЕРА ELEMENT

```
<?php
/**
 * Created by PhpStorm.
 * User: Aleks
 * Date: 04.12.2018
 * Time: 9:44
 */

namespace app\controllers;

use app\models\Element;
use yii\web\Controller;

class ElementController extends Controller
{
    public function actionIndex()
    {
        return $this->render('index');
    }

    public function action1lvl()
    {
        $element = Element::find()->asArray()->where(['level' => 1])->all();
        return $this->render('1lvl', compact('element'));
    }

    public function action2lvl()
    {
        $element = Element::find()->asArray()->where(['level' => 2])->all();
        return $this->render('2lvl', compact('element'));
    }

    public function action3lvl()

 {
        $element = Element::find()->asArray()->where(['level' => 3])->all();
        return $this->render('3lvl', compact('element'));
    }

    public function actionAdd($id)
    {
        $element = new Element();
        $element->addElement($id);
        return true;
    }
}
```

ПРИЛОЖЕНИЕ Е

ИСХОДНЫЙ КОД КОНТРОЛЛЕРА FEEDBACK

```
<?php

namespace app\controllers;

use Yii;
use app\models\FeedbackForm;
use yii\web\Controller;

class FeedbackController extends Controller
{
    public function actionIndex()
    {
        $feedbackForm = new FeedbackForm();
        if (isset($_POST['FeedbackForm'])) {
            $feedbackForm->attributes = Yii::$app->request->post('FeedbackForm');
            if ($feedbackForm->validate()) {
                $feedbackForm->sendComment();
            }
            $this->redirect('/feedback/index');
        }
        return $this->render('index', compact('feedbackForm'));
    }

}
```

ПРИЛОЖЕНИЕ Ж

ИСХОДНЫЙ КОД КОНТРОЛЛЕРА FIGHT

```
<?php

namespace app\controllers;

use app\models\Exercise;
use app\models\FightCreatingForm;
use Yii;
use yii\web\Controller;
use app\models\Fight;


class FightController extends Controller
{
    public function actionUpcomingFights()
    {
        $fightModel = new Fight();
        $fights = Fight::find()->asArray()->all();
        $participants = $fightModel->getParticipants();
        $exercises = $fightModel->getExercises();
        return  $this->render('upcoming-fights',  compact('fights',  'participants',
'exercises'));
    }

    public function actionCompletedFights()
    {
        $fightModel = new Fight();
        $fights = Fight::find()->asArray()->all();
        $participants = $fightModel->getParticipants();
        $exercises = $fightModel->getExercises();
        return  $this->render('completed-fights',  compact('fights',  'participants',
'exercises'));
    }

    public function actionCreating($id)
    {

         $this->layout = false;
        $creatingForm = new FightCreatingForm();

        $exercises = array();
        $exercisesQuery = Exercise::find()->asArray()->all();
        for ($i = 0; $i < count($exercisesQuery); $i++) {
            $exercises[$i + 1] = $exercisesQuery[$i]['name_exercise'];
        }

        if (isset($_POST['FightCreatingForm'])) {
            $creatingForm->attributes              =              Yii::$app->request-
>post('FightCreatingForm');
            if ($creatingForm->validate() && $creatingForm->signUp($id)) {
                $url = '/user/' . $id;
                return $this->redirect($url);
            }
        }
```

```
  return $this->renderAjax('creating', compact('creatingForm', 'id', 'exercises'));
    }
}
```

ПРИЛОЖЕНИЕ И
ИСХОДНЫЙ КОД КОНТРОЛЛЕРА MAP

```
<?php

namespace app\controllers;

use Yii;
use app\models\SportsFacility;
use app\models\SportsFacilityCreatingForm;
use app\models\ImageUploading;
use app\models\SportsFacilityMark;
use app\models\CommentForm;
use yii\web\Controller;
use yii\web\UploadedFile;

class MapController extends Controller
{
    public function actionIndex()
    {
        $sportsFacilityModel = new SportsFacility();
        $sportsObjects = $sportsFacilityModel->getSportsObjects();

        return $this->render('index', compact('sportsObjects'));
    }

    public function actionView($id)
    {
        $sportsFacilityModel = new SportsFacility();
        $sportsObject = $sportsFacilityModel->getSportsObjectById($id);

        $averageMarkFacility  =  SportsFacilityMark::find()->where(['id_facility'  =>
$id])->average('value');
        $averageMarkFacility = round($averageMarkFacility, 1);

        $commentForm = new CommentForm();
        if (isset($_POST['CommentForm'])) {
            $commentForm->attributes = Yii::$app->request->post('CommentForm');
            if ($commentForm->validate()) {
                $commentForm->writeFacilityComment($id);
            }
        }
        $commentFacility = $sportsFacilityModel->getFacilityComment($id);
        return $this->render('view', compact('sportsObject',
            'commentForm', 'commentFacility', 'averageMarkFacility'));
    }

    public function actionCreating()
    {
        $creatingForm = new SportsFacilityCreatingForm();
        $imageUploadingModel = new ImageUploading();

        if (isset($_POST['SportsFacilityCreatingForm'])) {
            $creatingForm->attributes              =              Yii::$app->request-
>post('SportsFacilityCreatingForm');
```

```
            $typeImage = 'sportsFacility';
            $file = UploadedFile::getInstance($creatingForm, 'main_image');
            $nameImage = $imageUploadingModel->uploadImage($file, $typeImage);
            $creatingForm->main_image = $nameImage;

            if ($creatingForm->validate() && $creatingForm->createFacility()) {
                return $this->redirect('/map/index');
            }

 }
        return $this->render('creating', compact('creatingForm'));
    }

    public function actionRateFacility($id, $mark)
    {
        $markFacility = SportsFacilityMark::findOne([
            'id_user' => Yii::$app->user->identity['id'],
            'id_facility' => $id
        ]);

        if (!isset($markFacility)) {
            $facilityMarkModel = new SportsFacilityMark();
            $facilityMarkModel->id_facility = $id;
            $facilityMarkModel->id_user = Yii::$app->user->identity['id'];
            $facilityMarkModel->value = $mark;
            $facilityMarkModel->save();
        } else {
            $markFacility->value = $mark;
            $markFacility->save();
        }

        return true;
    }
}
```

ПРИЛОЖЕНИЕ К

ИСХОДНЫЙ КОД КОНТРОЛЛЕРА SITE

```
<?php

namespace app\controllers;

use app\models\ChangingPasswordForm;
use app\models\FightOfferExercise;
use app\models\ImageUploading;
use Yii;
use app\models\EditProfileForm;
use app\models\FightOffer;
use app\models\RegistrationForm;
use app\models\User;
use app\models\Article;
use yii\web\Controller;
use app\models\AuthorizationForm;
use yii\web\UploadedFile;


class SiteController extends Controller
{
    public function actionIndex()
    {
        $articles = Article::find()->where(['id_category' => 1])->asArray()->all();
        return $this->render('index', compact('articles'));
    }

    public function actionRegistration()
    {
        $registrationForm = new RegistrationForm();
        if                    (isset($_POST['RegistrationForm'])                    &&
$_POST['RegistrationForm']['acceptingRules'] == 1) {
                $registrationForm->attributes            =            Yii::$app->request-
>post('RegistrationForm');
                $time = (string)time();
                $token = md5($time);

     if ($registrationForm->validate() && $registrationForm->signUp($token) &&
                    $registrationForm->verifyUser($token)) {
                    return $this->redirect('authorization');
                }
            }
            return $this->render('registration', compact('registrationForm'));
        }

        public function actionRules()
        {
            return $this->render('rules');
        }

        public function actionLogin()
        {
            if (!Yii::$app->user->isGuest)
                return $this->goHome();
```

```
            $authorizationForm = new AuthorizationForm();
                if (isset($_POST['AuthorizationForm'])) {
                    $authorizationForm->attributes            =            Yii::$app->request-
>post('AuthorizationForm');
                    if ($authorizationForm->validate()) {
                        Yii::$app->user->login($authorizationForm->getUser());
                        return $this->goHome();
                    }
                }
                return $this->render('authorization', compact('authorizationForm'));
            }

            public function actionLogout()
            {
                if (!Yii::$app->user->isGuest) {
                    Yii::$app->user->logout();

                      return $this->goHome();
                }
            }

            public function actionProfile()
            {
                $user = new User();
                $books = $user->getBooksByUserId(Yii::$app->user->identity['id']);
                $elements = $user->getElementsByUserId(Yii::$app->user->identity['id']);
                $offers  =  FightOffer::find()->where(['id_second_user'  =>  Yii::$app->user-
>identity['id']])->all();
                $ownOffers  =  FightOffer::find()->where(['id_first_user'  =>  Yii::$app->user-
>identity['id']])->all();
                $imageName = Yii::$app->user->identity['image_avatar'];
                return $this->render('profile', compact('books', 'elements',
                    'offers', 'ownOffers', 'imageName'));
            }

            public function actionEditProfile()
            {
                $editProfileForm = new EditProfileForm();
                $imageUploadingModel = new ImageUploading();

                if (isset($_POST['EditProfileForm'])) {
                    $editProfileForm->attributes            =            Yii::$app->request-
>post('EditProfileForm');

                    $typeImage = 'editingProfile';
                    $file = UploadedFile::getInstance($editProfileForm, 'image_avatar');
                    if (!empty($file)) {

          $nameImage = $imageUploadingModel->uploadImage($file, $typeImage);
                        $editProfileForm->image_avatar = $nameImage;
                    }

                    if ($editProfileForm->validate() && $editProfileForm->edit()) {
                        return $this->goHome();
                    }
                }
```

```
        return $this->render('edit-profile', compact('editProfileForm'));
        }

        public function actionShowBooks()
        {
            $this->layout = false;
            $user = new User();
            $books = $user->getBooksByUserId(Yii::$app->user->identity['id']);
            return $this->render('book-cart', compact('books'));
        }

        public function actionDeleteBook($id)
        {
            $this->layout = false;
            $user = new User();
            $user->deleteBook($id);
            $books = $user->getBooksByUserId(Yii::$app->user->identity['id']);
            return $this->render('book-cart', compact('books'));
        }

        public function actionShowElements()
        {
            $this->layout = false;


            $user = new User();
            $elements = $user->getElementsByUserId(Yii::$app->user->identity['id']);
            return $this->render('window-elements', compact('elements'));
        }

        public function actionDeleteElement($id)
        {
            $this->layout = false;
            $user = new User();
            $user->deleteElement($id);
            $elements = $user->getElementsByUserId(Yii::$app->user->identity['id']);
            return $this->render('window-elements', compact('elements'));
        }

        public function actionViewUser($id)
        {
            $user = User::findOne($id);
            return $this->render('view-user', compact('user'));
        }

        public function actionShowFightOffers()
        {
            $this->layout = false;
            $user = new User();
            $fightOfferExercisesModel = new FightOfferExercise();

            $offers         =         $user->getFightOffersBySecondUserId(Yii::$app->user-
>identity['id']);
            $exercises = $fightOfferExercisesModel->getFightOfferExercises();
            return $this->render('window-fightOffers', compact('offers', 'exercises'));
```

```
            }

        public function actionDeleteFightOffer($id)
        {
            $this->layout = false;
            $user = new User();
            $user->deleteFightOfferById($id);
            $offers         =         $user->getFightOffersBySecondUserId(Yii::$app->user-
>identity['id']);

            $fightOfferExercisesModel = new FightOfferExercise();
            $exercises = $fightOfferExercisesModel->getFightOfferExercises();

            return $this->render('window-fightOffers', compact('offers', 'exercises'));
        }

        public function actionAcceptFight($id)
        {
            $this->layout = false;
            $user = new User();
            $user->acceptOfferById($id);
            $offers         =         $user->getFightOffersBySecondUserId(Yii::$app->user-
>identity['id']);

            $fightOfferExercisesModel = new FightOfferExercise();
            $exercises = $fightOfferExercisesModel->getFightOfferExercises();
            return $this->render('window-fightOffers', compact('offers', 'exercises'));
        }

        public function actionShowOwnFightOffers()
        {
                  $this->layout = false;
            $user = new User();
            $fightOfferExercisesModel = new FightOfferExercise();

            $ownOffers        =        $user->getFightOffersByFirstUserId(Yii::$app->user-
>identity['id']);
            $exercises = $fightOfferExercisesModel->getFightOfferExercises();
            return       $this->render('window-ownFightOffers',       compact('ownOffers',
'exercises'));
        }

        public function actionDeleteOwnFightOffer($id)
        {
            $this->layout = false;
            $user = new User();
            $user->deleteFightOfferById($id);
            $ownOffers        =        $user->getFightOffersByFirstUserId(Yii::$app->user-
>identity['id']);
            return $this->render('window-ownFightOffers', compact('ownOffers'));
        }

        public function actionVerify($token)
        {
            $user = User::findOne(['token' => $token]);
```

```
            if (!empty($user)) {
                  if ($user->token == $token)
                      $user->verified = '1';
                  $user->save();
                  return $this->render('verification');
              } else {
                  return $this->goHome();
              }
          }


           public function actionChangePassword($token)
          {
              $user = User::findOne(['token' => $token]);
              $changingPasswordForm = new ChangingPasswordForm();

              if (isset($_POST['ChangingPasswordForm'])) {
                  $changingPasswordForm->attributes          =          Yii::$app->request-
>post('ChangingPasswordForm');
                  if    ($changingPasswordForm->validate()    &&    $changingPasswordForm-
>changePassword($user)) {
                      return $this->goHome();
                  }
              }

              if (!empty($user)) {
                  return                                  $this->render('changing-password',
compact('changingPasswordForm'));
              } else {
                  return $this->goHome();
              }
          }

          public function actionSendMailChangePassword()
          {
              $changingPasswordForm = new ChangingPasswordForm();
              $statusMail = $changingPasswordForm->sendMail();
              return $this->render('mail-changing-password', compact('statusMail'));
          }
```

ПРИЛОЖЕНИЕ Л
ИСХОДНЫЙ КОД МОДЕЛИ ARTICLE

```
<?php

namespace app\models;

use Yii;
use yii\behaviors\TimestampBehavior;
use yii\db\ActiveRecord;
use yii\db\Expression;

class Article extends ActiveRecord
{
    public function behaviors()
    {
        return [
            [
                'class' => TimestampBehavior::className(),
                'attributes' => [
                    ActiveRecord::EVENT_BEFORE_INSERT => ['date_publication'],
                ],
                'value' => new Expression('NOW()'),
            ],
        ];
    }

    public function getArticles()
    {
        $articles = Yii::$app->db->createCommand('SELECT * FROM `article`
            JOIN `user` ON `article`.`id_author` = `user`.`id`
            WHERE `article`.`id_category` = 2 AND `article`.`is_checked` = \'1\'
            ORDER BY `article`.`date_publication`')->queryAll();

         return $articles;
    }

    public function getArticleById($id)
    {
        $article = Yii::$app->db->createCommand('SELECT * FROM `article`
            JOIN `user` ON `article`.`id_author` = `user`.`id`
            WHERE `article`.`id_article` = :id_article',
            [':id_article' => $id])->queryAll();
        return $article;
    }

    public function getFoodArticles()
    {
        $articles = Yii::$app->db->createCommand('SELECT * FROM `article`
            JOIN `user` ON `article`.`id_author` = `user`.`id`
            WHERE `article`.`id_category` = 4 AND `article`.`is_checked` = \'1\'
            ORDER BY `article`.`date_publication`')->queryAll();
        return $articles;
    }
}
```

ПРИЛОЖЕНИЕ М

ИСХОДНЫЙ КОД МОДЕЛИ ARTICLECREATINGFORM

```
<?php

namespace app\models;

use Yii;
use yii\base\Model;

class ArticleCreatingForm extends Model
{
    public $name_article;
    public $annotation;
    public $content;
    public $main_image;
    public $is_advertising;

    public function attributeLabels()
    {
        return [
            'name_article' => 'Название статьи:',
            'annotation' => 'Аннотация',
            'content' => 'Контент:',
            'main_image' => 'Главная картинка:',
            'is_advertising' => 'Рекламная статья',
        ];
    }

    public function rules()
    {
        return [
            [['name_article', 'content', 'annotation'], 'required'],
            ['content', 'string', 'min' => 10, 'max' => 6000],
            ['name_article', 'string', 'min' => 4, 'max' => 30],
            ['annotation', 'string', 'min' => 4, 'max' => 60],

  ['main_image', 'file', 'extensions' => ['png', 'jpg']],
            [['is_advertising'], 'safe'],
        ];
    }

    public function createArticle($categoryId)
    {
        $article = new Article();
        $article->name_article = $this->name_article;
        $article->annotation = $this->annotation;
        $article->content = $this->content;
        $article->id_author = Yii::$app->user->identity['id'];
        $article->id_category = $categoryId;
        $article->main_image = $this->main_image;
        $article->is_advertising = $this->is_advertising;
        return $article->save();
    }
}
```

ПРИЛОЖЕНИЕ Н

ИСХОДНЫЙ КОД МОДЕЛИ AUTHORIZATIONFORM

```
<?php
/**
 * Created by PhpStorm.
 * User: Aleks
 * Date: 05.10.2018
 * Time: 9:03
 */

namespace app\models;

use yii\base\Model;
use app\models\User;
use Yii;

class AuthorizationForm extends Model
{
    public $login;
    public $password;

    public function attributeLabels()
    {
        return [
            'login' => 'Логин:',
            'password' => 'Пароль:',
        ];
    }

    public function rules()
    {
        return [
            [['login', 'password'], 'required'],
            ['password', 'validatePassword'],
        ];
    }

    public function validatePassword($attribute)
    {
        $user = $this->getUser();
        if (!$user || !Yii::$app->security->validatePassword($this->password, $user-
>password)) {
            $this->addError($attribute, 'Логин/пароль введены неверно');
        }
    }

    public function getUser()
    {
        return User::findOne(['login' => $this->login]);
    }
}
```

ПРИЛОЖЕНИЕ П

ИСХОДНЫЙ КОД МОДЕЛИ CHALLENGE

```
<?php

namespace app\models;

use Yii;
use yii\behaviors\TimestampBehavior;
use yii\db\ActiveRecord;
use yii\db\Expression;

class Challenge extends ActiveRecord
{
    public function behaviors()
    {
        return [
            [
                'class' => TimestampBehavior::className(),
                'attributes' => [
                    ActiveRecord::EVENT_BEFORE_INSERT => ['date_creation'],
                ],
                'value' => new Expression('NOW()'),
            ],
        ];
    }

    public function getChallenges(){
        $challenges = Yii::$app->db->createCommand('SELECT * FROM `challenge`
            JOIN `user` ON `challenge`.`id_creator` = `user`.`id`
            WHERE `challenge`.`is_checked` = \'1\'
            ORDER BY `challenge`.`date_creation`')->queryAll();
        return $challenges;
    }


public function getChallengeById($id){
        $challenge = Yii::$app->db->createCommand('SELECT * FROM `challenge`
            JOIN `user` ON `challenge`.`id_creator` = `user`.`id`
            WHERE `challenge`.`id_challenge` = :id_challenge',
            [':id_challenge' => $id])->queryAll();
        return $challenge;
    }
}
```

ПРИЛОЖЕНИЕ Р

ИСХОДНЫЙ КОД МОДЕЛИ CHALLENGECREATINGFORM

```
<?php

namespace app\models;

use Yii;
use yii\base\Model;

class ChallengeCreatingForm extends Model
{
    public $name_challenge;
    public $video_reference;
    public $description;

    public function attributeLabels()
    {
        return [
            'name_challenge' => 'Название челленджа:',
            'video_reference' => 'Ссылка на видео:',
            'description' => 'Описание челленджа:',
        ];
    }

    public function rules()
    {
        return [
            [['name_challenge', 'video_reference', 'description'], 'required'],
            ['name_challenge', 'string', 'min' => 4, 'max' => 50],
            ['video_reference', 'string', 'min' => 7, 'max' => 1000],
            ['description', 'string', 'min' => 10, 'max' => 2000],
        ];
    }


 public function signUp()
    {
        $challenge = new Challenge();
        $challenge->name_challenge = $this->name_challenge;
        $challenge->video_reference = $this->video_reference;
        $challenge->description = $this->description;
        $challenge->id_creator = Yii::$app->user->identity['id'];
        return $challenge->save();
    }
}
```

ПРИЛОЖЕНИЕ С

ИСХОДНЫЙ КОД МОДЕЛИ CHALLENGECREATINGFORM

```
<?php

namespace app\models;

use Yii;
use yii\base\Model;

class ChallengeParticipantForm extends Model
{
    public $video_reference;

    public function attributeLabels()
    {
        return [
            'video_reference' => 'Ссылка на видео:',
        ];
    }

    public function rules()
    {
        return [
            [['video_reference'], 'required'],
            ['video_reference', 'string', 'min' => 7],
        ];
    }

    public function takePart($id)
    {
        $challengeParticipation = new ChallengeParticipation();
        $challengeParticipation->id_challenge = $id;
        $challengeParticipation->id_participant = Yii::$app->user->identity['id'];
        $challengeParticipation->video_reference = $this->video_reference;
        return $challengeParticipation->save();

 }
}
```

ПРИЛОЖЕНИЕ Т

ИСХОДНЫЙ КОД МОДЕЛИ CHANGINGPASSWORDFORM

```
<?php

namespace app\models;

use Yii;
use yii\base\Model;

class ChangingPasswordForm extends Model
{
    public $login;
    public $password;
    public $newPassword;
    public $newRepeatPassword;

    public function attributeLabels()
    {
        return [
            'login' => 'Логин:',
            'password' => 'Текущий пароль:',
            'newPassword' => 'Новый пароль:',
            'newRepeatPassword' => 'Повторный новый пароль:',
        ];
    }

    public function rules()
    {
        return [
            [['login', 'password', 'newPassword', 'newRepeatPassword'], 'required'],
            ['newPassword', 'string', 'min' => 5],
            ['newRepeatPassword',    'compare',    'compareAttribute'=>'newPassword',
'message'=>"Пароли не совпадают"],
            ['password', 'validatePassword'],
        ];
    }


   public function validatePassword($attribute){
          $user = $this->getUser();
          if(!$user  ||  !Yii::$app->security->validatePassword($this->password,  $user-
>password)){
              $this->addError($attribute, 'Логин/пароль введены неверно');
          }
    }

    public function getUser(){
        return User::findOne(['login' => $this->login]);
    }

    public function changePassword($user){
        $user->password       =       Yii::$app->security->generatePasswordHash($this-
>newPassword);
        return $user->save();
    }
```

```
        public function sendMail(){
              $user = User::findOne(['id' => Yii::$app->user->identity['id']]);
              if(!empty($user['token'])) {
                  $url = 'street-workout.ru/site/change-password?token=' . $user['token'];
                  $login = $user['login'];
                  Yii::$app->mailer->compose('changing-password', compact('url', 'login'))
                      ->setFrom(['ninja_241296@mail.ru'     =>     'Street     Workout     и
саморазвитие'])
                      ->setTo($user['email'])
                      ->setSubject('Changing Password')
                      ->send();
                  return true;
              }
             return false;
          }
      }
```

ПРИЛОЖЕНИЕ У

ИСХОДНЫЙ КОД МОДЕЛИ COMPETITIONCREATINGFORM

```
<?php

namespace app\models;

use yii\base\Model;
use Yii;

class CompetitionCreatingForm extends Model
{
    public $competition_name;
    public $city;
    public $id_type_competition;
    public $date;
    public $age_from;
    public $age_to;
    public $id_creator;

    public function attributeLabels()
    {
        return [
            'competition_name' => 'Название соревнования:',
            'city' => 'Город:',
            'id_type_competition' => 'Тип соревнования:',
            'date' => 'Дата:',
            'age_from' => 'Возраст от:',
            'age_to' => 'до:',
        ];
    }

    public function rules()
    {
        return [



   [['competition_name', 'city', 'id_type_competition', 'date', 'age_from','age_to'],
'required'],
            ['competition_name', 'string', 'min' => 3],
            ['city', 'string', 'min' => 3, 'max' => 50],
            ['age_from', 'integer', 'max' => 100],
            ['age_to', 'integer', 'max' => 100],
        ];
    }

    public function signUp(){
        $competition = new Competition();
        $competition->competition_name = $this->competition_name;
        $competition->city = $this->city;
        $competition->id_type_competition = $this->id_type_competition;
        $competition->id_creator = Yii::$app->user->identity['id'];
        $competition->date = $this->date;
        $competition->age_from = $this->age_from;
```

```
  $competition->age_to = $this->age_to;
        return $competition->save();
    }
}
```

ПРИЛОЖЕНИЕ Ф

ИСХОДНЫЙ КОД МОДЕЛИ FIGHT

```
<?php

namespace app\models;

use yii\db\ActiveRecord;
use Yii;

class Fight extends ActiveRecord
{
    public function getFights()
    {
        $fights = Yii::$app->db->createCommand('SELECT * FROM `fight`
            JOIN `exercise` ON `fight`.`id` = `exercise`.`id_fight`
            JOIN `user_fight` ON `fight`.`id`= `user_fight`.`id_fight`
            JOIN `user` ON `user_fight`.`id_user`= `user`.`id`
            WHERE `user_fight`.`is_winner` IS NULL')->queryAll();
        return $fights;
    }

    public function getParticipants()
    {
        $participants = Yii::$app->db->createCommand('SELECT * FROM `user_fight`
            JOIN `user` ON `user_fight`.`id_user`= `user`.`id`
            WHERE `user_fight`.`is_winner` IS NULL')->queryAll();
        return $participants;
    }

    public function getExercises()
    {
     $exercises = Yii::$app->db->createCommand('SELECT * FROM `exercise_fight`
            JOIN `fight` ON `exercise_fight`.`id_fight`= `fight`.`id`
            JOIN  `exercise`  ON  `exercise_fight`.`id_exercise`=  `exercise`.`id`')-
>queryAll();
        return $exercises;
    }
}
```

ПРИЛОЖЕНИЕ Х

ИСХОДНЫЙ КОД МОДЕЛИ FIGHTCREATINGFORM

```
<?php

namespace app\models;

use yii\base\Model;
use Yii;

class FightCreatingForm extends Model
{
    public $id_first_user;
    public $id_second_user;
    public $city;
    public $date_fight;
    public $id_exercise1;
    public $id_exercise2;
    public $id_exercise3;
    public $playground_address;


    public function attributeLabels()
    {
        return [
            'city' => 'Город:',
            'date_fight' => 'Дата зарубы:',
            'id_exercise1' => 'Первое упражнение:',
            'id_exercise2' => 'Второе упражнение:',
            'id_exercise3' => 'Третье упражнение:',
            'playground_address' => 'Адрес площадки'
        ];
    }

    public function rules()
    {
        return [
            [['city', 'date_fight', 'playground_address',

'id_exercise1','id_exercise2','id_exercise3'], 'required'],
            ['city', 'string', 'min' => 3, 'max' => 70],
            ['playground_address', 'string', 'min' => 5, 'max' => 100],
        ];
    }

    public function signUp($id){
        $isSave = true;
        $fight_offer = new FightOffer();
        $fight_offer->id_first_user = Yii::$app->user->identity['id'];
        $fight_offer->id_second_user = $id;
        $fight_offer->city = $this->city;
        $fight_offer->date_fight = $this->date_fight;
        $fight_offer->playground_address = $this->playground_address;
        if(!$fight_offer->save()){
            $isSave = false;
        }
```

```
            $lastIdFightOffer         =         FightOffer::find()->select(['id_offer'])-
>orderBy(['id_offer' => SORT_DESC])->one();

            $fight_offer_exercise = new FightOfferExercise();
            $fight_offer_exercise->id_fight_offer = $lastIdFightOffer['id_offer'];
            $fight_offer_exercise->id_exercise = $this->id_exercise1;
            if(!$fight_offer_exercise->save()){
                $isSave = false;
            }

            $fight_offer_exercise = new FightOfferExercise();
         $fight_offer_exercise->id_fight_offer = $lastIdFightOffer['id_offer'];
            $fight_offer_exercise->id_exercise = $this->id_exercise2;
            if(!$fight_offer_exercise->save()){
                $isSave = false;
            }

            $fight_offer_exercise = new FightOfferExercise();
            $fight_offer_exercise->id_fight_offer = $lastIdFightOffer['id_offer'];
            $fight_offer_exercise->id_exercise = $this->id_exercise3;
            $fight_offer_exercise->bonus_exercise = '1';
            if(!$fight_offer_exercise->save()){
                $isSave = false;
            }

            if($isSave) {
                return true;
            }
        }
    }
```

ПРИЛОЖЕНИЕ Ц

ИСХОДНЫЙ КОД МОДЕЛИ IMAGEUPLOADING

```
<?php

namespace app\models;

use Yii;
use yii\base\Model;
use yii\web\UploadedFile;

class ImageUploading extends Model
{
    public function uploadImage(UploadedFile $file, $typeImage)
    {
        $filename = strtolower(md5(uniqid($file->baseName)) . '.' . $file->extension);

        if($typeImage == 'article') {
            $file->saveAs(Yii::getAlias('@web') . 'images/uploads/articles/' . $filename);
        }

        if($typeImage == 'editingProfile') {
            $file->saveAs(Yii::getAlias('@web') . 'images/uploads/profile/' . $filename);
        }

        if($typeImage == 'sportsFacility') {
            $file->saveAs(Yii::getAlias('@web') . 'images/uploads/sports_objects/' . $filename);
        }

        return $filename;
    }

}
```

ПРИЛОЖЕНИЕ Ш

ИСХОДНЫЙ КОД МОДЕЛИ REGISTRATIONFORM

```
<?php
/**
 * Created by PhpStorm.
 * User: Aleks
 * Date: 05.10.2018
 * Time: 7:33
 */

namespace app\models;

use yii\base\Model;
use app\models\User;
use Yii;

class RegistrationForm extends Model
{
    public $login;
    public $email;
    public $password;
    public $repeatPassword;
    public $acceptingRules;

    public function attributeLabels()
    {
        return [
            'login' => 'Логин:',
            'email' => 'E-mail:',
            'password' => 'Пароль:',
            'repeatPassword' => 'Повторный пароль:',
            'acceptingRules' => 'Принять правила сообщества',
        ];
    }

    public function rules()
    {
        return [


       [['login', 'email', 'password', 'repeatPassword'], 'required'],
                ['login', 'string', 'min' => 4],
                ['login', 'unique', 'targetClass'=>'app\models\User'],
                ['email', 'email'],
                ['email', 'unique', 'targetClass'=>'app\models\User'],
                ['password', 'string', 'min' => 5],
                ['repeatPassword',       'compare',       'compareAttribute'=>'password',
'message'=>"Пароли не совпадают"],
                ['acceptingRules',   'compare',   'compareValue'   =>   1,   'message'   =>
'Необходимо принять правила сообщества!']
        ];
    }

    public function signUp($token){
        $user = new User();
```

```
        $user->login = $this->login;
         $user->email = $this->email;
         $user->password = Yii::$app->security->generatePasswordHash($this->password);
         $user->token = $token;
         return $user->save();
    }

    public function verifyUser($token){
         $url = 'street-workout.ru/site/verify?token='.$token;
         $login = $this->login;
         Yii::$app->mailer->compose('verification', compact('url', 'login'))
             ->setFrom(['ninja_241296@mail.ru'=>'Street Workout и саморазвитие'])
             ->setTo($this->email)

     ->setSubject('Verification')
             ->send();
         return true;
    }
}
```

ПРИЛОЖЕНИЕ Щ

ИСХОДНЫЙ КОД МОДЕЛИ SPORTSFACILITY

```
<?php

namespace app\models;

use Yii;
use yii\db\ActiveRecord;

class SportsFacility extends ActiveRecord
{
    public static function tableName()
    {
        return 'sports_facility';
    }

    public function getSportsObjectById($id)
    {
        $sportsObject = Yii::$app->db->createCommand('SELECT * FROM `sports_facility`
            JOIN `user` ON `sports_facility`.`id_author` = `user`.`id`
            WHERE `sports_facility`.`id_facility` = :id_facility',
            [':id_facility' => $id])->queryAll();
        return $sportsObject;
    }

    public function getSportsObjects()
    {
        $sportsObjects     =     Yii::$app->db->createCommand('SELECT     *     FROM
`sports_facility`
            JOIN `user` ON `sports_facility`.`id_author` = `user`.`id`
            WHERE `sports_facility`.`is_checked` = \'1\'')->queryAll();
        return $sportsObjects;
    }



   public function getFacilityComment($id)
    {
        $commentFacility  =  Yii::$app->db->createCommand('SELECT  `content`,`id_user`,
`login`
            FROM `sports_facility_comment`
            JOIN `user` ON `sports_facility_comment`.`id_user` = `user`.`id`
            WHERE `sports_facility_comment`.`id_facility` = :id_facility',
            [':id_facility' => $id])->queryAll();
        return $commentFacility;
    }

}
```

ПРИЛОЖЕНИЕ Э

ИСХОДНЫЙ КОД МОДЕЛИ SPORTSFACILITYCREATINGFORM

```
<?php

namespace app\models;

use Yii;
use yii\base\Model;

class SportsFacilityCreatingForm extends Model
{
    public $name_facility;
    public $address;
    public $latitude;
    public $longitude;
    public $main_image;
    public $description;

    public function attributeLabels()
    {
        return [
            'name_facility' => 'Названиe:',
            'address' => 'Адрес:',
            'latitude' => 'Широта:',
            'longitude' => 'Долгота:',
            'main_image' => 'Главная картинка:',
            'description' => 'Описание:'
        ];
    }

    public function rules()
    {
        return [
            [['name_facility', 'latitude', 'longitude', 'address'], 'required'],
            ['description', 'safe'],
            ['name_facility', 'string', 'min' => 4, 'max' => 50],


 ['address', 'string', 'min' => 4, 'max' => 50],
            ['description', 'string', 'min' => 4, 'max' => 200],
            ['main_image', 'file', 'extensions' => ['png', 'jpg']],
            ['latitude', 'double'],
            ['longitude', 'double'],
        ];
    }

    public function createFacility()
    {
        $facility = new SportsFacility();
        $facility->name_facility = $this->name_facility;
        $facility->address = $this->address;
        $facility->id_author = Yii::$app->user->identity['id'];
        $facility->latitude = $this->latitude;
        $facility->longitude = $this->longitude;
        $facility->main_image = $this->main_image;
```

```
   if ($this->description != null) {
            $facility->description = $this->description;
        }
        return $facility->save();
    }
}
```

ПРИЛОЖЕНИЕ Ю
ИСХОДНЫЙ КОД МОДЕЛИ USER

```
<?php

namespace app\models;

use yii\db\ActiveRecord;
use yii\web\IdentityInterface;
use Yii;

class User extends ActiveRecord implements IdentityInterface
{
    public static function findIdentity($id)
    {
        return self::findOne($id);
    }

    public static function findIdentityByAccessToken($token, $type = null)
    {

    }

    public function getId()
    {
        return $this->id;
    }

    public function getAuthKey()
    {

    }

    public function validateAuthKey($authKey)
    {

    }


   /**
     * @param $id
     * @return string
     */
    public function getBooksByUserId($id)
    {
        $books = Yii::$app->db->createCommand('SELECT `book`.`name`, `book`.`image`,
`book`.`id` FROM `user`
            JOIN `user_book` ON `user`.`id` = `user_book`.`id_user`
            JOIN `book` ON `user_book`.`id_book`= `book`.`id`
            WHERE `user`.`id` = :id', [':id' => $id])->queryAll();
        return $books;
    }

    public function deleteBook($id)
    {
        Yii::$app->db->createCommand('DELETE FROM `user_book`
```

```
      WHERE `user_book`.`id_user` = :id_user AND `user_book`.`id_book` = :id_book',
                ['':id_book'  =>  $id,  ':id_user'  =>  Yii::$app->user->identity['id']])-
>query();
            return true;
        }

        public function getElementsByUserId($id)
        {
            $elements = Yii::$app->db->createCommand('SELECT `element`.`name`,
                    `element`.`image`, `element`.`id`, `element`.`level`
                FROM `user`
                JOIN `user_element` ON `user`.`id` = `user_element`.`id_user`
                JOIN `element` ON `user_element`.`id_element`= `element`.`id`

        WHERE `user`.`id` = :id', [':id' => $id])->queryAll();
            return $elements;
        }

        public function deleteElement($id)
        {
            Yii::$app->db->createCommand('DELETE FROM `user_element`
            WHERE  `user_element`.`id_user`  =  :id_user  AND  `user_element`.`id_element`  =
:id_element',
                [':id_element'  =>  $id,  ':id_user'  =>  Yii::$app->user->identity['id']])-
>query();
            return true;
        }

        public function getFightOffersBySecondUserId($id)
        {
            $offers = Yii::$app->db->createCommand('SELECT *
                FROM `fight_offer`
                JOIN `user` ON `fight_offer`.`id_first_user` = `user`.`id`
                WHERE `fight_offer`.`id_second_user` = :id',
                [':id' => $id])->queryAll();
            return $offers;
        }

        public function deleteFightOfferById($id_offer)
        {
            Yii::$app->db->createCommand('DELETE FROM `fight_offer_exercise`
            WHERE `fight_offer_exercise`.`id_fight_offer` = :id',
                [':id' => $id_offer])->query();
            Yii::$app->db->createCommand('DELETE FROM `fight_offer`
            WHERE `fight_offer`.`id_offer` = :id',

         [':id' => $id_offer])->query();
            return true;
        }

        public function getFightOffersByFirstUserId($id)
        {
            $ownOffers = Yii::$app->db->createCommand('SELECT *
                FROM `fight_offer`
                JOIN `user` ON `fight_offer`.`id_second_user` = `user`.`id`
                WHERE `fight_offer`.`id_first_user` = :id',
```

```
            [':id' => $id])->queryAll();
                return $ownOffers;
            }

            public function acceptOfferById($id_offer)
            {
                $offer = FightOffer::findOne($id_offer);
                $fightModel = new Fight();
                $fightModel->city = $offer->city;
                $fightModel->playground_address = $offer->playground_address;
                $fightModel->date_fight = $offer->date_fight;
                $fightModel->save();

                $fightOfferExercises = FightOfferExercise::find()->where(['id_fight_offer' =>
$id_offer])
                    ->asArray()->all();
                $lastFight = Fight::find()->orderBy(['id' => SORT_DESC])->one();
                for ($i = 0; $i < count($fightOfferExercises); $i++) {
                    $exerciseFightModel = new ExerciseFight();
                    $exerciseFightModel->id_fight = $lastFight['id'];
            $exerciseFightModel->id_exercise = $fightOfferExercises[$i]['id_exercise'];
                    if ($i != 2) {
                        $exerciseFightModel->bonus_exercise = 0;
                    } else {
                        $exerciseFightModel->bonus_exercise = 1;
                    }
                    $exerciseFightModel->save();
                }

                $userFightModel = new UserFight();
                $userFightModel->id_fight = $lastFight['id'];
                $userFightModel->id_user = $offer->id_first_user;
                $userFightModel->save();

                $userFightModel = new UserFight();
                $userFightModel->id_fight = $lastFight['id'];
                $userFightModel->id_user = $offer->id_second_user;
                $userFightModel->save();

                Yii::$app->db->createCommand('DELETE FROM `fight_offer_exercise`
                WHERE `fight_offer_exercise`.`id_fight_offer` = :id',
                    [':id' => $id_offer])->query();
                Yii::$app->db->createCommand('DELETE FROM `fight_offer`
                WHERE `fight_offer`.`id_offer` = :id',
                    [':id' => $id_offer])->query();
                return true;
            }
        }
```